HIGH RESOLUTION ALGORITHMS FOR THE NAVIER STOKES EQUATIONS FOR GENERALIZED DISCRETIZATIONS

by

Curtis Randall Mitchell

Dissertation submitted to the Faculty of the

Virginia Polytechnic Institute and State University

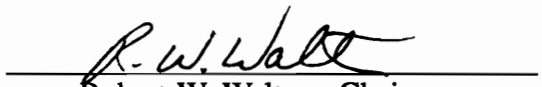
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

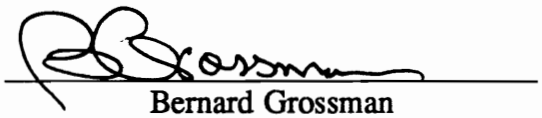
in

Aerospace Engineering

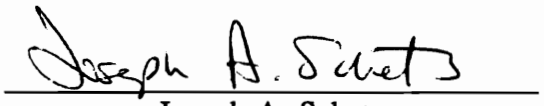
APPROVED:



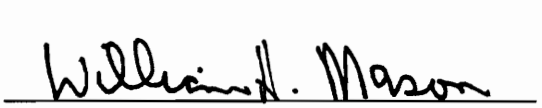
Robert W. Walters, Chairman



Bernard Grossman



Joseph A. Schetz



William H. Mason

Wing F. Ng

December 1992

Blacksburg, Virginia

HIGH RESOLUTION ALGORITHMS FOR THE NAVIER STOKES EQUATIONS FOR GENERALIZED DISCRETIZATIONS

by

Curtis Randall Mitchell

Committee Chairman: Robert W. Walters

Aerospace Engineering

(ABSTRACT)

Accurate finite volume solution algorithms for the two dimensional Navier Stokes equations and the three dimensional Euler equations for both structured and unstructured grid topologies are presented. Results for two dimensional quadrilateral and triangular elements and three dimensional tetrahedral elements will be provided. Fundamental to the solution algorithm is a technique for generating multidimensional polynomials which model the spatial variation of the flow variables. Cell averaged data is used to reconstruct pointwise distributions of the dependent variables. The reconstruction errors are evaluated on triangular meshes. The implementation of the algorithm is unique in that three reconstructions are performed for each cell face in the domain. Two of the reconstructions are used to evaluate the inviscid fluxes and correspond to the right and left interface states needed for the solution of a Riemann problem. The third reconstruction is used to evaluate the viscous fluxes. The gradient terms that appear in the viscous fluxes are formed by simply differentiating the polynomial. By selecting the appropriate cell control volumes, centered, upwind and upwind-biased stencils are possible. Numerical calculations in two dimensions include solutions to elliptic boundary value problems,

Ringleb's flow, an inviscid shock reflection, a flat plate boundary layer, and a shock induced separation over a flat plate. Three dimensional results include the ONERA M6 wing. All of the unstructured grids were generated using an advancing front mesh generation procedure. Modifications to the three dimensional grid generator were necessary to discretize the surface grids for bodies with high curvature. In addition, mesh refinement algorithms were implemented to improve the surface grid integrity. Examples studied include a Glasair fuselage, High Speed Civil Transport, and the ONERA M6 wing. The role of reconstruction as applied to adaptive remeshing is discussed and a new first order error estimator is presented. Numerical examples of the remeshing procedure include both smooth and discontinuous flows.

ACKNOWLEDGMENTS

The author wishes to express his sincerest thanks to his academic advisor Dr. Robert Walters for providing this academic and research opportunity. His vast technical knowledge and keen insight in his area of expertise has resulted in a very positive and rewarding experience. The author would also like to thank Dr. Ray Cosner and the McDonnell Douglas Corporation for sponsoring this three and a half year research effort. Without his loyal and financial support, this education would not have been possible. The author also wishes to thank the personnel and students of the Aerospace Engineering department for their friendship and invaluable discussions pertaining to his research.

The author would also like to thank his parents and family for sharing in his academic dreams. Their encouragement, enthusiasm and confidence towards their son's capabilities were truely a source of inspiration. Most of all, I wish to thank my wife Suzanne and my two daughters Morgan and Kelsey for their loving support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

$C_{i,j,l}$	$k-exact$ polynomial coefficients
e	internal energy per unit mass
e_0	total energy per unit mass
$\bar{e}$	solution error
f, g, h	inviscid flux vectors
f_v, g_v, h_v	viscous flux vectors
h	enthalpy per unit mass
h_0	stagnation enthalpy per unit mass
J	Jacobian of transformation
k	degree of reconstruction polynomial
$k(m)$	list of faces surrounding cell m
k_i	degree of the inviscid reconstruction polynomial
k_v	degree of the viscous reconstruction polynomial
L_p	standard p norm
$\hat{n}$	outward facing unit normal
N	number of quadrature points
N_f	number of quadrature points used in flux integration
N_r	number of quadrature points used in reconstruction integration
n_x, n_y, n_z	direction cosines in the x, y, z directions

P^k	polynomial of degree k
p	thermodynamic pressure
Pr	Prandtl number
$\bar{Q}$	conservative cell average fluid dynamic variable
$\bar{q}$	primitive cell average fluid dynamic variable
Q_L, Q_R	left and right fluid dynamic states
R	radius of curvature
T	thermodynamic temperature
R_m	stencil set for cell m
R_c	number of polynomial coefficients
Re	Reynolds number
S_m	surface bounding cell m
u, v, w	Cartesian velocity components
W	cell average weighting factors
x, y, z	Cartesian coordinates
$\bar{\mathbf{x}}$	vector of Cartesian coordinates
ω_f	quadrature weights used in flux integration
Ω_m	domain for cell m
ω_r	quadrature weights used in reconstruction integration
ξ, η, ζ	reference frame coordinates
ξ_f, η_f	quadrature coordinates used in flux integration
ξ_r, η_r, ζ_r	quadrature coordinates used in reconstruction integration
$\bar{\xi}$	vector of reference frame coordinates

ρ	density
ϕ, κ	MUSCL differencing higher order accuracy parameters
Ψ	matrix function for grid topology
Φ	vector of cell averages
∞	free stream conditions
μ	molecular viscosity
λ	bulk viscosity
γ	ratio of specific heats
τ	shear stress

Subscripts

f	face identifier
L	Left state variable
n	node identifier
R	Right state variable
v	viscous quantity
x	Component in the x direction
y	Component in the y direction
z	Component in the z direction
0	Stagnation quantity or cell centroid

Accents

$\hat{(\cdot)}$	Unit vector
$\tilde{(\cdot)}$	Roe averaged quantity
$\bar{(\cdot)}$	cell average quantity or a transformed space

Chapter 1

Introduction

1.1 Overview

In the field of computational fluid dynamics (CFD), researchers have been dedicated to developing efficient and accurate numerical procedures to determine the flow field behavior over arbitrary aerodynamic configurations. The intricate relationship between grid generation and flow solver capabilities has somewhat dictated the successes and failures during the evolution of this technology. The quality of the discretized domain directly affects the numerical stability and accuracy of the flow predictions. Grid quality measurements including mesh smoothness and uniformity are necessary to assess the state of the grid distribution. To date, the numerical techniques which solve the governing equations of fluid mechanics are sensitive to mesh irregularities whereby grid skewness and distortion adversely affects algorithm robustness and solution accuracy. Often times, grid skewness cannot be avoided due to the complex physical features associated with the body. This is especially true for structured grid generators that rely on hexahedral discretizations to describe the interior domain. Structured mesh generation procedures depend on mathematically based algorithms to determine the point distribution by solving algebraic or differential equations. While this process generally produces a smooth variation of cells, the lack of flexibility due to mesh connectivity requirements complicates the task and constitutes the majority of the grid generation effort.

On the other hand, very powerful numerical procedures for solving the governing hyperbolic systems of conservation laws on structured meshes have been developed and

made available for the past several years. Collectively, the schemes are referred to as upwind or flux-split schemes and encompass a variety of upwind formulations based on exact or approximate solutions to the Riemann problem [1] [2] [3] [5] [4] . The spatial accuracy of these methods is determined by the accuracy of the interpolating polynomial used to fit the data. An alternative approach to the upwind methods is based on the pioneering work of Jameson et al. [29] . He developed explicit Runge Kutta schemes with residual smoothing in order to control the unbounded growth of high frequency waves. At the forefront of high order accurate shock capturing schemes is an approach based on Essentially Non-oscillatory (ENO) interpolating techniques. Notable among this group were the efforts of Chakravarthy [6] , Osher [7] and Harten [8] . A predecessor to this technology is the total variation diminishing (TVD) scheme [27] which, by design, avoids numerical oscillations, thus capturing discontinuities with high resolution. However, in order to prevent the total variation from increasing, the accuracy is reduced to first order at local extrema via limiters. This local effect restricts the maximum global accuracy obtainable. An alternative approach is to allow the total variation to increase in a bounded fashion. The ENO formulation permits this by specifying reconstruction criteria that result in uniform accuracy over the entire domain. The procedure involves forming higher order terms consistent with Taylor series expansions. Arbitrary accuracy levels are achievable provided that the domain is large enough to support the stencil of cells used in the reconstruction step. Non-oscillatory uniform accuracy behavior is possible for hyperbolic systems of equations only if this stencil changes from cell to cell and from time level to time level [9] .

The performance of these methods are all subjected to the non-uniformities associated with the meshes. In theory, their claimed accuracy is applicable to Cartesian domains where truly one-dimensional interpolation procedures can be developed. The issue of interpolation accuracy on one-dimensional non-uniform meshes has been addressed in

the past. The most common implementation of one dimensional interpolation for multidimensional applications is a procedure based on one-dimensional interpolation along computational coordinate lines (constant η, ξ, and ζ). For bodies with high curvature, the discretization limitations and inadequacies are directly affecting the solution quality regardless of the choice of solution algorithm described above.

It's time for a change in philosophy. If we are to improve our ability to accurately predict the flow physics around arbitrarily complex bodies, we must first address the issue of generating the best possible grid which most accurately represents the computational domain. The grid skewness and non-uniformities must be eliminated over regions of high curvature, and the flow solver algorithms must be valid for the discretization of the generated meshes.

1.2 The Unstructured Arena

As an alternative, unstructured grid generation techniques and unstructured solution algorithms are becoming increasingly popular in the CFD community. The inherent features of unstructured grids offer improved control over the point distribution, mesh size, and mesh shape. In addition, these grids are more suitable for adaptive refinement/remeshing techniques and incorporating moving boundaries due to the generalized indexing strategies. In short, their lack of directionality offers greater flexibility for discretizing complicated bodies. With the availability of supercomputers, complete unstructured flow field computations over complex aircraft configurations are possible. Flow over a High Speed Civil Transport at Mach 3.0 shown in Figures 1.1 and 1.2 is one example. This aircraft was discretized using a three dimensional unstructured advancing front grid generator.

Although the grid generation and flow solver technologies are still in their infancy for unstructured grids, this approach is receiving attention and has progressed consider-

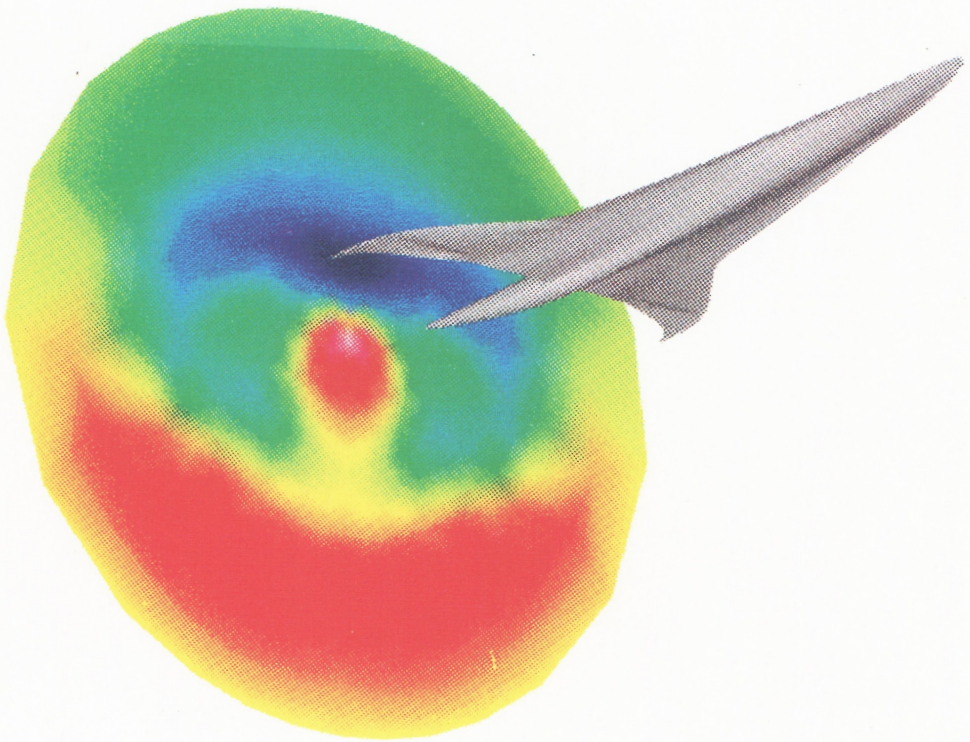


Fig. 1.1 – Outflow pressure contours for flow over a High Speed Civil Transport, $M_\infty = .84$ and $\alpha = 3.06°$

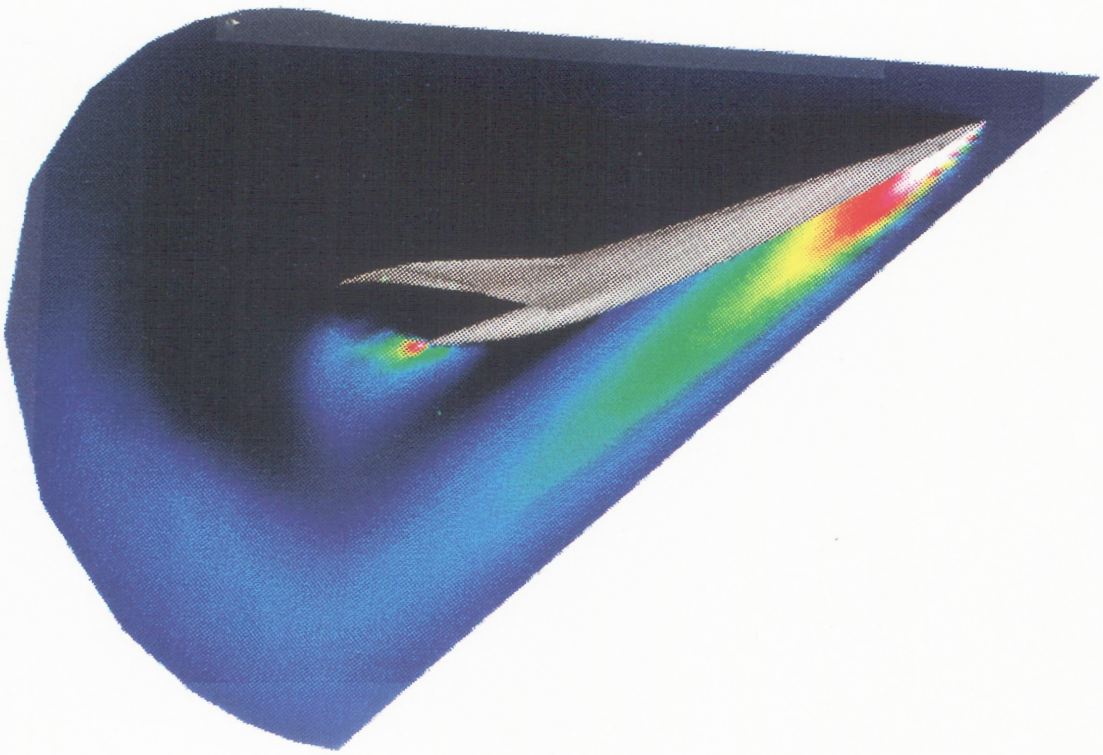


Fig. 1.2 – Symmetry plane pressure contours for flow over a High Speed Civil Transport, $M_\infty = .84$ and $\alpha = 3.06°$

ably over the past few years. Common to this grid generation technique is the creation of an unstructured background grid that stores the grid parameters which determine the mesh spacing characteristics. These quantities are stored at the nodes of the background grid and interpolated to the desired coordinates during the mesh generation phase. However, the user's effort in generating the background grid and specifying the grid control parameters becomes prohibitively expensive. The recent achievements in the advancing front grid generation arena are attributed to the efforts of Pirzadeh [10] [11] , and thus involve an alternative approach which greatly simplifies the creation of the background grid. He substitutes the complicated triangular/tetrahedral background meshes in favor of a simple Cartesian mesh where the solution to a Poisson equation with specified point and lines sources determines the mesh spacing parameters. The location of the point and line sources correspond to the important features of the geometry. For example, the leading edge of a wing could be modeled as a line source which would result in a concentration of grid points around the leading edge. The solution of the elliptic equations are generally quite smooth. As a result of his efforts, the background grid generation process provides a smooth distribution of grid parameters which not only improves the grid quality but also increases the robustness and computational efficiency of the grid generator.

The unstructured flow solver algorithms are not nearly as refined as their structured counterpart due, in part, to the lack of confidence in the grid generation area. In fact, innovative improvements have been scarce since 1990. The unstructured algorithms based on an approach developed by Barth [12] [13] represent the vast majority of higher order procedures on unstructured meshes in two and three dimensions. The approaches of [20] and [51] are also popular.

1.3 Motivation

There is a need to extend the accuracy and dependability of structured flow solver algorithms to an arbitrary grid setting. It is the generalization of these concepts into a single technology and methodology that directly extends itself to domains of arbitrary origin, including unstructured environments, that is the subject of this dissertation.

In describing this work, it is necessary to highlight and focus on the basic ideas common to a large class of finite volume methods. Within this formulation, there exists a common principle of balancing the time rate of change of conserved quantities with a flux integration through the surface bounding the cell. However, there is a division of approach for modeling the inviscid and viscous fluxes. The numerical modeling of the inviscid fluxes proceeds by evaluating the fluid dynamic states on both sides of an edge and computing the interface flux with an approximate Riemann solver. For structured grid environments, the interface quantities are commonly obtained with the traditional $\phi - \kappa$ MUSCL extrapolation formula [27] [28] . The underlying principle which derives this expression and is often unrealized, is a technique known as reconstruction. In general, the reconstruction step generates a multidimensional pointwise distribution of a function from the known average values of the function with the condition that the integration of the reconstruction function recovers the cell average exactly. A more complete and formal definition by Barth will be recalled at a later time. For the $\phi - \kappa$ expression, the reconstruction step reduces to generating a locally one-dimensional interpolating polynomial in terms of specific cell averages. However, the resulting expression which provides a pointwise value of the function (for various combinations of ϕ and κ) at the cell interface, assumes that these strategically selected cell averages are uniformly spaced, although is it straightforward to generalize this formula. Thus, this simple reconstruction upholds its accuracy only in computational space (or in the case of a uniformly spaced

grid). Nevertheless, the idea of separate reconstructions for the interface quantities as well as the cell stencil choice demonstrating upwind and upwind-biased interpolation capabilities needs to be retained.

The viscous terms are formed by considering the mathematical nature of the heat flux and shear stress terms. The most common discretization for numerically modeling these gradient terms is simply forming central differences [27] . Although extending this approach to an arbitrary grid environment is not directly applicable, we can still utilize the concepts of differentiating a function and selecting a support stencil which surrounds the face.

Current unstructured flow solver technologies utilize a few of these ideas, however, are limited in performance as compared to their structured counterparts. Several unstructured algorithms currently exist for solving the Euler and Navier Stokes equations [12-22]. The methodologies incorporate both finite volume and finite element techniques. Common to most of these algorithms is a first and higher order reconstruction for the inviscid flux calculation. Specifically, Barth [12] introduced a multidimensional linear reconstruction of cell averaged data which estimates the solution gradient within a cell from surrounding centroid data. This approximation is considered as a higher order correction to an otherwise first order Taylor series expansion. Frink [17] utilizes geometric features common to both triangles and tetrahedra and illustrates an elegant formula for calculating the interface quantities to higher order using the gradient concept. However, the averaging processes and geometrical constraints which redistribute the centroidal cell quantities to the nodes are questionable, that is to say, the accuracy issue has been left unanswered.

High order accurate finite volume schemes for the Euler equations on unstructured meshes were first introduced by Barth and Frederickson [13] . They derived and applied $k-exact$ reconstruction as a means of reconstructing cell average data to an arbitrary

accuracy. In summarizing their efforts, Barth's reconstruction criterion is designed to use polynomial functions that integrate the correct cell average and that for polynomials of degree k or less, the reconstruction is exact ($k-exact$), i.e. this procedure is exact for polynomials of degree k or less. Smooth numerical examples for the Euler equations were provided to validate the methodology including reconstructing a high order polynomial and solutions to Ringleb's flow.

Godfrey, Mitchell, and Walters [9] , extended the method to compute flows with discontinuities. In addition, they required that the reconstruction polynomial integrates the cell average for *all* cells used in the reconstruction step. In contrast, Barth's implementation compromises this feature. Second, the number of support cells within the stencil corresponds to the number of degrees of freedom prescribed by the polynomial. Barth's criterion insists on a locality constraint whereby the stencil is confined to a relatively small neighborhood. This results in an over-constrained linear system to yield the polynomial coefficients which can be solved only approximately, thus the conservation of the mean system of equations does not yield the correct cell averages. The idea of a predetermined stencil size eliminates this disturbing issue of selecting the appropriate "neighborhood" of cells. The highlights, however, included assessing the spatial accuracy of $k-exact$ reconstruction and combining this procedure with a moving stencil algorithm, similar to ENO techniques, for computing solutions for flows with discontinuities.

Common to both implementations of $k-exact$ reconstruction is the idea of performing one reconstruction per cell for each dependent variable in the domain. As a result, the polynomials are discontinuous along the cell interface where two distinct values can be obtained. These values composed the two states in resolving the Riemann problem. If we contrast the reconstruction principles developed for unstructured meshes compared to structured grids, we can expose one major difference. Structured algorithms require two separate reconstructions for "each" interface to calculate the inviscid flux contribution.

The disadvantage of evaluating all of the interface properties surrounding a cell with a single reconstruction lies in the inability to select one stencil which adequately reproduces the spatial variation of the flow variables. In addition, the ability and flexibility of biasing the stencil in a preferred direction e.g. upwind, upwind-biased, etc., relative to a cell face is compromised. It is this principle that is exploited in developing a multidimensional solution algorithm for the Navier Stokes equations.

1.4 Scope of the Investigation

The purpose of this research effort is to investigate high resolution solution algorithms for the Navier Stokes equations for arbitrary topologies including unstructured grids. Emphasis is focused on the fundamental reconstruction techniques that apply to an arbitrary grid setting. A finite volume formulation is derived using the integral expressions for the governing equations of gas dynamics. Issues including numerical flux functions, flux integration requirements, and time integration schemes are briefly summarized.

Next, multidimensional linear reconstruction algorithms which determine the pointwise variation of the state variables are described including two approaches. The first technique is a higher order correction to an otherwise first order method that is formulated from a Taylor series expansion. The additional expression requires the evaluation of the solution gradient for each cell in the domain. The gradient calculation is an approximation to an exact expression and involves the numerical solution of a boundary integral. Several options for the path of integration are given. The second algorithm is a procedure for reconstructing the spatial variation of the flow properties using polynomial functions. This method, known as $k-exact$ reconstruction, specifies a design criteria for generating k^{th} degree polynomials. Although this method is capable of high accuracy, the previous implementation is believed to possess inherent weaknesses. As a result, a

new and unique approach to this method is introduced.

The improved algorithm retains the important concepts of reconstruction that are well known in one dimension but are rarely discussed and often violated in multiple dimensions. The numerical error resulting from the reconstruction step is quantitatively evaluated on triangular meshes using test functions. By performing separate reconstructions for the right and left interface states using stencil selection algorithms that offer directional biasing, smooth as well as discontinuous flows can be computed. Before this research effort, only smooth flow patterns were computed using $k-exact$ reconstruction. Perhaps the most credible result of this unique approach lies in the ability to compute viscous flows on highly distorted meshes. Several researchers have claimed that viscous solutions on unstructured triangular grids have geometrical limitations [24] which limits the maximum interior angle to approximately 105°; however, by using precise reconstruction techniques and proper selection of the stencil, the viscous problem can be solved.

The flexibility of the method is demonstrated by considering two and three dimensional inviscid and viscous flows on quadrilateral, triangular, and tetrahedral meshes. The inviscid cases include Ringleb's flow, which has an analytical solution available to determine solution accuracy, and a simple shock reflection problem. The viscous test cases include incompressible flow over a flat plate and a high speed shock wave boundary layer interaction that exhibits laminar separation. Results for $k = 0, 1, 2$ degree polynomials are provided for these cases, and they are compared with experimental data. The three dimensional inviscid test case is transonic flow over the ONERA M6 wing. Comparisons for the pressure coefficient at several spanwise stations demonstrate the accurate flow field predictions.

The unstructured grids are generated using the advancing front technique in two and three dimensions. Modifications to the three dimensional algorithm were required in

order to generate the surface grids. A summary of these efforts is highlighted including surface discretization procedures and grid refinement algorithms. Examples include a Glasair fuselage, High Speed Civil Transport configuration, and the ONERA M6 wing.

The final research topic involves the role of reconstruction for adaptive remeshing algorithms. Specifically, a first order error estimator based on reconstruction principles is introduced and validated for discontinuous and shock free flows. Qualitative and quantitative analyses verifying the similar behavior between the exact error and approximate error is shown.

Chapter 2

The Finite Volume Formulation

2.1 Governing Integral Equations

The integral form of the three dimensional Navier Stokes equations can be written as:

$$\frac{\partial}{\partial t}\iiint_{\Omega} Q \, d\Omega + \iint_{S} \vec{F} \cdot \hat{n} \, dS - \iint_{S} \vec{G} \cdot \hat{n} \, dS = 0. \tag{2.1}$$

where

$$Q \equiv \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e_0 \end{Bmatrix}$$

$$\vec{F} \equiv f\hat{i} + g\hat{j} + h\hat{k}$$

$$\vec{G} \equiv \frac{1}{Re}(f_v\hat{i} + g_v\hat{j} + h_v\hat{k})$$

and

$$f \equiv \begin{Bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ \rho uh_0 \end{Bmatrix}, g \equiv \begin{Bmatrix} \rho v \\ \rho vu \\ \rho v^2 + p \\ \rho vw \\ \rho vh_0 \end{Bmatrix}, h \equiv \begin{Bmatrix} \rho w \\ \rho wu \\ \rho wv \\ \rho w^2 + p \\ \rho wh_0 \end{Bmatrix}$$

and

$$f_v \equiv \begin{Bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ \bar{\tau_x} - q_x \end{Bmatrix}, g_v \equiv \begin{Bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ \tau_{yz} \\ \bar{\tau_y} - q_y \end{Bmatrix}, h_v \equiv \begin{Bmatrix} 0 \\ \tau_{xz} \\ \tau_{yz} \\ \tau_{zz} \\ \bar{\tau_z} - q_z \end{Bmatrix}$$

$$\bar{\tau_x} = u\tau_{xx} + v\tau_{xy} + w\tau_{xz}$$

$$\bar{\tau_y} = u\tau_{xy} + v\tau_{yy} + w\tau_{yz}$$

$$\bar{\tau_z} = u\tau_{xz} + v\tau_{yz} + w\tau_{zz}$$

$$\tau_{xx} = \mu(2u_x - \frac{2}{3}(u_x + v_y + w_z))$$

$$\tau_{yy} = \mu(2v_y - \frac{2}{3}(u_x + v_y + w_z))$$

$$\tau_{zz} = \mu(2w_z - \frac{2}{3}(u_x + v_y + w_z))$$

$$\tau_{xy} = \mu(u_y + v_x)$$

$$\tau_{xz} = \mu(u_z + w_x)$$

$$\tau_{yz} = \mu(v_z + w_y)$$

and

$$q_x = -\frac{\mu}{Pr(\gamma - 1)}\frac{\partial a^2}{\partial x}$$

$$q_y = -\frac{\mu}{Pr(\gamma - 1)}\frac{\partial a^2}{\partial y}$$

$$q_z = -\frac{\mu}{Pr(\gamma - 1)}\frac{\partial a^2}{\partial z}$$

where $\hat{n}$ is the outward facing unit normal and Ω is the volume of the domain bounded by the surface S. The perfect-gas equations are non-dimensionalized with respect to density ρ_∞, velocity u_∞, reference length L and molecular viscosity μ_∞. In the equations above $\rho, u, v, w, e_0, h_0, p, a, \gamma$ are the density, (x, y, z) components of velocity, total energy per unit mass, stagnation enthalpy per unit mass, pressure, speed of sound, and ratio of specific heats, respectively. The molecular viscosity, μ, is calculated using Sutherland's law and the bulk viscosity, $\lambda = \frac{-2\mu}{3}$, is based on Stoke's hypothesis. Re and Pr are the Reynolds number and Prandtl number, respectively and are defined as

$$Re = \frac{\rho_{ref} u_{ref} x_{ref}}{\mu_{ref}}$$

$$Pr = \frac{\mu_{ref} C_p}{k}$$

For a perfect gas the equations are closed by the perfect gas law

$$p = (\gamma - 1)\rho(e_0 - \frac{1}{2}(u^2 + v^2 + w^2))$$

The first surface integral in Equation 2.1 represents the projection of the Euler flux onto a unit vector defined to be normal to the surfaces surrounding each cell. High order

accurate spatial schemes for generalized discretizations are developed by extending Godunov's approach [25] to include arbitrary control volume shapes. Numerical evaluation of this integral expression begins by examining the functional form of $\vec{F}$. Since $F = F(Q)$ the functional form of Q will at some point need to be addressed. At this stage, however, a Godunov extension proceeds by assuming a functional form of Q that describes the spatial variation of the flow properties within each cell of the domain. By establishing a piecewise representation for the dependent variables which appear in the governing equations, the interface properties along a cell boundary become discontinuous. In fact, two distinct values are possible corresponding to the functional descriptions for the two cells that share the common boundary. As a result of the non-uniqueness, the Euler flux function is approximated with a numerical flux function according to

$$\iint_{S_m} \vec{F} \cdot \hat{n}\, dS \approx \iint_{S_m} F(Q_L, Q_R, \hat{n})\, dS$$

where Q_L and Q_R describe the fluid dynamic states on both sides of the path of integration. Developing a discretization procedure that numerically approximates the surface integrals reveals two separate issues which have a combined effect on the accuracy. The first concentrates on the numerical estimate for the inviscid and viscous fluxes. The second focuses on the accuracy of the flux integration rule.

A finite volume discretization algorithm for the integral form of the Navier Stokes equations provided in Equation 2.1 proceeds by introducing cell averages of the conserved variables of mass momentum and energy valid within each control volume. Therefore, for each cell m bounded by domain Ω_m, the volume integral can be directly represented in terms of it's cell average according to the following definition:

$$\bar{Q}_m \equiv \frac{1}{\Omega} \iiint_m Q\, d\Omega_m. \qquad 2.2$$

This equation forms the basis of reconstruction methods. That is to say, given the cell

averages via a time evolution algorithm, it will be necessary to reconstruct the pointwise distribution to a given level of accuracy on an arbitrary grid.

2.2 Flux Difference Splitting

Consider Roe's flux difference splitting technique [1] [4] as the approximation for the inviscid flux function. Within this method, the surface integral has been approximated as:

$$\iint_{S_m} \vec{F} \cdot \hat{n}\, dS \approx \iint_{S_m} F(Q_L, Q_R, \hat{n})\, dS$$

and

$$\iint_{S_m} F(Q_L, Q_R, \hat{n})\, dS = \iint_{S_m} \frac{1}{2}\Big[F(Q_L) + F(Q_R) - \mid \tilde{A} \mid (Q_R - Q_L)\Big]\, dS. \qquad 2.3$$

The surface path of integration bounding the control volumes for finite volume schemes is simply a set of edges which encompass the cell. Therefore, the integral above can be rewritten to represent a summation process over the number of edges surrounding each cell as

$$\iint_{S_m} F(Q_L, Q_R, \hat{n})\, dS = \sum_{k(m)} \Bigg[\int_{k(m_i)} \frac{1}{2}\Big[F(Q_L) + F(Q_R) - \mid \tilde{A} \mid (Q_R - Q_L)\Big]\, dS \Bigg].$$

Roe's implementation of his flux function is commonly referred to as a flux-difference-splitting technique which approximates the solution to the exact one-dimensional Riemann problem. The Riemann problem is an initial value problem with piecewise constant initial data. The time dependent details of the interaction from the two states, designated as the right and left states respectively, approximates the existence of three characteristic waves. These three waves are identified as either a shock wave, rarefaction, or contact discontinuity. The eigenvalues of the Jacobian matrix for the one dimensional Euler equations reveals the existence of these three transitional states. While the exact solution to the Riemann problem requires a computationally expensive iterative technique [26] , Roe's derivation models the propagation of the wave information in terms of the interface fluxes. Specifically, the flux normal to a cell face is a function of the two different fluid dynamic states, Q_L and Q_R, and represents the initial data for the Riemann states.

In Equation 2.3 , the matrix $\tilde{A}$ is derived by linearizing the flux F with respect to Q as

$$A = \frac{\partial F}{\partial Q}.$$

The matrix is evaluated at a derived average state that satisfies the Rankine-Hugoniot jump conditions. Roe determined these average conditions to be:

$$\tilde{\rho} = (\rho_R \rho_L)^{1/2}$$

$$\tilde{u} = \frac{\rho_L^{1/2} u_L + \rho_R^{1/2} u_R}{\rho_R^{1/2} + \rho_L^{1/2}} \tilde{v} = \frac{\rho_L^{1/2} v_L + \rho_R^{1/2} v_R}{\rho_L^{1/2} + \rho_R^{1/2}} \qquad \tilde{w} = \frac{\rho_L^{1/2} w_L + \rho_R^{1/2} w_R}{\rho_L^{1/2} + \rho_R^{1/2}}$$

$$\tilde{h}_0 = \frac{\rho_L^{1/2} h_{0L} + \rho_R^{1/2} h_{0R}}{\rho_L^{1/2} + \rho_R^{1/2}}$$

$$\tilde{a}^2 = (\gamma - 1)(\tilde{h}_0 - \frac{(\tilde{u}^2 + \tilde{v}^2 + \tilde{w}^2)}{2})$$

as a result

$$F(Q_R) - F(Q_L) = \tilde{A}(Q_L, Q_R)(Q_R - Q_L).$$

If the diagonalizing matrices $\tilde{S}$ and $\tilde{S}^{-1}$ represent the left and right eigenvectors of $\tilde{A}$ and $\tilde{\Lambda}$ represents the eigenvalues of $\tilde{A}$ then

$$\tilde{A}(Q_L, Q_R) = \tilde{S}\tilde{\Lambda}\tilde{S}^{-1}$$

and

$$| \tilde{A} | (Q_R - Q_L) = \tilde{S} | \tilde{\Lambda} | \tilde{S}^{-1} \Delta Q.$$

This expression can be separated into three ΔF flux components resulting from the three distinct eigenvalues.

$$\tilde{S} | \tilde{\Lambda} | \tilde{S}^{-1} \Delta Q = |\Delta \tilde{F}_1| + |\Delta \tilde{F}_2| + |\Delta \tilde{F}_3|$$

with

$$|\Delta \tilde{F}_1| = |\tilde{U}| \left(\Delta \rho - \frac{\Delta p}{\tilde{a}^2} \right) \begin{bmatrix} 1 \\ \tilde{u} \\ \tilde{v} \\ \tilde{w} \\ (\tilde{u}^2 + \tilde{v}^2 + \tilde{w}^2)/2 \end{bmatrix} + |\tilde{U}| \tilde{\rho} \begin{bmatrix} 0 \\ \Delta u - n_x \Delta \bar{U} \\ \Delta v - n_y \Delta \bar{U} \\ \Delta w - n_z \Delta \bar{U} \\ \tilde{u}\Delta u + \tilde{v}\Delta v + \tilde{w}\Delta w - \tilde{U}\Delta \bar{U} \end{bmatrix}$$

$$|\Delta \tilde{F}_{2,3}| = \left|\tilde{U} \pm \tilde{a}\right| \left[\frac{\Delta p}{2\tilde{a}^2} \pm \frac{\tilde{\rho}\Delta \bar{U}}{2\tilde{a}}\right] \begin{bmatrix} 1 \\ \tilde{u} \pm n_x \tilde{a} \\ \tilde{v} \pm n_y \tilde{a} \\ \tilde{w} \pm n_z \tilde{a} \\ \tilde{h}_0 \pm \bar{U}\tilde{a} \end{bmatrix}$$

where $\Delta(\cdot) = (\cdot)_R - (\cdot)_L$ and $\Delta \bar{U} = n_x \Delta u + n_y \Delta v + n_z \Delta w$.

The functional form of Q_L and Q_R is generally represented by polynomials of varying degree k. Two dimensional inviscid numerical experiments of Barth [13] suggest that N point Gauss quadrature, where $N \geq \frac{k+1}{2}$, is adequate for evaluating the inviscid flux integral. Using k degree multidimensional polynomials as a representation of the spatial variation of Q_L and Q_R, Equation 2.3 is evaluated according to:

$$\iint_{S_m} \frac{1}{2}\left[F(Q_L^{k_i}) + F(Q_R^{k_i}) - \mid \tilde{A} \mid (Q_R^{k_i} - Q_L^{k_i})\right] dS \approx \sum_{k(m)} \left[\sum_{f=1}^{N_f} \omega_f \left[\mathbf{F}(Q_L^{k_i}, Q_R^{k_i}, \hat{n}, \xi_f, \eta_f)\right] \right] \tag{2.4}$$

where

$$Q_{L,R}^{k_i}(x,y,z) = \sum_{i=0}^{k} \sum_{j=0}^{k-i} \sum_{l=0}^{k-(i+j)} C_{i,j,l}\, x^i y^j z^l. \tag{2.5}$$

where $C_{i,j,l}$ is the polynomial coefficients and $k(m)$ is a list of faces bounding surface S and k_i designates the degree of the polynomial used in the inviscid calculation. N_f is the number of quadrature points used in the flux integration for each face in $k(m)$, ω_f and ξ_f, η_f are the weights and coordinates, respectively.

The viscous fluxes are approximated by representing k degree polynomials and their polynomial derivatives as the viscous flux function. This can be expressed as:

$$\iint_{S_m} \vec{G} \cdot \hat{n}\, dS \approx \iint_{S_m} G(Q^{k_v}, \frac{\partial Q^{k_v}}{\partial x}, \frac{\partial Q^{k_v}}{\partial y}, \frac{\partial Q^{k_v}}{\partial z}, \hat{n}) dS$$

where k_v represents the degree of the polynomial used to reconstruct the properties needed to calculate the viscous fluxes. Proceeding as before, the flux can be evaluated as:

$$\iint_{S_m} G(Q^{k_v}, \frac{\partial Q^{k_v}}{\partial x}, \frac{\partial Q^{k_v}}{\partial y}, \frac{\partial Q^{k_v}}{\partial z}, \hat{n})\, dS \approx \sum_{k(m)} \left[\sum_{f=1}^{N_f} \omega_f \left[\mathbf{G}(Q^{k_v}, \frac{\partial Q^{k_v}}{\partial x}, \frac{\partial Q^{k_v}}{\partial y}, \frac{\partial Q^{k_v}}{\partial z}, \hat{n}, \xi_f, \eta_f)\right] \right]. \tag{2.6}$$

Combining these results yields a semi-discrete formulation for Equation 2.1 as:

$$\Omega_m \frac{\partial \bar{Q}}{\partial t} + \sum_{k(m)} \Bigg[\sum_{f=1}^{N} \omega_f \Big[\mathbf{F}(Q_L^{k_i}, Q_R^{k_i}, \hat{n}, \xi_f, \eta_f) - \mathbf{G}(Q^{k_v}, \frac{\partial Q^{k_v}}{\partial x}, \frac{\partial Q^{k_v}}{\partial y}, \frac{\partial Q^{k_v}}{\partial z}, \hat{n}, \xi_f, \eta_f) \Big] \Bigg] = 0. \tag{2.7}$$

The solution procedure for the above equation is divided into three common steps. These processes are standard extensions of Godunov's scheme [12] [13] [27] [28] .

1.0 The Reconstruction Problem: Given cell average values of the dependent variables reconstruct pointwise polynomial expressions that describe their spatial behavior. This process is described in detail in Chapter 3.

2.0 Flux Integration: Perform the surface integration for the flux functions with sufficient numerical procedures. This issue is discussed in Chapter 4.

3.0 Time Evolution: Update the solution using a time integration scheme.

2.4 Time Integration

Implicit and explicit time integration schemes have been implemented and are of the form:

$$\Omega_m \frac{\partial \bar{Q}}{\partial t} = R(\bar{Q}) \tag{2.8}$$

where the residual $R(\bar{Q})$, defined by

$$R(\bar{Q}) = - \sum_{k(m)} \Bigg[\sum_{f=1}^{N} \omega_f \Big[\mathbf{F}(Q_L^{k_i}, Q_R^{k_i}, \hat{n}, \xi_f, \eta_f) - \mathbf{G}(Q^{k_v}, \frac{\partial Q^{k_v}}{\partial x}, \frac{\partial Q^{k_v}}{\partial y}, \frac{\partial Q^{k_v}}{\partial z}, \hat{n}, \xi_f, \eta_f) \Big] \Bigg]$$

discretely represents the steady state terms. The notation $R(\bar{Q})$ implies that the pointwise values are obtained as a function of the cell averages, i.e. $Q = Q(\bar{Q}, x, y, z)$, which is the reconstruction problem. The distinction between explicit and implicit time integration lies in the time level evaluation of $R(\bar{Q})$. Explicit schemes solve Equation 2.8 in delta form as

$$\Omega_m \frac{\Delta \bar{Q}}{\Delta t} = R(\bar{Q})^n$$

where n represents the current time level and

$$\Delta\bar{Q} = \bar{Q}^{n+1} - \bar{Q}^{n}.$$

An $m - stage$ Runge-Kutta time stepping method developed by Jameson [29] has been applied for this class of time integration algorithms. An implicit formulation begins with evaluating Equation 2.8 at time level $n+1$ as

$$\Omega_m \frac{\Delta\bar{Q}}{\Delta t} = R(\bar{Q})^{n+1}$$

Expanding this equation in a Taylor series yields the following system of linear equations

$$[\frac{\Omega_m}{\Delta t} I - \frac{\partial R(\bar{Q})^n}{\partial Q}]\Delta\bar{Q} = R(\bar{Q})^n. \qquad 2.9$$

Within the approach lies the need to calculate the linearization of R with respect to conservative variables. However, by selecting primitive variables, the computational effort is reduced and can be accomplished via the following transformation:

$$\frac{\partial\bar{Q}}{\partial t} = \frac{\partial\bar{Q}}{\partial\bar{q}}\frac{\partial\bar{q}}{\partial t} = \bar{M}\frac{\partial\bar{q}}{\partial t}$$

where

$$\bar{q} = (\rho, u, v, w, p)^T.$$

Using this result Equation 2.9 is rewritten as

$$[\bar{M}\frac{\Omega_m}{\Delta t} - \frac{\partial R(\bar{q})^n}{\partial\bar{q}}]\Delta\bar{q} = R(\bar{q})^n \qquad 2.10$$

Direct solvers are available which can solve this system of equations exactly, however due to excessive computational effort and memory requirements, an iterative block Jacobi technique was implemented. Representing Equation 2.10 as

$$Ax = b$$

the matrix A can be divided into diagonal D and off diagonal L and U matrices, where $A = L + D + U$. The following iterative technique is then employed

$$x^{(n+1)} = D^{-1}(b - (L + U)x^n)$$

The number of iterations to adequately approximate the solution of the linear problem will vary, however 6 iterations were performed for the numerical results for the ONERA M6 wing. This corresponds to a one-half order reduction in a RMS norm based on the sum of Δx. All other calculations used Euler explicit time integration.

A subset of the iterative technique applied to Equation 2.10 is a point Jacobi method which is computationally efficient and effective during the initial transient modes of the calculation. In addition, the block Jacobi scheme is used to initialize the inner iteration method.

Chapter 3

Multidimensional Reconstruction Schemes

3.1 The Reconstruction Problem

In the previous section, a procedure was outlined for advancing the cell averages $\bar{Q}$ in time, contingent on evaluating the numerical fluxes. If we examine the semi-discrete form of the governing equations carefully, we reveal that the cell averages $\bar{Q}$ are updated in time using expressions that require pointwise values of Q; specifically the values at pre-determined Gauss points for use in the flux integration step. As a consequence, procedures must be developed to model the spatial variation of Q in terms of known cell averages $\bar{Q}$. This is the reconstruction problem.

In words, the reconstruction problem is the process of determining a pointwise distribution function given cell averaged data with the restriction that integrating the function recovers the cell average exactly. This restriction was imposed in the first step of formulating a finite volume procedure, namely that

$$\frac{\partial}{\partial t}\iiint_{\Omega} Q \, d\Omega = \frac{\partial}{\partial t}(\bar{Q}\Omega)$$

where for cell m and volume Ω which is independent of time

$$\bar{Q}_m \equiv \frac{1}{\Omega}\iiint_{m} Q(x,y,z) \, d\Omega_m.$$

The key to the success of the finite volume scheme is accurately producing the pointwise values for use in evaluating the flux quantities. For steady state solutions, the accuracy of the scheme is limited by the accuracy of the expression used to determine the interface values. In addition, assuming exact time integration schemes, the accuracy

of the time dependent solution is still a function of the approximation of the spatial variables.

The functional form of choice for $Q(x,y,z)$ is polynomial expressions of varying degree k written in one, two, and three dimensions as

$$Q^{k_i}(x) = \sum_{i=0}^{k} C_i\, x^i$$

$$Q^{k_i}(x,y) = \sum_{i=0}^{k} \sum_{j=0}^{k-i} C_{i,j}\, x^i y^j$$

$$Q^{k_i}(x,y,z) = \sum_{i=0}^{k} \sum_{j=0}^{k-i} \sum_{l=0}^{k-(i+j)} C_{i,j,l}\, x^i y^j z^l,$$

respectively.

There are a few reasons for choosing piecewise polynomial functions. First of all, polynomial expressions are relatively easy to integrate over the control volumes. Secondly, in order to compute "weak" solutions, the model must permit discontinuities. Piecewise constant polynomials satisfy this requirement as two independent states Q_L and Q_R are evaluated at the cell interface. The discontinuous behavior is resolved with the Riemann solver. Multidimensional higher order methods result in non-constant interface quantities, however, the one dimensional Riemann solver may be used in combination with sufficient quadrature. Multidimensional Riemann solvers would be ideal for this situation, however, the solution along the interface at the instant of initial contact can be adequately resolved by solving an acceptable number of one-dimensional Riemann problems. For smooth data, the difference between the values Q_L and Q_R decreases with increasing polynomial degree thereby decreasing the importance of sophisticated Riemann solvers [13].

One dimensional highly accurate solution algorithms that exploit the reconstruction process are abundant with the research in ENO schemes [6] [7] [8] [30] . The extension of the basic first order accurate methods to arbitrarily high orders of accuracy are described

within the framework of ENO. There are several ways in which the polynomial pointwise behavior can be derived using cell averages in one dimension. They are

1. reconstruction by deconvolution
2. reconstruction using the primitive function
3. reconstruction by matching cell averages.

Given a uniform one dimensional stencil of cell averages, all of these methods will reproduce the same unique polynomial. These approaches have been extended to multiple dimensions for structured grid domains [31] via reconstruction along local generalized coordinate lines. While the higher order methods reflect the high accuracy reconstruction technology for multidimensional structured flow solver algorithms, the extension of these procedures to arbitrary domain discretizations is, in most cases, not possible. For these reasons, they will not be discussed in this work. However, as an introduction into multidimensional methods valid for arbitrary discretizations, a simple one dimensional problem will be shown to visualize the concepts.

Consider the following test function which is shown in Figure 3.1

$$f(x) = sin(\pi x).$$

The reconstruction process for a polynomial of degree two is demonstrated by subdividing and initializing the domain by area averaging the exact solution onto the mesh. This can be accomplished by directly integrating the following equation:

$$\bar{Q}_m = \frac{1}{L_m} \int_{L_m} f(x)dx$$

the results are provided in part *a* of Figure 3.2.

Next, generate an equation for each cell that satisfies conservation of cell averaged quantities. This can be achieved by substituting a second degree polynomial written as

$$Q_m(x) = P^2(x) = C_{0_m} + C_{1_m} x + C_{2_m} x^2 \qquad 3.1$$

into the previous equation yielding

$$\bar{Q}_m = \frac{1}{L_m} \int_{L_m} (C_{0_m} + C_{1_m} x + C_{2_m} x^2) dx.$$

Three equations are needed to solve for the coefficients C_0, C_1, and C_2 for each cell m. For this example, we select the 2 nearest neighbors and generate the following equations:

$$\begin{aligned}
\bar{Q}_{m-1} &= \frac{1}{L_{m-1}} \int_{L_{m-1}} (C_{0_m} + C_{1_m} x + C_{2_m} x^2) dx \\
\bar{Q}_m &= \frac{1}{L_m} \int_{L_m} (C_{0_m} + C_{1_m} x + C_{2_m} x^2) dx \\
\bar{Q}_{m+1} &= \frac{1}{L_{m+1}} \int_{L_{m+1}} (C_{0_m} + C_{1_m} x + C_{2_m} x^2) dx
\end{aligned}$$

which can be rewritten as

$$\begin{bmatrix} \bar{Q}_{m-1} \\ \bar{Q}_m \\ \bar{Q}_{m+1} \end{bmatrix} = \begin{bmatrix} \frac{1}{L_{m-1}} \int_{L_{m-1}} dx & \frac{1}{L_{m-1}} \int_{L_{m-1}} x dx & \frac{1}{L_{m-1}} \int_{L_{m-1}} x^2 dx \\ \frac{1}{L_m} \int_{L_m} dx & \frac{1}{L_m} \int_{L_m} x dx & \frac{1}{L_m} \int_{L_m} x^2 dx \\ \frac{1}{L_{m+1}} \int_{L_{m+1}} dx & \frac{1}{L_{m+1}} \int_{L_{m+1}} x dx & \frac{1}{L_{m+1}} \int_{L_{m+1}} x^2 dx \end{bmatrix} \begin{bmatrix} C_{0_m} \\ C_{1_m} \\ C_{2_m} \end{bmatrix}$$

We now have our equation set which can be solved using standard LU decomposition. At this point, we have determined the coefficients $C_{0_m}, C_{1_m}, C_{2_m}$ for use in Equation 3.1 to yield a pointwise distribution of Q for the cell m. Figure 3.2*b* shows the results of the reconstruction applied at cell 3.

3.2 Gradient Based Linear Reconstruction

In this section, the development of a general approach for obtaining higher order accuracy based on piecewise linear reconstruction of cell averaged data is presented. The approach applies to arbitrary polygonal control volumes and follows the work by Barth presented in Ref. [12] .

3.2.1 Two Dimensional Algorithms

Consider the two dimensional mesh discretizations shown in Figure 3.3. Developing a linear reconstruction method for modeling the spatial dependence of the state variables within the domain m proceeds by expressing the pointwise variation in terms of a Taylor

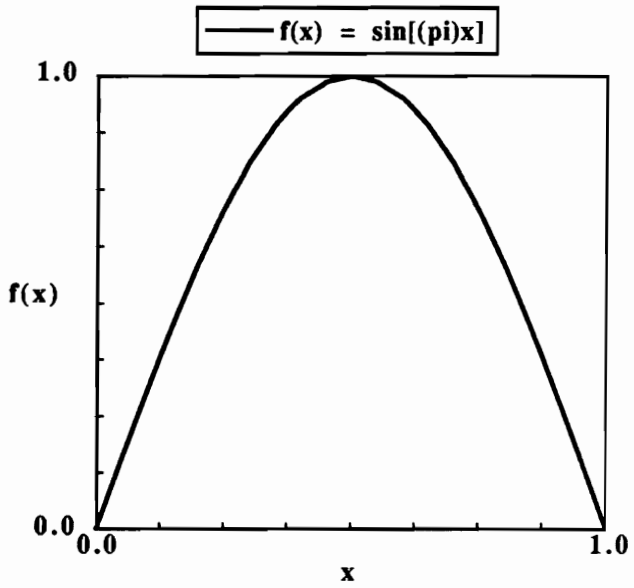
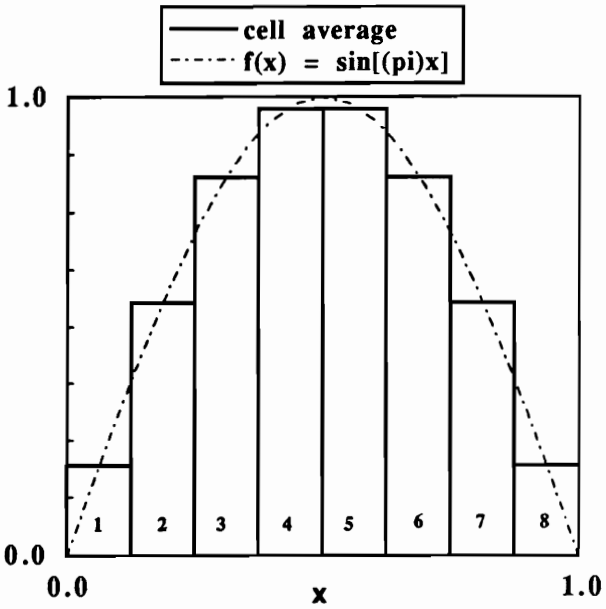
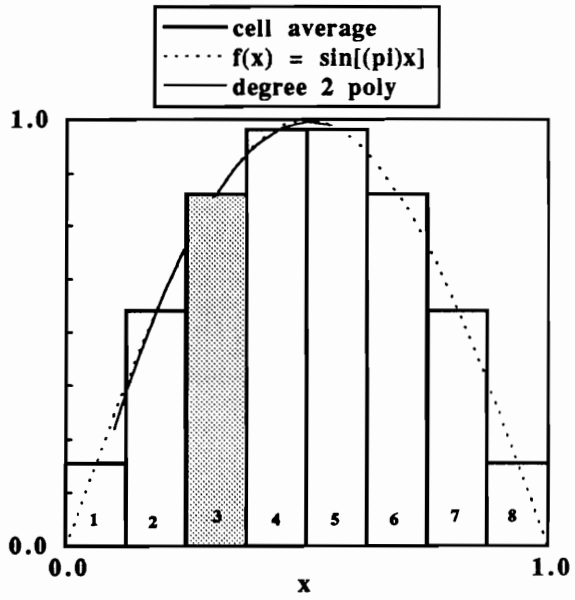


Fig. 3.1 – Test function for one dimensional reconstruction



a – Test function cell averaged



b – $K = 2$ Reconstruction of test function

Fig. 3.2 – One dimensional reconstruction using $k = 2$

series. Specifically, a linear distribution of Q expanded about the centroid (x_0, y_0) can be written as

$$\begin{aligned} Q(x,y) &= Q(x_0,y_0) + \frac{\partial Q}{\partial x}(x-x_0) + \frac{\partial Q}{\partial y}(y-y_0) \\ &= Q(x_0,y_0) + \nabla Q \cdot \triangle r \end{aligned} \tag{3.2}$$

Recall that we are restricted to using cell averaged data to form the terms in Equation 3.2. However, by applying the integral definition of the cell average written as

$$\bar{Q}_m \equiv \frac{1}{\Omega}\iint_m Q(x,y)\, dxdy$$

and substituting the linear function for $Q(x,y)$ from Equation 3.2 yields

$$\bar{Q}_m \equiv \frac{1}{\Omega}\iiint_m \left[Q(x_0,y_0) + \frac{\partial Q}{\partial x}(x-x_0) + \frac{\partial Q}{\partial y}(y-y_0)\right] d\Omega_m$$

where for a constant gradient vector, i.e. linear distribution function, the integral can be shown to be

$$\bar{Q}_m = Q(x_0,y_0)$$

where x_0, y_0 is the geometric centroid of the control volume.

The vector ∇Q represents the solution gradient for the cell and can be computed using the following exact relation:

$$\iiint_{\Omega_m} \nabla Q\, d\Omega_m = \iint_{S_m} Q\hat{n}\, dS.$$

For a linear reconstruction of Q, the expression becomes

$$\nabla Q = \frac{1}{\Omega_m}\iint_{S_m} Q\hat{n}\, dS. \tag{3.3}$$

The gradient calculation is calculated by computing the surface integral defined by a closed path centered around cell m. If the function Q varies linearly, the gradient rule is exact and will yield an exact calculation of ∇Q as long as the integral expression is evaluated without error.

Several strategies for selecting the boundary paths of integration have been investigated by previous researchers [14] [12][17] . The simplest surface integration path referred to as type 1 is shown in Figure 3.3 and corresponds to a path defined by the actual cell control volume. Using this boundary path, two viable alternatives exist for solving Equation 3.3 and will be identified as Type $1-A$ and Type $1-B$.

A. Estimate the mid-point pointwise value of Q by forming an arithmetic average of the two cells that share a common face, i.e. the cell pairs connected by a dashed line, (see Figure 3.3*a*). Using the mid-point trapezoidal rule, calculate the gradient.

B. Estimate the nodal values of Q by averaging the surrounding cell data to the vertex points. The corresponding groups of cells that influence the nodal approximation are grouped by dashed lines in Figure 3.3*b*. The gradient calculation can be computed via the trapezoidal rule. Frink [17] chooses to weight the influence of the surrounding cells relative to the inverse distance from the cell to the node as

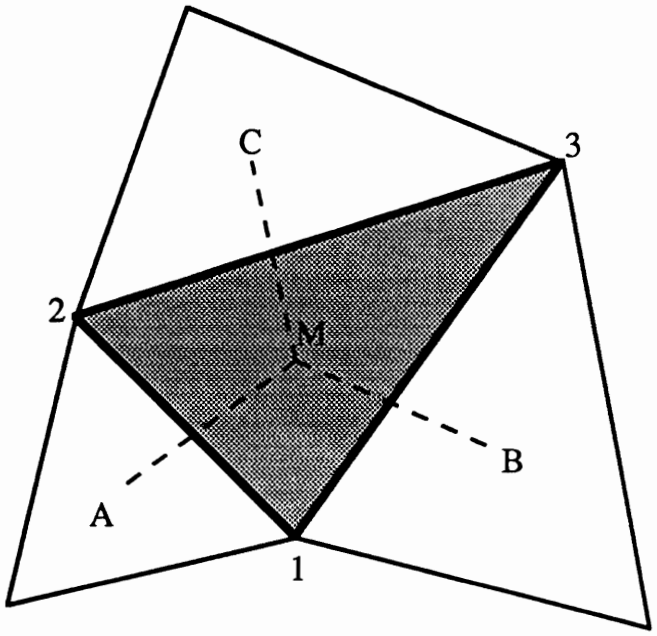
$$Q_n = \frac{\sum_{i=1}^{N} \frac{Q_i}{r_i}}{\sum_{i=1}^{N} \frac{1}{r_i}}$$

where

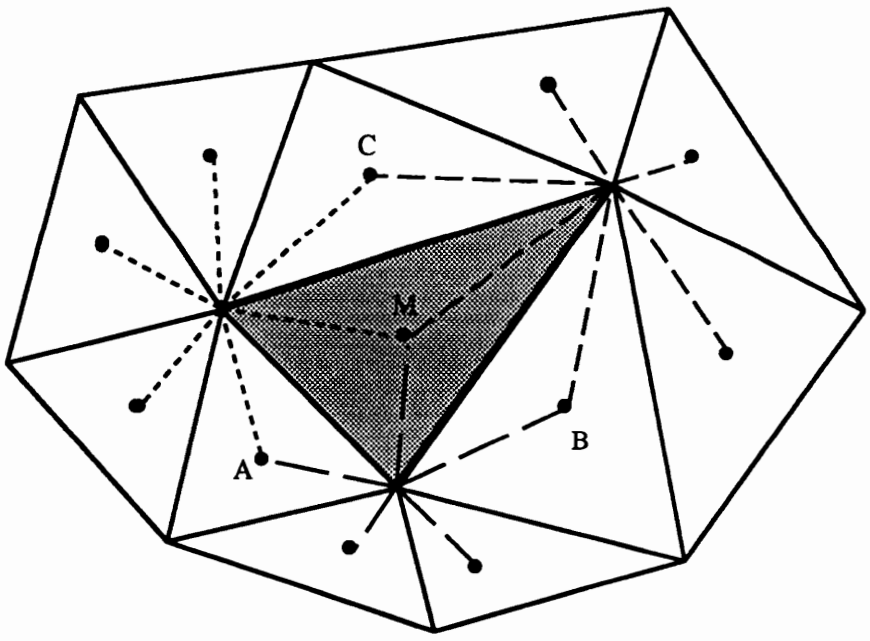
$$r_i = [(x_n - x_i)^2 + (y_n - y_i)^2)]^{\frac{1}{2}}.$$

and N is the number of cells surrounding node n.

These techniques are very attractive due to their ease of implementation. The information describing the integration paths are available, and the issue of a valid integration path is never in question. However, both of these techniques have an inherent weakness. As a result of the averaging process, an exact calculation of ∇Q when Q varies linearly is most often not possible. Specifically, the Taylor series expansion and the trapezoidal rule are formally second order accurate, however the accuracy of the averaging process is second order accurate in special cases. For example, the line connecting the cell pairs must bisect the edge at the midpoint for approach A to preserve the correct gradient.



a – Approach A



b – Approach B

Fig. 3.3 – Typical triangle for Type 1 path integration

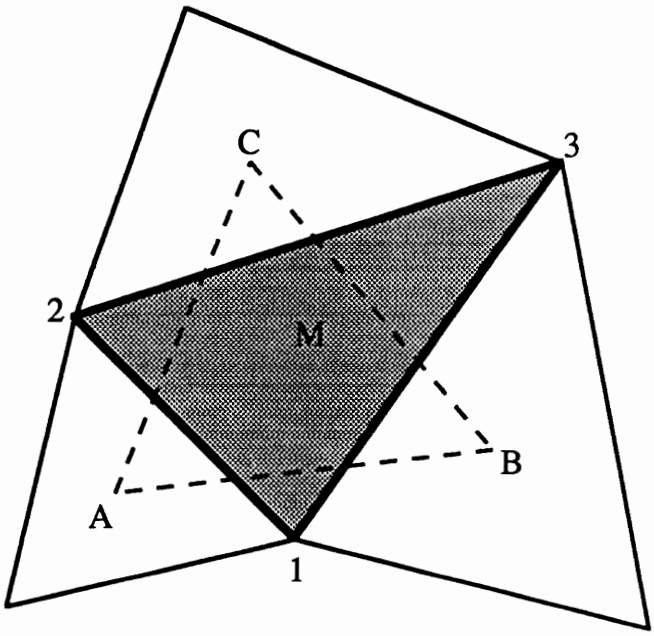
Similarly for approach B, the interpolated nodal value of Q must be computed based on linear variations from the surrounding cell data. While the accuracy of the averaging process on the global level is in question, the two algorithms have been shown to produce adequate results for both cell centered and cell vertex schemes for the Euler equations on smooth grids [17] [32] . Their effectiveness on grids which are highly distorted and skewed has not been demonstrated.

A second boundary path that upholds the criteria of calculating the gradient exactly for linear variations in Q is a method used by Slack et al. [14] . Figure 3.4a depicts a typical type 2 path of integration that includes the first cell neighbors of M. The path formed by connecting the cell centroids for cells A, B, and C constitutes the boundary for calculating Equation 3.3 using the trapezoidal rule. Unlike the previous method, this approach requires additional computational and storage requirements to include the boundary path characteristics. Due to geometrical considerations, this approach has a weakness. Consider part b of this Figure. The results from this boundary integral is a poor representation for the solution gradient to be used for cell M. In fact, the cell centroid is not bounded within the path of integration.

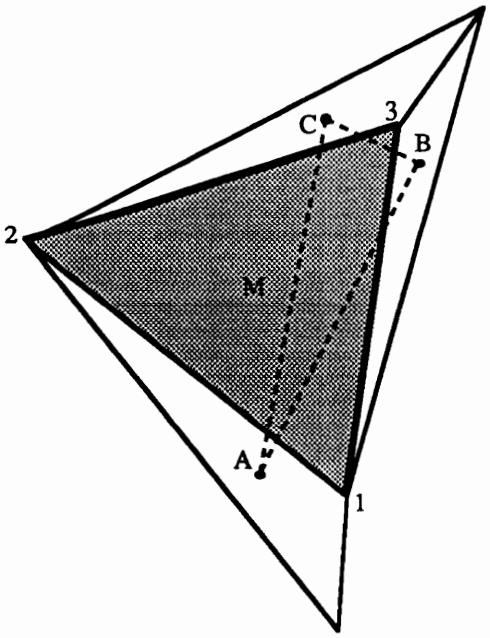
A method proposed by Barth in [12] overcomes both of the difficulties described above. His approach, shown in Figure 3.5, is based on a centroid-centroid path which traces the convex hull of a neighboring set of cells that share a common vertex with cell M. The gradient is calculated using the trapezoidal rule and does not exhibit geometrical restrictions. As in the previous method by Slack, additional work is required to evaluate Equation 3.3 due to the complexity of the boundary surfaces.

3.2.2 Three Dimensional Algorithms

The three dimensional extension of linear reconstruction algorithms based on a higher order Taylor series correction is straightforward. Consider the following Taylor series



a – Typical triangle



b – Distorted triangle

Fig. 3.4 – Boundary description for Type 2 path integration

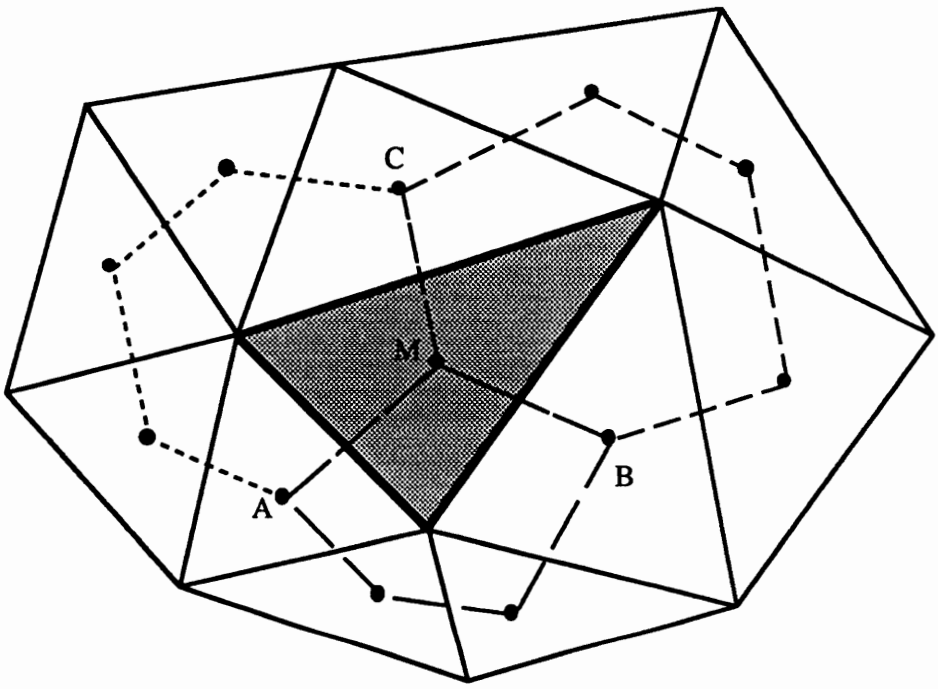


Fig. 3.5 – Typical triangle for Type 3 path integration

expanded about point (x_0, y_0)

$$\begin{aligned} Q(x,y,z) &= Q(x_0,y_0,z_0) + \frac{\partial Q}{\partial x}(x-x_0) + \frac{\partial Q}{\partial y}(y-y_0) + \frac{\partial Q}{\partial z}(z-z_0) \\ &= Q(x_0,y_0,z_0) + \nabla Q \cdot \triangle r \end{aligned} \tag{3.4}$$

The solution gradient is calculated via Equation 3.3 using a three dimensional boundary surface as the integration path. In two dimensions, several path options were provided, however, the three dimensional extension of a few of these approaches is difficult and not investigated in practice. The method of choice is the cell control volume, i.e. type 1, where the path of integration is easily identified. In the two dimensional example, the control volume was a triangle. The most common and simplest three dimensional element is the tetrahedral shown in Figure 3.7 where the 4 triangular sides constitute the closed path of integration. High order reconstruction algorithms are derived in a fashion analogous to their two dimensional counterpart.

3.2.3 Simple Linear Reconstruction Formulas

Although the numerical effort for evaluating the gradient expressions for use in Equation 3.2 is not computationally intensive, simplifications can be made to a few of the linear reconstruction processes that reduces the memory and numerical requirements. The semi-discrete form derived for the finite volume algorithm outlined in Chapter 2 reveals that only the interface values of Q used in the flux integration step are required. Consequently, the Type 1 path integration formulas can be reduced to yield the value of Q at the cell mid face in terms of the geometry and Q. This is a consequence of the geometrically invariant features of triangles. Consider reconstructing an estimate of Q at the mid face of the edge marked with a cross ($\times$) shown in Figure 3.6. Assuming that we have average values of Q at the midpoint for all three faces, $f1, f2$, and $f3$, the algebraic expression for Q according to Equation 3.2 reduces to:

$$Q_x = Q_c + \frac{1}{3}\Big[2Q_{f1} - (Q_{f2} + Q_{f3})\Big] \qquad 3.5$$

This result can be easily derived using a one dimensional Taylor series expansion

$$\begin{aligned} Q_x &= Q(s_0) + \frac{\partial Q}{\partial s}(s - s_0) \\ &= Q(s_0) + \nabla Q \cdot \triangle s \end{aligned}$$

where s_0 is the cell centroid. The higher order correction term is derived by using simple one dimensional finite difference techniques and is written as

$$\nabla Q \cdot \triangle s \approx \frac{Q_{f1} - \frac{1}{2}(Q_{f2} + Q_{f3})}{\frac{3}{2}\triangle s} \triangle s$$

This formula is identical to the higher order correction term in Equation 3.5.

A similar analysis can be performed for the case of averaging Q to the cell vertices. The algebraic expression in Equation 3.2 simplifies to

$$Q_x = Q_c + \frac{1}{3}\Big[\frac{1}{2}(Q_{n1} + Q_{n2}) - Q_{n3}\Big]. \qquad 3.6$$

The terms in the Taylor series expansion are formed consistent with Figure 3.6 as

$$\nabla Q \cdot \triangle s \approx \frac{\frac{1}{2}(Q_{n1} + Q_{n2}) - Q_{n3}}{3\triangle s} \triangle s$$

which reduces to the correct expression in Equation 3.6.

Both of the three dimensional higher order algorithms can be derived using a one-dimensional Taylor series approach by examining Figure 3.7 and exposing the geometrically invariant features of tetrahedrals. Solving the integral expression for the gradient using Equation 3.3, the higher order formula for the mid-point approximation of Q assuming cell values averaged to the mid-face (Type 1 approach A) is

$$Q_x = Q_c + \frac{1}{4}\Big[3Q_{f1} - (Q_{f2} + Q_{f3} + Q_{f4})\Big]. \qquad 3.7$$

For the case of averaging the properties to the nodes, the expression which was first introduced by Frink [17] is

$$Q_{\times} = Q_c + \frac{1}{4}\left[\frac{1}{3}(Q_{n1} + Q_{n2} + Q_{n3}) - Q_{n4}\right]. \qquad 3.8$$

The higher order correction term for the first case is modeled as

$$\nabla Q \cdot \triangle s \approx \frac{Q_{f1} - \frac{1}{3}(Q_{f2} + Q_{f3} + Q_{f4})}{\frac{4}{3}\triangle s} \triangle s$$

which is equivalent to the higher order correction term in Equation 3.7.

The finite difference approximation for the one dimensional gradient using nodal values of Q can be deduced from Figure 3.7 to be

$$\nabla Q \cdot \triangle s \approx \frac{\frac{1}{3}(Q_{n1} + Q_{n2} + Q_{n3}) - Q_{n4}}{4\triangle s} \triangle s.$$

This expression also recovers the formula derived in Equation 3.8.

3.2.3 Monotonicity

The linear reconstruction techniques can introduce new extrema into the solution as a result of the higher order correction involving the gradient terms. A monotonicity principle that limits the maximum and minimum values for the linearly reconstructed function can be enforced to control the overshoots and undershoots. This is accomplished by modifying the functional form of Equation 3.2 to include a weighting factor on the higher order correction term in the Taylor series expansion. A limited form can be written for cell M as

$$Q(x,y)_M = Q(x_0,y_0)_M + \Phi_M \nabla Q_M \cdot \triangle r_M, \quad \Phi \in [0,1]$$

For linear reconstruction, the extrema occur at the vertices of the cell. The monotonicity idea can be enforced by guaranteeing that the limited quantities will be bounded by the neighboring cell centroid values. The quantity Φ is determined by the following steps:

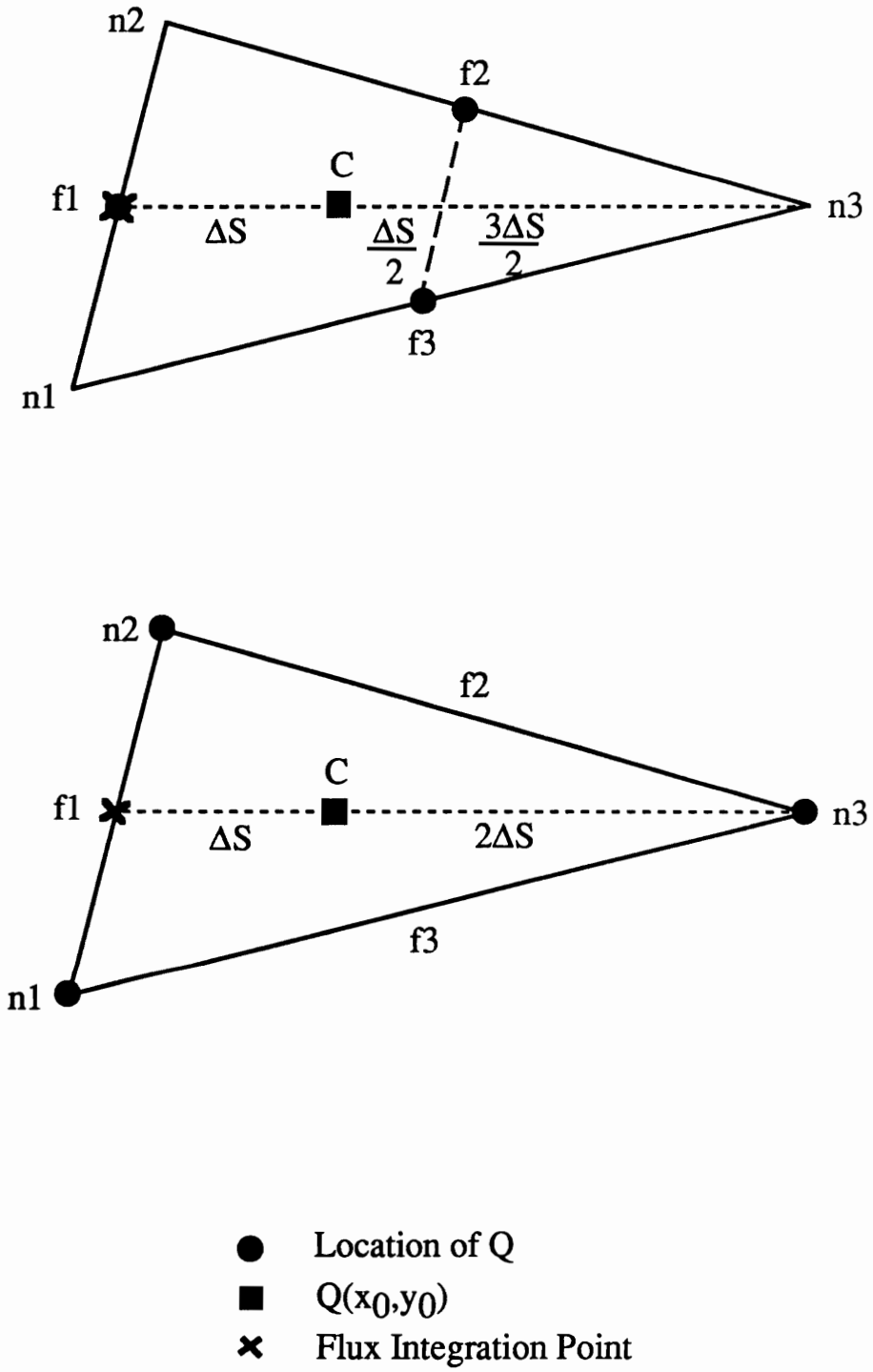


Fig. 3.6 – Geometrically invariant features of triangles

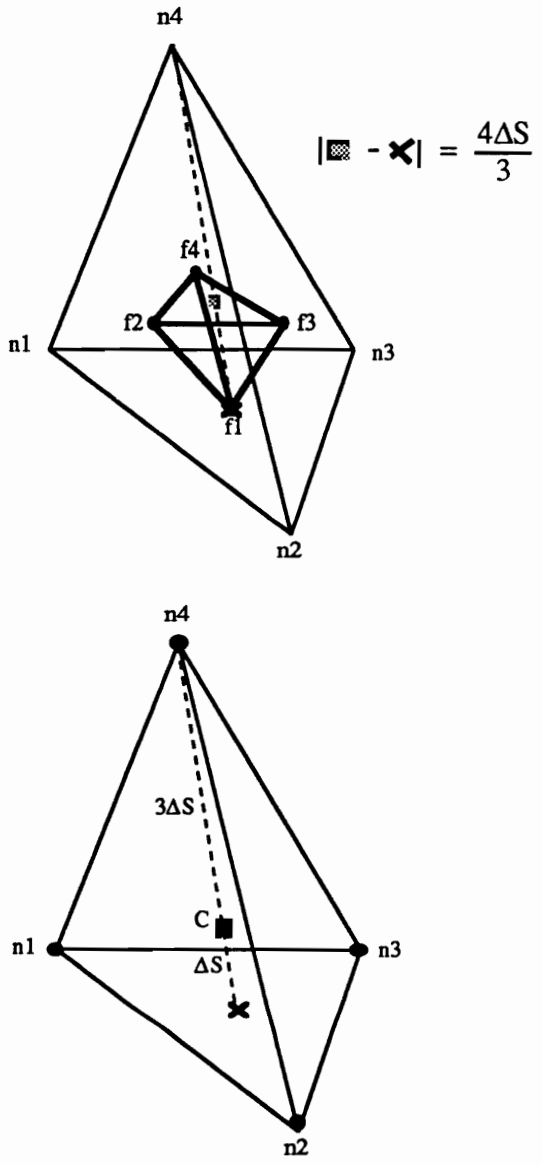


Fig. 3.7 – Geometrically invariant features of tetrahedra

1. Calculate the minimum and maximum values of Q from the immediate cell neighbors according to:

$$Q_M^{min} \leq Q(x,y)_M \leq Q_M^{max}$$

where

$$Q_M^{min} = min(Q_M, Q_{neighbors})$$

$$Q_M^{max} = max(Q_M, Q_{neighbors}).$$

2. Calculate Φ for each vertex j by using

$$\Phi_{M_j} = \begin{cases} min(1, \frac{Q_M^{max} - Q_M}{Q_j - Q_M}), & \text{if } Q_j - Q_M > 0; \\ min(1, \frac{Q_M^{min} - Q_M}{Q_j - Q_M}), & \text{if } Q_j - Q_M < 0; \\ 1, & \text{if } Q_j - Q_M = 0 \end{cases}$$

3. $\Phi_M = min(\Phi_{M_j})$

Although this procedure satisfies the monotonicity principle throughout cell M, a slight variation of this approach implemented by Slack et al. [14] imposes a less restrictive condition. The limiting value of Φ from item 2 is based on the computed values of the function at the midpoint of each face.

3.3 K-exact Reconstruction

3.3.1 Design Guidelines

The process of generating accurate interpolating polynomials in multiple dimensions is vital to the success of developing high order accurate solution algorithms for arbitrary grid topologies. However, the lack of polynomial uniqueness is an indication that these methods have noticeable weaknesses. For example, if we attempt to construct a multidimensional linear polynomial where the data points lie along a straight line, we will not be successful. In addition, if we reconstruct the data for cell A by selecting a set of points too far away from A, the accuracy will eventually degrade. The reconstruction development follows the basic reconstruction design criteria that are well known in

one dimension, however they are rarely discussed and often violated in multiple dimensions. The following is a list of guidelines which is used in the reconstruction process as specified by Barth [13].

1. *K-exactness:* A polynomial in two independent variables of degree k has the form

$$P^k(x,y) = \sum_{i=0}^{k} \sum_{j=0}^{k-i} C_{i,j}\, x^i y^j .$$

   The reconstruction step is $k-exact$ if the process reconstructs polynomials of degree k or less exactly. Notice that in two dimensions we have $\frac{(k+1)(k+2)}{2}$ coefficients, therefore we need this number of independent equations to reconstruct the polynomial exactly.

2. *Conservation of cell averaged quantities:* Given cell average quantities, construct a k degree polynomial which satisfies the following:

$$\bar{Q}_m \equiv \iiint_m Q \; d\Omega_m .$$

   i.e. the polynomial chosen must reconstruct the mean exactly.

3. *Stencil Selection:* The stencil selection process is perhaps the most difficult and challenging criteria to overcome. The support set of cells necessary to perform the reconstruction step that will best represent the solution space is not obvious. In fact, depending of the nature of the flow physics and of the terms we are modeling, separate approaches are necessary. This issue will be addressed in greater detail in Chapter 4.

3.3.1 Satisfying the Guidelines

Barth's approach to satisfying the criteria of conserving cell averaged quantities proceeds by performing a transformation local to the parent cell that involves a rotation and scaling into a new coordinate system $(\bar{x}, \bar{y})$ according to

$$\begin{bmatrix} \bar{x} \\ \bar{y} \end{bmatrix} = \begin{bmatrix} D_{1,1} & D_{1,2} \\ D_{2,1} & D_{2,2} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

where the matrix D satisfies

$$\iiint_m \bar{x}^2 \, d\Omega_m = 1; \quad \iiint_m \bar{y}^2 \, d\Omega_m = 1; \quad \iiint_m \bar{x}\bar{y} \, d\Omega_m = 0; \quad \iiint_m \bar{y}\bar{x} \, d\Omega_m = 0.$$

Conservation of the mean is guaranteed by transforming the standard basis polynomials written as

$$\bar{P} = [1, \bar{x}, \bar{y}, \bar{x}^2, \bar{x}\bar{y}, \bar{y}^2, ...]$$

into the $zero - mean$ basis polynomials $\bar{P}^0$. The transformed polynomial basis is now represented as:

$$\bar{P}^0 = [1, \bar{x}, \bar{y}, \bar{x}^2 - 1, \bar{x}\bar{y}, \bar{y}^2 - 1, ...]$$

where all but the first have zero cell averages. The details of the transformation are provided in Ref. [13] .

The transformation process is applied to every cell in the domain resulting in separate expressions that guarantee the mean conservation principle. In order to generate the reconstruction polynomial, $\frac{(k+1)(k+2)}{2}$ cells are required if the reconstruction is to be $k - exact$. Barth's approach is to include a larger supporting set to reconstruct his polynomial expressions. The stencil choice is based on the equation set that minimizes the Frobenius norm. Recall the Frobenius norm, $A_n$is defined as

$$A_n = \left[\sum_{i,j} |a_{ij}|^2\right]^{\frac{1}{2}}$$

where i, j are the row and column indices, respectively. As a result, the increased cell neighbor set that creates the matrix equations that yield the polynomial coefficients is over-constrained. The solution is computed using a modified Gram-Schmidt algorithm [33] .

It should be mentioned that the consequence of over specifying the matrix equation to build the reconstruction polynomial is that the basic principle of conserving the cell averaged quantities is violated. In addition, the justification for the stencil selection

strategy is weak. In the next section, an improved approach to $k - exact$ reconstruction is discussed.

Chapter 4

K-Exact Reconstruction, A New Approach

4.1 Conservation of Cell Averages

The reconstruction process is defined by examining Equation 2.2 and addressing the inverse problem of accurately approximating the pointwise function $Q(x, y, z)$ to a desired accuracy given cell average data, $\bar{Q}$.

Consider a general three-dimensional polynomial P of degree k written as:

$$P^k(x,y,z) = \sum_{i=0}^{k} \sum_{j=0}^{k-i} \sum_{l=0}^{k-(i+j)} C_{i,j,l}\, x^i y^j z^l. \tag{4.1}$$

In three dimensions, a polynomial of degree k contains $\frac{(k+1)(k+2)(k+3)}{6}$ degrees of freedom. Recall that $1-D$ and $2-D$ polynomials contain $k+1$ and $\frac{(k+1)(k+2)}{2}$ coefficients, respectively. If we apply the polynomial representation of Equation 4.1 to evaluate Equation 2.2 we conclude, after careful examination, that for cell m:

$$\bar{Q}_m = \sum_{i=0}^{k} \sum_{j=0}^{k-i} \sum_{l=0}^{k-(i+j)} \frac{1}{\Omega_m} C_{i,j,l} \iiint_{\Omega_m} x^i y^j z^l \, d\Omega. \tag{4.2}$$

The volume integral above is a function only of the control volume discretization which bounds cell m. Direct integration formulas in the form of Gauss quadrature rules are readily available to evaluate the integral for various elements including triangles, quadrilaterals, hexahedra, and tetrahedra. A summary of the formulas are provided in Ref. [34] . For example, a function integrated over the volume of the tetrahedron shown in Figure 4.1 can be evaluated according to,

$$\int_0^1 \int_0^{1-\xi} \int_0^{1-\xi-\eta} f(\xi,\eta,\zeta) d\xi d\eta d\zeta \cong \sum_{r=1}^{N_r} \omega_r f(\xi_r, \eta_r, \zeta_r),$$

where N_r is the number of points, ξ_r, η_r, ζ_r, are the coordinates, and ω_r are the weights used in the quadrature formulas. Notice that if $f(\xi, \eta, \zeta)$ is a polynomial of degree k, sufficient N_r pairs of coordinates and weights can be selected to integrate the function exactly.

The transformation of the volume integral from the real geometrical domain to the simpler integration space of the reference element is easily accomplished with a change in variables which yields the following relationship:

$$\int_{\Omega^{real}} f(\bar{\mathbf{x}}) dx dy dz = \int_{\Omega^{ref}} f(\bar{\mathbf{x}}(\bar{\xi}))\, det(J)\, d\xi d\eta d\zeta,$$

where J is the determinant of the Jacobian matrix, $\bar{\mathbf{x}}$ represents the spatial coordinates x, y, z and $\bar{\xi}$ represents the reference coordinates ξ, η, ζ. In the case of a tetrahedral element, the following relationships apply:

$$det(J) = 6\Omega_m$$

$$\bar{\mathbf{x}}(\xi, \eta\, \zeta) = (1 - \xi - \eta - \zeta)\bar{\mathbf{x}}_{n1} + \xi\bar{\mathbf{x}}_{n2} + \eta\bar{\mathbf{x}}_{n3} + \zeta\bar{\mathbf{x}}_{n4}$$

where Ω_m is the real element volume.

Substituting the above expressions into Equation 4.2 yields the following linear equation for the coefficients $C_{i,j,l}$, which can be generated for each cell within the domain:

$$\bar{Q}_m = \sum_{i=0}^{k} \sum_{j=0}^{k-i} \sum_{l=0}^{k-(i+j)} C_{i,j,l}\; 6 \sum_{r=1}^{N_r} \omega_r x_m^i(\xi_r, \eta_r, \zeta_r) y_m^j(\xi_r, \eta_r, \zeta_r) z_m^l(\xi_r, \eta_r, \zeta_r).$$

As a reminder, this equation contains $\frac{(k+1)(k+2)(k+3)}{6}$ unknown polynomial coefficients $C_{i,j,l}$. Therefore, the final tasks in completing the reconstruction step are selecting the support stencil of equal number and solving the linear system for the coefficients. For example, if $R^k_{(m)}$ is a collection of cells to be used in reconstructing a k degree pointwise distribution of density ρ for cell m, the resulting linear system to yield the unknown polynomial coefficients is:

$$\left[\bar{\rho}_m = \sum_{i=0}^{k} \sum_{j=0}^{k-i} \sum_{l=0}^{k-(i+j)} C_{i,j,l}\; 6 \sum_{r=1}^{N_r} \omega_r x_m^i(\bar{\xi}) y_m^j(\bar{\xi}) z_m^l(\bar{\xi})\right]$$

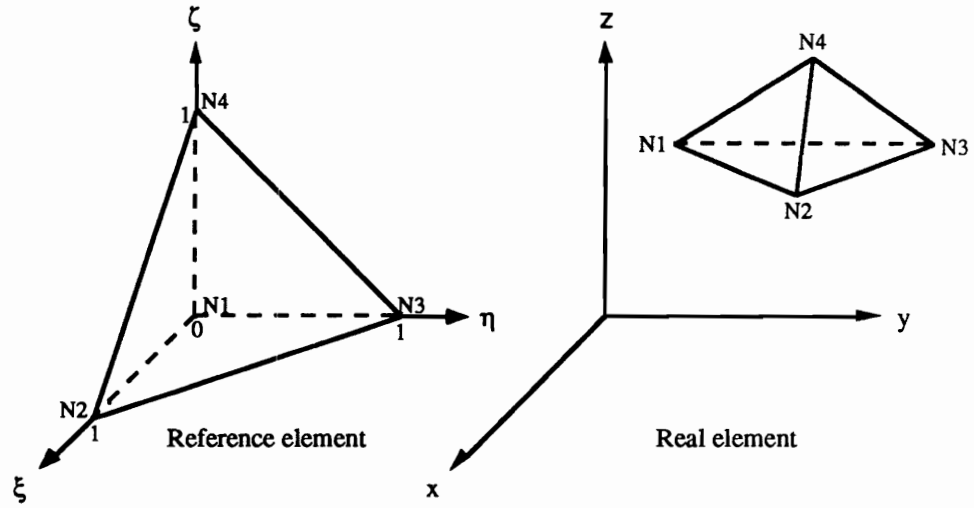


Fig. 4.1 – Reference and real domains for tetrahedral element.

$$m = R_m^k(1),\ R_m^k(\frac{(k+1)(k+2)(k+3)}{6})$$

or written compactly:

$$[\Psi][C_{ijl}] = [\Phi] \tag{4.3}$$

As was mentioned earlier, the possibility of selecting a stencil that results in a singular set of equations is very likely. When this occurs, the stencil is altered by substituting additional cells from the support cell list. An alternative approach for handling the singular equation set is to utilize Singular Value Decomposition algorithms [35] which will invert this type of matrix. In fact, another useful feature of this algorithm is that it allows more equations than unknowns permitting the support stencil to be larger than the required polynomial coefficients (or degrees of freedom). However, as a consequence, the conservation of the mean criteria will be violated, and the computational costs increase. In addition, as the stencil set increases in size, less valid polynomial data is introduced into the reconstruction process which may affect the accuracy. For example, a two dimensional reconstruction for cell A using a polynomial of degree $k = 1$ would require at least 3 cell averages. If we include cells A, B and C and the resulting stencil yields a non-singular set of equations, then conservation of the mean is guaranteed for the entire domain bounded by the three cells. However, if the equation set is singular, then the resulting polynomial does not integrate the mean values and merely represents a "best" fit for the data.

Returning to Equation 4.3 and examining this system of equations, we note a few important points. Ψ is strictly a function of the grid topology, and if the support stencil remains unchanged in time, Ψ is fixed. Φ however is time varying requiring the system to be solved every iteration. This procedure can be implemented by decomposing Ψ into lower and upper triangular matrices and saving the decomposition. Both LU decomposition and Singular Valued Decomposition have this feature. For each time level's

cell averages, the coefficients can be obtained by simple forward and backward substitution. In fact, all of the reconstructions which share a common stencil can be computed simultaneously using the same factored Ψ. If a stencil should change during the solution advancement process, as with ENO schemes and rotated schemes, a complete LU decomposition would be necessary.

Solving Equation 4.3 yields the polynomial coefficients used to determine the pointwise variation for the function defined by Equation 2.5. In all of these results, we select the set of primitive variables ρ, u, v, w, p as the functions. The inviscid and viscous flux integrals can be evaluated at the Gauss points using Equations 2.4 and 2.6, respectively. The gradients of the properties that appear in the viscous flux vectors are easily obtained through polynomial differentiation.

4.2 Efficient Reconstruction

An efficient implementation of the reconstruction algorithm proceeds by realizing that the reconstruction polynomials are evaluated only at the Gauss points. As a result, the work in generating the polynomial expression in Equation 2.5 can be greatly reduced. Notice that if the polynomial is evaluated at a prescribed value of x, y, z, (for example at the mid-point of the face) the only non-constant terms in the reconstruction algorithm are the stencil of cell averages, Φ. Consequently, these terms can be individually grouped resulting in the following expression:

$$Q_{L,R}^{k_i} = \sum_{i=i}^{R_c} W_i \Phi_i.$$

where R_c is the number of polynomial coefficients and W_i is a constant weighting factor multiplying the cell average Φ_i. The constants, W_i, are precomputed at a one time expense and stored in memory throughout the calculation. The actual calculation of W_i can be performed efficiently as well. For example, consider a $k = 1$ reconstruction in three dimensions. Four cell averages are necessary to complete the stencil. The

computational effort in generating the weights W_i reduces to a simple 3 by 3 inversion, for which the solution can be directly coded. As a result, the LU decomposition and subsequent forward and backward substitution steps are eliminated. This results in a significant memory and CPU savings.

4.3 Stencil Selection Algorithms

Several factors must be considered when selecting a stencil to be used in the reconstruction algorithms. Although it is difficult to determine the "optimum" support set due to the virtually limitless combinations of possible stencils, we can key on two fundamental items: the nature of the governing equations and the geometrical features of the discretized domain.

The Euler equations are hyperbolic in nature and are typically modeled as a system of waves containing characteristic information. Upwind schemes have been developed to extract the information by extrapolating the flow properties from both sides of a cell face. A consistent idea for selecting a stencil to model the terms that appear in these equations is straightforward and is presented in this section. When the effects of viscosity are considered, the complete Navier Stokes equations are solved. The elliptic nature of the viscous terms reveals that information travels in all directions, hence there is a need to include a stencil which is not biased in any preferred direction.

Geometrical considerations are also important and play a role in determining the stencil. This is evident for structured flow solver algorithms where upwind interpolating polynomials are generated by identifying computational cells which are aligned in the direction normal to a cell face. Of course, the task of locating upwind cells is greatly simplified for structured grids, since incrementing the logical coordinate indexes yields the adjacent cell neighbors. While this is more difficult to accomplish in a general grid environment, search algorithms can be developed to provide an analogous result.

The inviscid stencil

The inviscid flux calculation proceeds by reconstructing the primitive variables $\rho, u, v, w,$ and p with polynomials of degree k_i (i denotes inviscid). Two separate reconstructions corresponding to the right and left fluid dynamic states will be performed for each face in the domain. The polynomials will be evaluated at the specified Gauss points according to Equation 2.4. Fixed stencils are investigated for $k_i = 0, 1,$ and 2 degree reconstructions and are illustrated in two dimensions in Figures 4.2 and 4.3. Notice that upwind and upwind-biased stencils are easily obtainable for both structured and unstructured grids using the following upwinding stencil selection algorithm.

Upwinding stencil algorithm

Consider the bold edge in Figure 4.4. We begin by identifying the upwind cell parent A to be used in reconstructing the left fluid dynamic state. For $k_i = 0_{upwind}$, the stencil is complete. For $k_i = 1_{upwind}$, we require two additional support cells to conclude the stencil. The following algorithm accomplishes this task:

1. Gather the first cell neighbors that surround the parent cell.
2. Construct unit vectors from the cell face center to the neighboring cell centroids.
3. Perform a dot product with the outward facing edge normal and the unit vectors to the cell neighbors.
4. Sort the list from smallest to largest.
5. Beginning at the top of the list, select the required number of entries.

The considerations which influenced the algorithm development included contiguous cell stencils, stencils which align in the upwind direction, and stencils that remain local to each face. Notice that negative dot products indicate an upwind cell, and applying this algorithm to Figure 4.4 would yield a stencil including cells A, C, and D. For a $k_i = 2_{upwind}$ reconstruction, we build on the $k_i = 1_{upwind}$ stencil by selecting three additional cells to complete the stencil. The algorithm above is repeated with a support

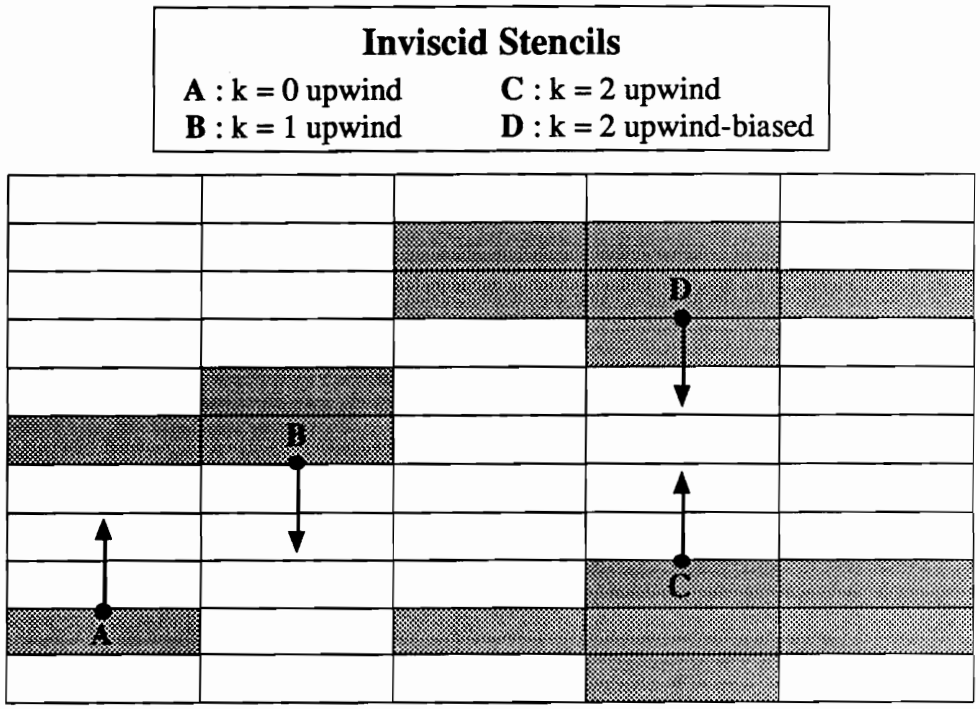


Fig. 4.2 – Sample inviscid stencils for quadrilateral elements.

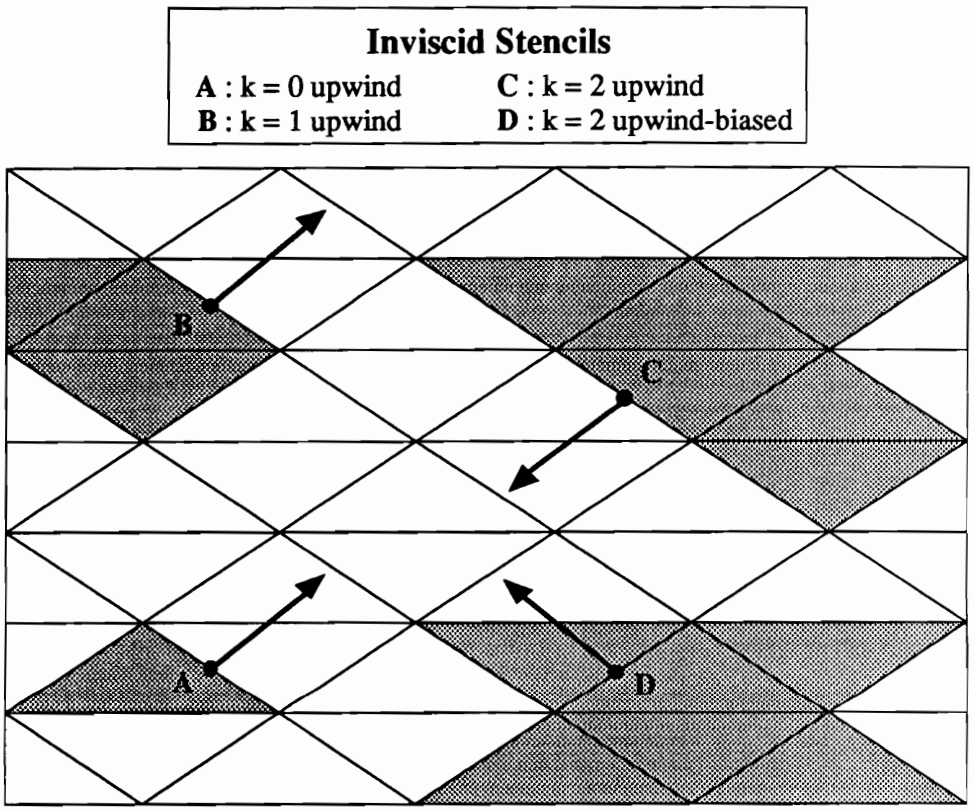


Fig. 4.3 – Sample inviscid stencils for triangular elements.

set including the first cell neighbors for cells A, C, and D. The first three entries from the sorted list are accepted (F, G, and H in our example). A $k_i = 2_{upwind-biased}$ stencil is constructed by including cells whose dot product yields a positive value. For all of our results, we restrict the level of biasing to include only 1 cell, and it is identified as the nearest downwind cell for the given edge (cell B replaces cell G in our example).

Viscous flux calculation

The viscous flux evaluation is also based on reconstructing the primitive variable set ρ, u, v, w, and p with polynomials of degree k_v (v denoting viscous) for each edge in the domain. The gradient quantities of ρ, u, v, w, and p are formed by simply differentiating the polynomials accordingly. Chain-rule differentiation was applied to the square of the speed of sound (a^2) gradients in terms of the density and pressure. For comparison, the quantity a^2 was reconstructed separately, however, no noticeable differences resulted.

The viscous stencil

The stencil used to generate the polynomials for the viscous flux calculation are based on a locality principle which attempts to align the edge at the geometrical center of the stencil (or as close as possible). This concept is consistent with the central difference approach used in structured algorithms. Fixed stencils are investigated for $k_v = 1$, and 2 degree reconstructions and are illustrated in two dimensions for both structured and unstructured grids in Figure 4.5.

4.4 Limiting

In order to control the numerical oscillations that occur in the presence of discontinuities, the reconstructed quantities will be bounded using the following rule,

$$Q_{L,R} \in [\Phi_L, \Phi_R]$$

which states that the interface properties must not introduce new extrema. Φ_L and Φ_R are the cell averages used in the left and right reconstructions.

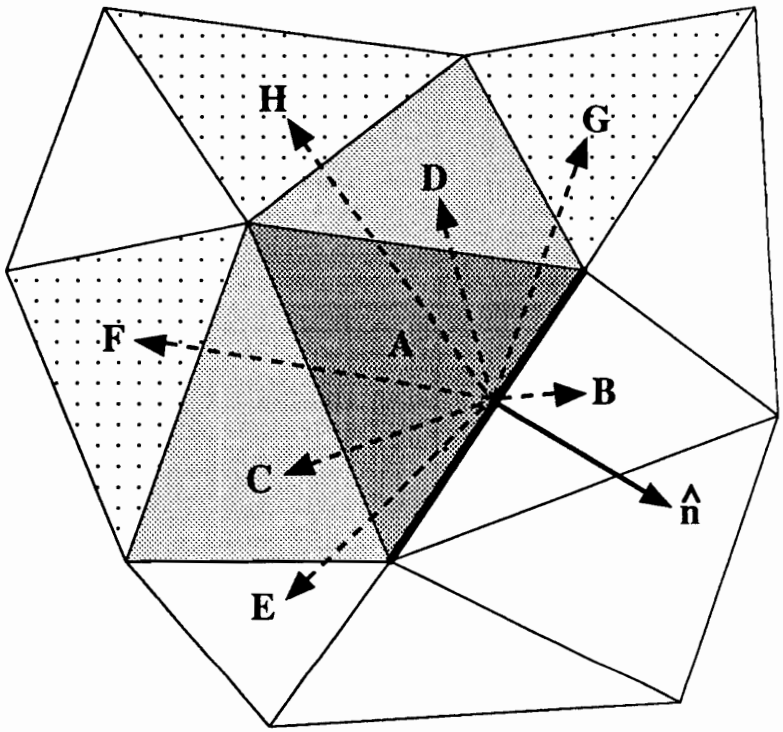


Fig. 4.4 – Example for upwind stencil selection algorithm.

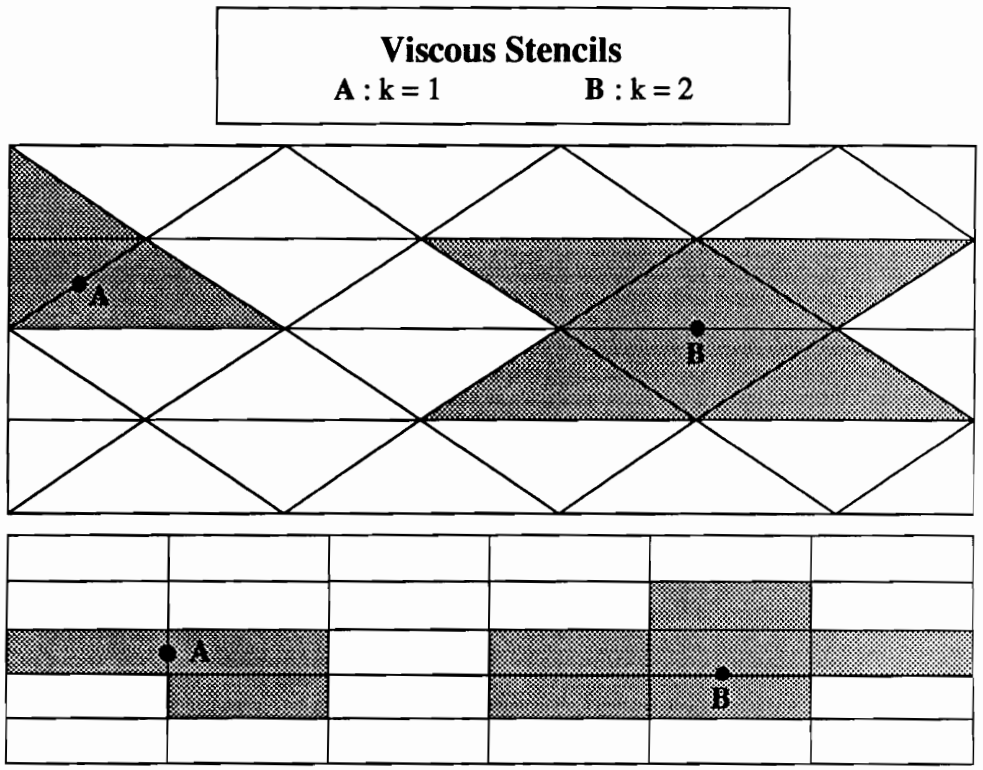


Fig. 4.5 – Sample viscous stencils for triangular and quadrilateral elements.

4.5 Reconstruction Accuracy

The accuracy of the reconstruction process is investigated by computing the exact cell averages of two dimensional functions on a mesh, then reconstructing the original functions with polynomials, and computing the error via norms. Recall that the standard norm L_p is defined as

$$L_p = \|F(exact) - F(approximate)\|_p \equiv \left[\int_\Omega |F(exact) - F(approximate)|^p \; da\right]^{\frac{1}{p}}$$

This expression represents the difference between the exact solution and the computed solution integrated over the entire domain Ω. For our purposes, the functional form of $F(approximate)$ is a polynomial expression, and $F(exact)$ will be known test functions. Direct integration formulas are readily available in the form of highly accurate Gauss quadrature rules to evaluate the above expression. Notice that if the test function is a polynomial, the integration rules are exact, however smooth non-polynomial functions can be predicted quite accurately using a large number of Gauss points. Realizing that the error integrated over the domain is equivalent to the summation of the integrated errors within each cell, the above equation can be rewritten as:

$$L_p = \left[\sum_{c(m)} \left[\int_{\Omega_m} |F(exact) - F^k|^p \; d\Omega\right]\right]^{\frac{1}{p}}$$

where k is the degree of the polynomial. Evaluating the expression using 12 point quadrature in two-dimensions will integrate the expression exactly for polynomials up to degree 6. The final form of the L_p is

$$L_p = \left[\sum_{c(m)} \left[\sum_{f=1}^{12} |\omega_f \mathbf{F}(\xi_f, \eta_f)|^p\right]\right]^{\frac{1}{p}} \qquad 4.4$$

where

$$\mathbf{F} \equiv F(exact) - F^k.$$

In one dimension, the L_p norm is $O(\Delta x^n)$ for an n^{th} order accurate solution. The accuracy n is determined by plotting the L_p norm versus Δx on a log-log plot. The slope

of the resulting line is the accuracy of the scheme. For an n^{th} order accurate solution on two dimensional uniform domains, the L_p norm is $O(\Omega \Delta s^n)$, where Δs is a uniform length scale representative of Δx and Δy. When assessing the reconstruction accuracy for two dimensional cases, it is critical to maintain uniform cell size during mesh refinement. For triangular discretizations, the length scale is difficult to determine since the control volumes do not directly align themselves with the two coordinate directions, however, if the domain is uniformly refined the relative change in cell size is equal for all cells. Notice that the slope, and hence accuracy, can be computed in a discrete sense as follows

$$n \equiv \frac{log(\frac{e_{\Delta s_1}}{e_{\Delta s_2}})}{log(\frac{\Delta s_1}{\Delta s_2})}. \tag{4.5}$$

If the triangles are equilateral we can use the distance from the centroid to any of the vertex points as the length Δs. The simple relationship holds

$$\Delta s \approx \sqrt{\frac{\Omega}{1.3}}$$

where Ω is the area of the equilateral triangle.

4.5 Validation of Reconstruction Algorithm

Two types of test functions are selected to validate the reconstruction algorithm: polynomials and a hyperbolic sin function. The criteria for validation is based on successfully satisfying the design criteria presented earlier. Specifically, these are demonstrating the ability to guarantee conservation of mean quantities and reconstructing k degree polynomials exactly. Reconstructing polynomial data using polynomial functions will demonstrate the $k-exactness$ criteria. Hyperbolic sin functions are smooth in nature and will show the performance of the method when reconstructing non-polynomial data. The stencil selection algorithm for this series of tests is not an important issue, as the purpose is to only show that for given functions, exact reconstruction procedures are possible. For this purpose, a stencil is selected based on a nearest neighbor criteria. The

performance of the method applied to a actual solution algorithm will be provided in the next section.

The following four polynomial functions of increasing degree k and the hyperbolic function selected for this test and are listed below and shown in Figure 4.6.

$$
\begin{aligned}
f(x,y) &= P^0(x,y) = 1.0 \\
f(x,y) &= P^1(x,y) = 1.0 + x + y \\
f(x,y) &= P^2(x,y) = 2.5 - 10x + y + 10x^2 + 2xy + 5y^2 \\
f(x,y) &= P^3(x,y) = 2.5 + 5.25x + .4y - 1.5x^2 + xy + 8y^2 + x^3 - 20.7x^2y + 8.3xy^2 - .2y^3 \\
f(x,y) &= \frac{sin(\pi y) \times sinh(\pi x)}{sinh(\pi)}.
\end{aligned}
\tag{4.6}
$$

Recall that reconstructing polynomials of degree k contain $\frac{(k+1)(k+2)}{2}$ coefficients. The test functions will be reconstructed using polynomials of degree $1, 2$, and 3, therefore the resulting polynomial coefficients will be represented as:

$$
\begin{aligned}
P^0(x,y) &= c_0 \\
P^1(x,y) &= c_0 + c_1 x + c_2 y \\
P^2(x,y) &= c_0 + c_1 x + c_2 y + c_3 x^2 + c_4 xy + c_5 y^2 \\
P^3(x,y) &= c_0 + c_1 x + c_2 y + c_3 x^2 + c_4 xy + c_5 y^2 + c_6 x^3 + c_7 x^2 y + c_8 xy^2 + c_9 y^3.
\end{aligned}
$$

The validation procedure involves three steps.

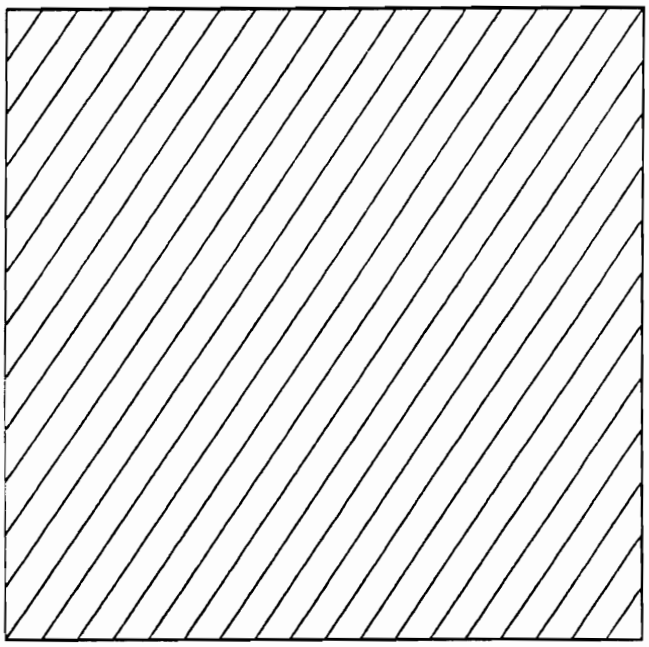
1. Area average the test functions onto the mesh by solving

$$\bar{f}_m = \frac{1}{\Omega_m} \iiint_{\Omega_m} f(x,y)\, d\Omega$$

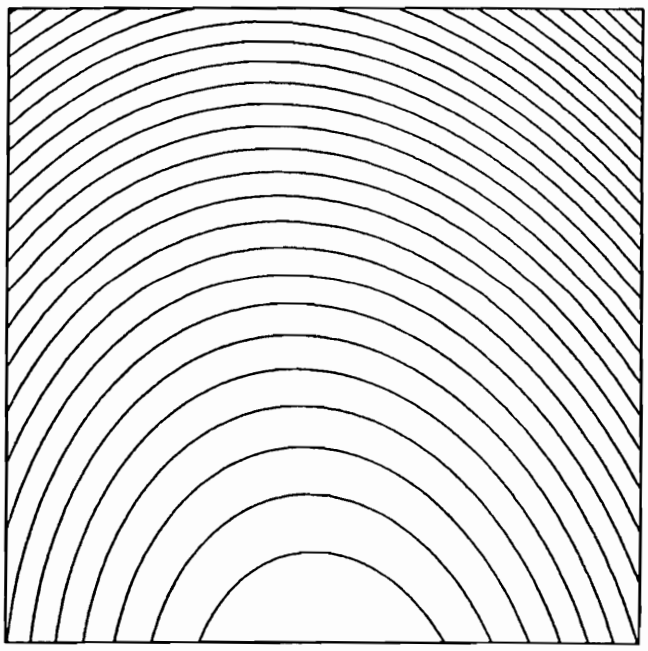
   for each cell m using the functions $f(x,y)$ listed in Equation 4.6 and sufficient quadrature. For all results, 12 point Gauss quadrature is used which will integrate two dimensional polynomials of degree 6 exactly.

2. Reconstruct polynomials of degree $1, 2$, and 3 using the procedure in section 4.1.

3. Compute the error norms and determine the accuracy of the scheme.

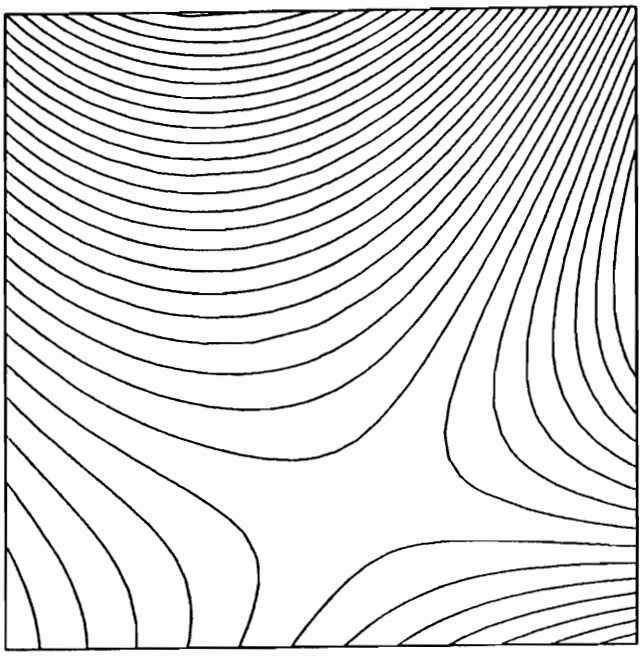


a – Degree 1 polynomial

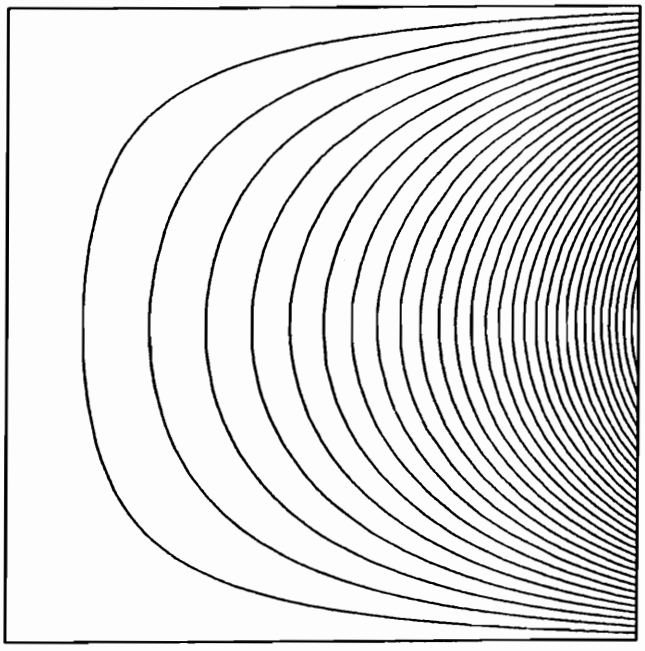


b – Degree 2 polynomial

Fig. 4.6 – Test functions used for reconstruction validation



c – Degree 3 polynomial



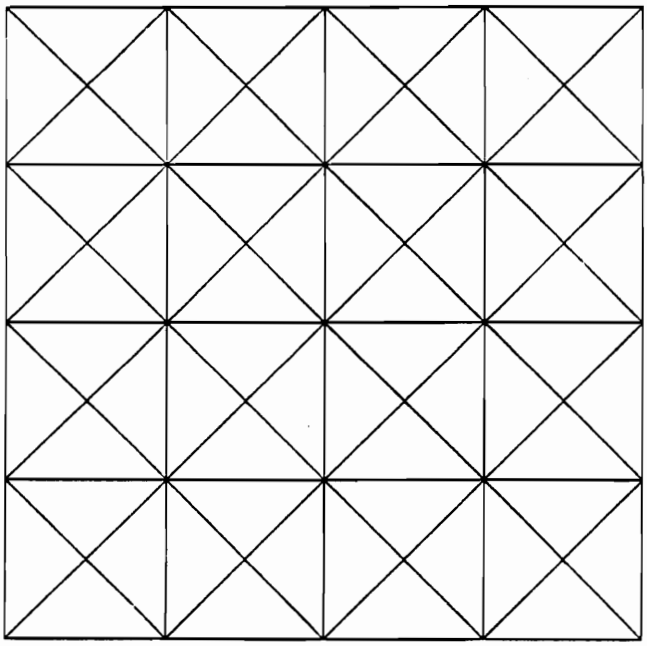
d – Hyperbolic function

Fig. 4.6 – Test functions used for reconstruction validation (Continued).

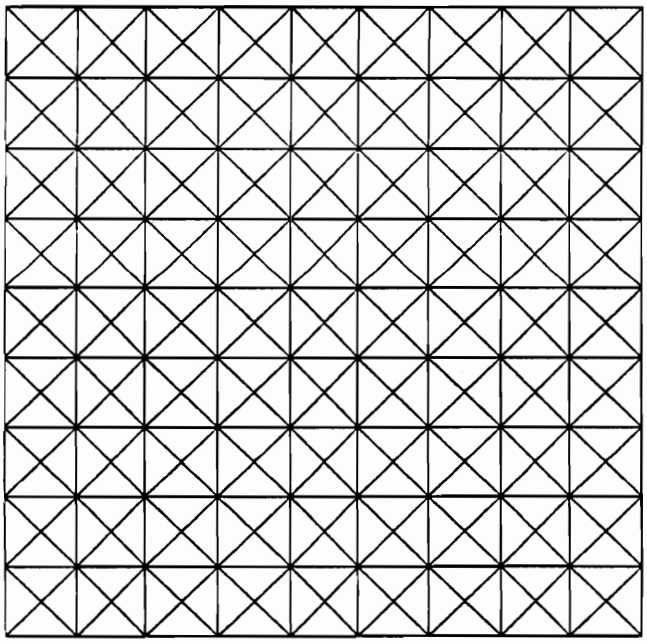
The three triangular grids used in this investigation are shown in Figure 4.7 and consist of 64, 324, and 1444 cells, respectively. They were constructed by subdividing a square domain $I \in [0,1] \times [0,1]$ into triangles. Example stencils used in the reconstruction step are provided in Figure 4.8 for the medium grid. The bold dot indicates the parent cell for the stencil.

A visual critique of the accuracy of the reconstruction process is realized using piecewise contouring. For each cell in the domain, the corresponding reconstruction polynomial is contoured within the cell bounds. The result is a discontinuous interface value between adjacent cells. This is a very powerful and intuitive approach to assessing the accuracy of the procedure. Consider Figures 4.9 and 4.10 which demonstrates the contouring procedure for the degree 3 polynomial and hyperbolic functions, respectively. Detailed in the figures are the result for degree 1, 2, and 3 polynomial reconstructions. Notice the improvement in the solutions for increasing k and that the algorithm reconstructed the first function exactly for $k = 3$. The linear, quadratic, and cubic behavior of the reconstructions are visually apparent within the cell contours. In both cases, the features of the discretized domain become difficult to distinguish with increased reconstruction accuracy.

The accuracy of the reconstruction is investigated by computing the L_1 error norm according to Equation 4.4 using highly accurate quadrature rules. Figure 4.11 shows the variation in solution error as a function of the reconstruction polynomial degree k for the degree 3 polynomial and the hyperbolic test functions. The error trends are consistent with expectations, as the level of error is reduced for increasing values of k (p refinement) and, for a given level of k, the error reduces with increased mesh densities (h refinement). Notice that the degree 3 polynomial test function is reconstructed exactly using $k = 3$ as the error calculation is effectively zero, i.e. machine zero, for all three meshes. In order to quantify the numerical accuracy of the method, the error versus mesh spacing is

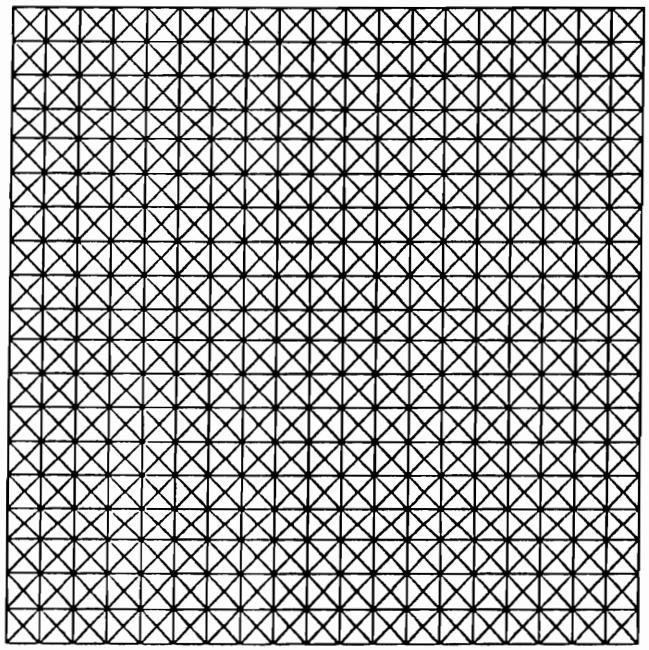


a – grid.1, 64 **cells**



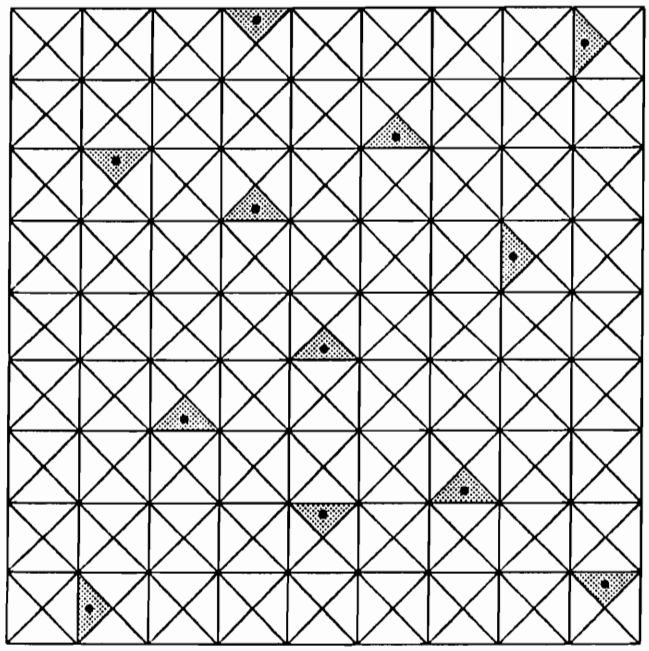
b – grid.2, 324 **cells**

Fig. 4.7 – Triangular grid domains for two dimensional test functions

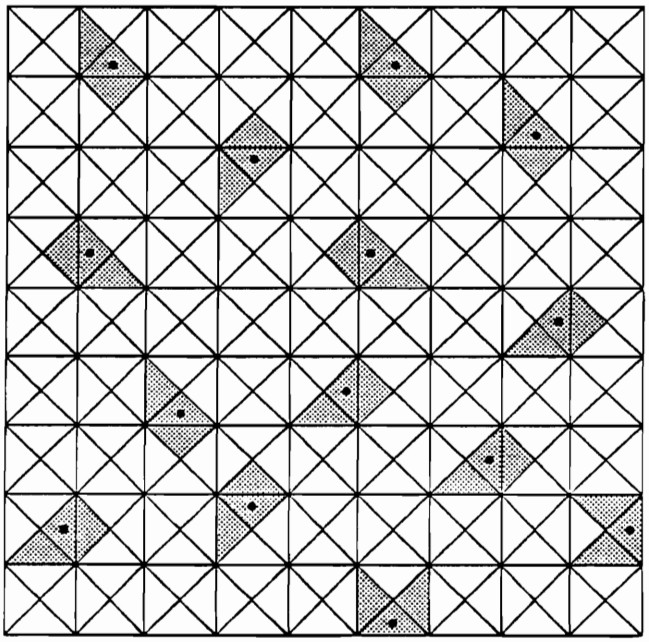


c – grid.3, 1444 **cells**

Fig. 4.7 – Triangular grid domains for two dimensional test functions (Continued).

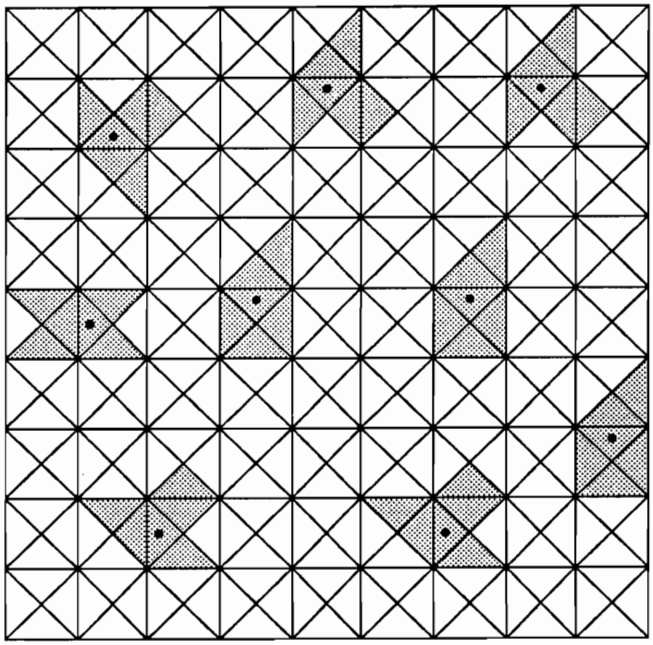


a – Stencil for $k = 0$

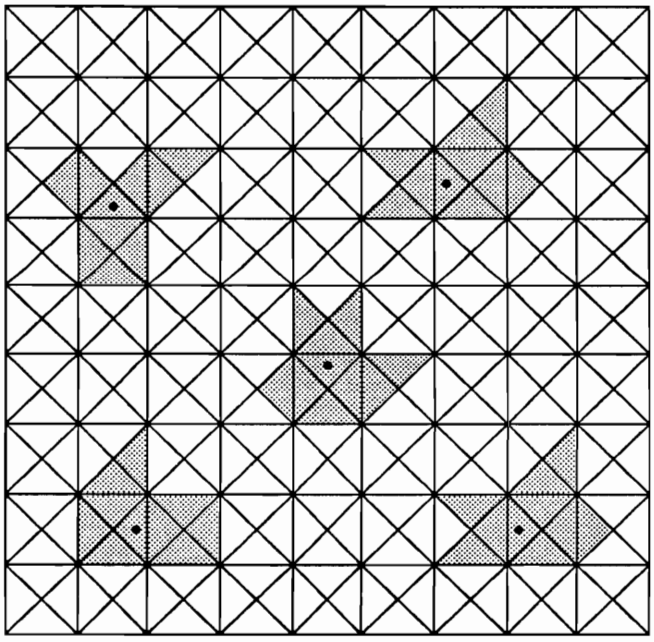


b – Stencil for $k = 1$

Fig. 4.8 – Support set of cells for grid.2

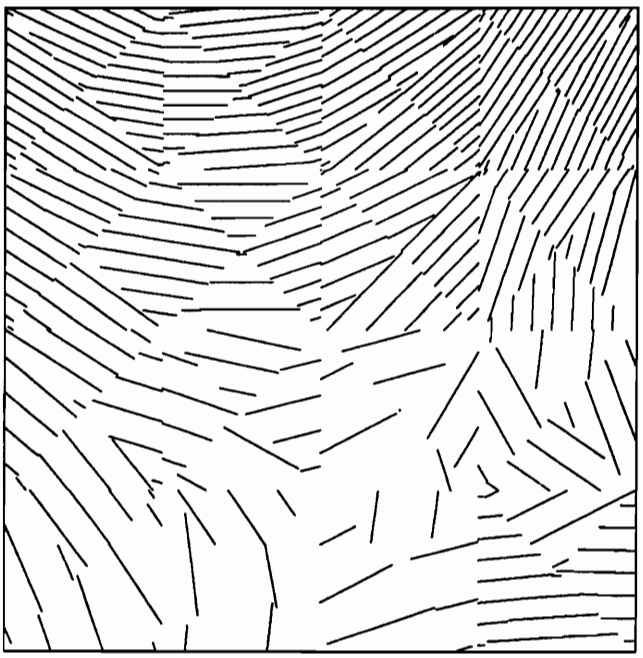


c – Stencil for $k = 2$

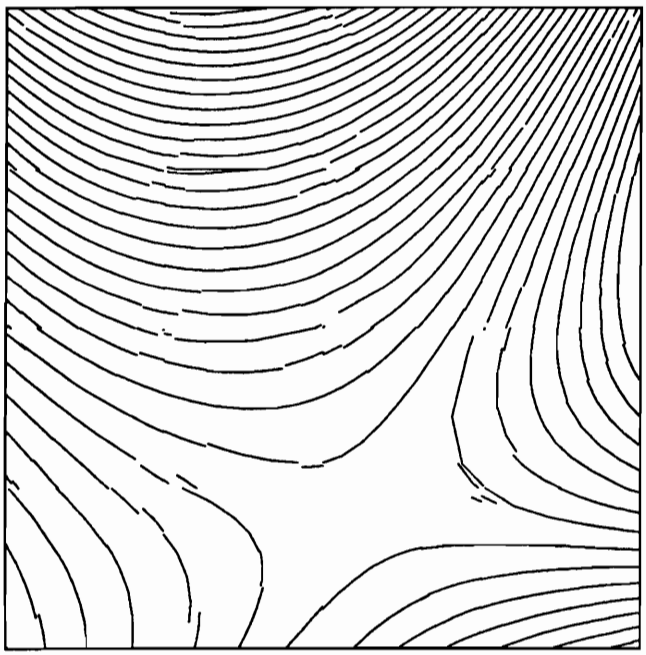


d – Stencil for $k = 3$

Fig. 4.8 – Support set of cells for grid.2 (Continued).

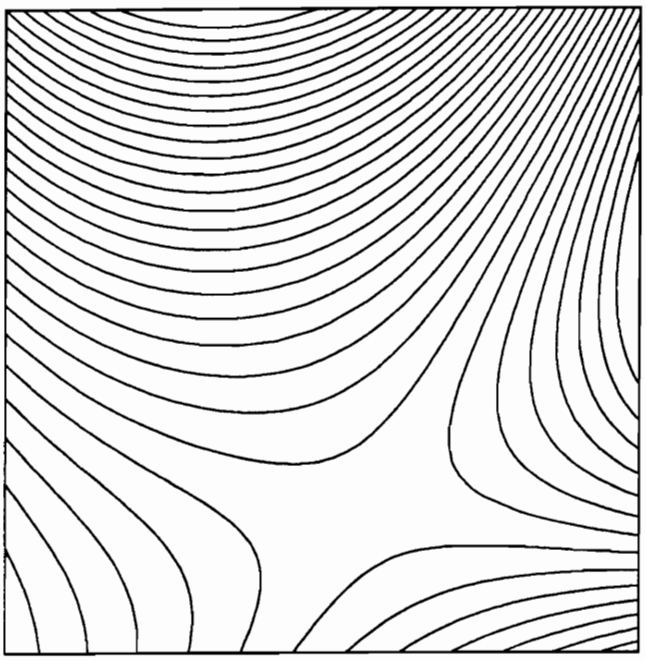


a – $K = 1$ reconstruction



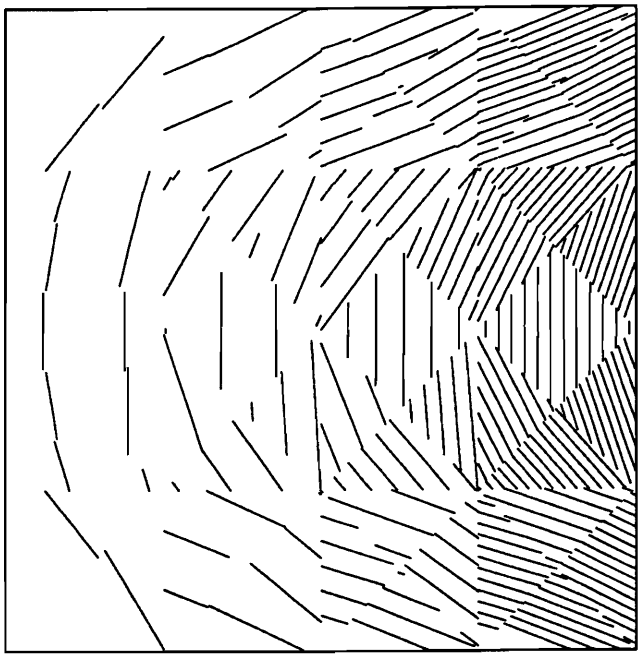
b – $K = 2$ reconstruction

Fig. 4.9 – Cell-wise contours for degree 3 test function

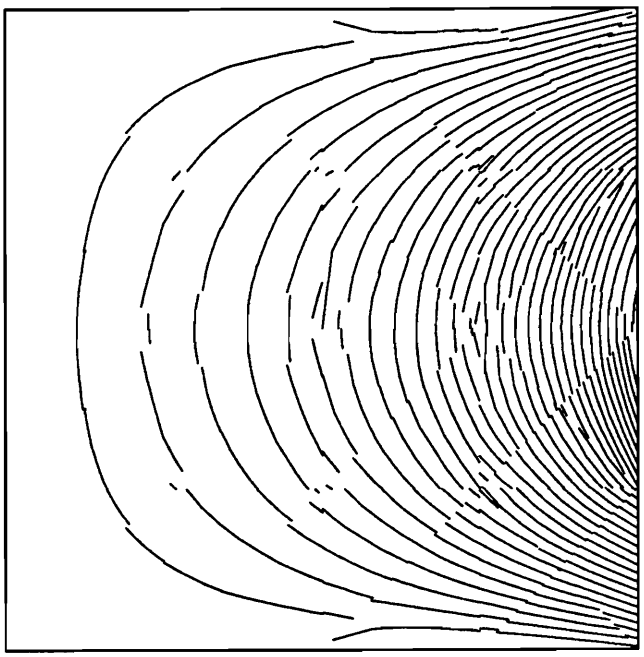


c – $K = 3$ reconstruction

Fig. 4.9 – Cell-wise contours for degree 3 test function (Continued).

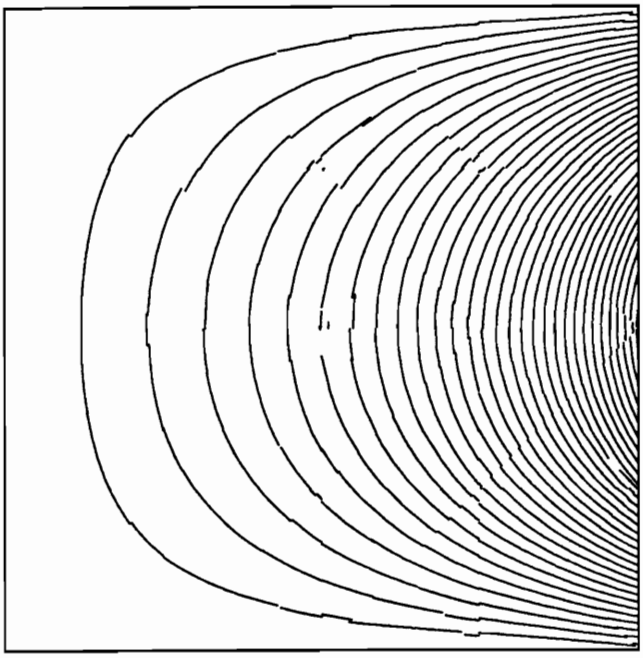


a – $K = 1$ **reconstruction**



b – $K = 2$ **reconstruction**

Fig. 4.10 – Cell-wise contours for hyperbolic test function



c – $K = 3$ reconstruction

Fig. 4.10 – Cell-wise contours for hyperbolic test function (Continued).

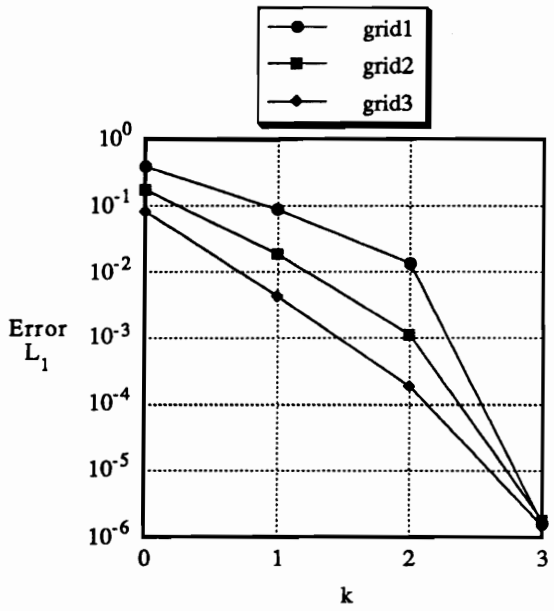
plotted on a log-log scale, and the slope of the line reveals the order of the accuracy. The results are plotted in Figure 4.12 and listed in Table 4.1. The numerical values presented in the table represent an average calculation of the slope using Equation 4.5. From this investigation, k degree polynomials are determined to be $O(k+1)$ which verifies that the reconstruction procedure is implemented correctly.

4.6 Validation of Solution Algorithm

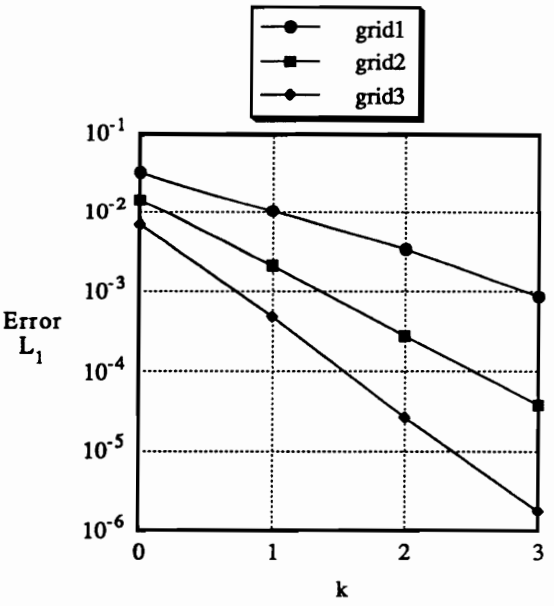
Two numerical experiments will be considered to evaluate the performance of the reconstruction procedure within a finite volume setting. Accurate solutions to elliptic boundary value problems will be a key capability to lay the framework for solving the Navier Stokes equations. Specifically, the shear stress and heat flux terms require the accurate discretization of gradient quantities. This requires careful stencil selection to insure a successful algorithm. In addition, the issue of the accuracy of the flux integration procedure needs to be addressed. In order to utilize the accuracy of the reconstruction process, it is necessary to implement flux integration techniques of sufficient order. That is to say, the accuracy of the flux quadrature should not degrade the accuracy obtained by the reconstruction step. In addition, it should not be overly accurate as this results in a waste of computer resources. Multi-point Gaussian quadrature of various orders of accuracy will be used and compared in the integration. As a reminder, for straight line intervals, N point Gaussian quadrature integrates polynomials of degree $2N-1$ exactly and is $2N$ order accurate.

As was mentioned previously, the gradient expressions common in elliptic boundary value problems as well as the viscous terms in the Navier Stokes equations are formed via polynomial differentiation computed as

$$\frac{\partial P^k(x,y)}{\partial x} = \frac{\partial}{\partial x}\left(\sum_{i=0}^{k}\sum_{j=0}^{k-i} C_{i,j}\, x^i y^j\right)$$

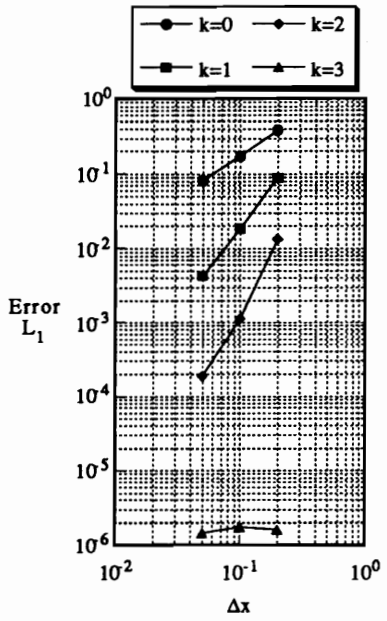


a – Degree 3 polynomial

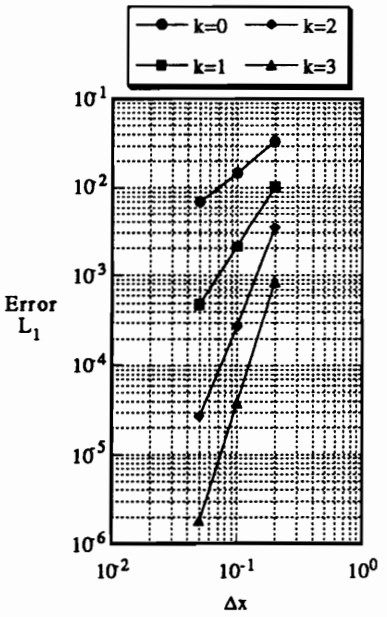


b – Hyperbolic function

Fig. 4.11 – Reconstruction error summary for test functions.



a – Degree 3 polynomial



b – Hyperbolic function

Fig. 4.12 – Reconstruction error comparison for test functions.

Table 4.1 – Numerical accuracy summary for test functions.

Degree of Reconstruction Polynomial	Accuracy Order
k = 0	.992
k = 1	1.946
k = 2	3.051
k = 3	-----

a – Degree 3 polynomial

Degree of Reconstruction Polynomial	Accuracy Order
k = 0	.999
k = 1	1.974
k = 2	3.102
k = 3	3.957

b – Hyperbolic function

$$\frac{\partial P^k(x,y)}{\partial y} = \frac{\partial}{\partial y}(\sum_{i=0}^{k} \sum_{j=0}^{k-i} C_{i,j}\, x^i y^j).$$

Consistent with structured algorithms, a separate reconstruction is performed for each face in the domain thus allowing greater flexibility as compared to a single reconstruction valid for all faces surrounding the cell. This becomes very apparent when we consider that the polynomial derivative for linear ($k = 1$) reconstruction is a constant in both the x and y directions. As a result, the value of the gradient is identical for all points in the cell domain. Not only is this a poor representation of the solution gradient at the cell mid-face, but the robustness of this procedure in a solution advancement algorithm is extremely sensitive to grid characteristics, so this is not a practical option. The alternative of separate calculations of the gradient expressions for each face in the domain provides the opportunity to choose the optimum stencil of cells local to the face.

4.6.1 Boundary Conditions

In common CFD applications, boundary values which are also in the form of average quantities, are available for use in the reconstruction step. For certain boundary condition types, the domain which the cell occupies varies. Two alternatives include a fictitious region outside the boundary of the mesh referred to as ghost cells and the surface defined by the boundary domain referred to as a face cell. The properties for physical boundaries such as a wall, are represented as face cells, as they are valid on the boundaries of the mesh. Axisymmetric boundary conditions assume the presence of a cell which is not directly discretized in the domain, but rather is a mirror image of the immediate cell above the surface. The key to including these cells in the reconstruction step is to identify the control volume with which to satisfy the mean conservation criteria. For all results presented, the face cell option applies. In two dimensions, an equation is derived that guarantees conservation of the mean for line averaged boundary values.

The global accuracy of the solution algorithms is not only determined by the accuracy

of the interior scheme, but also the boundary conditions as well. As will be demonstrated in a later example, if the boundary conditions are not modeled consistently (accurately) with the interior domain, the truncation errors introduced into the solution lead to less accurate results. For the purpose of validating the solution algorithm, the boundary condition issues are removed by imposing the exact conditions in an averaged sense.

4.6.1 Elliptic Boundary Value Problems

The numerical experiment selected to validate the method for elliptic boundary valued problems is a solution to the heat equation. The integral form of the governing equation is written as:

$$\frac{\partial}{\partial t}\iiint_{\Omega} Q \, d\Omega + \iint_{S} \vec{F} \cdot \hat{n} \, dS = 0.$$

with

$$Q = \rho C_p T$$

$$\vec{F} = f_i + g_j$$

$$f = -\kappa \frac{\partial T}{\partial x} \qquad g = -\kappa \frac{\partial T}{\partial y}$$

where κ is the thermal conductivity.

The solution domain is defined by

$$0 \leq x \leq 1 \qquad 0 \leq y \leq 1$$

with boundary conditions

$$T(x,0) = 0.0$$

$$T(x,1) = 0.0$$

$$T(0,y) = 0.0$$

$$T(1,y) = sin\pi y$$

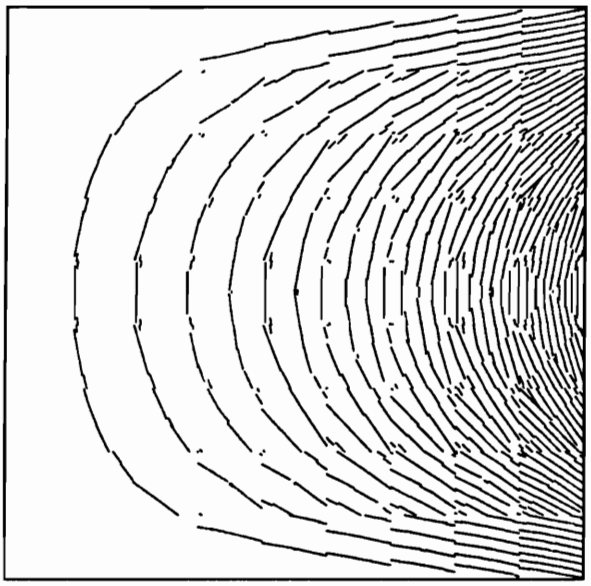
An exact solution to the problem is the hyperbolic test function used in the previous section and provided here for further reference.

$$T(x,y) = \frac{sin(\pi y) \times sinh(\pi x)}{sinh(\pi)}$$

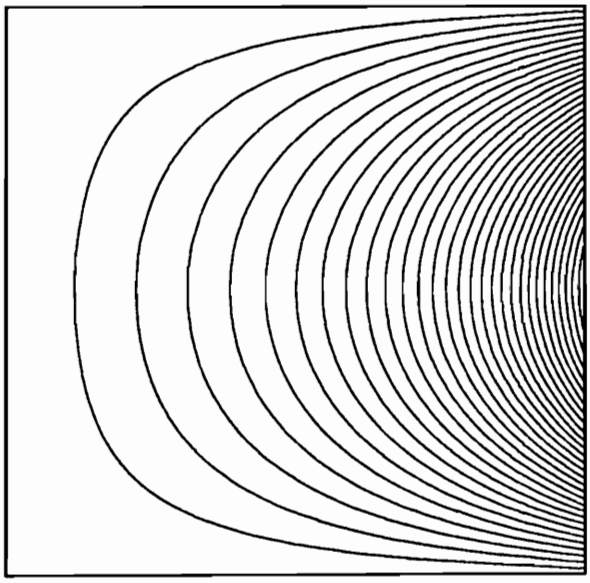
All solutions were integrated in time using a simple Euler explicit scheme. Steady state solutions were based on reducing the normalized L_2 norm of the residual 5 orders of magnitude or until it remains unchanged (worst case 2.8 orders). The three triangular meshes used in Sect. 4.5 are sufficient for this investigation. The results are provided in both graphical and tabular form for $k = 1, 2$, and 3. Temperature contours are shown in Figure 4.13 for grid.2 (medium grid) using the cell by cell contouring approach for visual clarity. The differences between the $k = 2$ and $k = 3$ results cannot be seen in the contours but are reflected in the error analysis presented in Figure 4.14 and Table 4.2. The slopes of the lines in Figure 4.14 indicate the correct trends associated with h/p refinement.

The results of the flux integration study reveals improvement in solution predictions only for the case of $k = 3$. Barth [13] claimed that $\frac{(k+1)}{2}$ Gauss points are necessary to integrate k degree reconstructions without error. However, his findings were based on solutions to the Euler equations. By focusing on the functional form of the integrand in the flux integral, we notice that gradient expressions are integrated as compared to the complete reconstruction polynomial. Consequently, a quadratic ($k = 2$) polynomial reduces to a linear behavior in gradient form, thus the quadrature rules need be only adequate to exactly integrate a linear function. Therefore, the identical results between $N = 1, 2$ and 3 point quadrature is justified for the $k = 1$ and $k = 2$ cases. However, for $k = 3$ the polynomial derivative is quadratic in form requiring only 2 Gauss points to solve the integral exactly. While the differences in Figure 4.14 are visually minor, the numerical results in part a of Table 4.2 reveal a 15% improvement in solution error. The

discrete accuracy summary in Table 4.2 upholds the $0(k+1)$ order accurate results for k degree reconstruction.

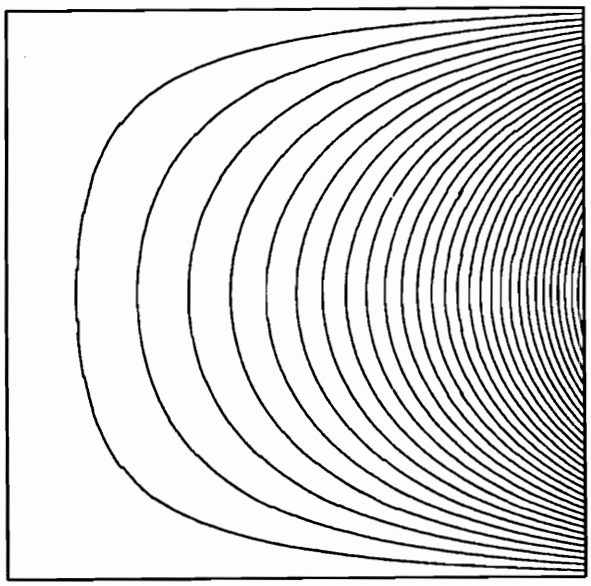


a – $K = 1$ **reconstruction**



b – $K = 2$ **reconstruction**

Fig. 4.13 – Cell-wise contours for heat equation results



c – $K = 3$ reconstruction

Fig. 4.13 – Cell-wise contours for heat equation results (Continued).

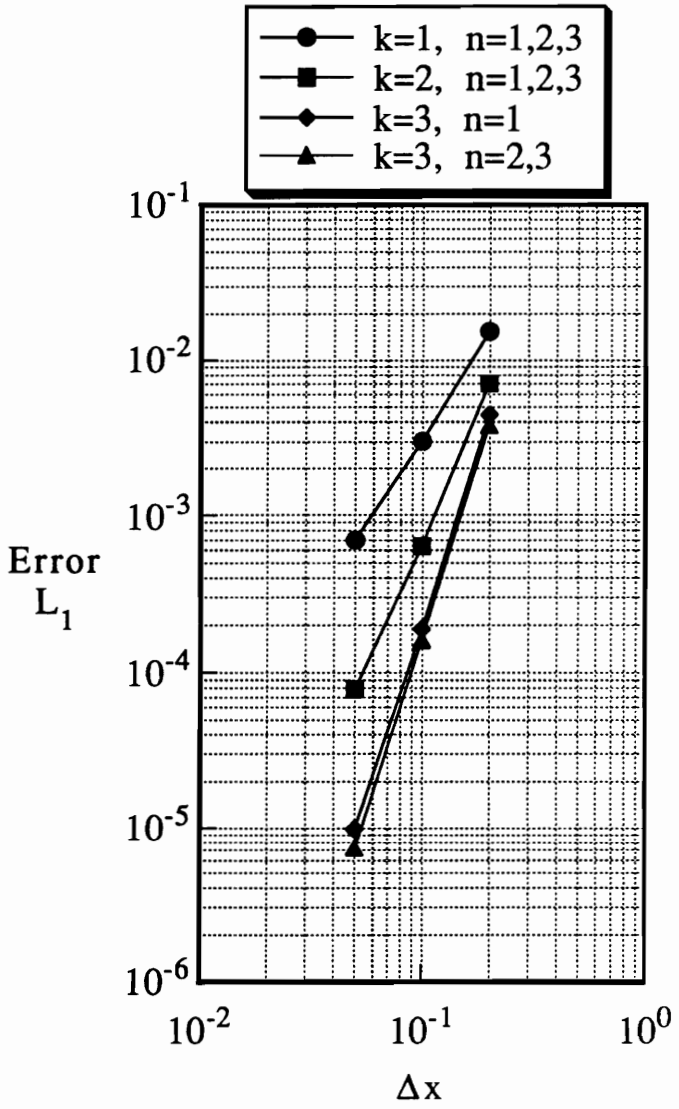


Fig. 4.14 – Solution error summary for heat equation

Table 4.2 – Solution accuracy and error norms for heat equation results using N point Gaussian quadrature.

Grid Number	Degree of Reconstruction Polynomial K	Number of Gauss Points in Flux Integration N	L1 Error Norm
1	1	1,2,3	1.53 e-02
1	2	1,2,3	7.08 e-03
1	3	1	4.42 e-03
1	3	2,3	3.78 e-03
2	1	1,2,3	3.00 e-03
2	2	1,2,3	6.38 e-04
2	3	1	1.89 e-04
2	3	2,3	1.61 e-04
3	1	1,2,3	6.96 e-04
3	2	1,2,3	7.84 e-05
3	3	1	9.84 e-06
3	3	2,3	7.21 e-06

a – Solution error norms

Degree of Reconstruction Polynomial	Number of Gauss Points in Flux Integration	Accuracy Order
k = 1	N = 1,2,3	1.984
k = 2	N = 1,2,3	2.891
k = 3	N = 1	3.920
k = 3	N = 2,3	4.021

b – Solution accuracy

4.6.2 Ringleb's Flow

Ringleb's flow presented by Chiocchia [36] is a shock-free transonic hodograph solution for two dimensional planar potential flow. The flow quantities are non-dimensionalized according to:

$$\bar{\rho} \equiv \frac{\rho}{\rho_0}$$

$$\bar{a} \equiv \frac{a}{a_0}$$

$$\bar{q} \equiv \frac{q}{a_0}$$

where ρ is the density, a is the speed of sound, and q is the velocity magnitude. The streamlines in the hodograph plane (q, θ) are given by

$$\psi = \frac{1}{\bar{q}} sin\theta$$

where θ is the local flow angle. The flow on the physical plane (x, y) corresponding to the streamlines given above is as follows:

$$x = \frac{1}{2}\frac{1}{\bar{\rho}}(\frac{1}{\bar{q}^2} - \frac{2}{k^2} + \frac{J}{2})$$

$$y = \pm \frac{1}{k\bar{\rho}\bar{q}}\sqrt{1 - (\frac{\bar{q}}{k})^2}$$

where

$$k = \frac{1}{\psi} \qquad \textit{constant on every streamline}$$

$$J = \frac{1}{\bar{c}} + \frac{1}{3\bar{c}^3} + \frac{1}{5\bar{c}^5} - \frac{1}{2} lg \frac{1+\bar{c}}{1-\bar{c}}$$

$$\bar{c} = \sqrt{1 - \frac{\gamma - 1}{2}\bar{q}^2}$$

$$\bar{\rho} = \bar{c}^{\frac{2}{\gamma - 1}}$$

where $\bar{c}$ and $\bar{\rho}$ are the stagnation speed of sound and density respectively. The constant velocity lines are circles defined as

$$(x - \frac{J}{2})^2 + y^2 = \frac{1}{4\bar{\rho}^2\bar{q}^2}$$

For this investigation, three grid levels were used to assess the numerical accuracy of the schemes. Two reconstructions were performed for each face in the domain corresponding to the right and left states, respectively. The boundary streamlines shown in Figure 4.15 vary from 1.47 yielding transonic flow conditions on the left wall to 0.8, a subsonic state on the right wall. The boundary conditions for this study include the exact conditions specified over the complete domain. Roe's approximate Riemann solver was used with a split flux condition at the inflow and outflow and full flux condition on the two walls. The meshes and density solutions are shown in Figure 4.15. The improvement in solution quality with increased mesh size (h refinement) is apparent. Cell-wise density contours for linear and quadratic reconstruction are shown in Figure 4.16. The smoothness in contour transition from cell to cell exemplifies the idea of p refinement. The upwind stencils for the $k = 1$ solution and the upwind-biased stencils for $k = 2$ are shown in Figure 4.17. The bold dot indicates the location of the parent face. Notice that the upwind-biased stencil includes the contributions from one downwind cell.

The results of the accuracy calculations are shown in Figure 4.18. The improvement with increased k and decreased h is clear. The behavior of the error levels indicate the first, second, and third order accuracies for $k = 0, 1$ and 2, respectively. The discrete error estimates are provided in Table 4.3 for various accuracies of flux integration. The greatest improvement is seen in the $k = 2$ solution using two point flux integration, and it amounts to an average 34% reduction in error levels. For a $k = 0$ reconstruction, the solution was independent of the flux integration rule, and little benefit was realized for the $k = 1$ case. For the cases presented here, the effect of the increased accuracy for the flux integration step is not linked to the global accuracy of the steady state solutions, i.e. the slope of the log-log error plot is upheld independent of the number of Gauss points. However, the solution errors are observed to decrease for several of these conditions, and they are, therefore, considered to be a better solution. These results are unique as they

are the first to formally demonstrate the accuracy of steady state solutions on triangular meshes using numerical techniques.

The importance of accurate boundary condition treatment is demonstrated by numerically evaluating the streamline boundaries using common CFD methods. A first order tangency condition is applied along the right and left boundaries using the following two alternatives for modeling the pressure.

$$\frac{\partial p}{\partial n} = 0$$

$$\frac{\partial p}{\partial n} = \frac{\rho Q^2}{R}$$

where R is the exact local radius of curvature. The results of this study are shown in Figure 4.19. For a $k = 1$ reconstruction and using first order boundary conditions, the realized global accuracy decreased from 2.07 to 1.7, and the error levels increased an alarming 600%. The solution errors associated with the coarse mesh using exact boundary conditions are lower than the first order boundary condition solutions on the medium mesh. The accuracy (slope) for the first order boundary condition treatment is not affected using the radius of curvature correction in the pressure gradient, however the errors are slightly less, indicating a physically more correct boundary treatment. While it has been commonly asserted that the global accuracy of the numerical solution is limited by the lowest level approximations, quantified verification of this effect has not been demonstrated in multiple dimensions until this work. As a result, the accuracy of the boundary conditions must be consistent with the interior domain to achieve maximum solution accuracy. For an improved modeling of the tangency boundary condition, the curvature-corrected symmetry technique of Dadone and Grossman [54] could be implemented.

Finally, a comparison of three linear reconstruction procedures is shown in Figure 4.20 including $k - exact$ reconstruction and the two Type 1 gradient based methods presented in Chapter 3. For the case of averaging the properties to the nodes, Frink's approach was

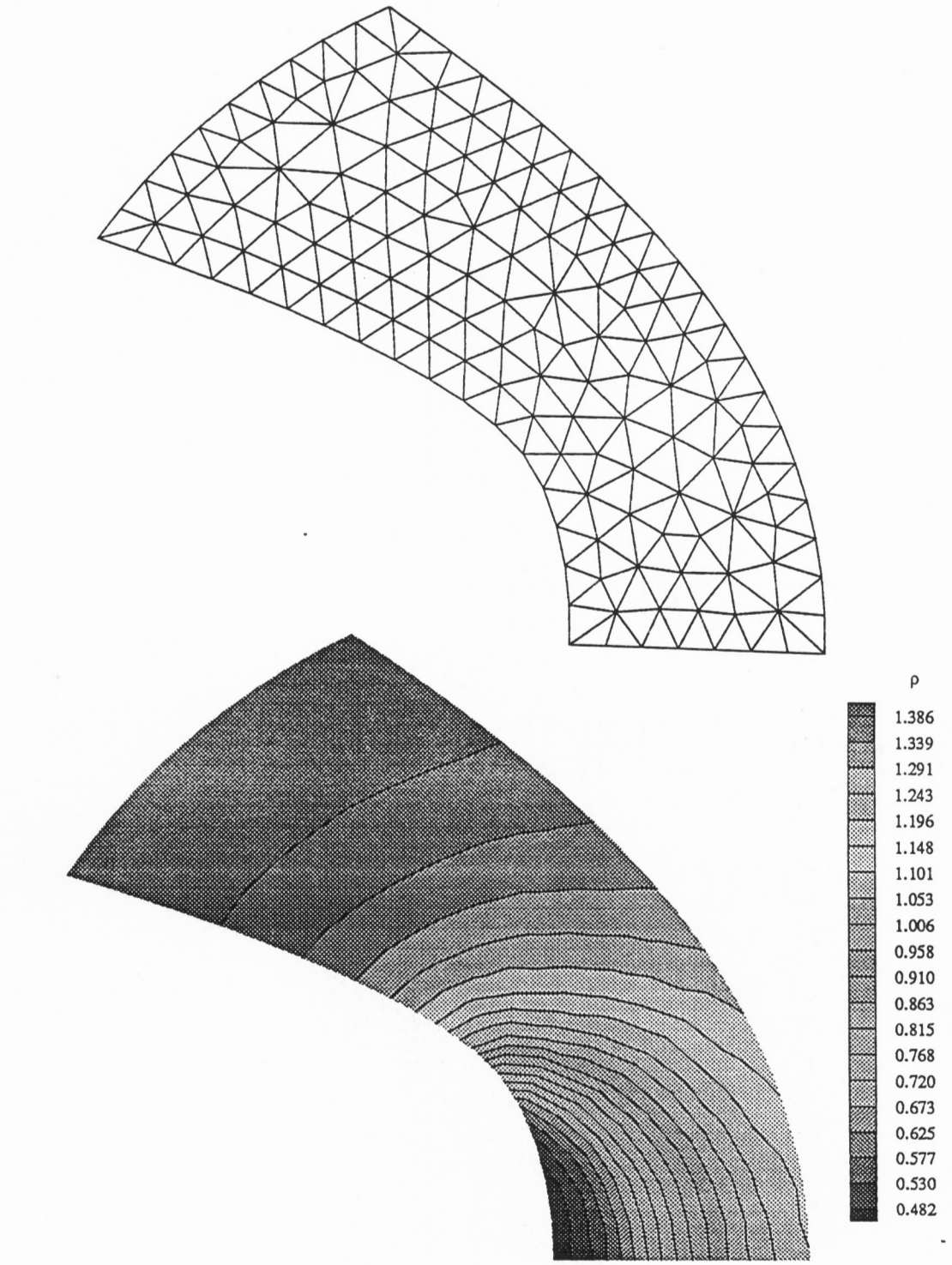


Fig. 4.15 – Coarse grid and density contours for Ringleb's flow $k = 1$.

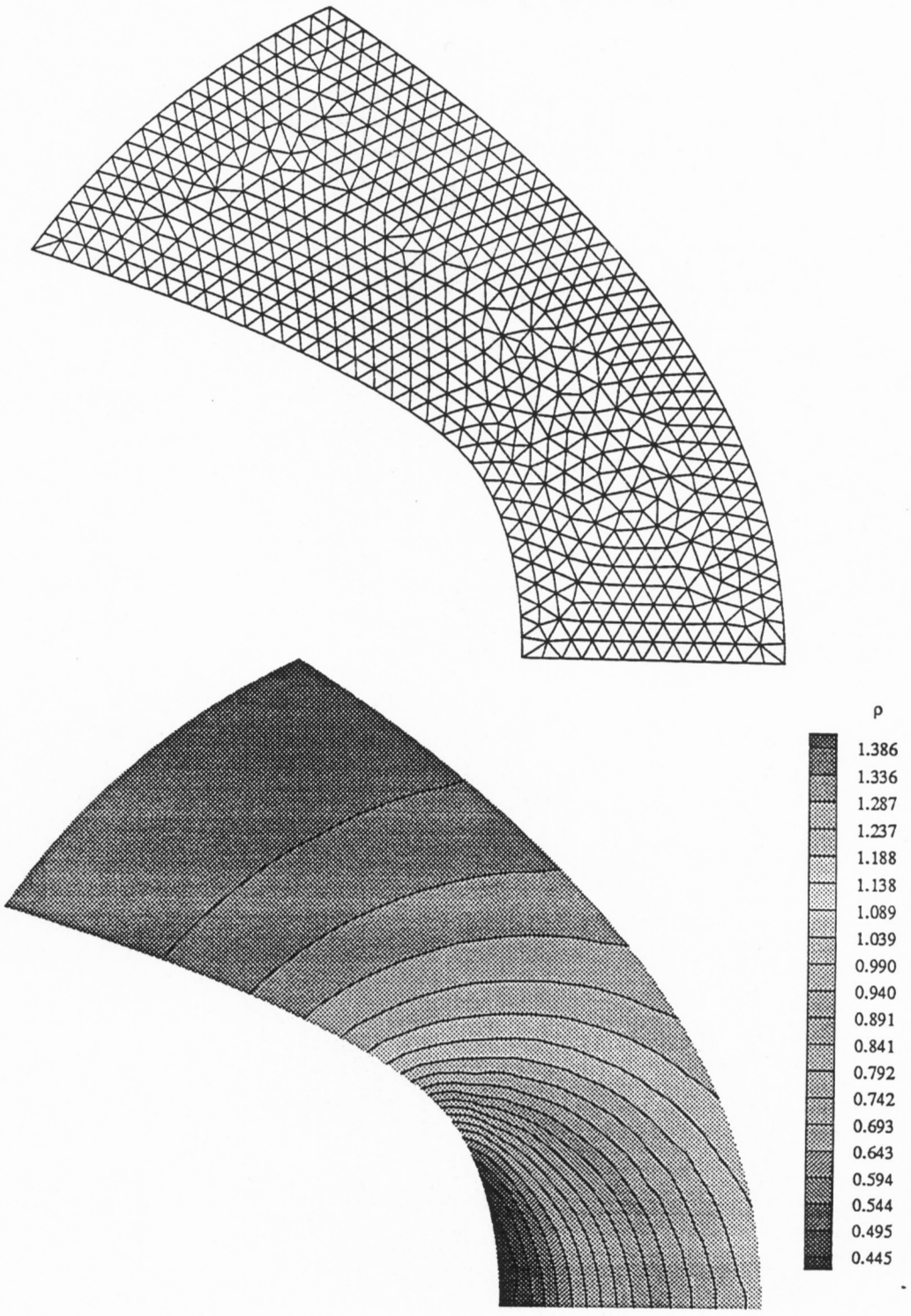


Fig. 4.15 – Medium grid and density contours for Ringleb's flow $k = 1$ (Continued).

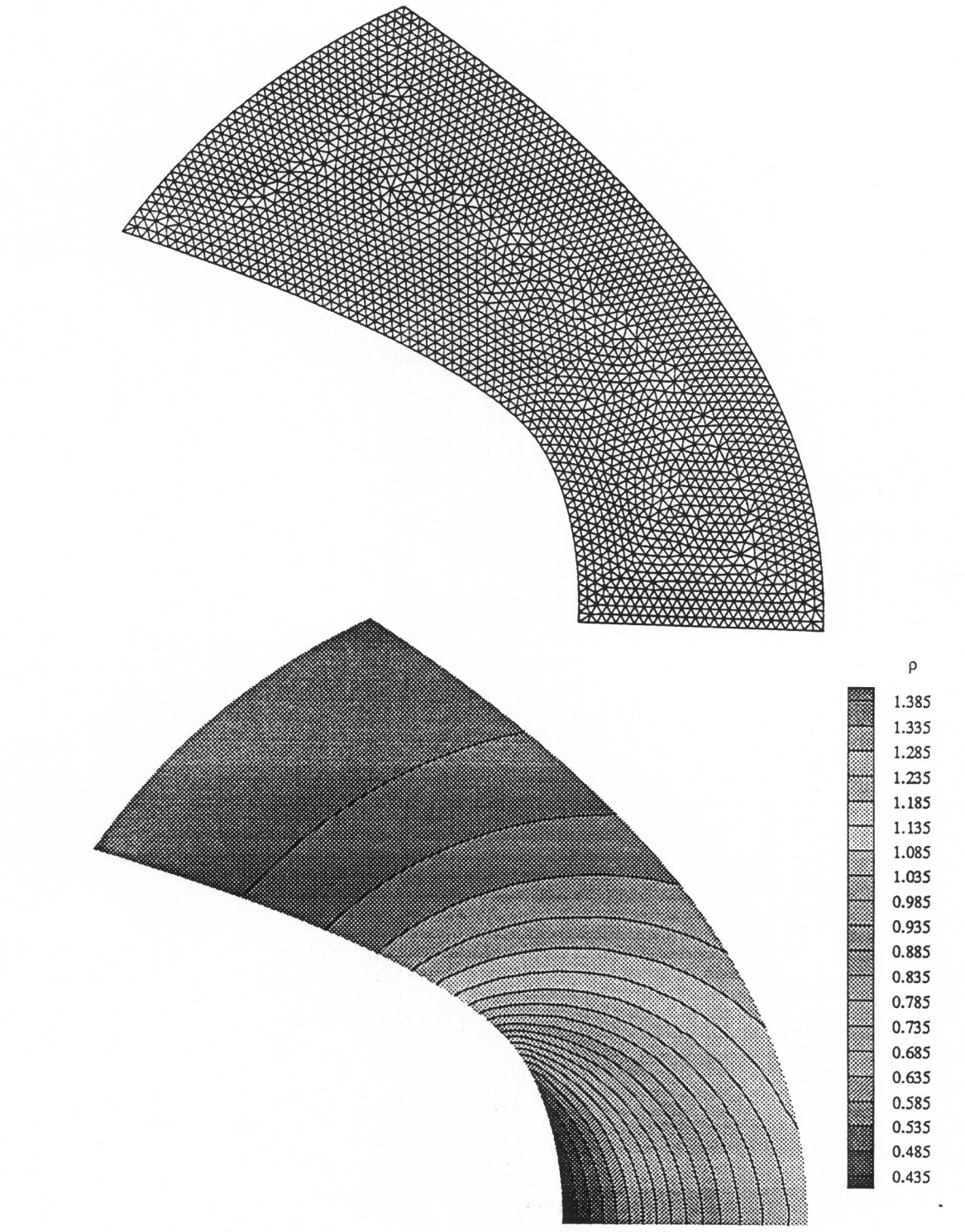
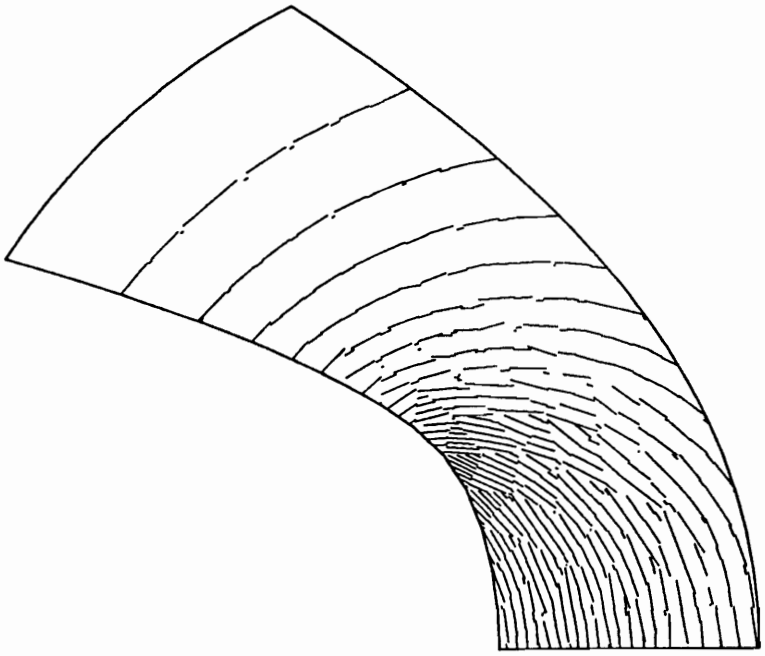
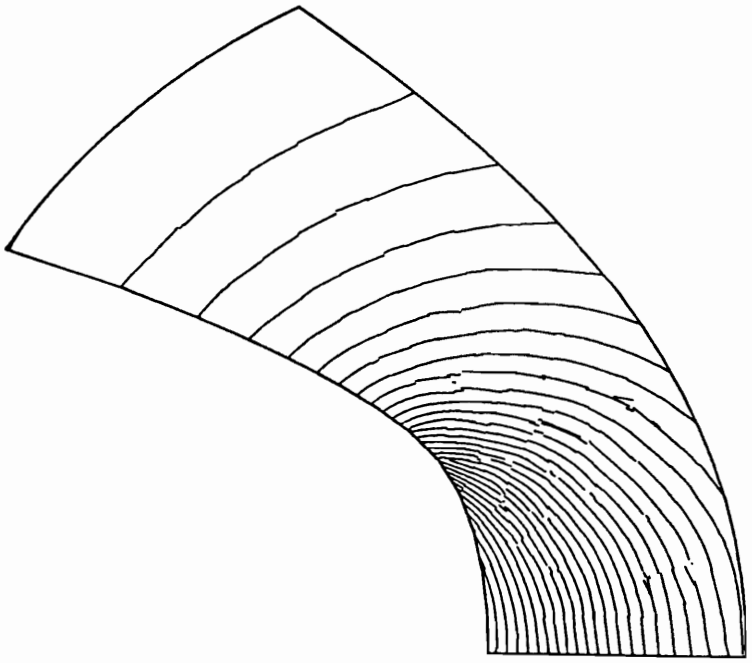


Fig. 4.15 – Fine grid and density contours for Ringleb's flow $k = 1$. (Continued).

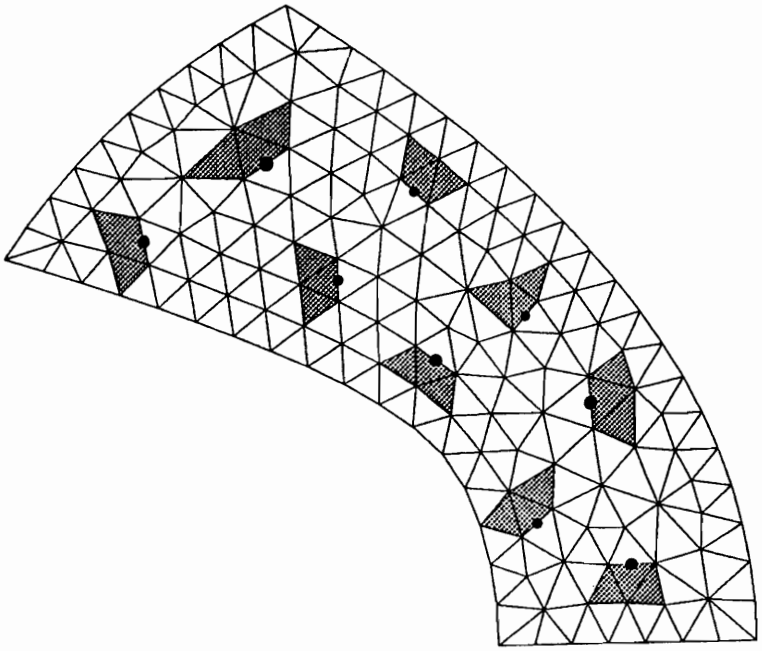


a – $K = 1$ reconstruction

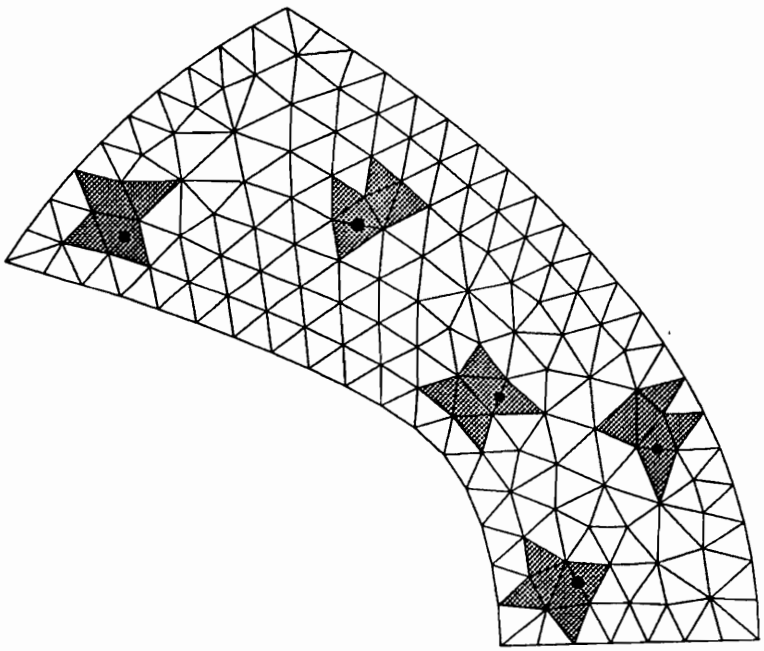


b – $K = 2$ reconstruction

Fig. 4.16 – Cell-wise contours for Ringleb's Flow



a – Stencil for $k = 1_{upwind}$



b – Stencil for $k = 2_{upwind-biased}$

Fig. 4.17 – Upwind and upwind-biased stencils for Ringleb's flow

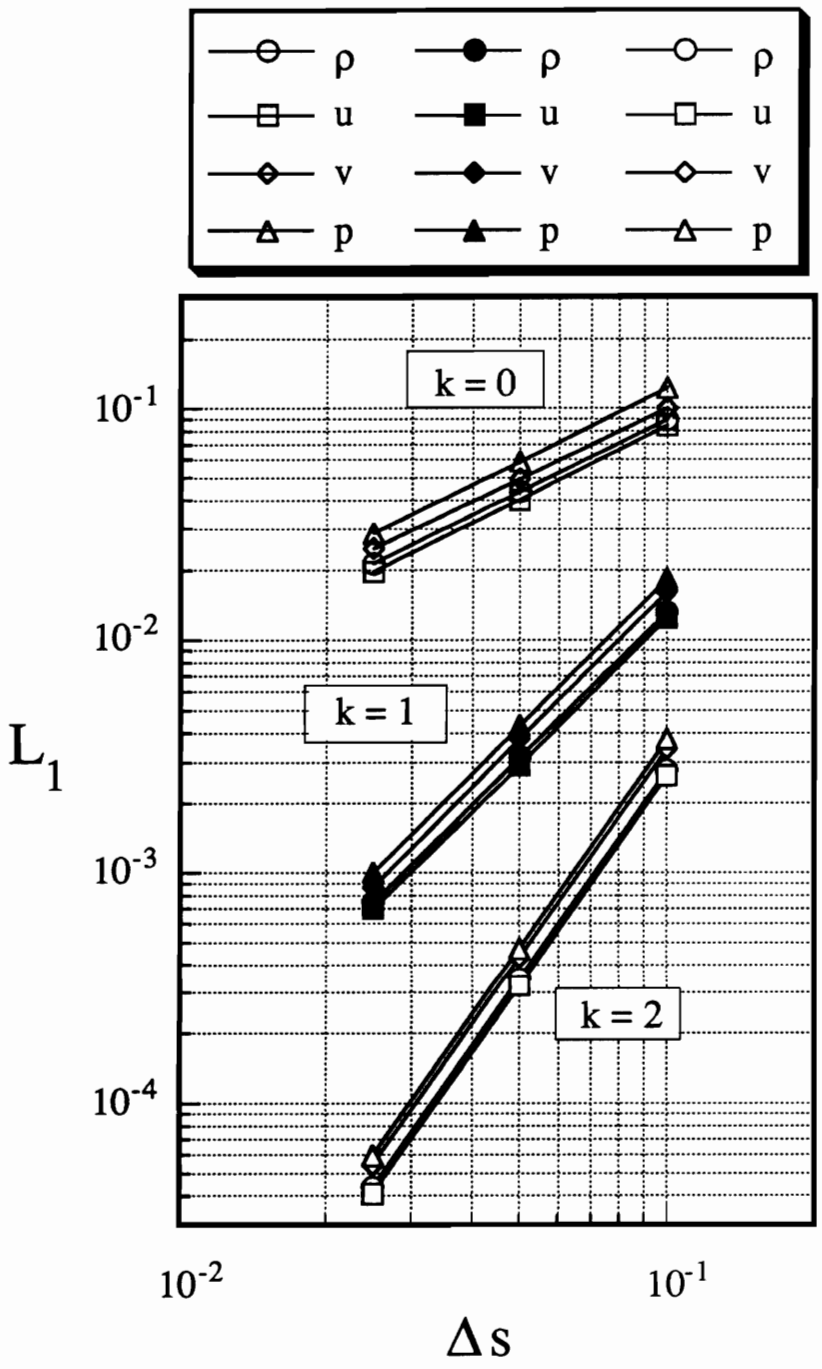


Fig. 4.18 – Solution error summary for Ringleb's flow

Table 4.3 – Solution accuracy and error norms for Ringleb's flow using N point Gaussian quadrature.

Grid Number	Degree of Reconstruction Polynomial K	Number of Gauss Points in Flux Integration N	L1 Error Norm for Density
1	0	1,2,3	8.89 e-02
1	1	1	1.31 e-02
1	1	2	1.26 e-02
1	1	3	1.27 e-02
1	2	1	4.13 e-03
1	2	2	2.76 e-03
1	2	3	2.75 e-03
2	0	1,2,3	4.32 e-02
2	1	1	3.14 e-03
2	1	2	3.11 e-03
2	1	3	3.10 e-03
2	2	1	5.23 e-04
2	2	2	3.45 e-04
2	2	3	3.46 e-04
3	0	1,2,3	2.12 e-02
3	1	1	7.42 e-04
3	1	2	7.40 e-04
3	1	3	7.41 e-04
3	2	1	6.91 e-05
3	2	2	4.36 e-05
3	2	3	4.35 e-05

a – Solution error norms

Degree of Reconstruction Polynomial	Number of Gauss Points in Flux Integration	Density Accuracy Order
k = 0	N = 1,2,3	1.034
k = 1	N = 1	2.071
k = 1	N = 2	2.045
k = 1	N = 3	2.049
k = 2	N = 1	2.949
k = 2	N = 2	2.991
k = 2	N = 3	2.993

b – Solution accuracy

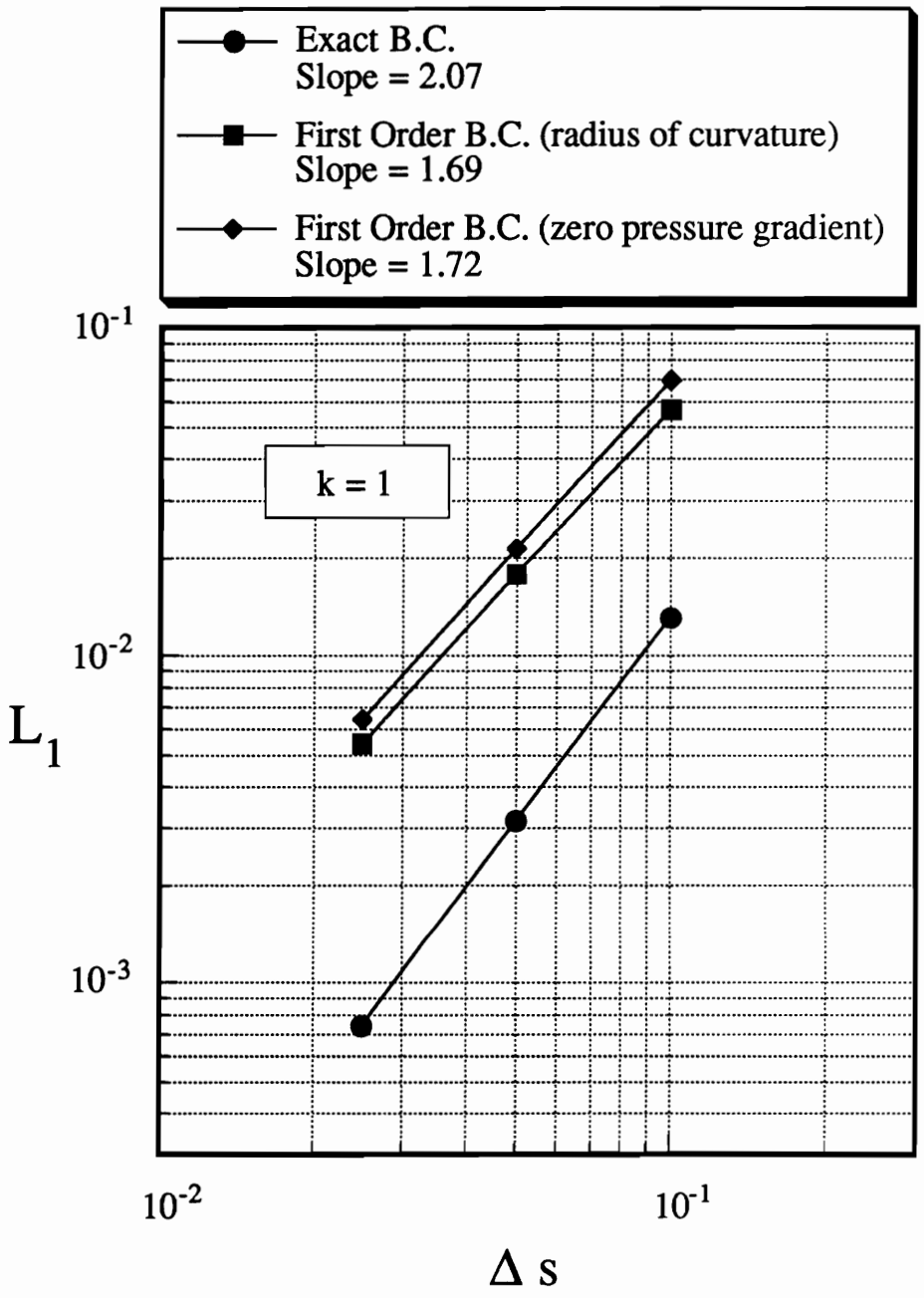


Fig. 4.19 – Comparison of boundary conditions for Ringleb's flow

chosen for the investigation. The spatial accuracy for the Type 1 methods were assessed by reconstructing their steady state solutions using a $k = 1$ reconstruction and computing the error norm. Using this approach, the error analysis procedure was consistent for all methods. The results indicate that the only formally second order accurate scheme is $k - exact$ reconstruction. The effects of averaging the properties was to reduce the accuracy for the Type 1 schemes. The discrete accuracy levels are 2.07, 1.85 and 1.56 for $k - exact$, node averaged, and face averaged methods, respectively. Although the accuracy characteristics appear to be similar for all three linear reconstruction techniques, the error levels indicate the relative difference between the methods. For example, the Type $1 - B$ medium mesh results indicate a 117% increase in solution error as compared to $k - exact$ reconstruction. Due to the accuracy reduction, the error differences grow radically during mesh refinement.

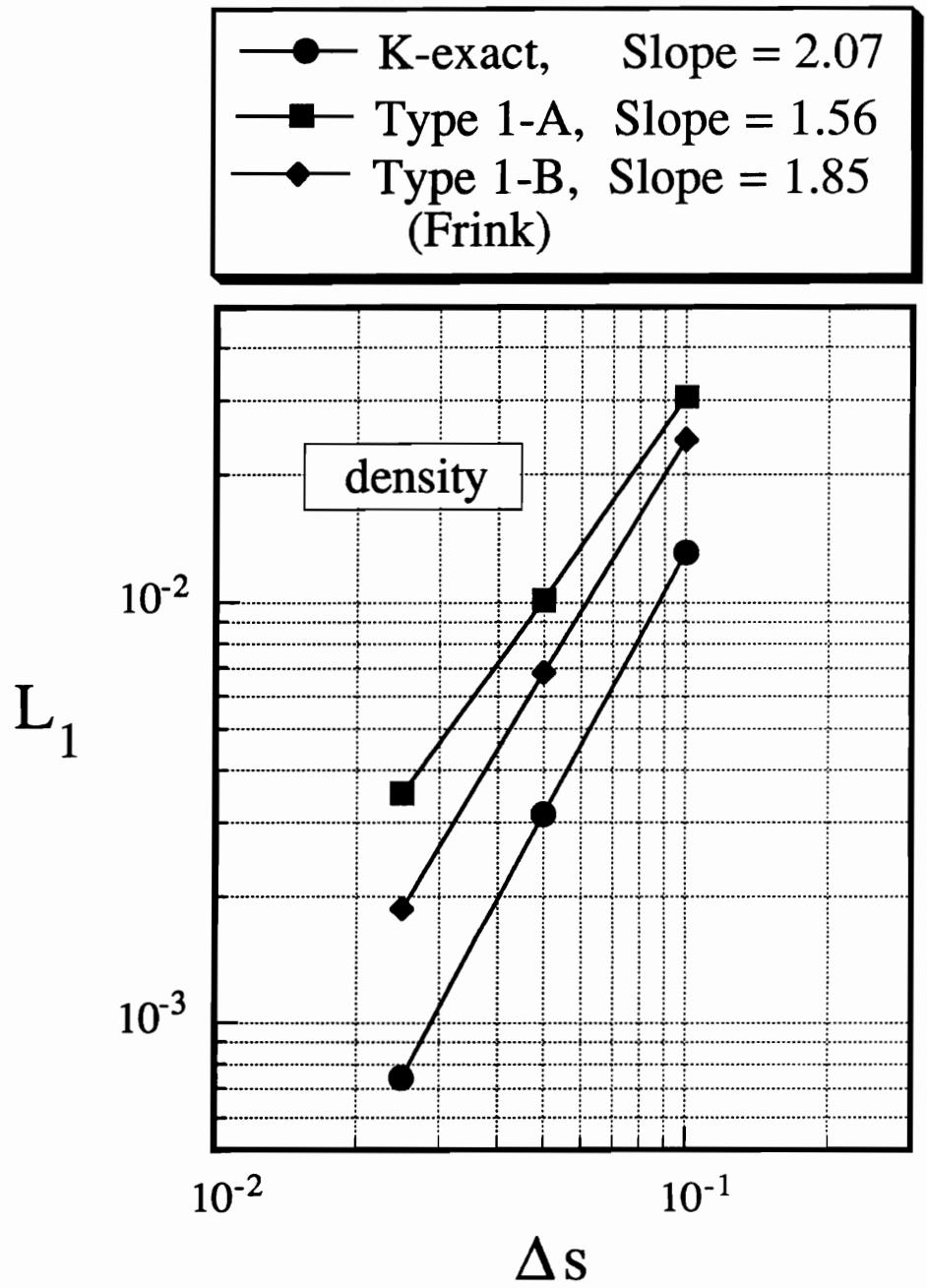


Fig. 4.20 – Comparison of linear reconstruction methods

Chapter 5

Advancing Front Grid Generation

5.1 Overview

Unstructured computational fluid dynamics techniques have been implemented with success for the past several years. Two dimensional triangular meshes and three dimensional tetrahedral meshes have arisen as a popular alternative to the traditional structured meshes. There are geometrical advantages to an unstructured grid approach which reveals the weaknesses of structured grids. Specifically, the flexibility of nodal placement and element size variations common with unstructured technologies results in more effective discretizations for arbitrary complex bodies. In addition, this approach is ideal for adaptation procedures which involve sophisticated strategies to redistribute the cells within the entire domain. In recent years, this level of control has been shown to improve solution quality for various flows.

Unstructured grids are generated in three different ways; subdividing structured grids into triangular or tetrahedral control volumes, Voronoi/Delaunay triangularization, and the advancing front technique. The first method of simply converting structured grids does not expose a discretization advantage, however this approach has been used to validate solution algorithms by comparing with structured solutions on meshes with similar characteristics [32] . The second approach, Voronoi/Delaunay triangularization [37] [40] , requires a nodal distribution to generate the mesh. Given the exact nodal placement, these algorithms connect the points to form the cells. While this approach has been implemented with success, it lacks the geometric flexibility to create the nodes

within the computational domain. Advancing front methods [10] [11] [38] [39] create the point distribution during the actual mesh generation process. An outline of the basic algorithm is highlighted in the next section following the work of McGrory [41] .

5.1.1 Advancing Front Algorithm

A two dimensional advancing front process begins by creating an initial front using prescribed boundary segments and grid spacing parameters provided by a background grid. The initial front must form a closed path with a connectivity strategy that guarantees that the surface normals point into the computational domain. This front is marched towards the interior by generating triangles and updating the front to reflect the changes. The triangle size and orientation are interpolated from the background grid. The ideal third point is computed and compared with existing nodes using the following criteria:

1. Gather all existing nodes within a prescribed radius from the ideal point.
2. Determine which points generate a usable triangle, i.e. one which does not jeopardize the integrity of the grid. Basic considerations include points that lie on the correct side of the front and points that create triangles that do not cross existing edges of the current front.
3. Select the optimum point using an ordering criteria. Options include closest node to the ideal point, the point that generates a triangle that closely matches the ideal element in unstretched local coordinates, and the Delaunay triangulation criteria.

If the above procedure does not produce a usable existing node, the ideal point is subjected to the tests in item 2. However, if the ideal point also fails the prescribed criteria, the search radius is increased, and the process is repeated until an acceptable point is found. As nodes are added, the front is updated by removing faces which are not part of the current front. In order to maintain the grid characteristics provided by the background grid, the smallest elements on the current front are generated first. This

scheme is repeated until all of the edges are removed, thus discretizing the entire domain.

The three dimensional advancing front algorithm generates an initial front consisting of a closed surface of triangles. This surface mesh is generated in three dimensions using the same principles defined above. The added complexity of the multidimensional surface meshes requires modifications to the skeleton approach outlined in two dimensions. Nevertheless, the basic use of background grid information to determine cell size and orientation and the algorithms for accepting points to the domain still applies. Special care must be taken to ensure that the surface mesh of triangles is closed and that the surface normals point into the computational domain before generating the interior mesh. The interior tetrahedra mesh is discretized by adding a new or existing node to a triangular face on the current front. Triangular faces are removed from the front during the process until the domain is completely meshed.

5.1.2 The Background Grid

Grid characteristics including cell size and orientation are provided through a background grid that totally encompasses the region to be discretized. The quality of the resulting meshes is strongly related to the smooth distribution of these grid parameters. In addition to sacrificing grid generation robustness, the solution accuracy is adversely affected by the lack of grid smoothness. For these reasons, the creation and subsequent modifications of the background grid is considered to be an inherent weakness for the advancing front technique. However, recent progress has been achieved in improving the accuracy, flexibility and efficiency of background grids used for Euler calculations [10] [11] . The extension of this method to provide a distribution necessary to compute viscous flows has not been documented in three dimensions and is rarely discussed or demonstrated in two dimensions.

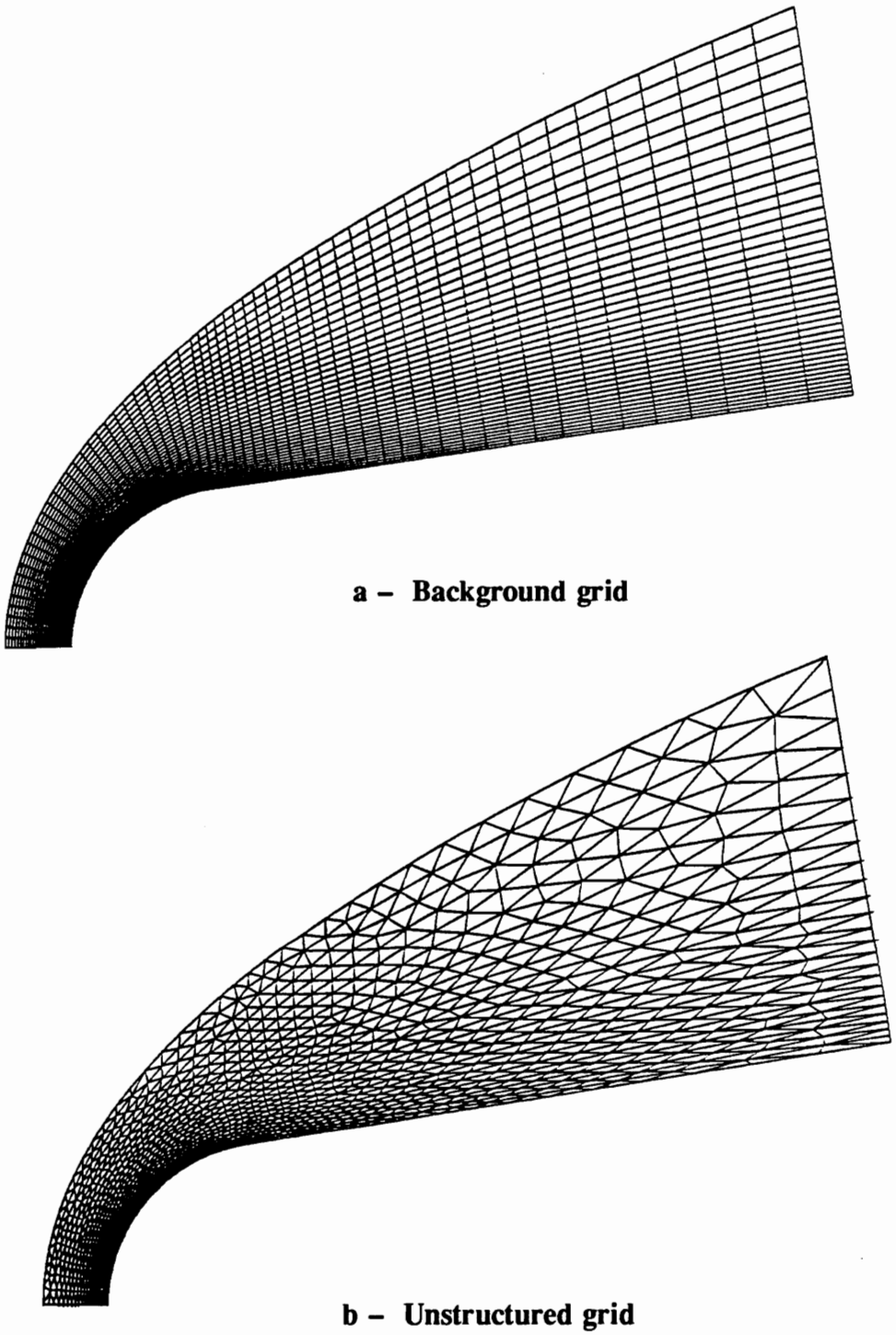
The background grid stores the distribution parameters at the nodes. During the

mesh generation procedure, the parameters are interpolated from the background grid to the local point of interest. Typically, these grids are unstructured and often require a considerable amount of user effort to construct and assign grid spacing parameters.

As was mentioned previously, viscous applications featuring grids which were generated using the advancing front technique are rare. This is primarily due to a lack of grid quality when generating highly distorted cells. The variation in cell spacing, stretching ratio, and stretching direction necessary to produce quality viscous grids requires a very smooth distribution that is difficult to produce manually. As a result of the poorly conditioned background grid parameters, the computational domain is likely to cause numerical instabilities and inaccuracies. A method implemented by McGrory [42] in two dimensions uses the smooth qualities of structured grids to compute smooth background grid parameters. The resulting mesh distribution is similar to the structured grid. The approach is as follows. First, a structured grid is created with the appropriate grid stretching and cell size. Then, through geometrical considerations, the unstructured grid parameters are computed at the nodes of the structured grid. Finally, the structured grid is divided into triangular elements and used as the background grid. Although this technique requires the work of creating a structured grid, it is a very effective alternative, and it provides a great deal of control when generating smooth, highly stretched triangular grids.

For example, Figure 5.1 shows a structured grid which was subdivided into triangles and the resulting unstructured grid for a blunt body using a two dimensional advancing front grid generator. The cell size, stretching ratio, and stretching direction were computed as a function of the structured grid geometry. As can be seen in the figure, the distributions are very similar. The grid integrity is maintained as the variation in cell characteristics throughout the domain is smooth.

In three dimensions, the grid parameters include the cell size, two stretching direc-



a – Background grid

b – Unstructured grid

Fig. 5.1 – Background grid and unstructured grid for blunt body

tions and stretching ratios. The size of the tetrahedra is determined by the cell size parameter and corresponds to the radius of a sphere which circumscribes an equilateral tetrahedron in transformed coordinates. The transformed space removes all stretching variations via simple mapping procedures. After locating the points in transformed space that produces an equilateral tetrahedral element, the true physical coordinates are obtained by a reverse mapping procedure which includes the effects of two stretching directions and their corresponding stretching ratios.

5.2 NUKE, A Three Dimensional Grid Generator

Three dimensional unstructured meshes are generated using an improved version of an Advancing Front grid generator developed by McGrory [41] . Code named NUKE, this software was originally written to run on a Silicon Graphics Iris Workstation using a sophisticated graphical interface, however a user-specified non-graphical mode is also available for Cray or Convex machines. Written entirely in the "C" programming language, this single program allows the user to visually critique the entire grid generation process.

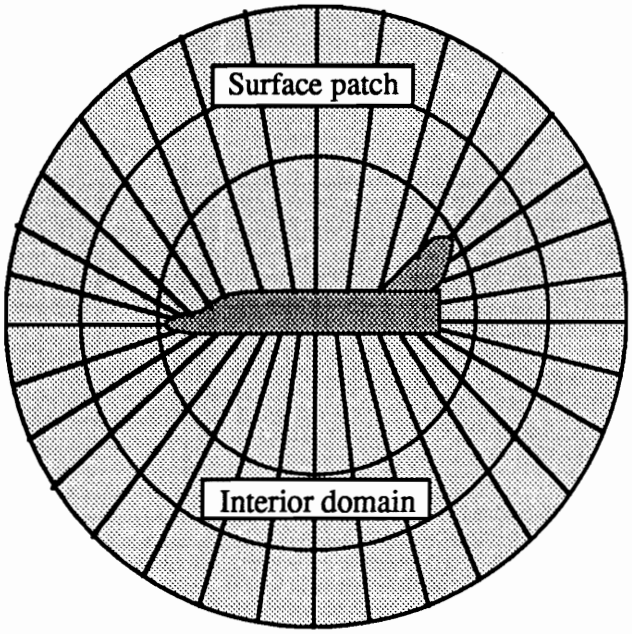
The enhancements to the grid generation capabilities are in the areas of surface description and surface discretizations. Before this research effort, three dimensional surface grids for bodies with curvature was not possible with this software. A discussion of the relevant features follows.

5.2.1 Surface Description

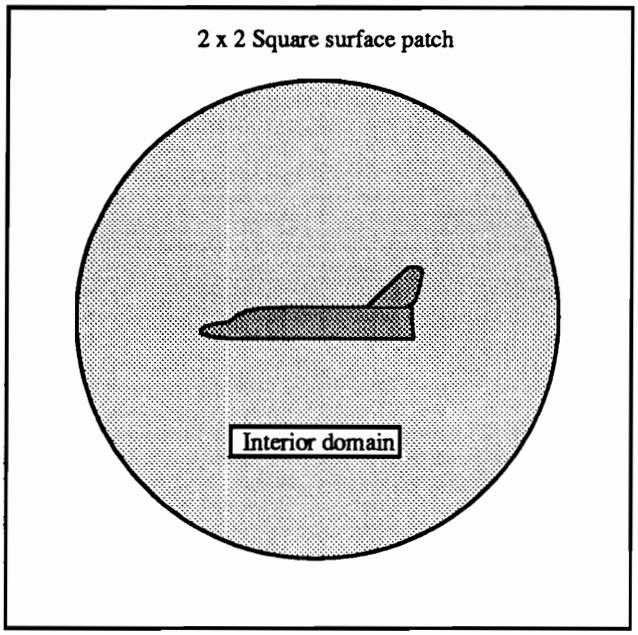
The algorithm requires three pieces of information to generate an unstructured grid around a body: lines, corner points, and surfaces. The line information describes the physical boundaries of the domain used when generating the initial fronts. The corner points are helpful to avoid creating redundant end points where multiple lines coincide. The boundary descriptions, including surface and far field, are provided in the form

of structured oriented patches. A patch is the simplest mathematical element to model a surface. These patches only define the surface definition and are not necessarily discretized in their entirety. The portion of the patch which is used in generating the surface grid encompasses any number of line segments which combine to form a closed curve. For example, a square planar patch could be used to define the surface of an arbitrary planar body by simply defining a closed line segment (in the shape of the body), which is physically within the bounds of the square patch. This greatly simplifies the structured patch surface description approach. Consider Figure 5.2 which shows the symmetry plane for a space shuttle configuration. The region to be meshed consists of the area between the closed outline of the body and the closed circular outer boundary. The surface which describes the physical description for the computational domain is a planar structured patch. Two options exist for defining the interior surface. The first is provided in part *a* in the same figure. As can be seen, the surface is constructed using a common structured grid. While this two dimensional planar grid can be created using existing structured grid generation tools, the simple alternative presented in part *b* is more appealing, requires less work, and provides the necessary surface details to generate the unstructured mesh.

Figures 5.3 and 5.4 shows two examples of three dimensional patch descriptions including a Glasair fuselage and a High Speed Civil Transport (HSCT), respectively. Notice that both configurations are composed of several structured patches which form the complete shape. The dimensions of the patches are independent from one another allowing non-contiguous point distributions between patch interfaces. The only restriction currently imposed relating to surface patches is that the closed loops defining the region to be meshed must not cross a patch boundary. While this is currently a limitation, provisions can be included to allow loops containing multiple patches. Consequently, the fuselage surface and the HSCT will be discretized in multiple portions each of which



a – Complicated structured patch



b – Simple structured patch

Fig. 5.2 – Patch descriptions for symmetry plane of space shuttle configuration

corresponds to the local patch. Also included in the figures are the unstructured surface meshes. The ability to generate smooth, high quality surface meshes is essential before proceeding with the interior grid generation.

5.2.2 Surface Discretization

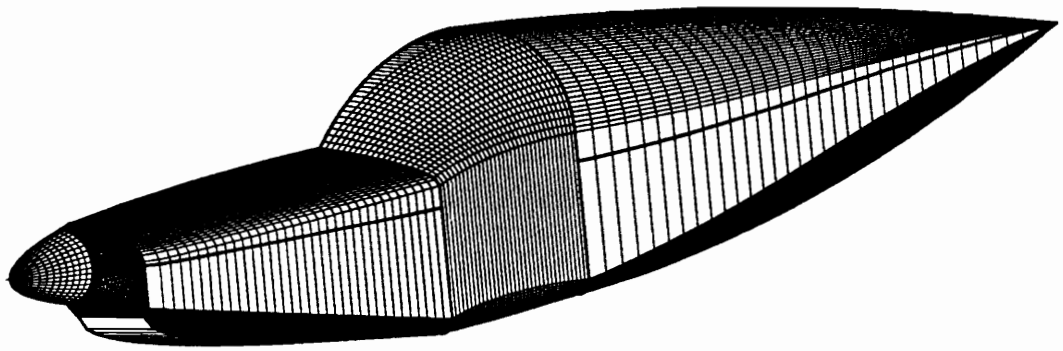
The surface grids are generated using an algorithm which advances an initial front by moving along the physical three dimensional surfaces. An alternative approach would involve mapping (projecting) the three dimensional domain to a two dimensional computational space, generating a mesh using the advancing front method in two dimensions and reverse mapping the mesh back into the three dimensional space. Although the grid generation procedure in two dimensions is more refined and robust, selecting the mapped space is often difficult for bodies of high curvature. In addition, the grid characteristics in mapped space can lead to inadequate distributions in physical space. All element information including two stretching directions, ratios and size are interpolated from an unstructured background grid which must encompass the entire domain to be discretized. Triangular surface elements are generated by adding interior points which form the base of an equilateral tetrahedral element in transformed coordinates. This transformation is a function of the background grid parameters discussed above and is calculated as follows:

$$\begin{bmatrix} P'_x \\ P'_y \\ P'_z \end{bmatrix} = \begin{bmatrix} \frac{\alpha_{x1}}{\delta_1} & \frac{\alpha_{y1}}{\delta_1} & \frac{\alpha_{z1}}{\delta_1} \\ \frac{\alpha_{x2}}{\delta_2} & \frac{\alpha_{y2}}{\delta_2} & \frac{\alpha_{z2}}{\delta_2} \\ \alpha_{x3} & \alpha_{y3} & \alpha_{z3} \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix}$$

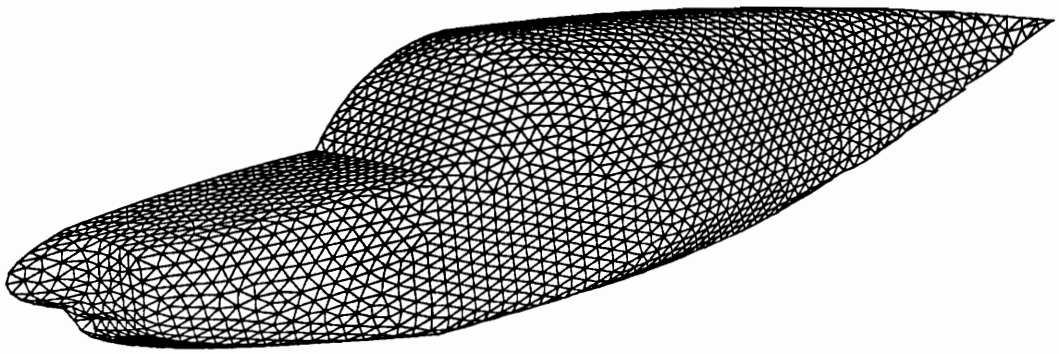
where the subscripts x, y, z refer to the components of the α vector and where $\vec{\alpha}_1$ and $\vec{\alpha}_2$ are the stretching directions provided by the background grid and $\vec{\alpha}_3$ is computed as

$$\vec{\alpha}_3 = \vec{\alpha}_1 \times \vec{\alpha}_2.$$

In addition, δ_1 and δ_2 represent the ratio of element size relative to their respective directions of stretching.

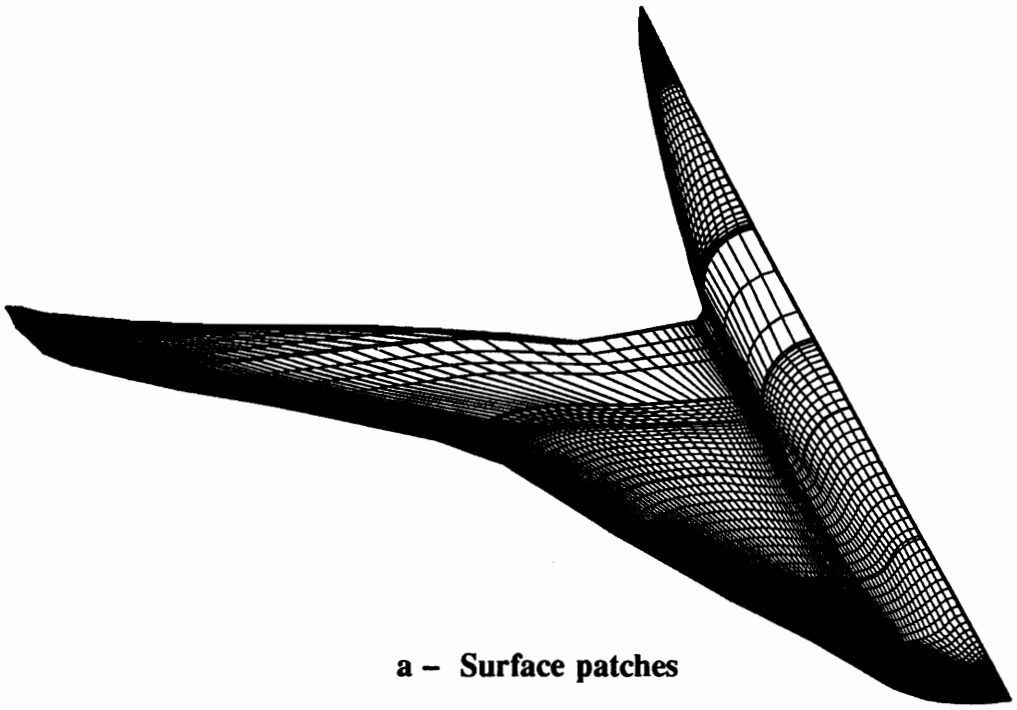


a – Surface patches

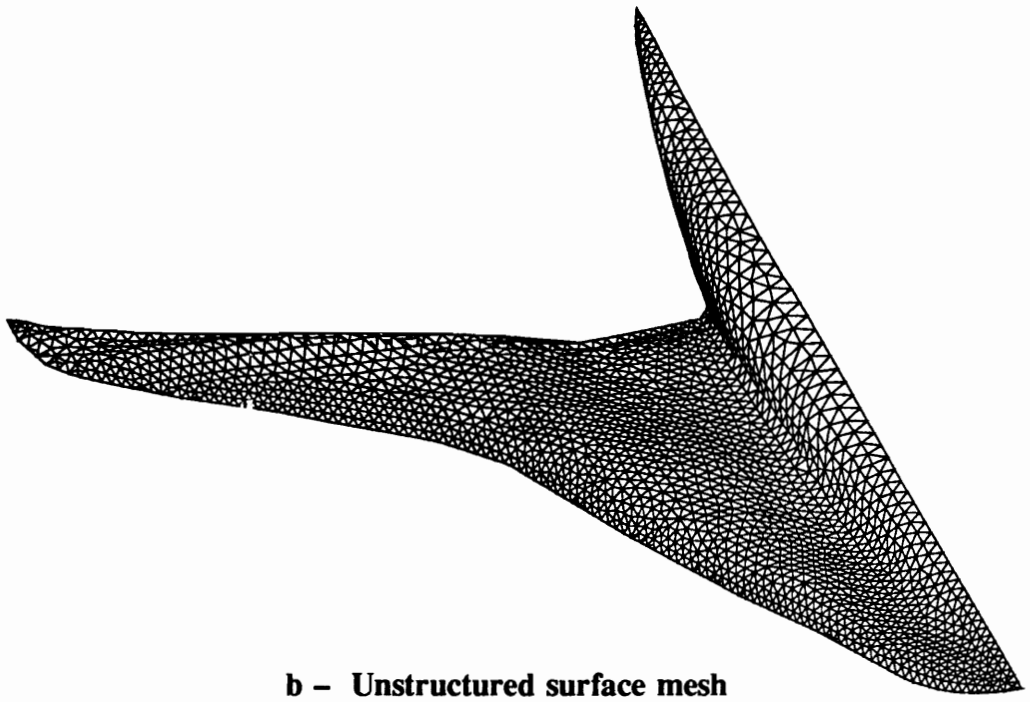


b – Unstructured surface mesh

Fig. 5.3 – Glasair fuselage surface patches and unstructured mesh



a – Surface patches



b – Unstructured surface mesh

Fig. 5.4 – High Speed Civil Transport (HSCT) surface patches and unstructured mesh

The procedure for producing surface grids involves three steps: generating initial boundary fronts in the form of line segments, advancing the front along the surface patches by adding points to create triangular cells, improving the integrity of the surface meshes using grid enhancement techniques. The algorithms for determining the location of the points that form the triangular surface elements often involves solving simultaneous equations using iterative procedures. Efficient numerical techniques are available to solve these equations. The most effective methods require parametric representations of the surface descriptions. The details of the parameterization are as follows.

The parametric patch

If we consider the structured surface grids to be a collection of patches defined by 4 points and 4 boundary edge curves, then each quadrilateral cell represents a separate patch. The coordinates for the patch are given by two-parameter, single-valued mathematical functions of the form

$$x = x(u,v); \quad y = y(u,v); \quad z = z(u,v); \quad u,v \in [0,1] \tag{5.1}$$

Figure 5.5 shows the representation of the parametric surface patch. Using the notation provided in this figure, the functional form for the parametric equations can be easily shown to be

$$\begin{aligned} x &= [x_{00}(1-u) + x_{10}u](1-v) + [x_{01}(1-u) + x_{11}u]v \\ y &= [y_{00}(1-u) + y_{10}u](1-v) + [y_{01}(1-u) + y_{11}u]v \\ z &= [z_{00}(1-u) + z_{10}u](1-v) + [z_{01}(1-u) + z_{11}u]v \end{aligned} \tag{5.2}$$

where x_{00}, x_{10}, x_{01}, and x_{11} represent the x coordinate values for the patch defined by points 00, 10, 01, and 11, respectively according to Figure 5.5.

The initial front

In order to generate the initial fronts, the line segments that define the body shape need to be mapped into the (u,v) space common to the surface patch. The physical

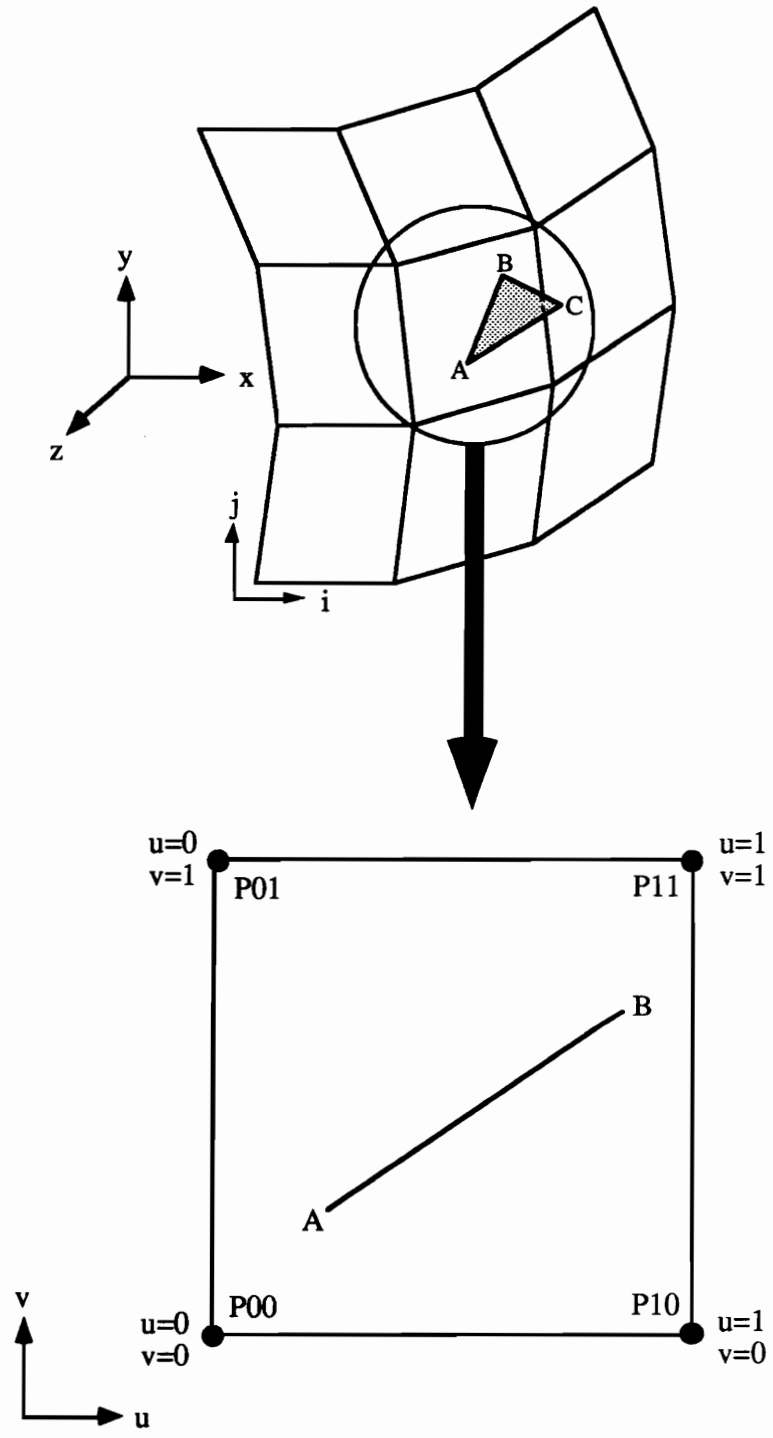


Fig. 5.5 – Parametric surface patch representation

(x, y, z) coordinates of the line segments and the surface patches are available; however, the (u, v) space is only a function of the surface patch indices. As a result, the (u, v) coordinates need to be computed for the line segments. The line segment $\overline{AB}$ in Figure 5.5 lies on the surface patch, however the parametric coordinates (u, v) are unknown. These values are obtained by implementing an algorithm that computes the minimum distance between a point and a surface in parametric space, see Ref. [43] . For our purposes, the distance is approximately zero since the line is bounded on the surface patch, however the algorithm will reveal this location in parametric space. Figure 5.6 portrays this situation where point A is a point defined on the line segment and point P is the location of this point on the surface patch. From this figure we can see that $\overrightarrow{P-A}$ must be in a direction normal to the surface at point P. Therefore the point P satisfies

$$(\overrightarrow{P-A}) \times (\overrightarrow{P^u} \times \overrightarrow{P^v}) = 0$$

or

$$(\overrightarrow{P-A}) = d(\overrightarrow{P^u} \times \overrightarrow{P^v})$$

where d is the minimum distance between the two points.

By representing $P(u, v)$ using Equation 5.2, this vector equation reveals three dependent scalar equations for the unknowns u, v, d. Newton's method can be used to efficiently solve this equation set as exact linearizations are easily calculated. The uniqueness of solution is guaranteed for this application as our lines are defined within the boundaries of the surfaces, however, in a general implementation, there may be multiple solutions or no solutions at all. After the input line segments are mapped into the parametric space of the surface patch, the initial front is generated by discretizing these boundary lines according to the background grid specifications.

The length of the line segment is calculated by determining the length of a line that forms an edge of an equilateral tetrahedral element in transformed coordinates. Figure

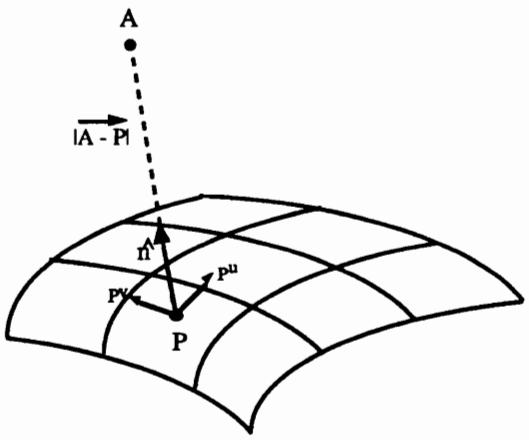


Fig. 5.6 – Minimum distance between a point and a surface

5.7 portrays an equilateral tetrahedron circumscribed in a sphere. Assume that we wish to generate the line segment $\overline{AB}$ starting from point A. The length of this line can be shown to be $\overline{AB} = \sqrt{8/3}s$, where s is the radius of the sphere. Using this information, point B can be determined and subsequently transformed back into physical coordinates. This procedure is continued until all of the boundary fronts are discretized thus completing the first step in generating the surface mesh.

Advancing the front

The surface elements are generated by locating points that form the triangular base of an equilateral tetrahedron in transformed coordinates. Returning to Figure 5.5, assume that line segment $\overline{AB}$ is on the initial front and we desire the location of point C to form a surface element. From the geometrical features of figure 5.7, we deduce that in unstretched coordinates

$$|\overline{A-C}| = |\overline{B-C}| = \sqrt{8/3}s$$

where s is the radius of the sphere.

An algorithm that reveals the location of this point involves solving two simultaneous equations for the parametric variables u, v using the secant method. An outline of the procedure is shown below.

1. Transform points A and B into unstretched coordinates.
2. Estimate the parametric location of point C to be at point A and the midpoint of A and B.
3. Solve the following equations:

$$f_1(u,v) = |\overline{A-C}| - \sqrt{8/3}s = 0$$

$$f_2(u,v) = \vec{\overline{AB}} \cdot \vec{\overline{M_{AB}C}} = 0$$

using the secant method where M_{AB} is the midpoint between points A and B. Recall that $(u,v) \in [0,1]$ according to Equation 5.1 and that during the solution process, the updated values of u, v will require that we move along the surface patches as we

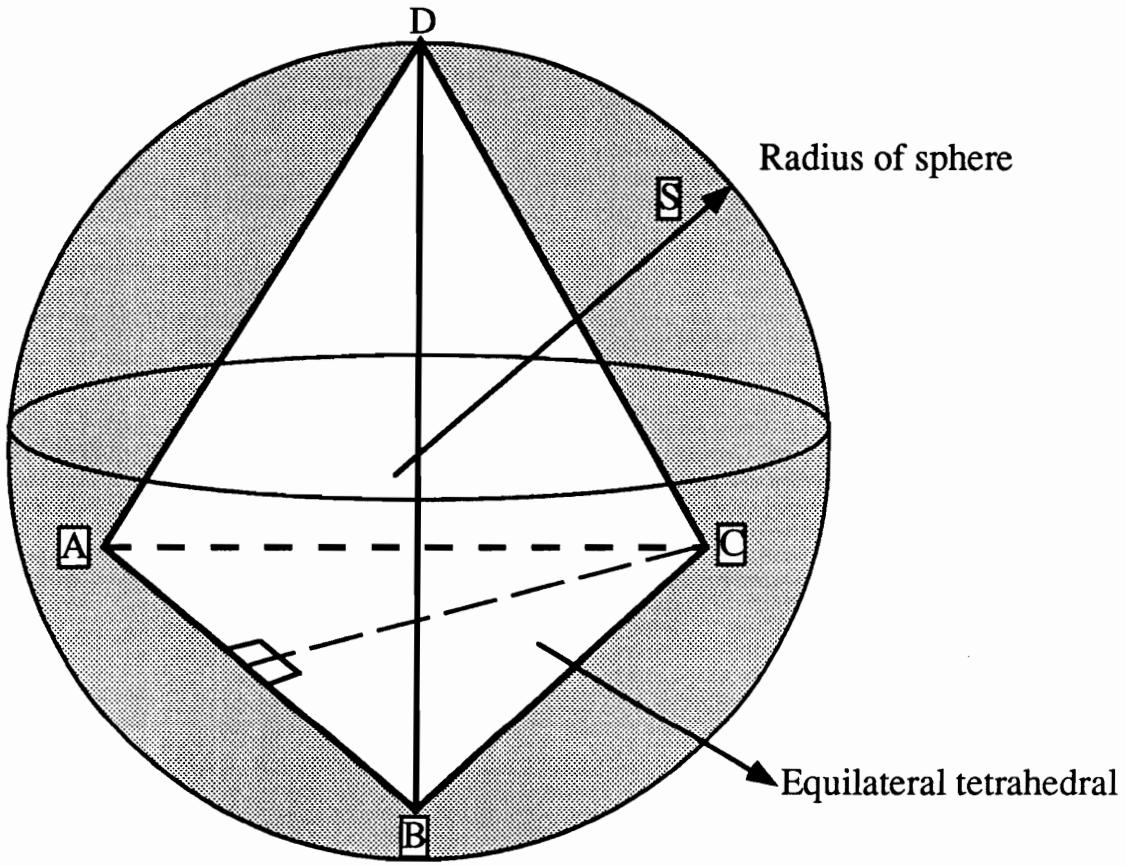


Fig. 5.7 – Sphere circumscribing equilateral tetrahedral

converge to a solution. For example, if point C in Figure 5.5 is the location of the unknown point (in stretch coordinates), then during the iterative procedure we will have moved across the initial surface patch 1 unit in the logical i direction.

4. Transform the point back into stretched coordinates. Recall that acceptance of this estimate for an interior point is subject to the criteria outlined in the advancing front algorithm above.

5.2.3 Surface Grid Refinement Algorithms

Upon completion of the boundary discretization, the surface grid quality is refined by performing a series of grid improvement algorithms. A tension spring analogy in combination with Delaunay triangularization criteria permits nodal movement and faces to be interchanged among the elements. For details of these methods in two-dimensions see Ref. [42] . Several iterations of this combined approach greatly improves the grid quality. When this process is complete, the final distribution of cells meets Delaunay criteria.

5.2.4 Examples

Several examples are provided to demonstrate the surface grid generation capabilities using the surface discretization and mesh improvement procedures previously outlined. The first involves the discretization of the ONERA M6 wing. Figure 5.8 shows the details of the surface description including the wing and symmetrical wing cap.

Several features of the mesh are highlighted in Figures 5.9 through 5.12 including the farfield, airfoil symmetry plane, leading edge, and wing tip cap. Notice the level of mesh improvement using the grid refinement techniques.

The second example details the surface and farfield mesh for the HSCT. The surface patches and completed surface grid was shown in Figure 5.4. The improvements using the criteria stated above can be seen in Figure 5.13 for a portion of the wing. The

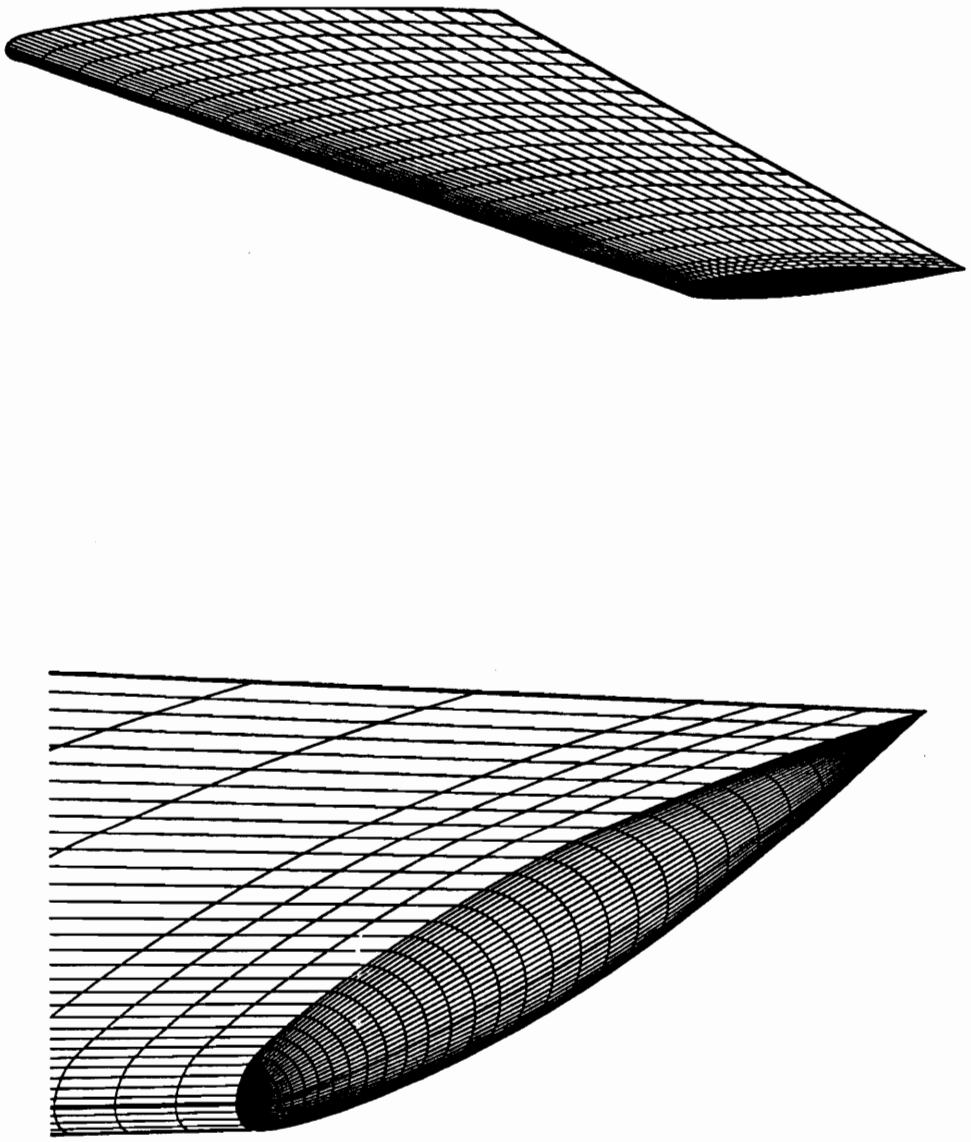
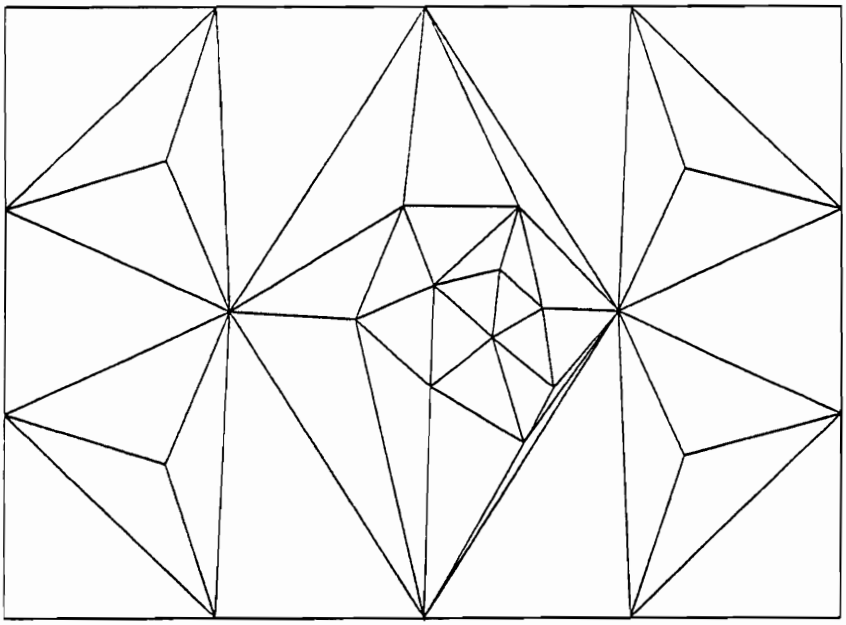
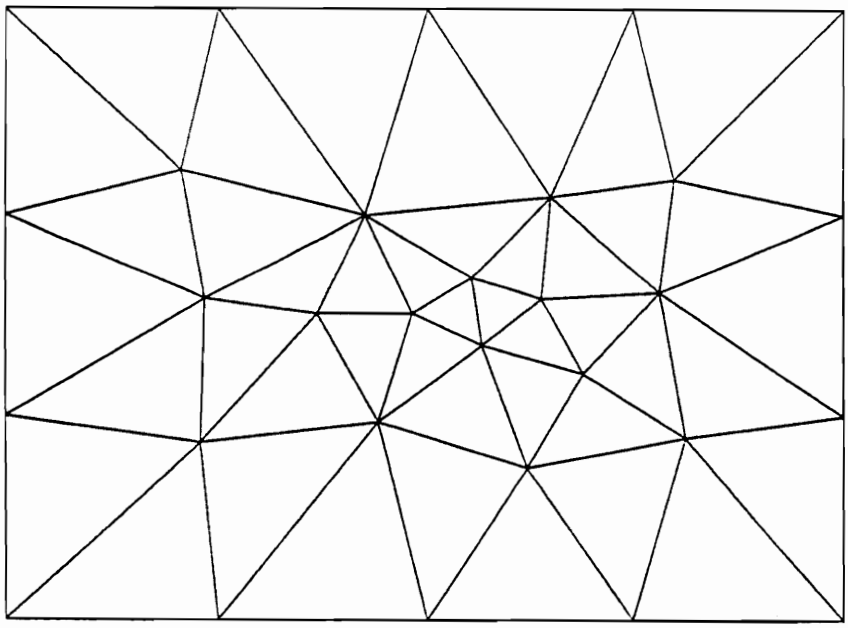


Fig. 5.8 – Surface description for the ONERA M6 wing

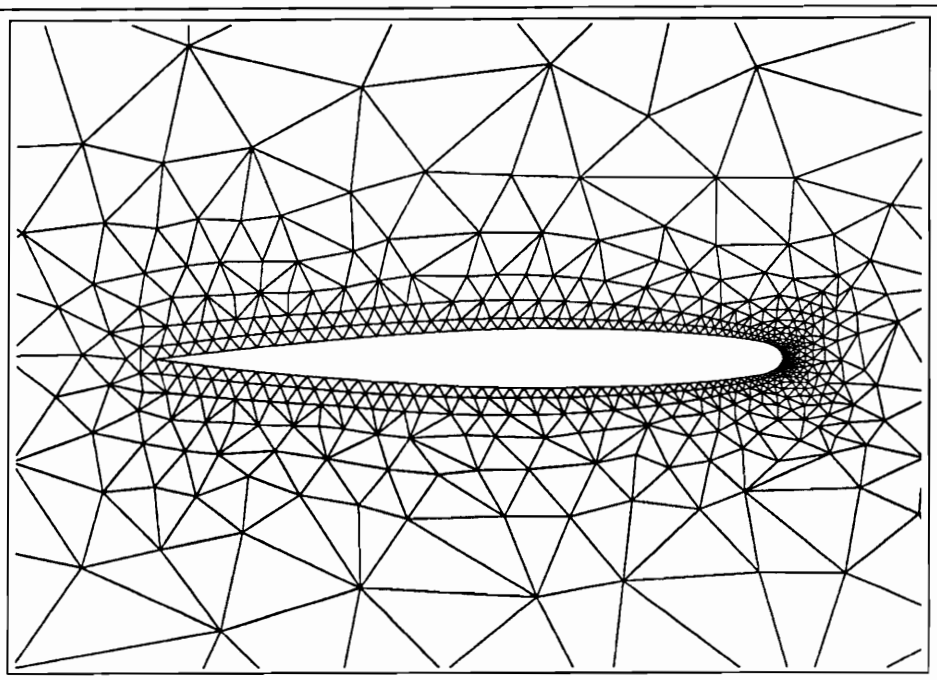


a – Before mesh improvement

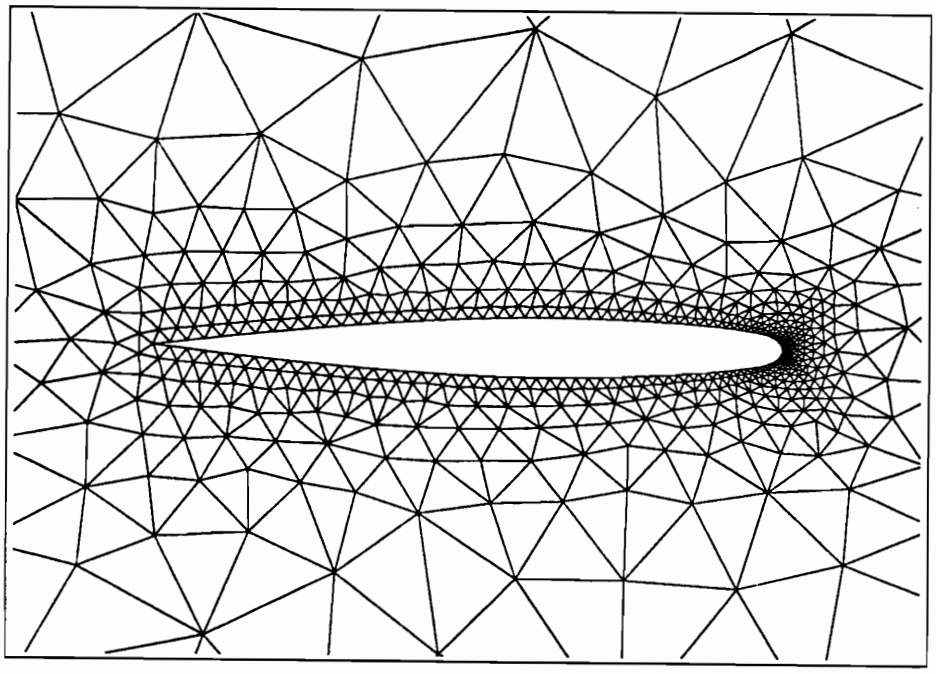


b – After mesh improvement

Fig. 5.9 – Comparison of farfield surface meshes for the ONERA M6 wing

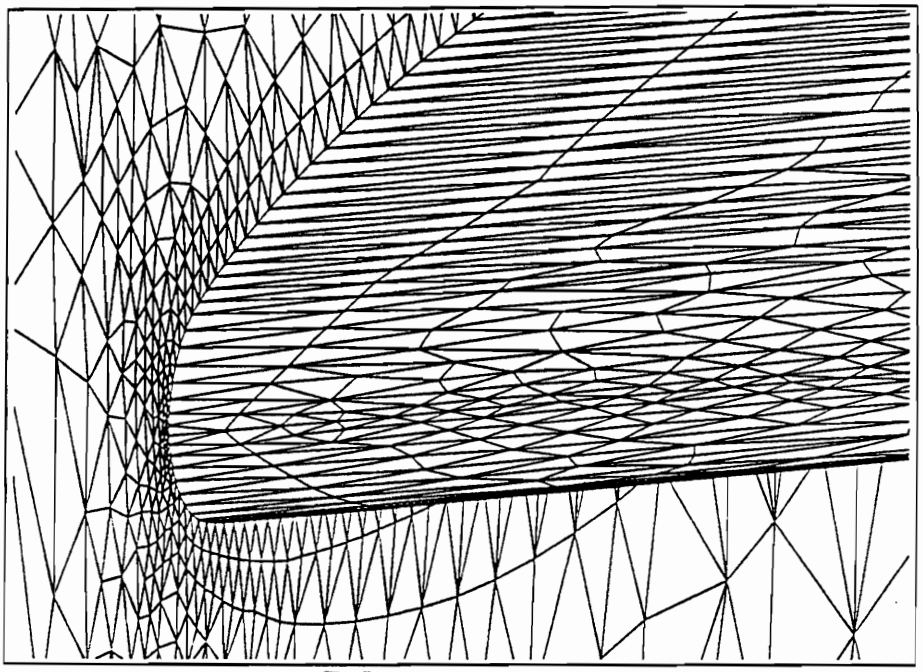


a – Before mesh improvement

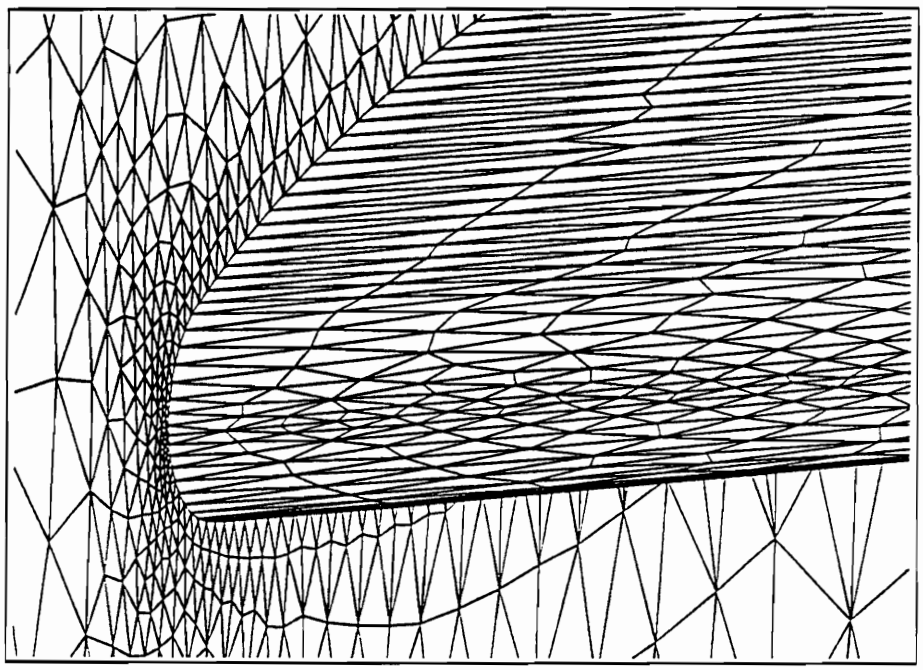


b – After mesh improvement

Fig. 5.10 – Comparison of airfoil symmetry plane surface meshes for ONERA M6 wing

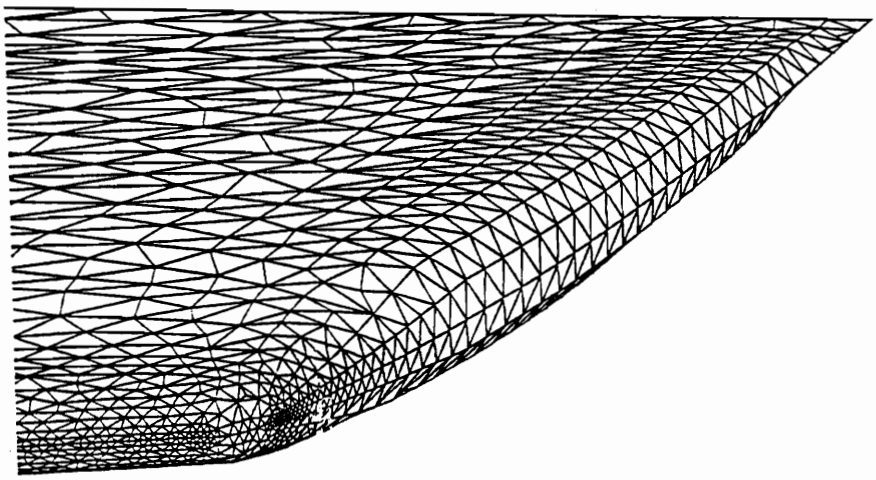


a – Before mesh improvement

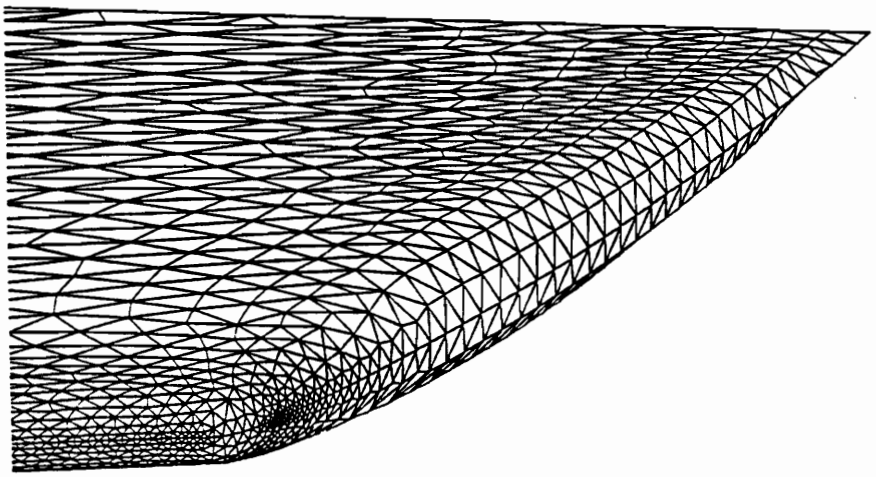


b – After mesh improvement

Fig. 5.11 – Comparison of leading edge surface meshes for ONERA M6 wing



a – Before mesh improvement



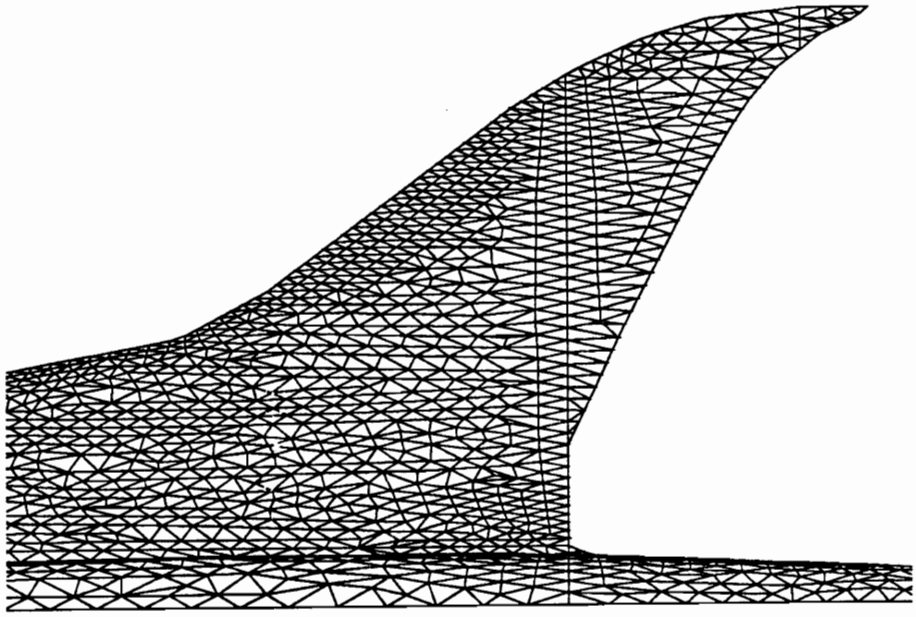
b – After mesh improvement

Fig. 5.12 – Comparison of wing tip cap surface meshes for ONERA M6 wing

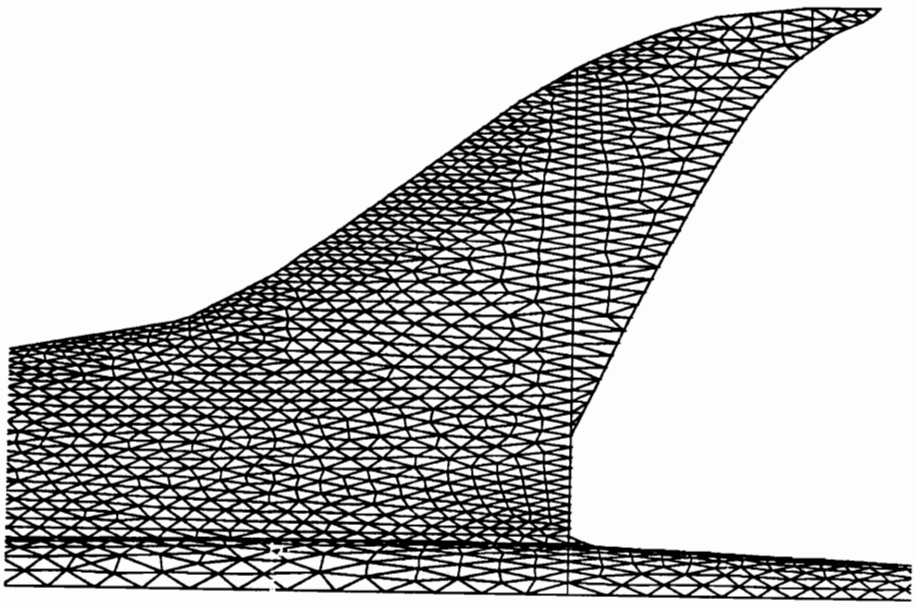
conical farfield boundary is presented in Figure 5.14 along with static pressure contours on the aircraft. This outer boundary shape is excellent for high speed calculations, and it required minimal effort to generate. Figure 5.15 is a view of the surface mesh looking from the tail of the aircraft towards the nose. The yellow grid lines represent the symmetry plane while the white lines show a portion of the outer boundary.

5.3 Interior Discretization

The interior mesh generation process is similar in concept to the surface grid generation approach. New points are added in the interior domain which form equilateral tetrahedra in unstretched coordinates. The primary difficulty in advancing the three dimensional front is the collapsing of the front to a point that prohibits completing the discretization. When this circumstance arises, faces are removed from the front that lie within a prescribed radius. This region is then re-generated. This approach has proven to be very successful in generating interior meshes. Interior mesh improvements similar to those described for the surface are currently being implemented.



a – Before mesh improvement



b – After mesh improvement

Fig. 5.13 – Comparison of wing surface meshes for the High Speed Civil Transport

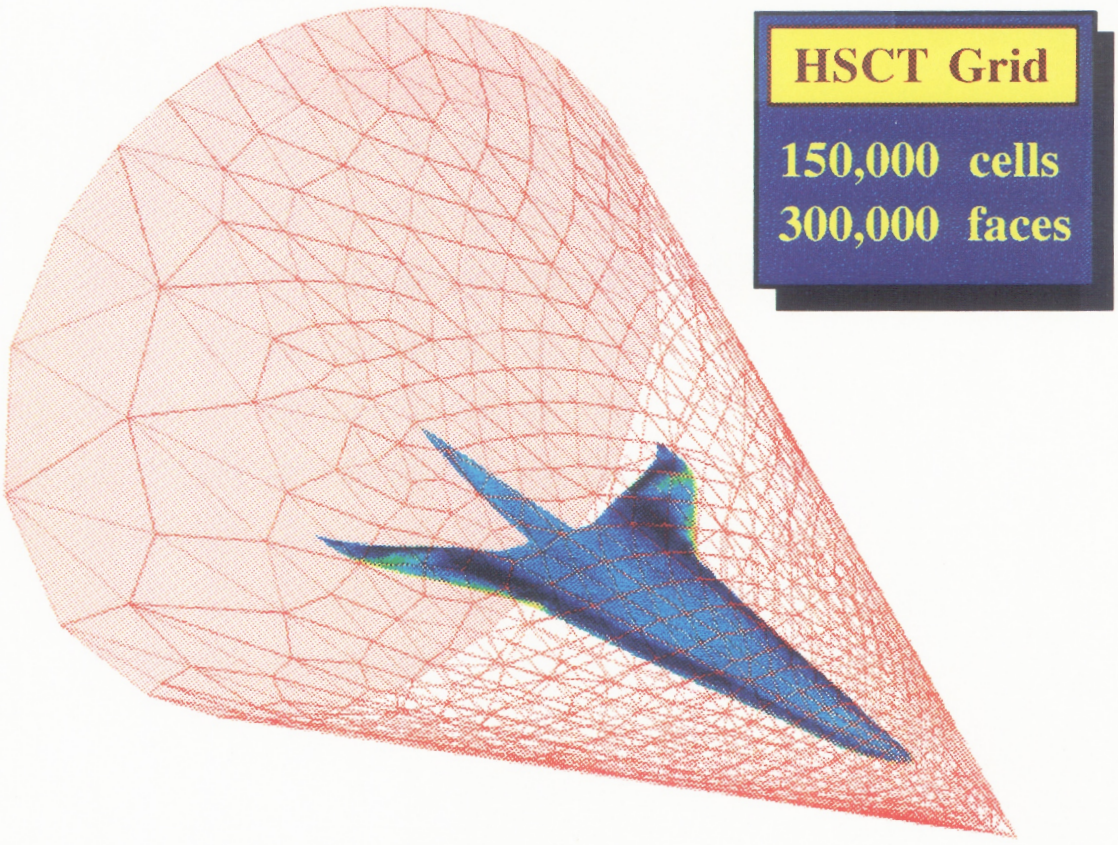


Fig. 5.14 – Conical outer boundary mesh for High Speed Civil Transport

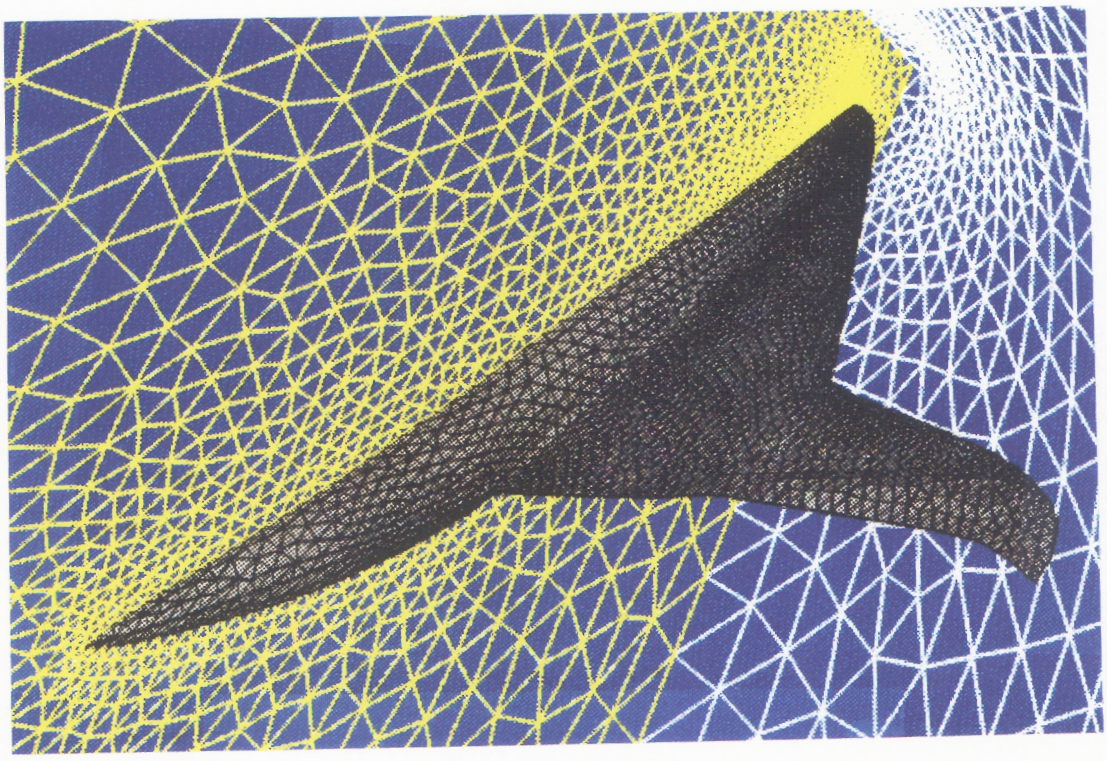


Fig. 5.15 – Interior surface mesh for High Speed Civil Transport

Chapter 6

Two Dimensional Navier Stokes Solutions

A description of the grids used for the two dimensional test cases is provided in Table 6.1. The numerical experiments were performed on both unstructured (triangular) and structured (quadrilateral) elements for comparison. The unstructured grids were generated using a modified Advancing Front technique [42] .

6.1 Inviscid Shock Reflection

A simple inviscid shock reflection is chosen as the first test case to validate the solution algorithm in the presence of discontinuities. Solutions with $k_i = 0_{upwind}$ and $k_i = 1_{upwind}$ were obtained for a $M_\infty = 2.9$ oblique shock at a wave angle of $\beta = 29°$ reflecting off a flat wall. A description of the problem, including the boundary condition specifications is provided in Figure 6.1. Steady state solutions were obtained, with and without the aid of limiting the primitive variables, for quadrilateral and triangular discretizations. Grid domains and pressure contours for both grid topologies are shown in Figures 6.2 and 6.3.

The improvement in using the higher degree reconstruction $k_i = 1_{upwind}$ as compared to the first order method, $k_i = 0_{upwind}$, is apparent in the contours. Considerably less diffusion results from the higher order method, and, as a result, the shocks are sharpened to nearly half their thickness as compared to the first order results. The effect of limiting can be observed in the wall pressure comparisons provided in Figure 6.4. The peak overshoot was greatly reduced, and the oscillations are well controlled.

Table 6.1 – Grid parameters for two-dimensional test cases.

TWO-D GRID PARAMETERS Shock reflection (SR) Flat plate boundary layer (FPBL) Shock boundary layer interaction (SBLI)						
Case	Grid #	# Cells	# Nodes	# Faces	Type	(AR)max
SR	grid.1	1243	676	1918	Tri	1
	grid.2	1176	1254	2429	Quad	1
FPBL	grid.1	800	861	1660	Quad	1190
	grid.2	400	441	840	Quad	550
	grid.3	853	469	1321	Tri	772
SBLI	grid.1	1680	1767	3446	Quad	103
	grid.2	1651	883	2533	Tri	34
	grid.3	6564	3398	9961	Tri	150

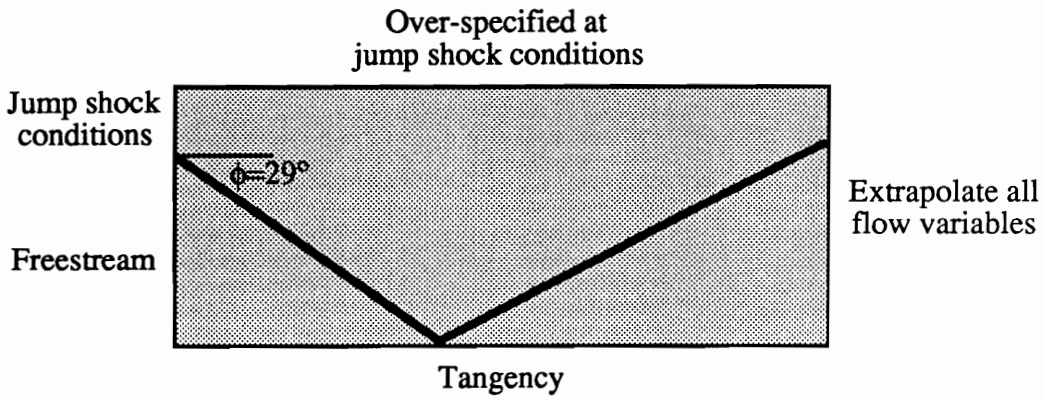
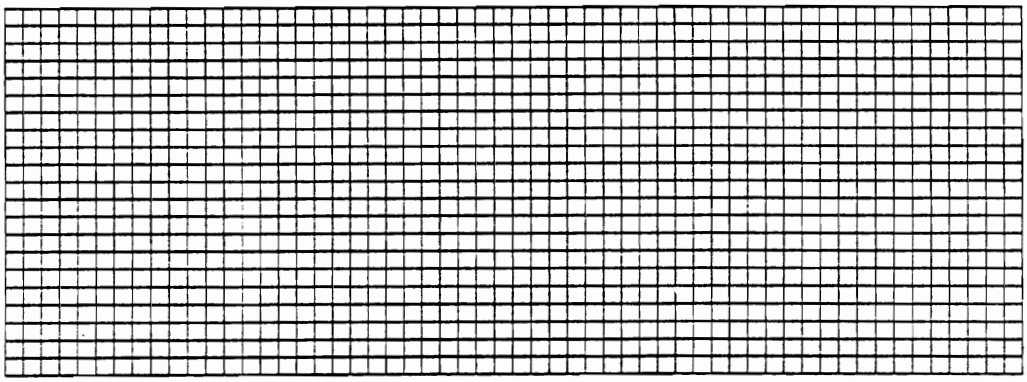
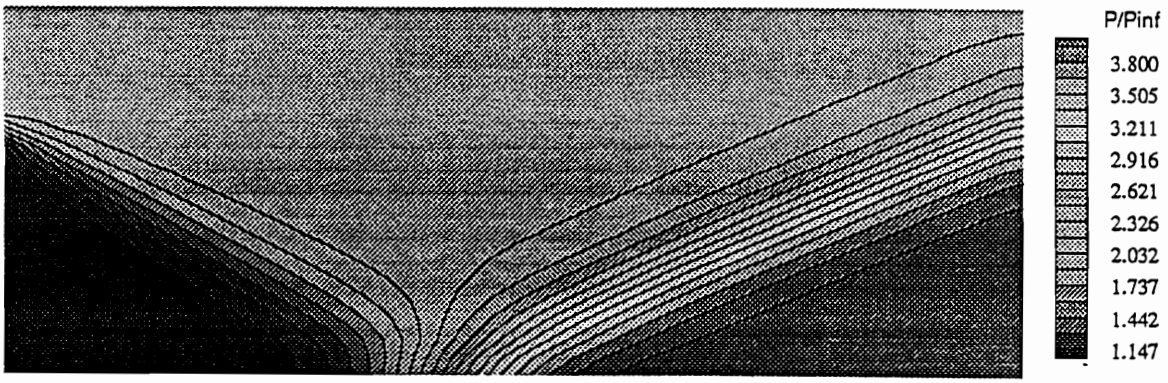


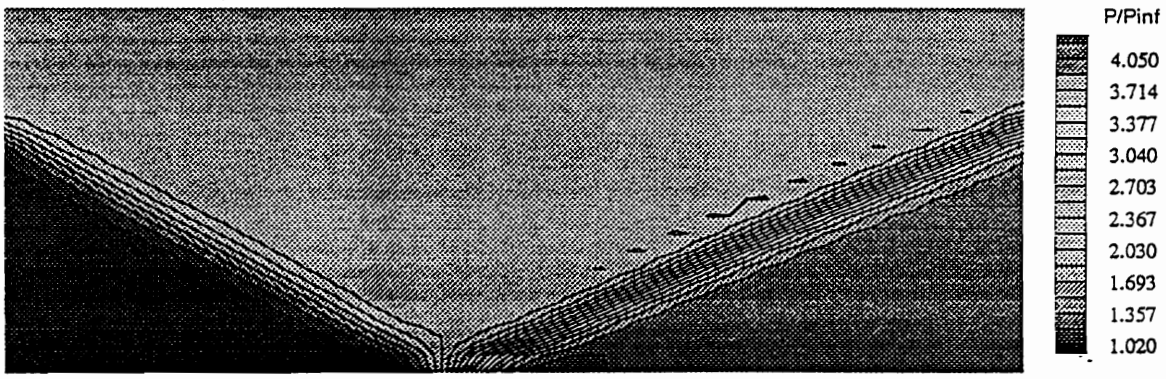
Fig. 6.1 – Problem description and boundary condition specification for the inviscid shock reflection problem.



a – Computational grid.

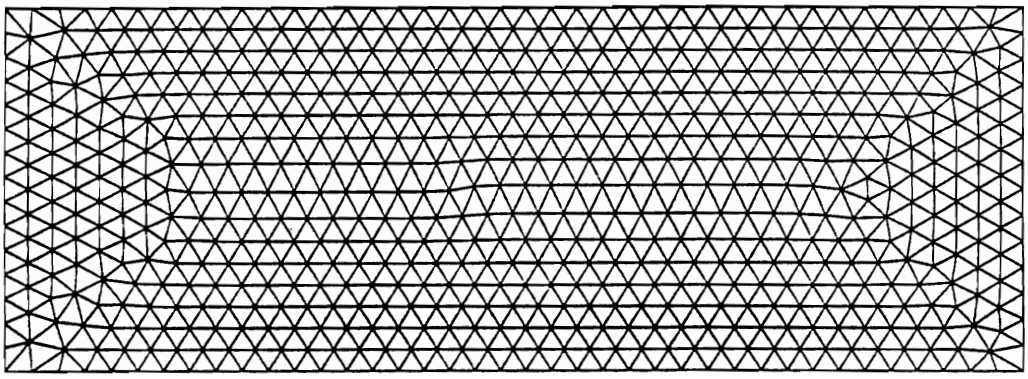


b – Static pressure contours, $k_i = 0$



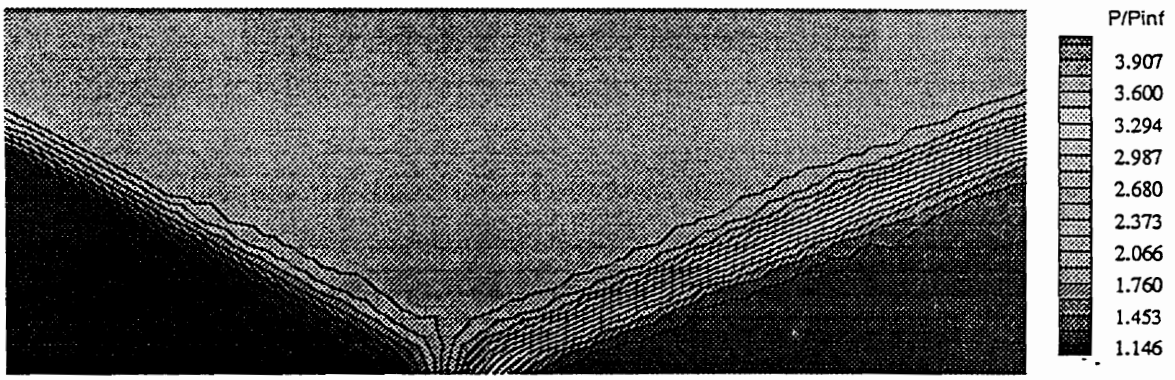
c – Static pressure contours, $k_i = 1$

Fig. 6.2 – Shock reflection problem, triangular discretization (grid.1).



a – Computational grid.

b – Static pressure contours, $k_i = 0$



c – Static pressure contours, $k_i = 1$

Fig. 6.3 – Shock reflection problem, quadrilateral discretization (grid.2).

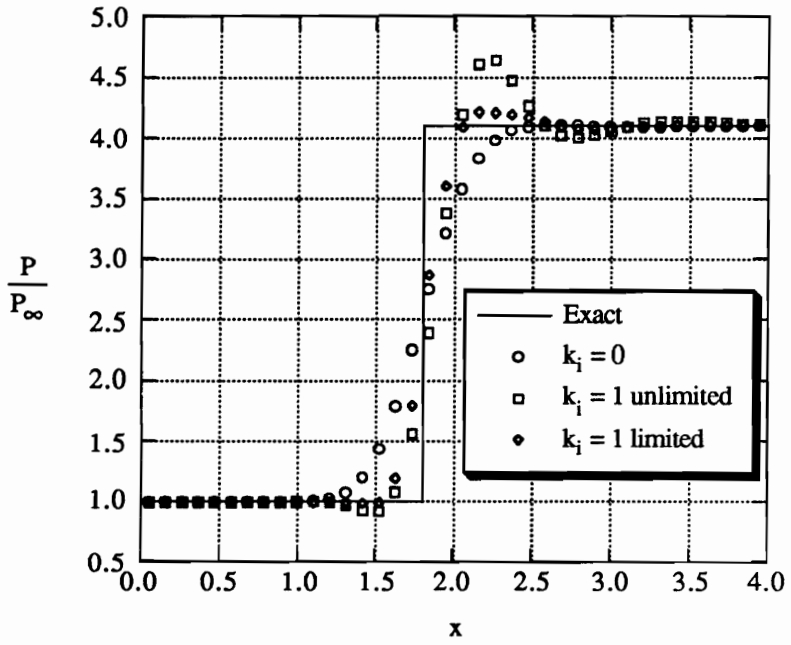


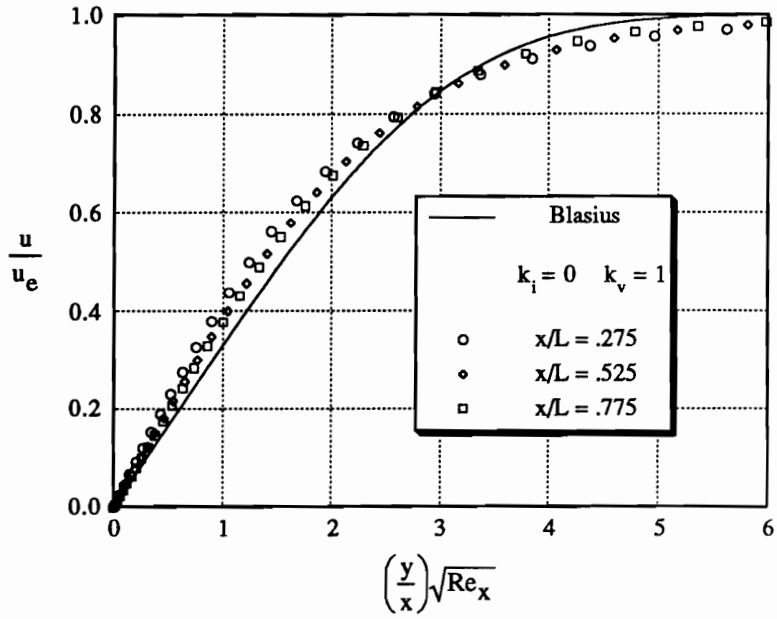
Fig. 6.4 – Wall pressure comparison for shock reflection, triangular discretization (grid.1).

6.2 Flat Plate Boundary Layer.

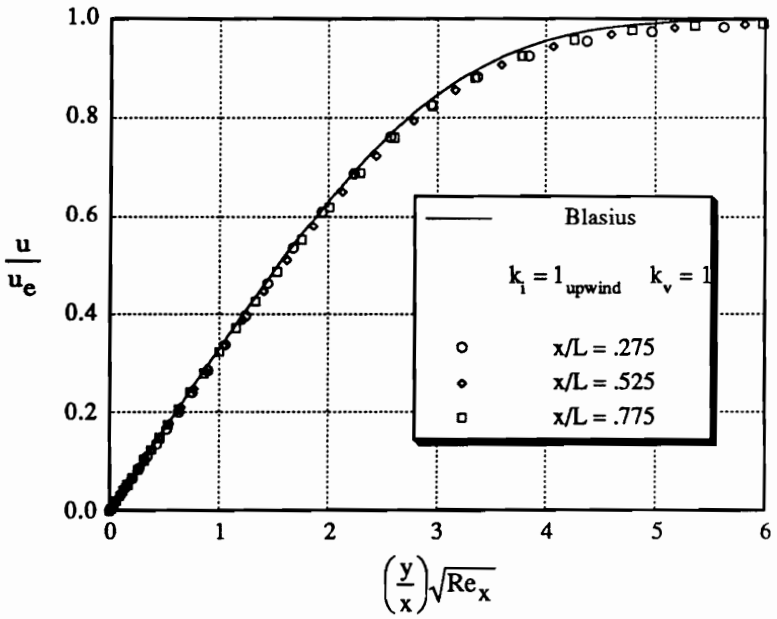
An assessment of the viscous capabilities of the reconstruction algorithm is investigated by considering subsonic flow over a flat plate and comparing to the velocity profiles from the Blasius solution [44] . Results for quadrilateral and triangular meshes are provided at a free stream Mach number $M_\infty = 0.3$ and Reynolds number based on freestream velocity and length of the plate of 2.0×10^5. Constant freestream entropy and total pressure and $v = 0$ were imposed at the inflow. In addition, the static pressure was extrapolated from the interior. No-slip and adiabatic wall conditions were applied on the plate. The pressure was specified along the outflow and top boundaries, all other properties were extrapolated. The results are shown in Figures 6.5 through 6.7.

The solution quality using $k_i = 0_{upwind}$ (first order) and $k_i = 1_{upwind}$ (second order) for the quadrilateral mesh in Figure 6.5 is similar to current structured algorithm results. Figure 6.6 compares $k_i = 0_{upwind}, 1_{upwind}$, and $2_{upwind-biased}$ reconstructions fixing $k_v = 1$ on a coarse quadrilateral mesh. The results do not provide visual evidence that $k_i = 2_{upwind-biased}$ reconstruction improves the solution. This is because the viscous and inviscid reconstructions are not of the same degree, hence global accuracy is dictated by the lowest order approximation. This concept was tested by performing consistent reconstructions, specifically, $k_i = 2_{upwind-biased}$ and $k_v = 2$ for grid.2, however the results were essentially equivalent. Coarser grids may be necessary to reveal any improvement with higher degree reconstructions for this test case. Figure 6.7 demonstrates the ability to calculate viscous flows on highly stretched triangular meshes in which all of the data for $x_L \geq .3$ was plotted. As can be seen in Table 6.1, the maximum aspect ratio for the triangles was as high as 772. Moreover, 25% of the angles were greater than 150 degrees and 18% were greater than 170 degrees. Many researchers have believed that it was not possible to accurately resolve the spatial variations on trianglar meshes with this amount

of distortion. However, by precise reconstruction and careful selection of the stencil, accurate calculations are possible.



a – $k_i = 0, k_v = 1$



b – $k_i = 1_{upwind}, k_v = 1$

Fig. 6.5 – Comparisons of velocity distributions with incompressible Blasius solution; $M_\infty = .3$, $Re_L = 2.0 \times 10^5$, Grid.1

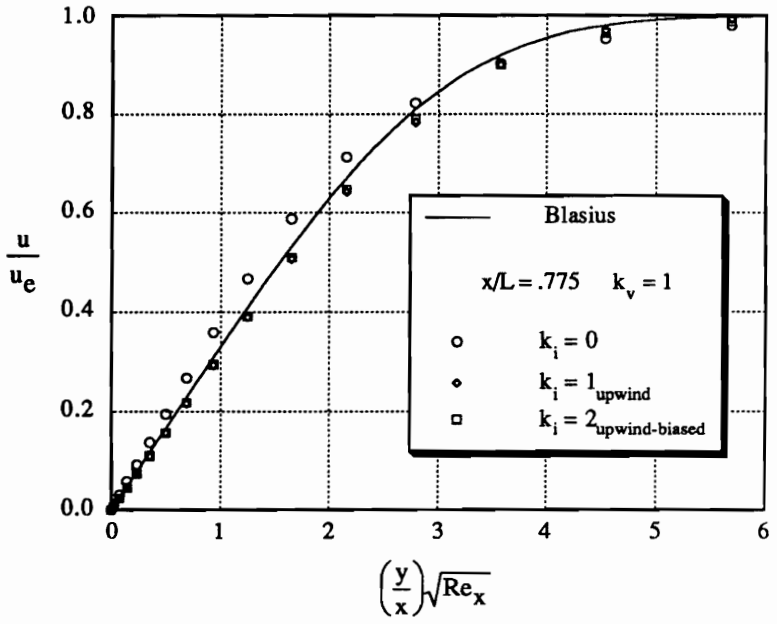
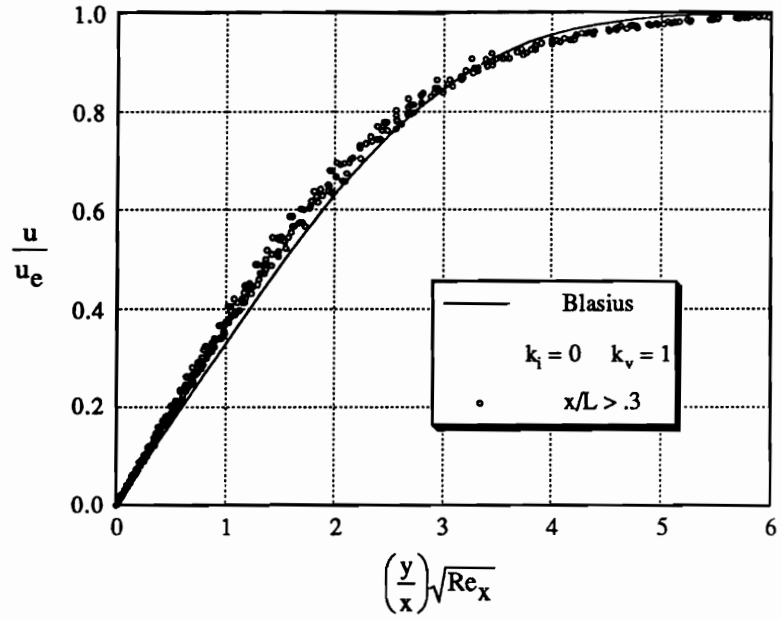
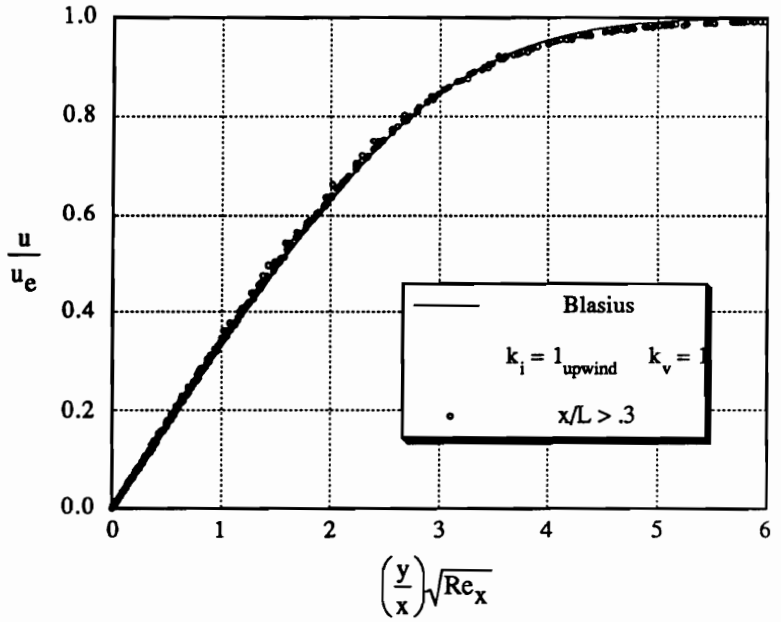


Fig. 6.6 – Comparisons of velocity distributions with incompressible Blasius solution; $M_\infty = .3$, $Re_L = 2.0 \times 10^5$, **Grid.2**



a – $k_i = 0, k_v = 1$



b – $k_i = 1_{upwind}, k_v = 1$

Fig. 6.7 – Comparisons of velocity distributions with incompressible Blasius solution; $M_\infty = .3$, $Re_L = 2.0 \times 10^5$, **Grid.3**

6.3 Shock Boundary Layer Interaction.

The third model problem selected to test the reconstruction algorithm represents the combined physics of an impinging shock wave with a laminar boundary layer resulting in separated flow. The experiments of Hakkinen [45] provide the details of a Mach 2.0 oblique shock wave impinging on a developing laminar boundary layer with sufficient strength to cause a region of separated flow before reattachment. The details of the computational domain and boundary condition treatment are shown in part *a* of Figure 6.8. The three grids used in this simulation depicted in Figure 6.9 demonstrate the quality of the unstructured grids using the background grid procedure defined in Chapter 5 section 5.1.2.

Comparisons of wall pressure and skin friction for the three grids used in this investigation are provided in Figure 6.10. The size of the separation region defined in the skin friction plot and the wall pressure distribution is captured by all three grids and is in agreement with the documented numerical results on structured grids [27] . In fact, the solution on the quadrilateral grid (grid.1) using $k - exact$ reconstruction captures the separation region and wall pressure distribution better than the structured algorithm in Ref. [27] on the same grid. Possible explanations include the accuracy of the computational space interpolations used in the inviscid flux calculation and the choice of Riemann solver. The unstructured grid refinement study reveals better prediction of peak wall pressure, however the separation region remained virtually unchanged. Static pressure contours are shown in part *b* of Figure 6.9. The details of the physics including the separating boundary layer, expansion over the separation bubble, and recompression wave caused by turning the flow at the wall are well represented for the fine unstructured grid. A final calculation was performed on the quadrilateral mesh (grid.1) comparing higher order reconstructions. The results are shown in Figures 6.11 for wall pressure and

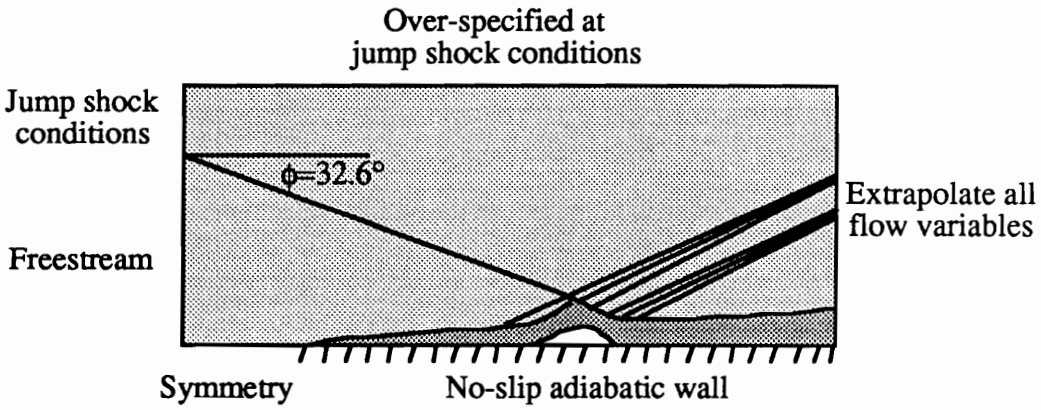
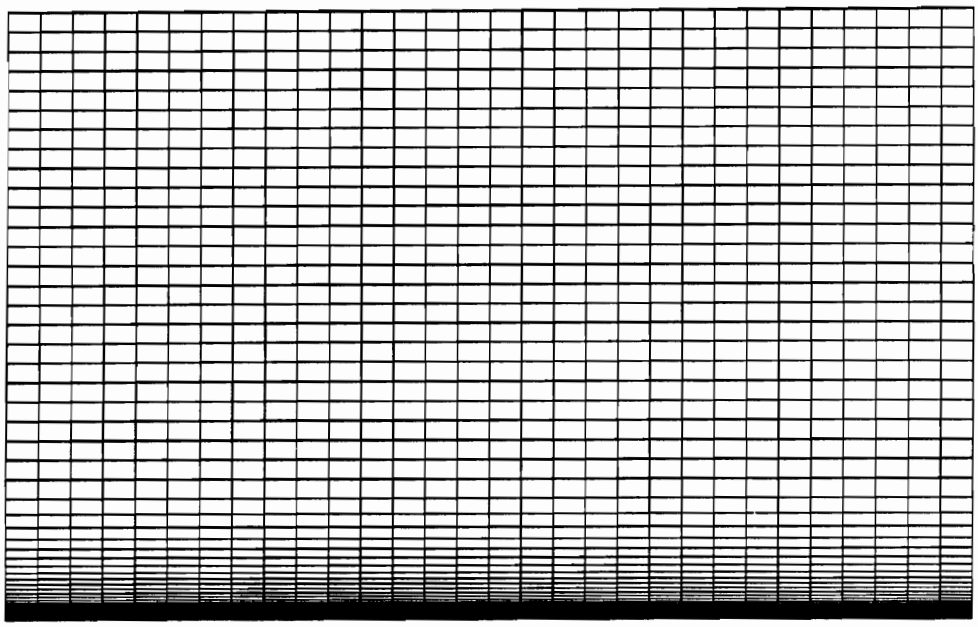
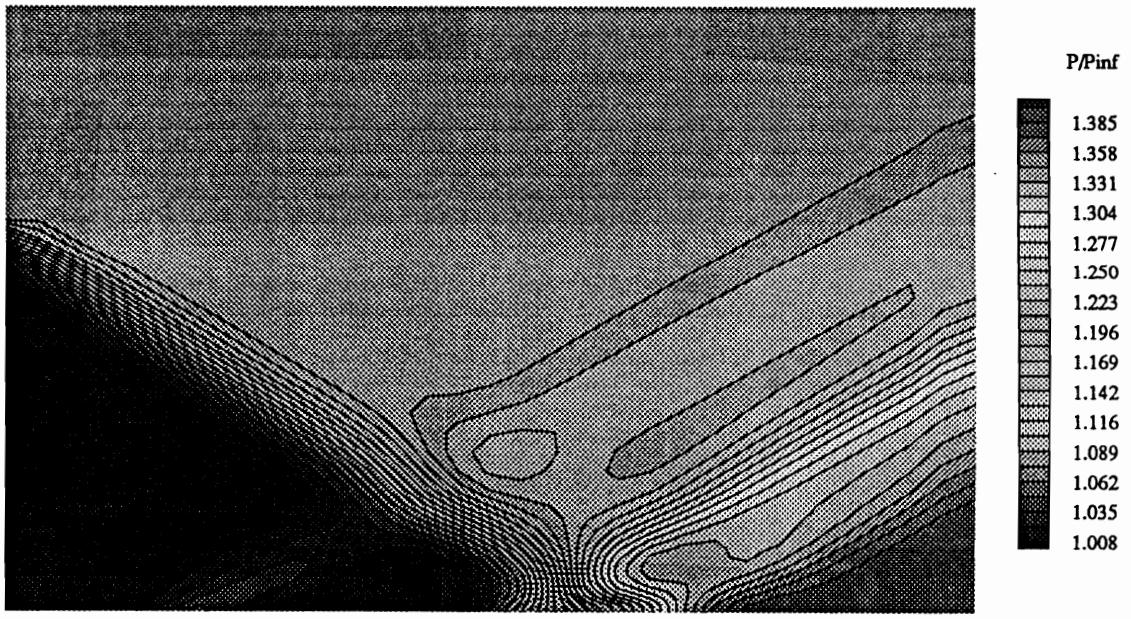


Fig. 6.8 – Problem description and boundary condition specification for the laminar shock boundary layer interaction problem.

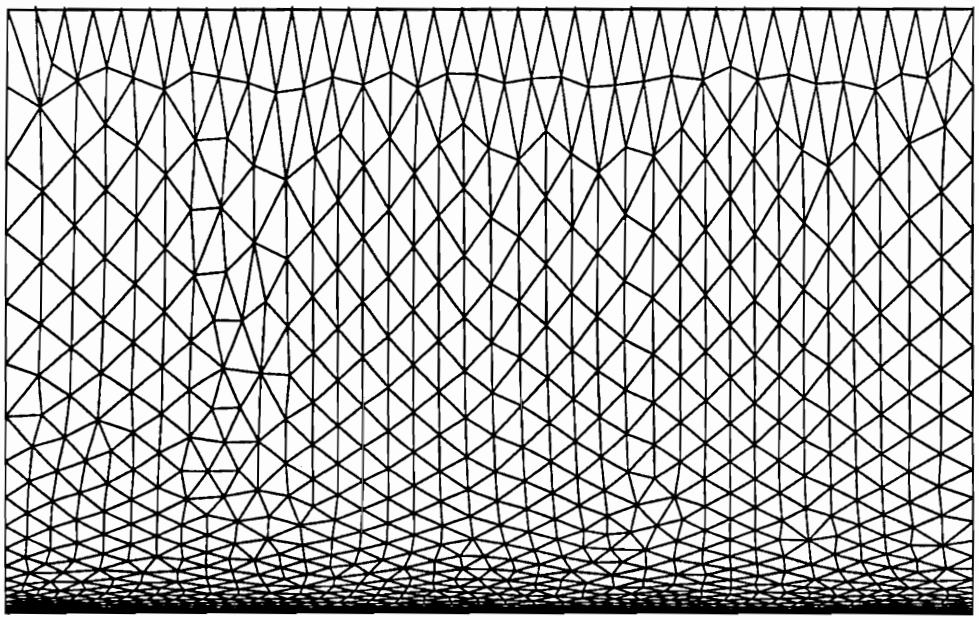


a – Grid.1

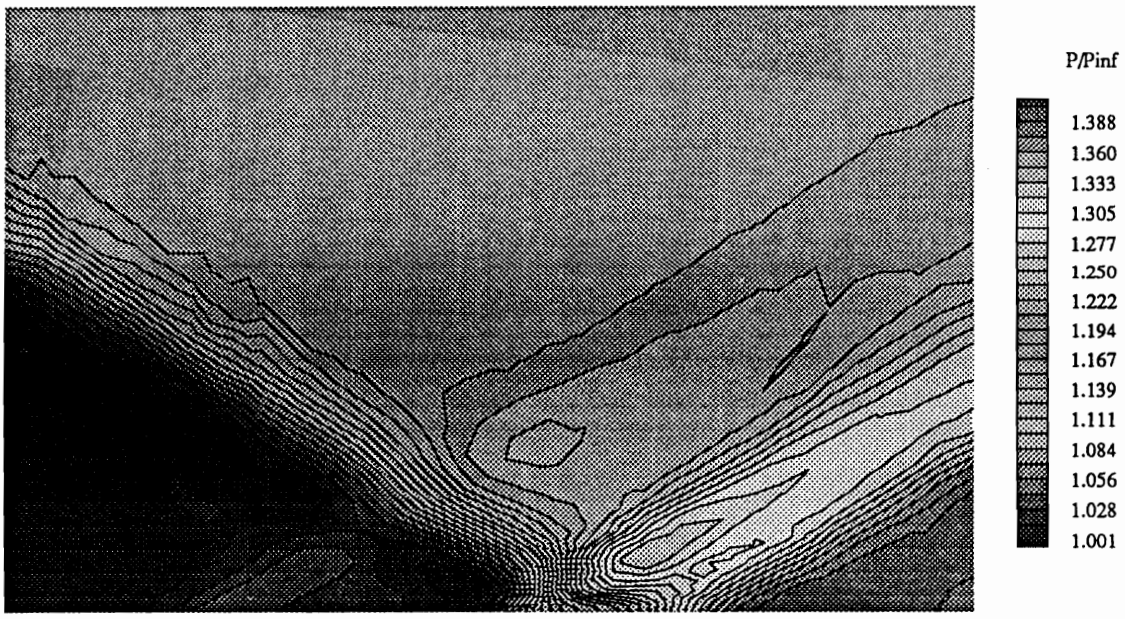


b – Static pressure contours

Fig. 6.9 – Computational grids and pressure contours for the laminar shock boundary layer interaction problem.

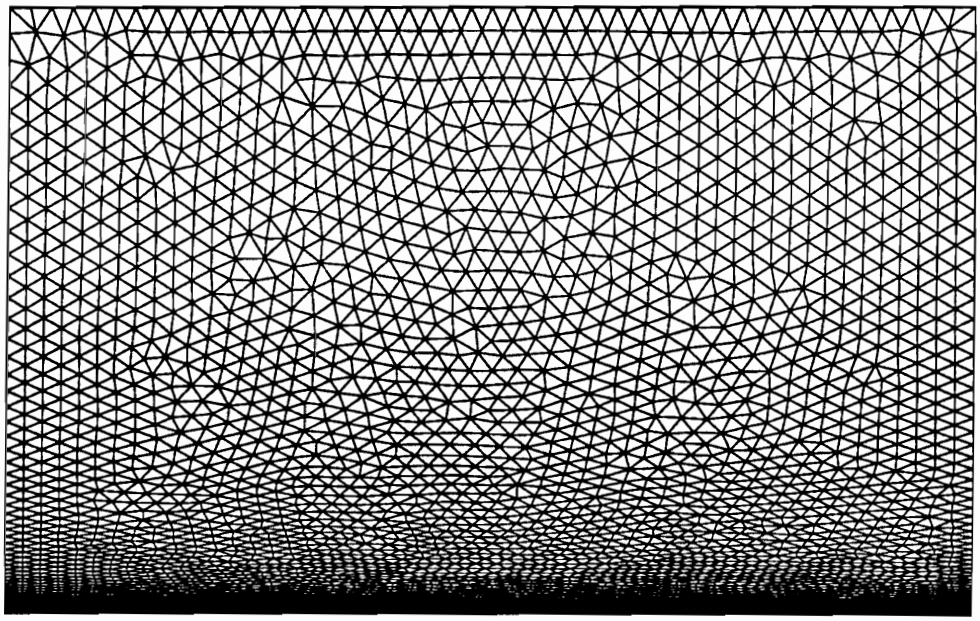


a – Grid.2

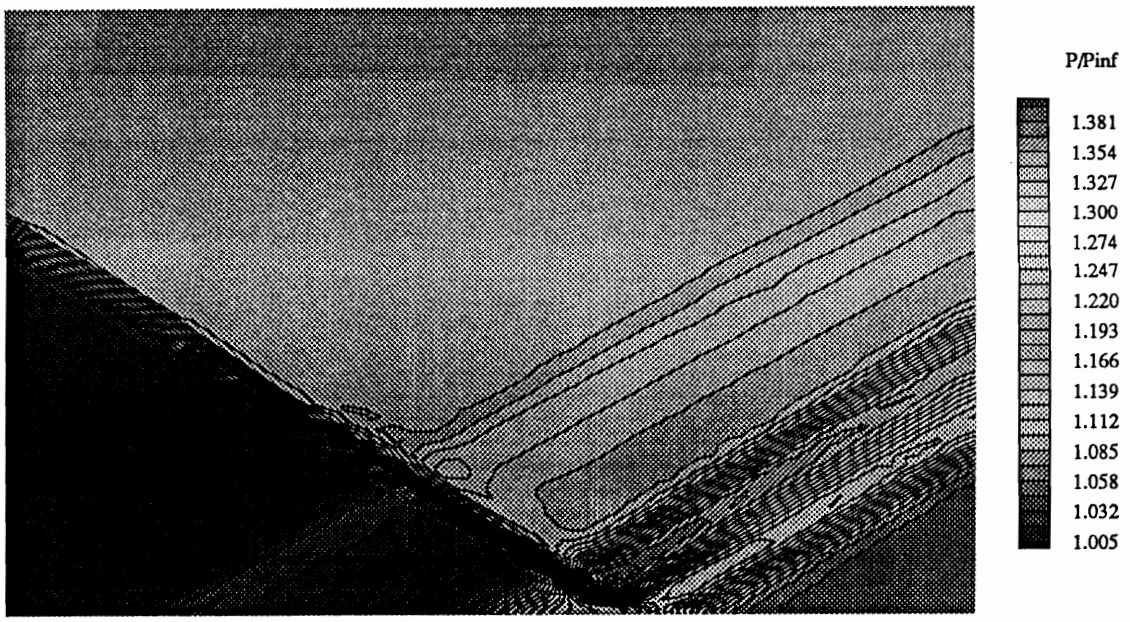


b – Static pressure contours

Fig. 6.9 – Computational grids and pressure contours for the laminar shock boundary layer interaction problem. (Continued)



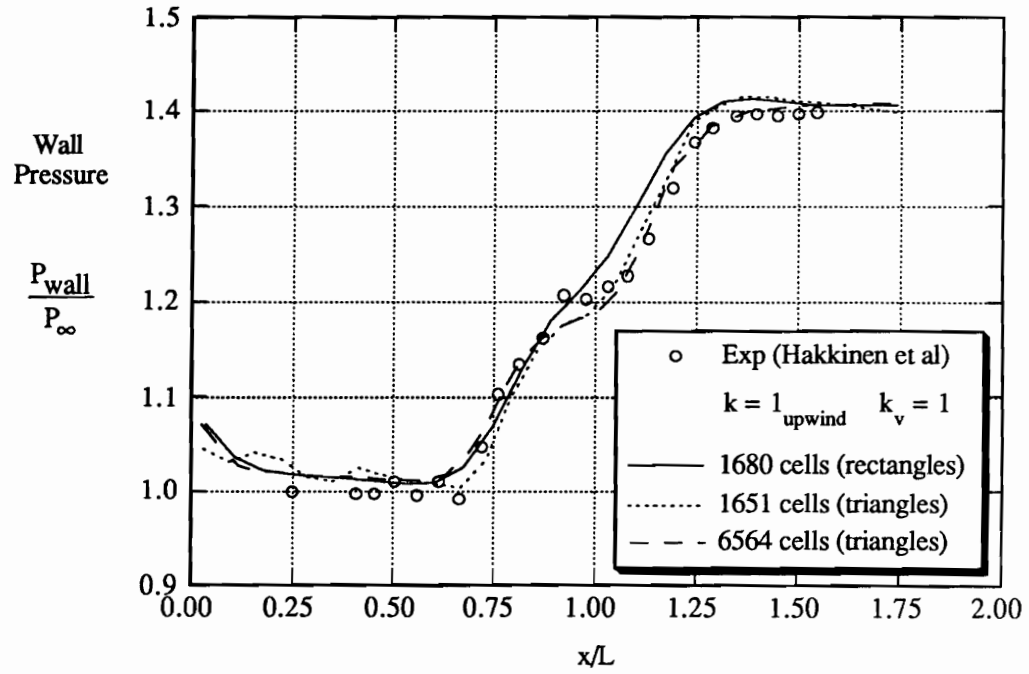
a – Grid.3



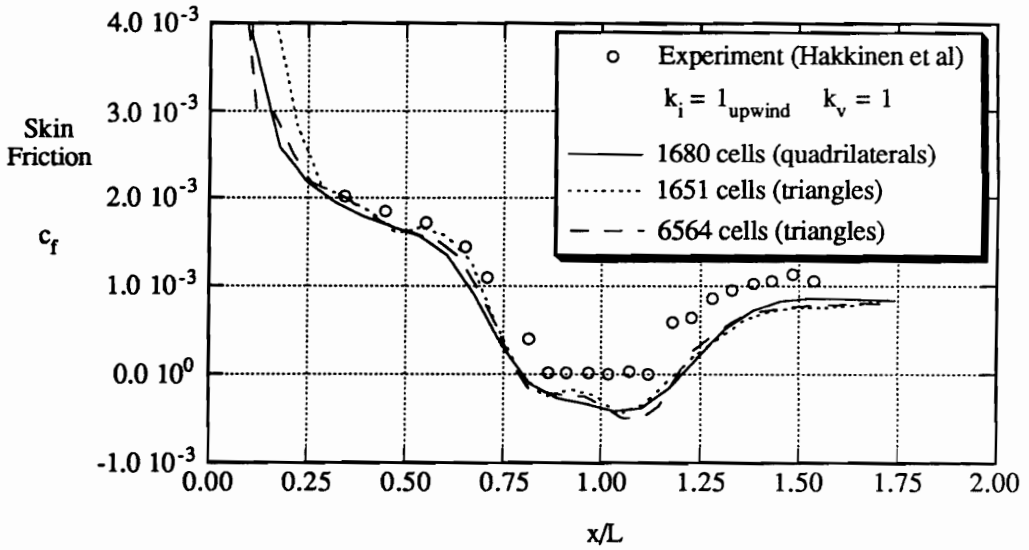
b – Static pressure contours

Fig. 6.9 – Computational grids and pressure contours for the laminar shock boundary layer interaction problem. (Continued)

skin friction. These higher order results are the first in the literature for this problem and demonstrates the improved accuracy with increased degree of reconstruction. The most noticeable improvement is the reattachment point following the separation region. The numerical results in Ref. [27] demonstrate this trend in a grid refinement study. The peak wall pressure is also observed to agree slightly better with the experimental data.

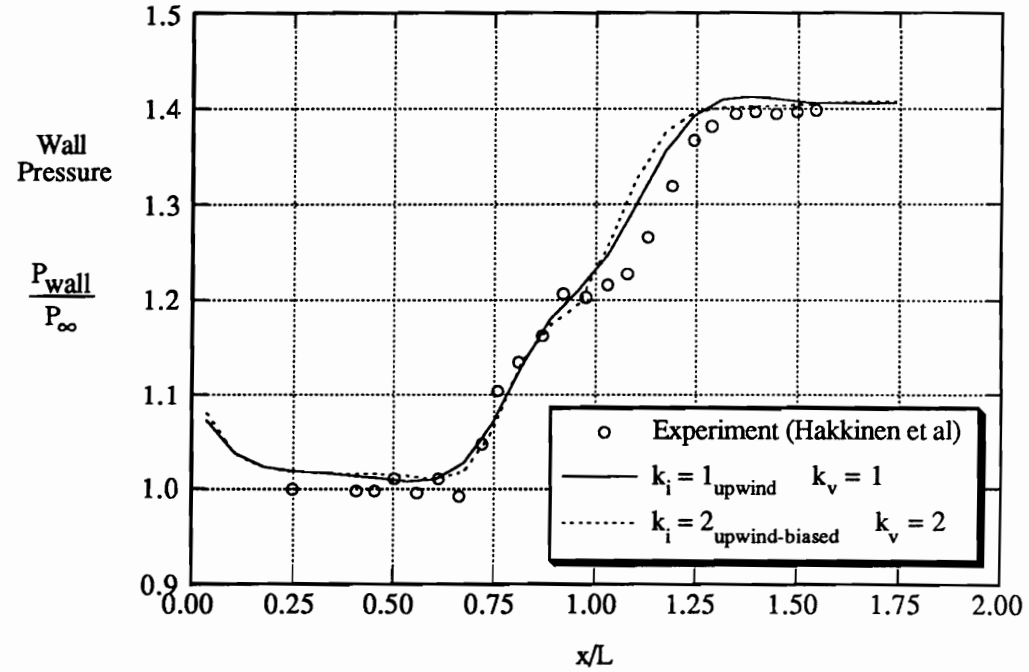


a – Wall pressure

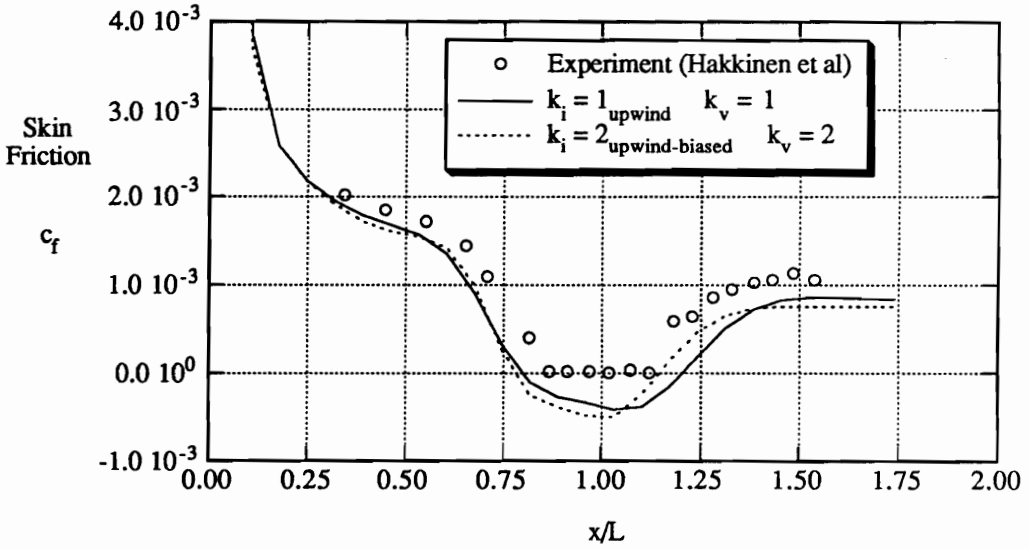


b – Skin friction

Fig. 6.10 – Wall pressure and skin friction comparisons;
$M_\infty = 2.0, Re_L = 2.96 \times 10^5$.



a – Wall pressure



b – Skin friction

Fig. 6.11 – Wall pressure and skin friction comparisons for quadrilateral mesh (grid.1); $M_\infty = 2.0, Re_L = 2.96 \times 10^5$.

Chapter 7

Three Dimensional Euler Solutions

7.1 ONERA M6 Wing

The test case selected to validate the reconstruction algorithm in three dimensions is the ONERA M6 wing [46] . This geometry has been frequently selected to determine the performance of structured [28] as well as unstructured [17] [32] flow solver algorithms. A summary of the grid specifications is provided in Table 7.1. Transonic flow predictions at $M_\infty = 0.84$ and $\alpha = 3.06°$ were performed on the two grids shown in Figure 7.1. Both grids exhibit spanwise stretching with a maximum value of six at the leading edge and reducing to an average value of three for the last half of the chord. This results in an order of magnitude reduction in grid size compared to its unstretched counterpart. No stretching was imposed along the outboard portion of the wing containing the spherical wing tip cap ($0.9 \leq \frac{y}{b} \leq 1.0$).

The boundary conditions imposed for this investigation include tangency along the wing surface, symmetry constraints along the center plane, and characteristic conditions in the form of Riemann invariants for all remaining boundaries.

The implicit time integration procedure outlined in Chapter 3 was used to obtain steady state solutions for this investigation. Implicit point Jacobi was performed using $k_i = 0$ (first order) until the normalized l_2 norm of the full residual decreased $1 - 1.5$ orders of magnitude. At this point, the inner iteration scheme using a higher order, $k_i = 1_{upwind}$, reconstruction reduced the residual to the desired 4 orders of magnitude. The CFL's for point Jacobi and the inner iteration scheme were 1.0 and 50.0, respectively.

Table 7.1 – Grid parameters for three-dimensional test case.

THREE-D GRID PARAMETERS
ONERA M6 Wing

Grid #	# Cells		# Nodes		# Faces	
	Boundary	Total	Boundary	Total	Boundary	Total
grid.1	3988	47004	1996	8088	3988	88026
grid.2	8476	105155	4240	18067	8476	197596

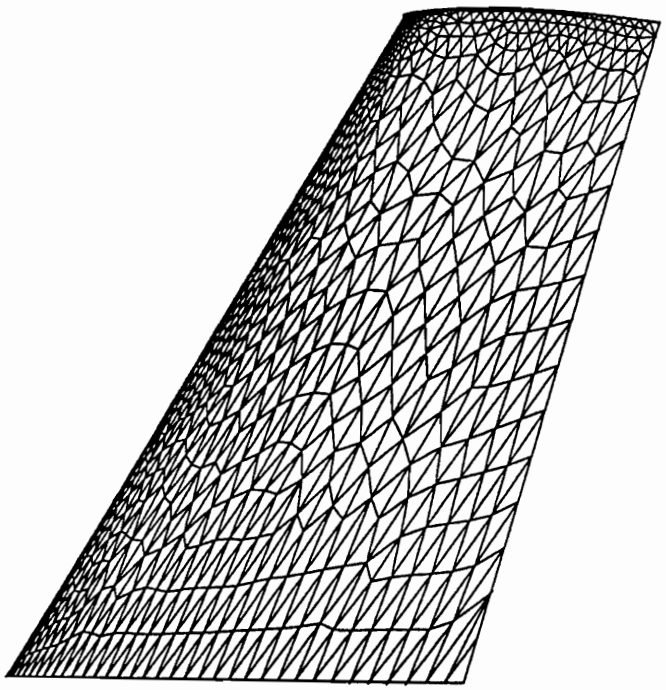
With the aid of blocking strategies, all computations were performed interactively on a Cray YMP. The coarse grid was run as one block, however the fine grid case was divided into several regions based on geometrical considerations. Specifically, the cells were sorted spanwise and grouped into 4 blocks. Solution histories are provided in Figures 7.2 and 7.3. The coarse grid solution converged in approximately 17 minutes using 10 mega-words of memory. On the fine grid (grid.2), the residual histories display the convergence trends as a result of varying the number of iterations performed on each block as they are sequentially cycled. The number varies from an initial value of 3 to a final value of 1. The slope remains fairly constant during the convergence process. The complete solution required 65 CPU minutes and 9 mega-words of memory.

The results for this study include surface pressure contours and streamwise surface C_p distributions. The basic flow features are shown in Figure 7.4 including the lambda type shock pattern on the top surface of the wing and the pressure distribution superimposed on the symmetry plane. Notice the grid clustering near the leading edge to resolve the initial gradients. Also included in this figure are surface contours at $\frac{y}{b} = 0.65$. The post-processing capabilities for displaying unstructured CFD results have greatly improved over the past two years. These results, as well as the results that follow, were obtained using the flow visualization packages FAST and TECPLOT.

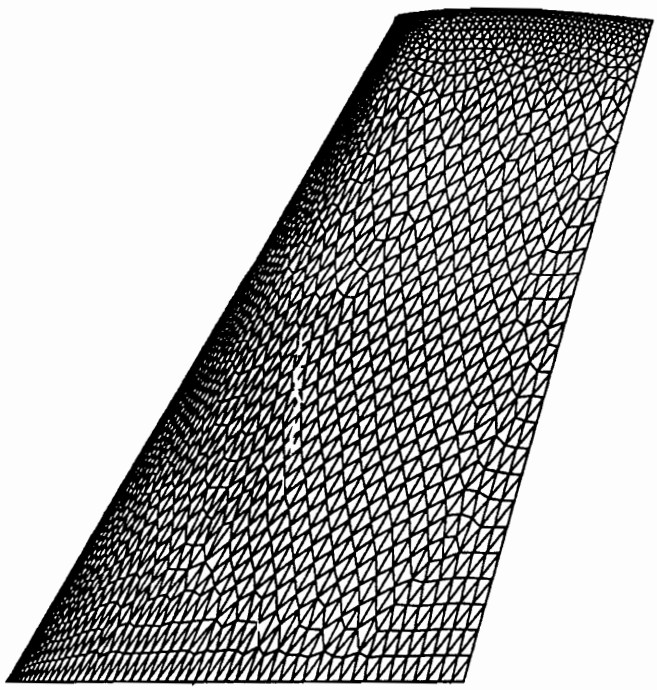
The static pressure contours for the suction and pressure side of the wing are shown in Figure 7.5. The contour levels indicate the relative shock strengths over the top portion of the wing. The increased resolution of the shock structure for grid.2 is apparent in the contour distribution.

Pressure coefficient results at 7 spanwise stations are provided in Figure 7.6. The experimental data is presented at a Reynolds number of 11.7 million. The properties are linearly interpolated to the data locations. The results on the fine mesh agree more favorably with experiment, however the basic Cp distribution is well captured on the

coarse grid. The lack of agreement at the 80% span location is common among inviscid solutions using this level of mesh spacing. Other researchers including Frink et al. [17] and Barth et al. [32] have validated their unstructured Euler solvers using this test case and show similar solution trends.



a – Grid.1



b – Grid.2

Fig. 7.1 – Computational grids for ONERA M6 wing.

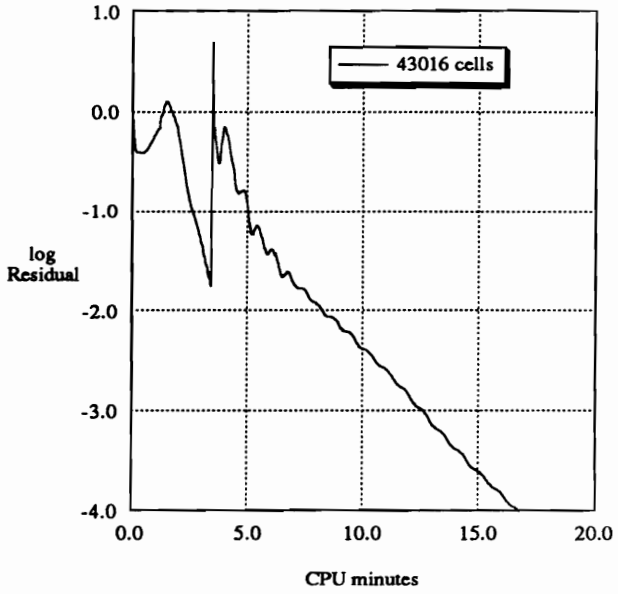
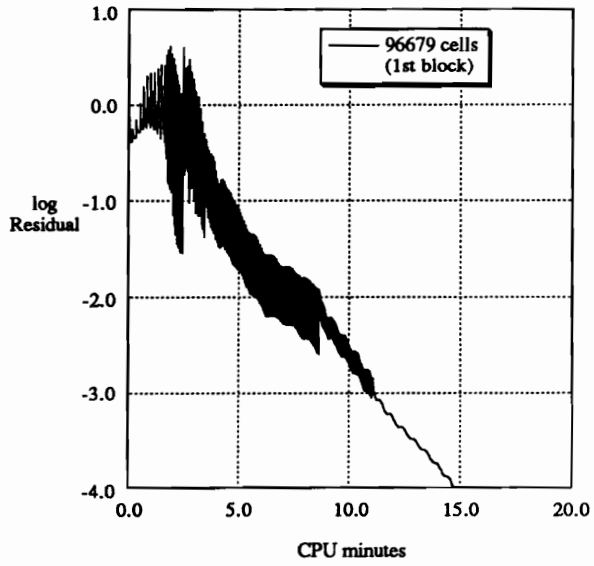
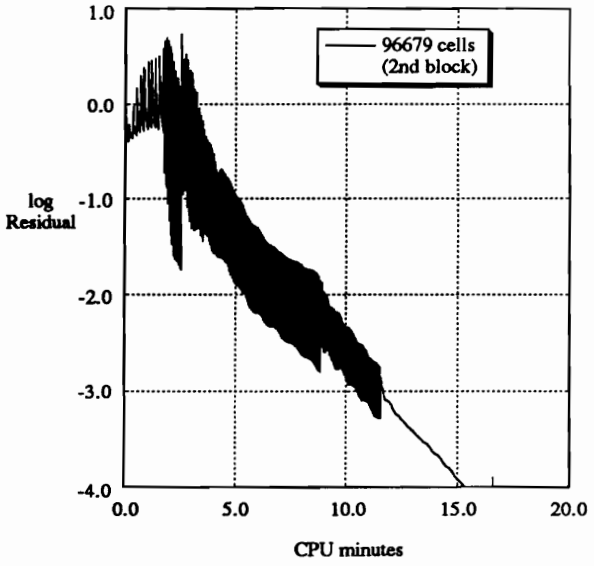


Fig. 7.2 – Solution histories for ONERA M6 wing; grid.1.

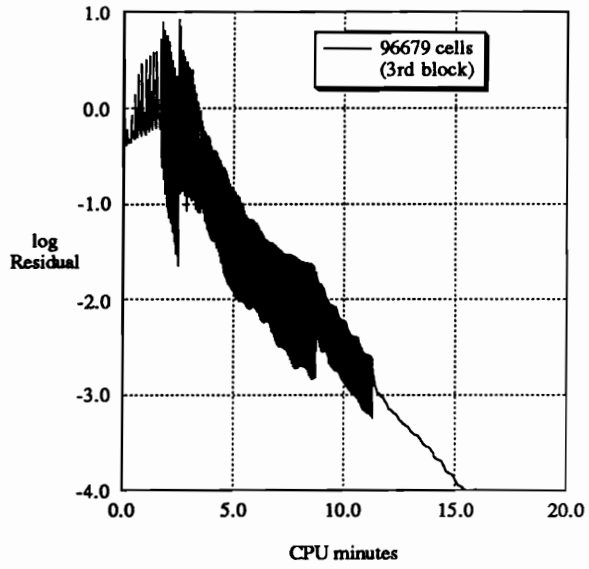


a – Block 1.

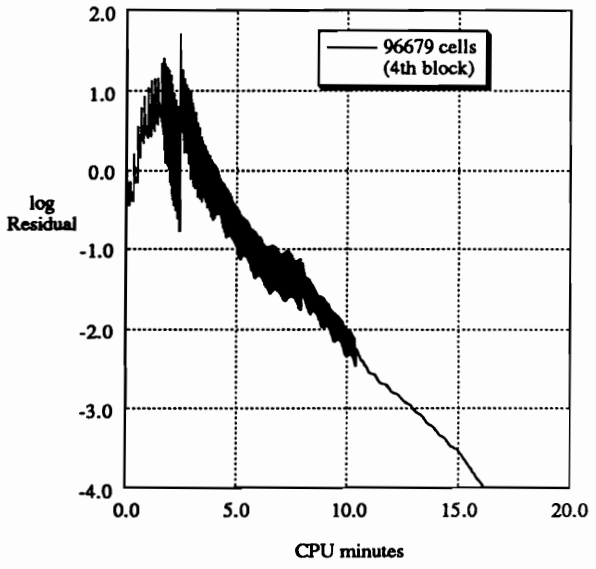


b – Block 2.

Fig. 7.3 – Solution histories for ONERA M6 wing; grid.2.



c – Block 3.



d – Block 4.

Fig. 7.3 – Solution histories for ONERA M6 wing; grid.2. (Continued).

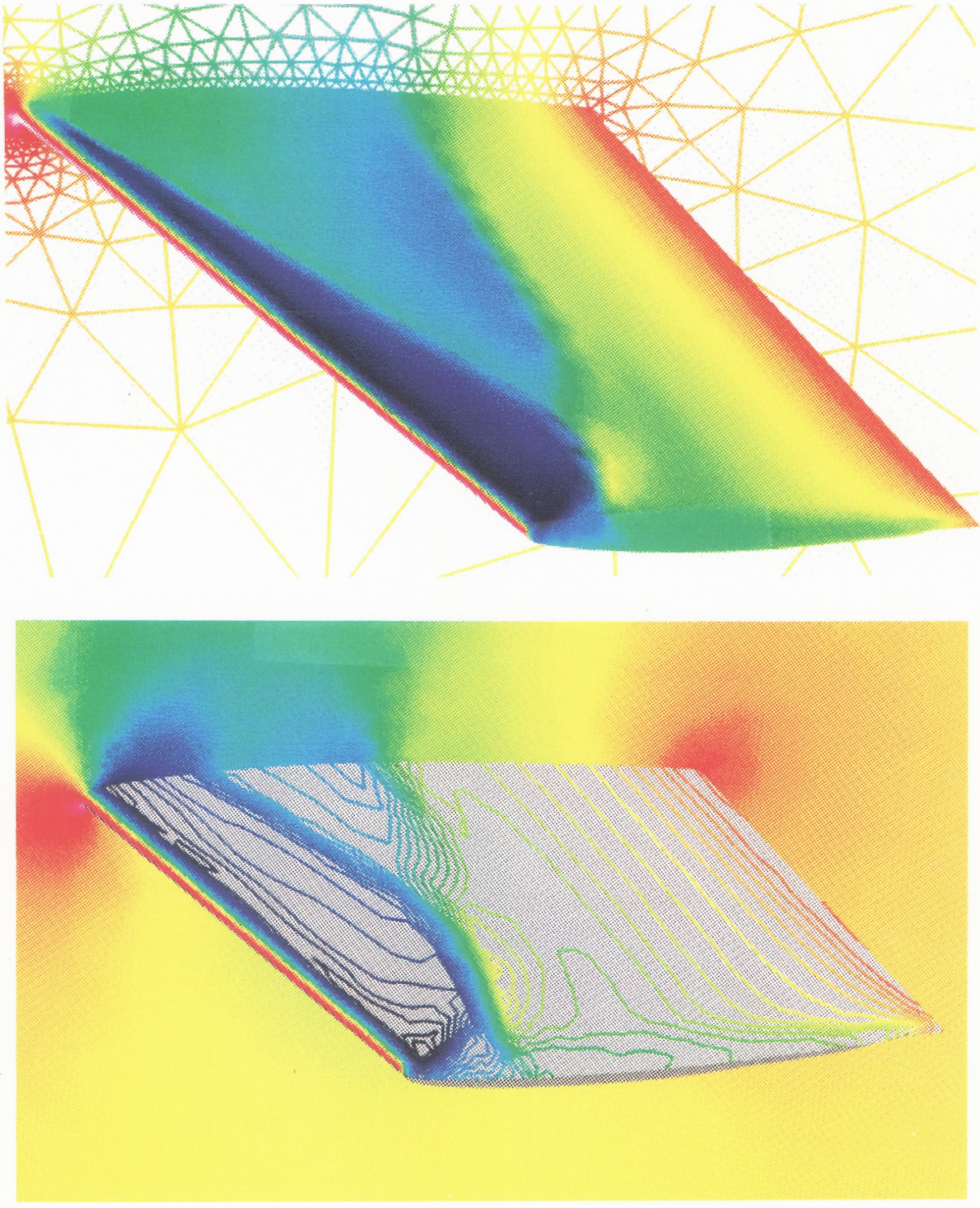
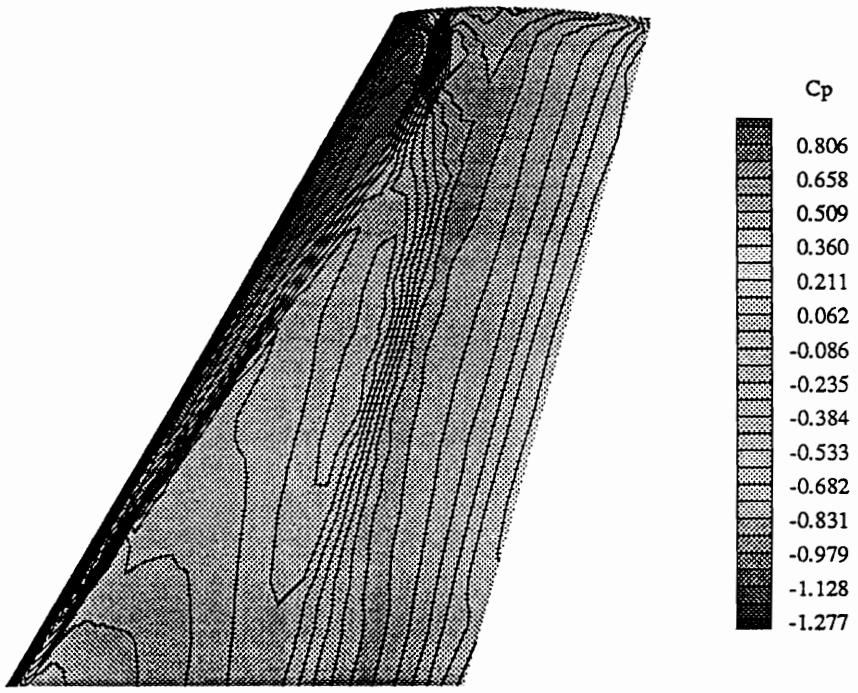
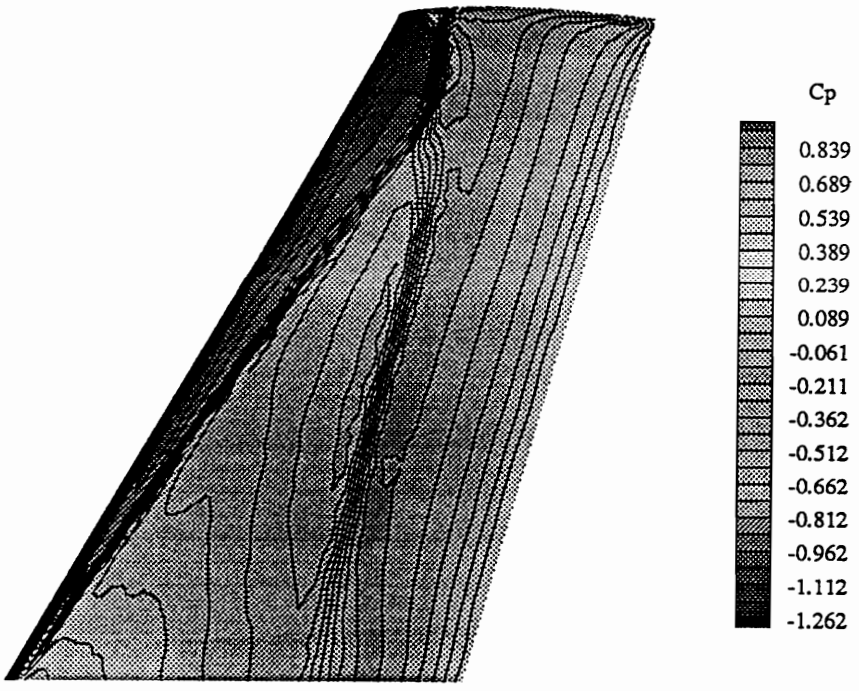


Fig. 7.4 – Static pressure contours for the ONERA M6 wing.

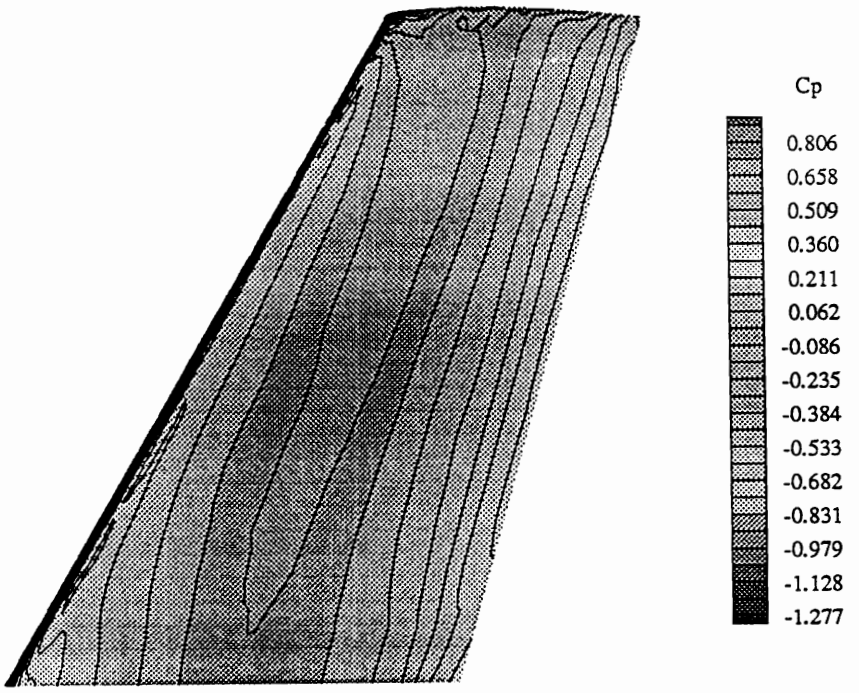


a – Grid.1

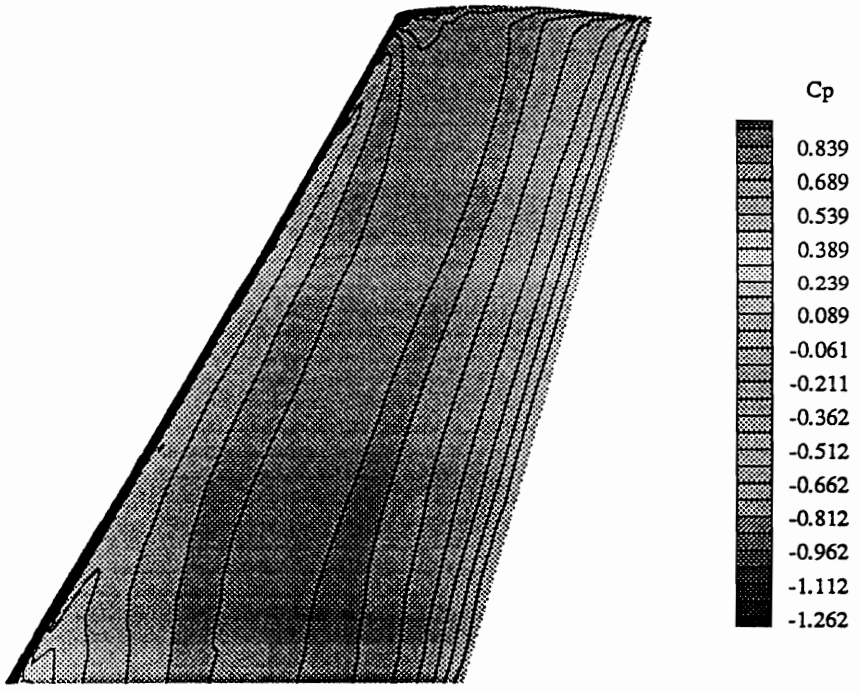


b – Grid.2

Fig. 7.5 – Suction side static pressure contours for the ONERA M6 wing.

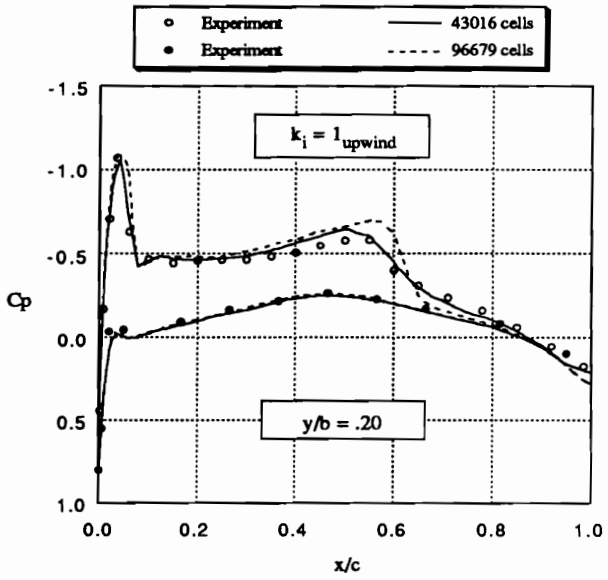


a – Grid.1

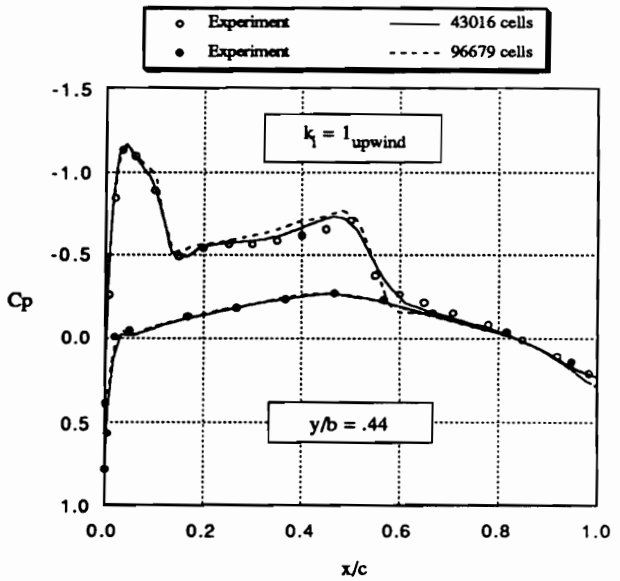


b – Grid.2

Fig. 7.5 – Pressure side static pressure contours for the ONERA M6 wing. (Continued)

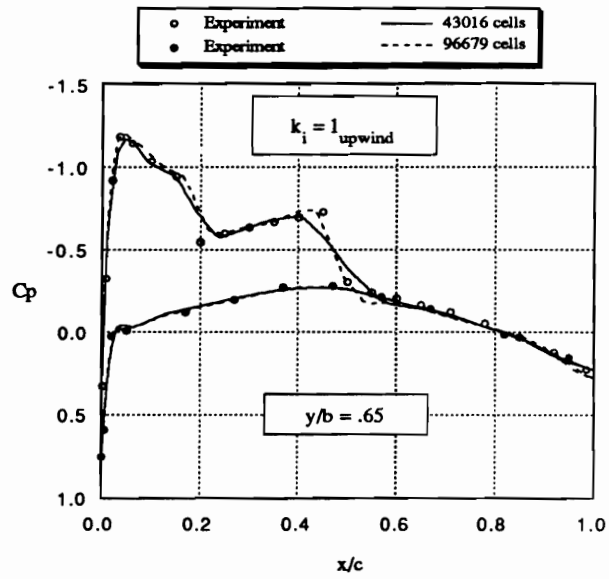


a – Station 1.

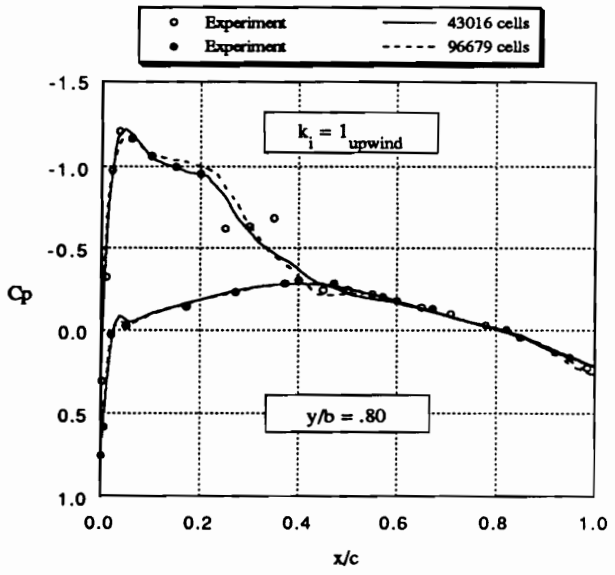


b – Station 2.

Fig. 7.6 – Pressure coefficient for the ONERA M6 wing, $M_\infty = .84$ and $\alpha = 3.06°$.

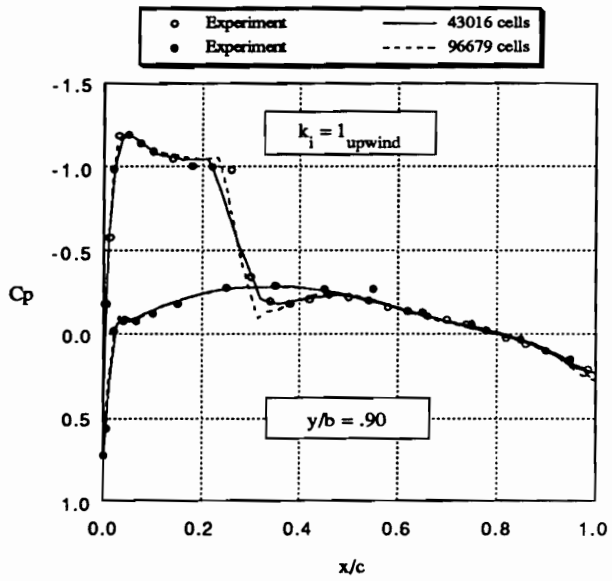


c – Station 3.

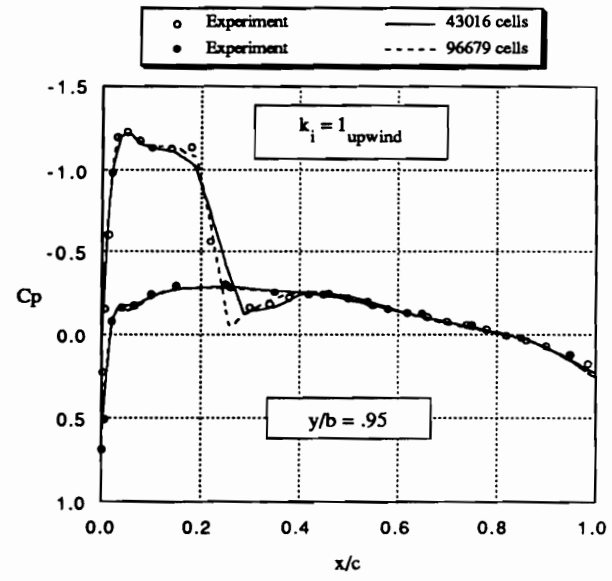


d – Station 4.

Fig. 7.6 – Pressure coefficient for the ONERA M6 wing, $M_\infty = .84$ and $\alpha = 3.06°$. (Continued).

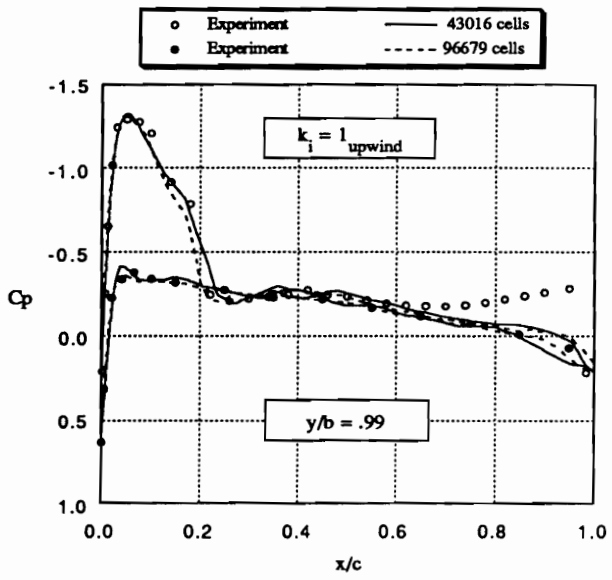


e – Station 5.



f – Station 6.

Fig. 7.6 – Pressure coefficient for the ONERA M6 wing, $M_\infty = .84$ and $\alpha = 3.06°$. (Continued).



g – Station 7.

Fig. 7.6 – Pressure coefficient for the ONERA M6 wing, $M_\infty = .84$ and $\alpha = 3.06°$. (Continued).

Chapter 8

Reconstruction For Adaptive Remeshing Algorithms

8.1 Introduction

Common to adaptive mesh methods is the principle of improving the flow field resolution by strategically refining and distributing the grid points within the domain. Geometrical complexity, as well as flow field complexity, must be considered in the discretization process. The geometrical features such as high curvature regions of the body can be adequately defined in the initial mesh or alternatively during the refinement process. Because the physical flow features are not available apriori, the characteristics of the initial internal mesh are usually uniformly spaced and stretched over the entire domain. As the flow patterns develop, the solutions are analyzed exposing regions in the domain that warrant improved resolution. By using adaptive mesh procedures, these important flowfield features, such as shock waves, can usually be resolved with a high degree of accuracy.

In general, there are three types of grid adaptation methods. These methods are categorized as either mesh movement, mesh enrichment, or adaptive remeshing. Mesh movement algorithms maintain a constant number of grid points and redistribute the nodes into regions of interest. This is accomplished by replacing the mesh edges with a spring of a certain stiffness and moving the nodal coordinates until the spring system is in equilibrium. As a result, nodal connectivity is preserved thus making the technique easy to incorporate into existing flow solvers. One draw back of this approach is the potential for an inadequate initial grid structure which results in highly skewed grids and limits the

final grid resolution. Alternatively, mesh enrichment schemes introduce more degrees of freedom into the system by introducing new points to the existing domain in regions of the flow where added resolution is required. Nodes are added in the domain by simply sub-dividing the computational cells which are marked for refinement [21]. The primary difficulty in adopting this approach is the need for more sophisticated data structures to store the connectivity information. Adaptive remeshing procedures compute a new mesh distribution over the entire computational domain. The distribution parameters include both cell size and cell stretching ratios and directions. Utilizing this information, the mesh is regenerated according to the prescribed specifications.

The use of adaptive remeshing techniques for the Euler equations has been successfully demonstrated [14] [47] [48] [49] [50] [51] [52] . Fundamental to the success of the methods is the establishment of an adaptation criterion that identifies various flow features. The classical approach constructs an error estimator by selecting a single property and forming its second order derivative tensor. Principle directions are identified and all scaling information is computed assuming a uniform value of the error over the entire domain. Paillere, Powell and Zeeuw [48] implemented a wave-model-based refinement criteria that detects regions of compressibility and regions of rotationality. Their efforts were a result of examining the flow features common to inviscid compressible flows. Hetu and Pelletier [52] introduced an adaptive remeshing strategy for solving complex viscous incompressible flows. One of the error estimators presented in their work was sensitive to shear layers and regions of high strain. In addition to selecting the appropriate remeshing functions, the issue of numerically modeling these terms on unstructured, and potentially highly stretched grids cannot be overlooked. Finally, the numerical implementation must be robust, efficient and accurate.

8.2 Error Estimators

The key to success in all adaptive mesh methods lies in the performance of the error estimators. Two classical approaches currently used in the computational fluid dynamics community either adapt to error estimates or to the physical flow features, i.e. gradients. Both of these methods have inherent difficulties and limitations. The most commonly used direct error estimator follows the work of Ref. [48] and is presented below.

$$| \tilde{e}_i | = h_i{}^2 \left(\left| \frac{\partial^2 q}{\partial x^2} \right| + \left| \frac{\partial^2 q}{\partial x \partial y} \right| + \left| \frac{\partial^2 q}{\partial y^2} \right| \right)$$

As can be seen, the error $\tilde{e}_i$ is based on a cell length parameter, h_i, and the second derivative values for property q. The most commonly selected property is density. A common adaptation parameter that indicates trends in solution errors based on physical flow features is

$$| \tilde{e}_i | = | \Delta q_i |$$

where $\tilde{e}_i$ is the error and Δq_i represents an undivided difference for property q.

The issue of the "optimum" arrangement of cells within the domain remains a very puzzling question. Once the error estimators have been utilized to determine the local cell errors, a criteria is necessary to ascertain which cells require continued refinement. The CFD community appears to be divided between two popular approaches. The first method calculates an error estimate for each cell in the domain using one of the two equations above or a simple variation of them. The errors are scaled from zero to one and compared against a predetermined target value. All cells which equal or exceed the threshold are flagged for further refinement. While this method is fairly easy to implement, it requires significant user interaction to determine the target value needed to resolve the flow physics. The second method is based on an equal cell error distribution. Typically, this approach is implemented by establishing a target error and flagging all cells whose error is within one standard deviation of this value.

The adaptive remeshing procedure is usually defined by the following sequence of steps:

1. Establish a grid refinement criteria.
2. Generate a steady state solution.
3. Calculate error estimates for every cell in the domain using an error indicator.
4. Calculate the remeshing parameters at the nodal points based on the prescribed target error.
5. Repeat steps 2 through 4 until the grid refinement criteria is satisfied.

Common to both methods is the concept of improving the quality of the computed solutions; however as the flow fields become increasingly complicated, more sophisticated remeshing functions are necessary. In fact, several researchers [52] have adopted the philosophy of identifying separate remeshing functions to detect key features in a flow using the same error estimators described above . While this may be a viable alternative for remeshing complicated flows, the fundamental issue of accurate error estimators is still critical to the success of reliable remeshing procedures. Without a thorough understanding of the flow physics and the numerical foundation of the error estimators relative to the physics of the solutions, erroneous results can occur. Warren [53] demonstrates inaccurate solution predictions using the above error estimators which indicates that the strategies are not adequate. He offers a modification to the error estimators which improves the performance and assesses his results by comparing to structured grid converged results. Comparisons of this nature are vitally important as they represent true tests of solution accuracy; however there is still no mathematical foundation that supports the validity of the error indicators. It is this principle that we focus on when applying reconstruction principles to obtain a more complete understanding of the behavior of solution errors.

8.3 The Role of Reconstruction

Developing procedures that attempt to reveal a relationship between the solution error and the computational domain represents the majority of the research effort in remeshing algorithms. However to date, these estimators lack rigorous validation. It is acknowledged that the methods in many cases have justifiable foundation but quantitative proof is simply not available. As a result, it is certainly conceivable that these methods described above do not actually model the solution errors or even behave as the real errors. The role of the reconstruction process, specifically the fundamentals of reconstruction accuracy, will serve as an analysis tool in determining the functional behavior of solution errors.

The two key features that error estimators should possess are:

1. The error estimator must behave as the true error, i.e. the difference between the exact error and the approximate error decreases during mesh refinement.
2. The numerical approximation for the error should be computed accurately regardless of mesh distortion and skewness.

8.4 A New First Order Error Estimator

A definition for the reconstruction accuracy and numerical error was provided in Chapter 3 and demonstrated in Chapter 4. The integral expression for the error norms were calculated using highly accurate Gauss quadrature and revealed solution accuracies as high as third order. This success is due entirely to the precise reconstruction process that was formally presented in Chapter 4. Using the definition for the L_p norm from Chapter 3, the solution error was calculated according to

$$L_p = \|F(exact) - F(approximate)\|_p \equiv \left[\int_\Omega |F(exact) - F(approximate)|^p \; da\right]^{\frac{1}{p}}.$$

Recall that the reconstruction accuracy results in Chapter 4 revealed that using polynomial approximations of degree k, the solution accuracy was $0(k+1)$.

Note that we are now taking a different viewpoint than truncation error estimates in classic finite difference schemes in which the derivatives of the partial differential equation are modeled with finite differences of a given order of accuracy. In the present work, we are concerned with the true error, i.e. the difference between the exact solution and our numerical approximation integrated over the domain. The pointwise value of the numerical approximation comes from the reconstruction process applied to the cell averaged data following a time integration algorithm.

The development of a first order error estimator begins by examining the error norm L_p. In two dimensions, the L_p norm is $O(\Omega \Delta s^n)$ for an n^{th} order accurate solution. The length scale Δs is an indicator of the mesh spacing and is $O(\Delta x, \Delta y)$. In Chapter 4, a $k = 0$ reconstruction was demonstrated to be first order accurate. In terms of the error norm

$$L_p = O(\Omega \Delta s) = O(\Delta x, \Delta y). \tag{8.1}$$

Recall that in terms of a two dimensional Taylor series, the first order truncation error for a function $U(x + \Delta x, y + \Delta y)$ expanded about $U(x, y)$ is

$$T.E. = \frac{\partial U}{\partial x}\Delta x \; + \; \frac{\partial U}{\partial y}\Delta y.$$

If we draw an analogy between the behavior of this pointwise error which is $0(\Delta x, \Delta y)$ and if we assume equilateral triangular meshes where $\Delta x \approx \Delta y$, then

$$T.E. \approx \Big(\frac{\partial U}{\partial x} \; + \; \frac{\partial U}{\partial y}\Big)\Delta x. \tag{8.2}$$

Relating Equation 8.1 to Equation 8.2 reveals a functional form for an error estimator that behaves qualitatively consistent with the solution errors. Specifically, the new error estimator is

$$\tilde{e}_i = \Big(\Big|\frac{\partial Q}{\partial x}\Big| \; + \; \Big|\frac{\partial Q}{\partial y}\Big|\Big)\Delta s \tag{8.3}$$

where Q represents a fluid property.

8.5 Validation of Error Estimator

Two test cases will be investigated to validate the first order error estimator. An exact steady state compressible solution in the absence of discontinuities is necessary to compare the exact solution errors to the estimated errors. This problem will also indicate the overall performance based on the key features outlines above. For these reasons, Ringleb's flow [36] is selected. Secondly, the remeshing scheme will be challenged by a 10 degree high speed ramp that exhibits both shock waves and expansion fans.

Ringleb's Flow

The first order accurate solutions for a $k = 0$ reconstruction were documented in Chapter 4 for this test case. The L_1 norm indicated a slope of 1 on a log-log plot which is consistent for first order accurate schemes. The first test for the error estimator is to compare the estimated error to the exact error. The numerical procedure for calculating the estimated error defined by Equation 8.3 is itemized as follows:

1. Reconstruct the gradient expressions in Equation 8.3 by performing a $k = 1$ reconstruction for each face in the domain using an upwind stencil. The resulting polynomial is represented as

$$Q(x, y) = c_o + c_1 x + c_2 y$$

and by simply differentiating the polynomial we conclude that

$$\frac{\partial Q}{\partial x} = c_1 \qquad \frac{\partial Q}{\partial y} = c_2.$$

The gradient expressions are arithmetically averaged for each cell.

2. The length scale Δs is the length from the cell centroid to the vertex point and was given in Chapter 3 as

$$\Delta s \approx \sqrt{\frac{\Omega}{1.3}}$$

where Ω is the area of the equilateral triangle.

3. The errors for each cell in the domain can now be calculated according to Equation 8.3. The total error is the sum of individual cell errors.

A comparison between the exact error and the computed error is provided in Figure 8.1. The first plot reveals that the approximate error approaches the exact error during mesh refinement. Part *b* for this figure is graphed on a log-log axis to compare the accuracy. Recall that the slope of this line determines the solution accuracy. As can be seen, the curves have nearly equal slopes. The offsets in the curves are due to several reasons including approximation of the gradient, scaling based on volume, and neglecting higher order terms. However, the results of the analysis provide the numerical evidence that this new error estimator behaves as the true error.

Implementing this error estimator in a remeshing procedure requires the selection of a cell remeshing criteria. For all of the following results, we adopt the philosophy of an equal distribution of error throughout the computational domain based on a user-specified cell target error E_t. The remeshing algorithm determines the cell size by comparing the target error to the calculated (estimated) error according to

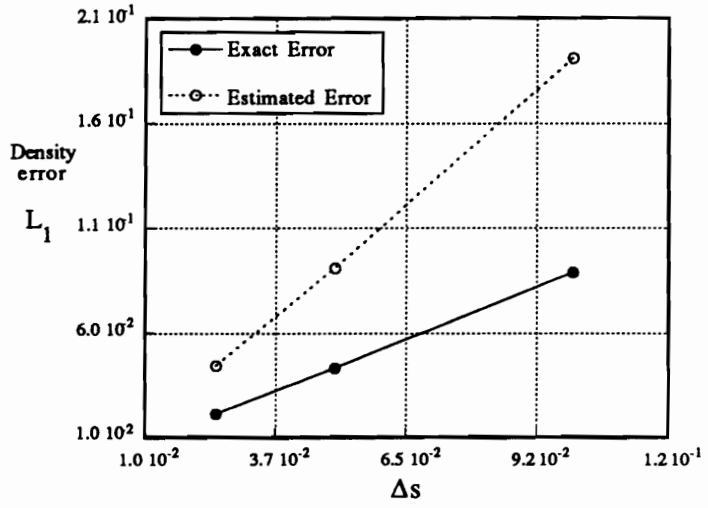
$$\Delta s_{new} = \frac{E_t}{\left| \frac{\partial U}{\partial x} \right| + \left| \frac{\partial U}{\partial y} \right|}$$

or

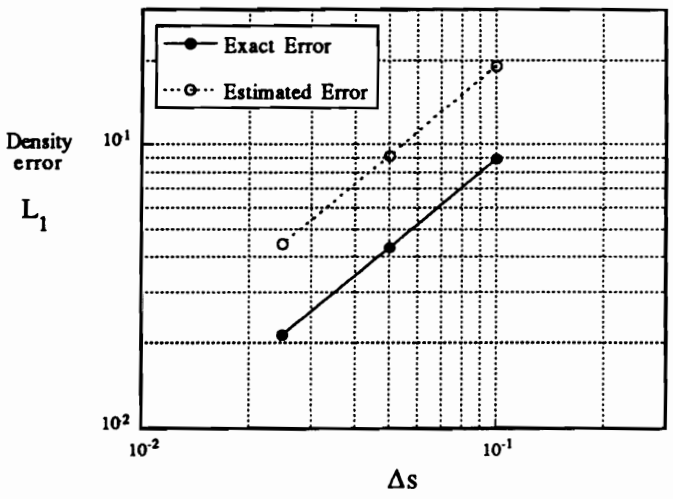
$$\Delta s_{new} = \frac{E_t}{\tilde{e}} \Delta s \qquad 8.4$$

As a note, the value of Δs_{new} is the radius of a circle that circumscribes the equilateral triangle. This quantity is directly used in the advancing front grid generator.

Figure 8.2 shows the results of the adaptive remeshing algorithm for 3 different target errors. The coarse mesh shown in Figure 4.15 served as the initial discretization, and the remeshing process was fully automated to perform any number of remeshing cycles without user intervention. As can be witnessed in the figures, the remeshing cycles indicate that the high curvature regions exhibit the maximum solution errors and as a



a – Error trend



b – Accuracy comparison

Fig. 8.1 – Comparison of the exact error to the estimated error

result, the mesh distribution is uniformly concentrated in this area. All solutions were interpolated onto the improved meshes resulting in an efficient remeshing process. A constraint was enforced during the cycle that limited the maximum change in Δs to

$$.5 \leq \Delta s \leq 2.0,$$

however this did not affect the geometrical features of the final mesh. A summary of the grid convergence for a typical remeshing cycle is shown in Figure 8.3. The remeshing history using the exact error $\tilde{e}_{exact}$ in Equation 8.4 is shown for comparison. Notice that after 7 remeshes, grid convergence had been achieved using the approximate method and that the differences between the $\tilde{e}_{estimated}$ and $\tilde{e}_{exact}$ are consistent with the results in Figure 8.1. Specifically, the exact error is numerically smaller than the approximate method for a given grid size thus the target error is achieved with fewer cells.

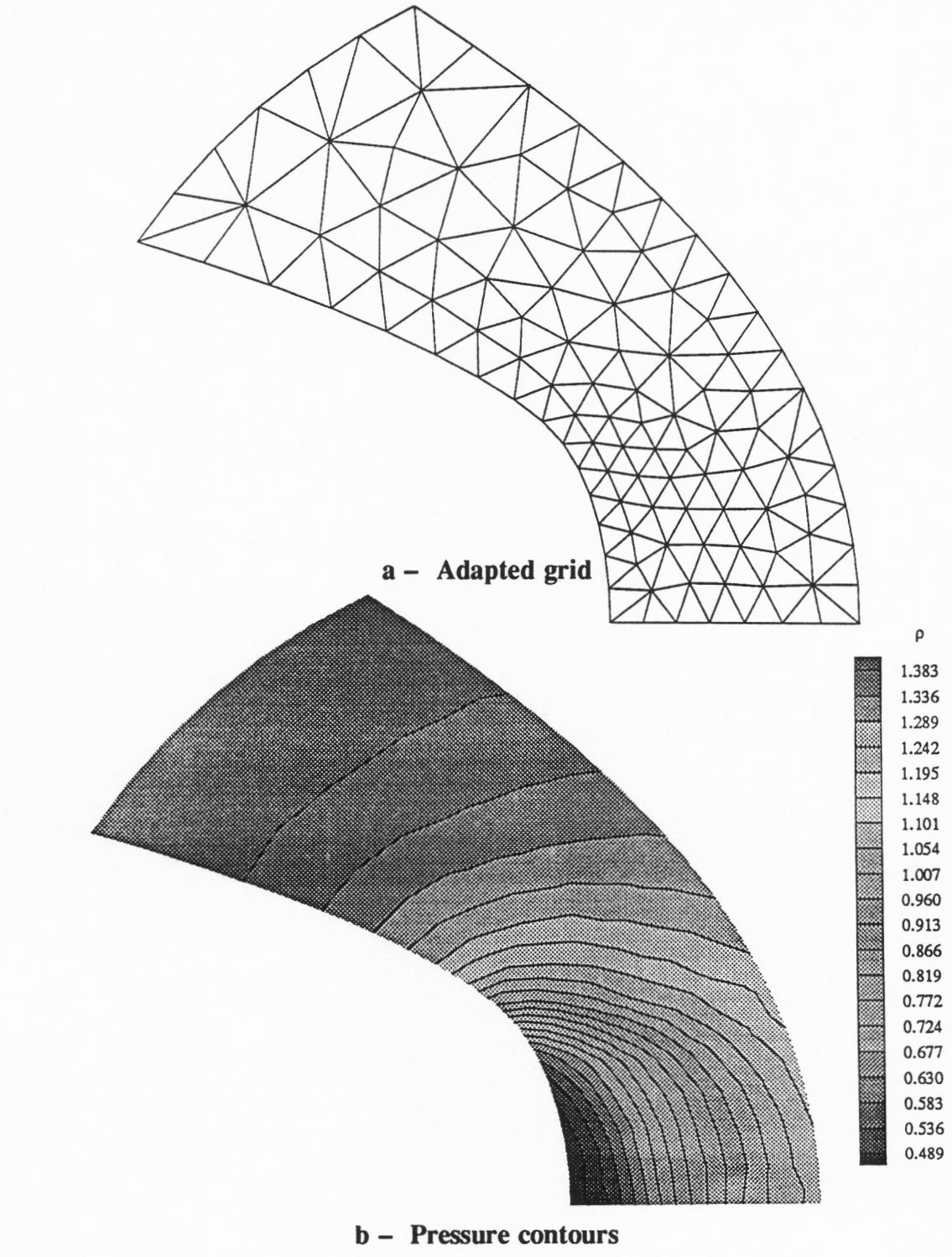


Fig. 8.2 – Adapted grid and pressure contours for Ringleb's flow, $k = 0$, Target error = $1.0e-03$

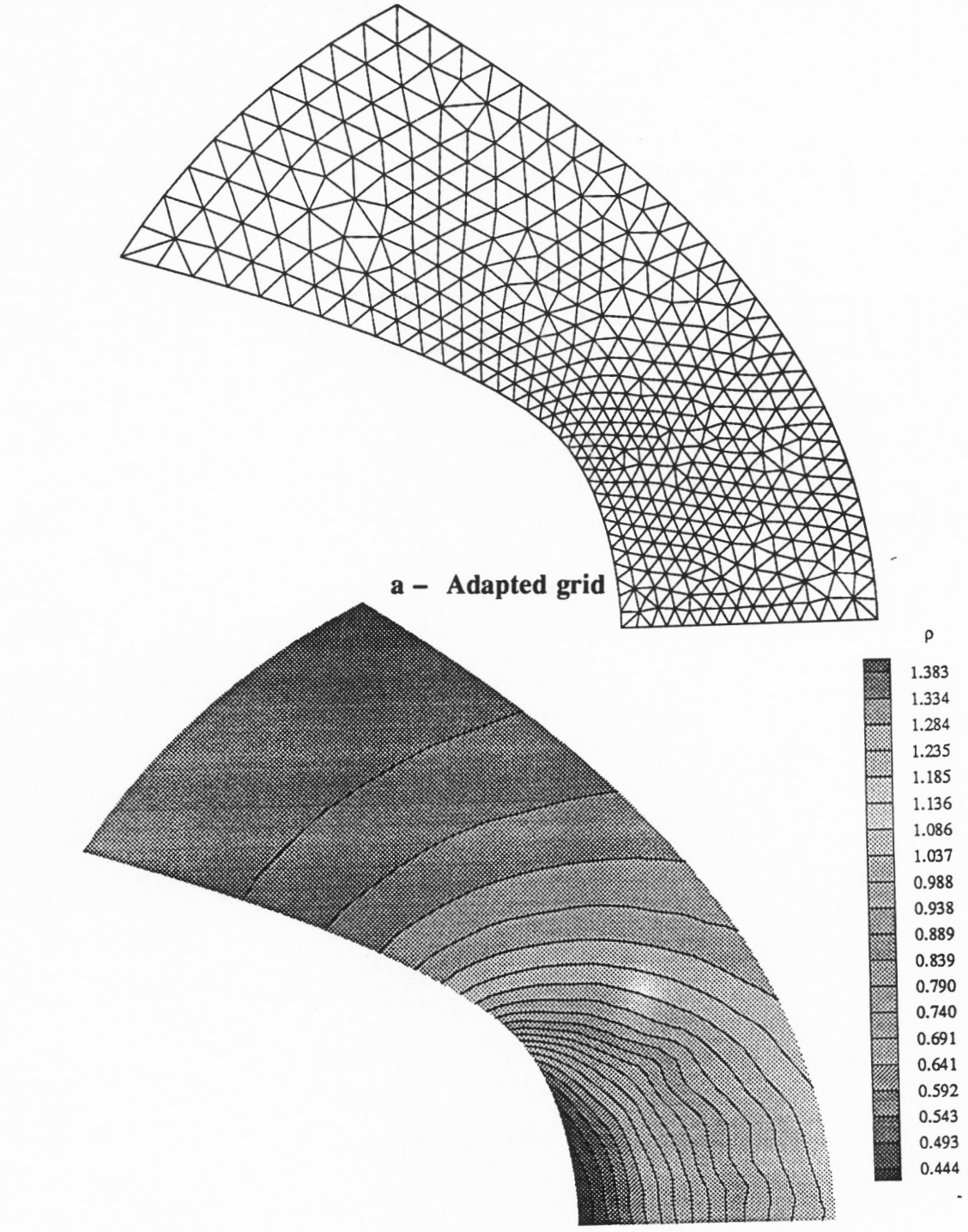


Fig. 8.2 – Adapted grid and pressure contours for Ringleb's flow, $k = 0$, Target error = $1.0e-04$ (Continued).

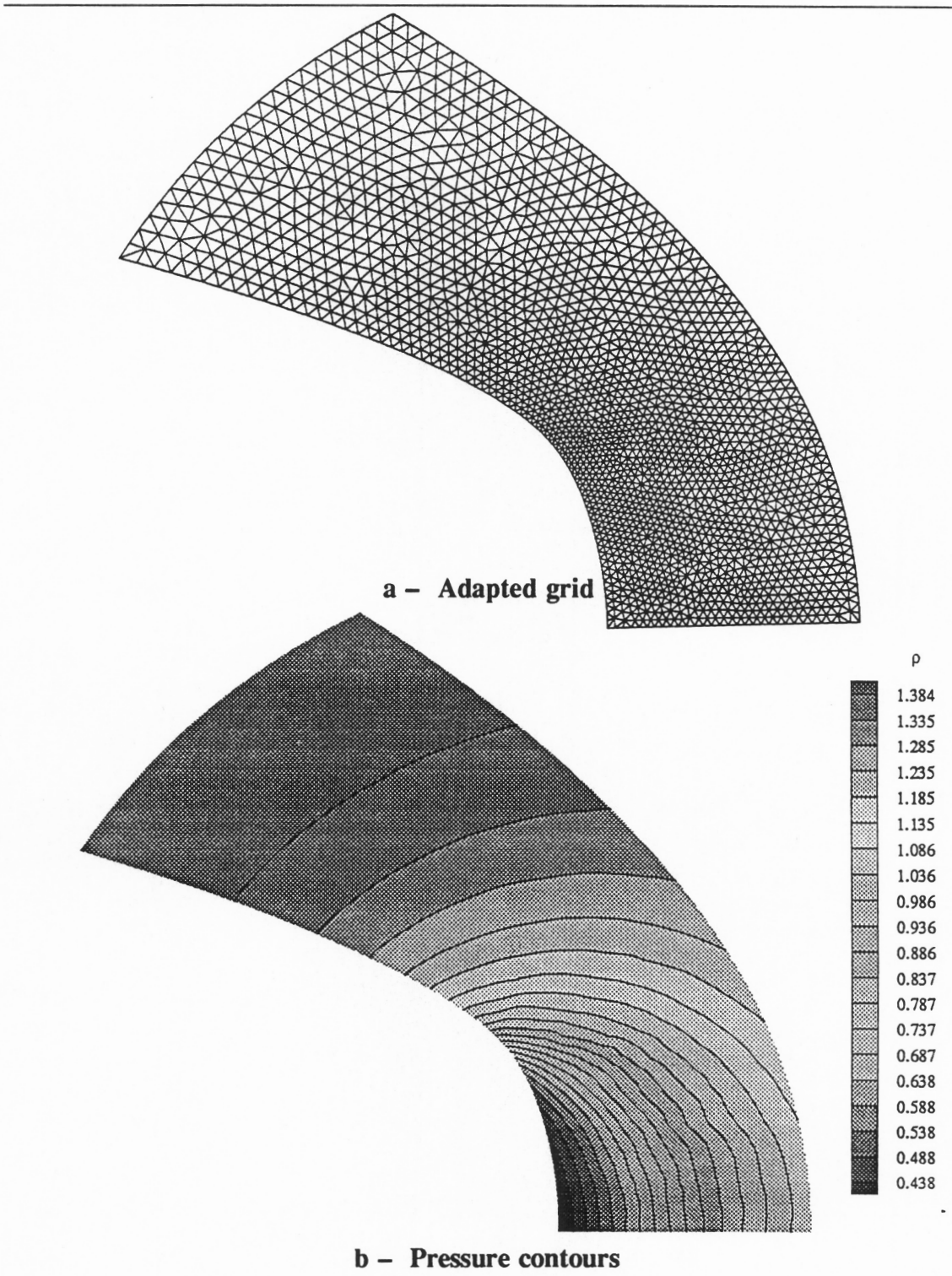


Fig. 8.2 – Adapted grid and pressure contours for Ringleb's flow, $k = 0$, Target error = $1.0e-05$ (Continued).

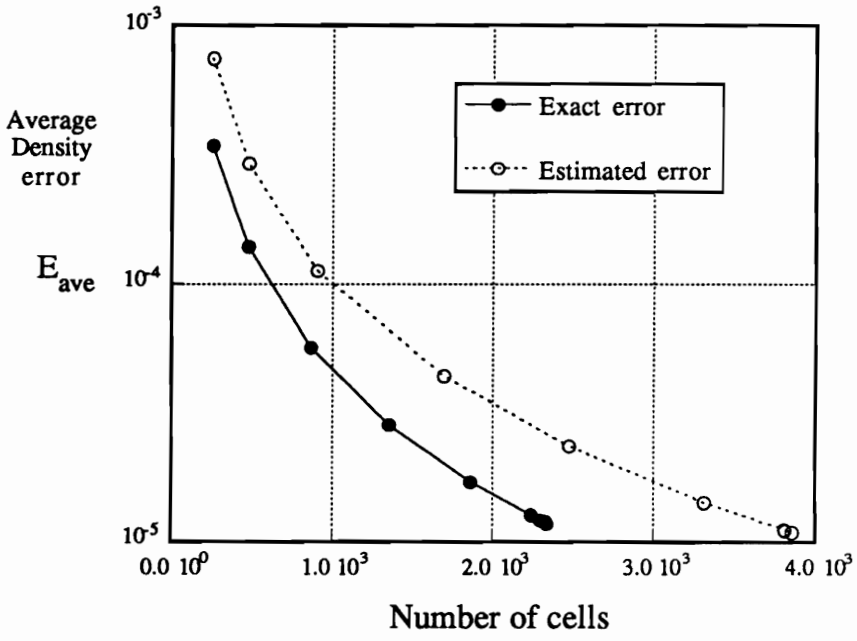


Fig. 8.3 – Grid convergence summary for Ringleb's flow, $k = 0$, **Target error** = $1.0e - 05$

10 Degree Ramp

The second test case is Mach 5 flow over a 10 degree ramp. Three target errors were selected to assess the performance of the remeshing algorithm in the presence of discontinuities. The resulting adapted meshes are provided in Figure 8.4 and reveal the relative level of solution error between the shock wave and the expansion fan. The peak pressure ratio relative to the free stream pressure increases from 2.938 in the initial mesh to 4.060 on the finest mesh. A comparison of the pressure on the lower and upper walls is shown in Figure 8.5. The improvment in solution quality during refinement is evident. The maximum errors consistently occur at the initial compression point on the ramp due to turning the uniform flow parallel to the ramp wall. The maximum error through the expansion fan occurs on the ramp wall as well. The resolution of the shock wave/expansion fan interaction is most clearly resolved on the finest grid.

The importance of these results lie within the relative error levels associated with shock waves and expansion fans. Remeshing strategies which are based strictly on gradient quantities require the correct scaling levels to force grid refinement in the shock free regions of the flow. Consequently, the relationship between the errors associated with shock waves as compared to expansion fans was not realized. The new first order error estimator used in this analysis reveals the relative errors between the two flow pattern types.

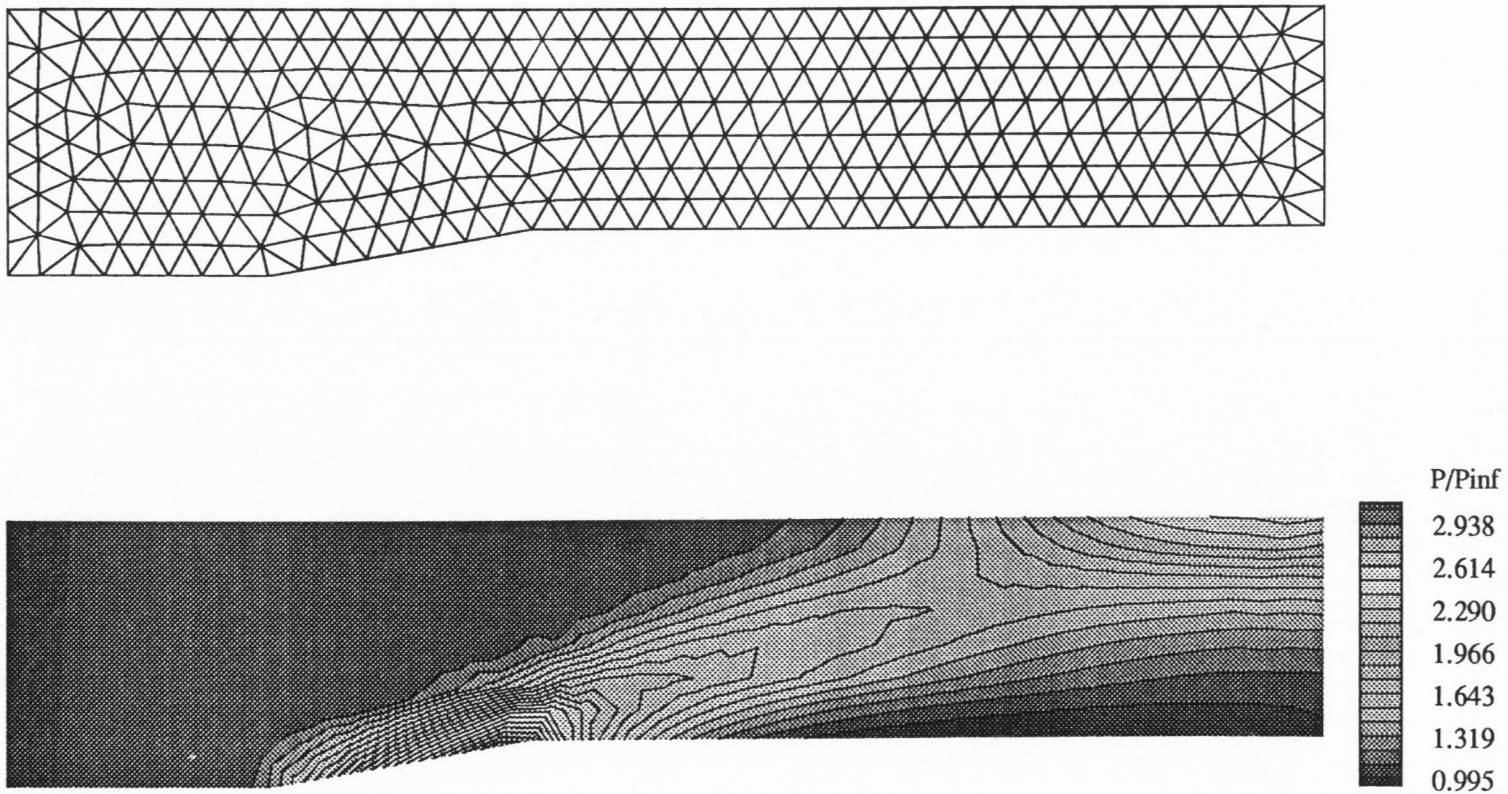


Fig. 8.4 – Original grid and pressure contours for 10 degree Ramp, $k = 0$, 555 cells

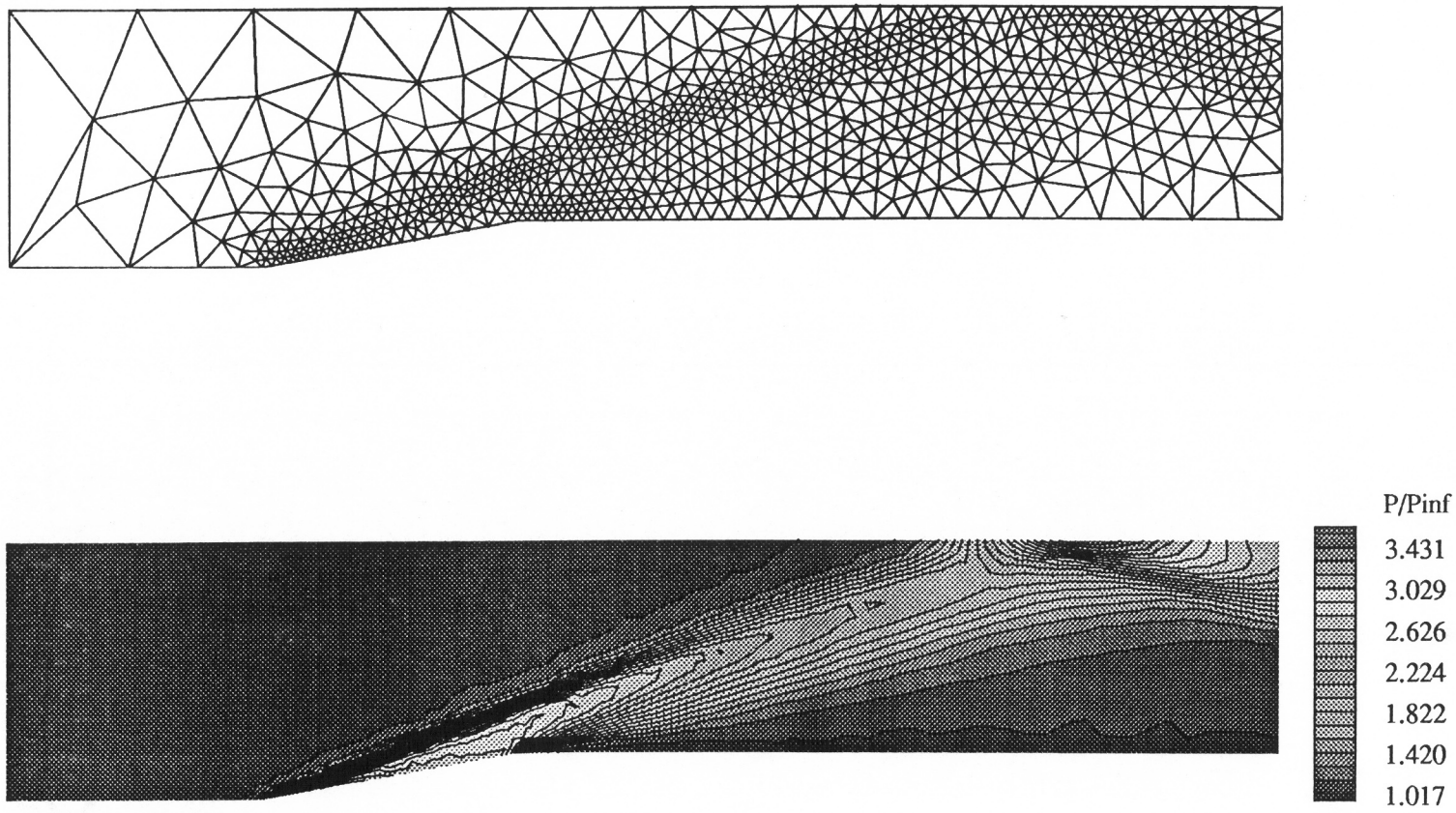


Fig. 8.4 – Adapted grid and pressure contours for 10 **degree Ramp,** $k = 0$, 1599 **cells, Target error** = $1.0e-07$ **(Continued).**

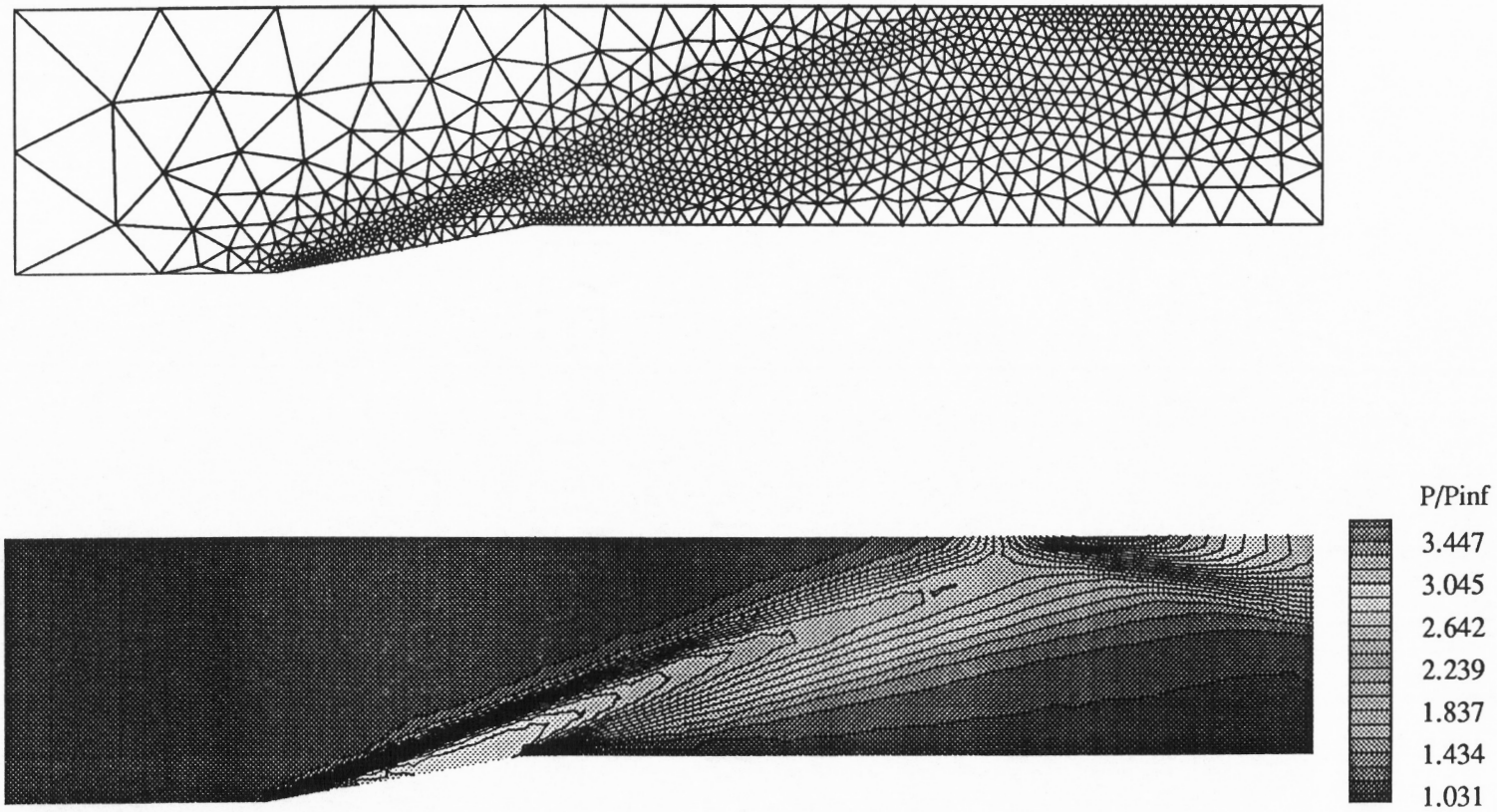


Fig. 8.4 – Adapted grid and pressure contours for 10 degree Ramp, $k = 0$, 2274 cells, Target error = $5.0e-08$ (Continued).

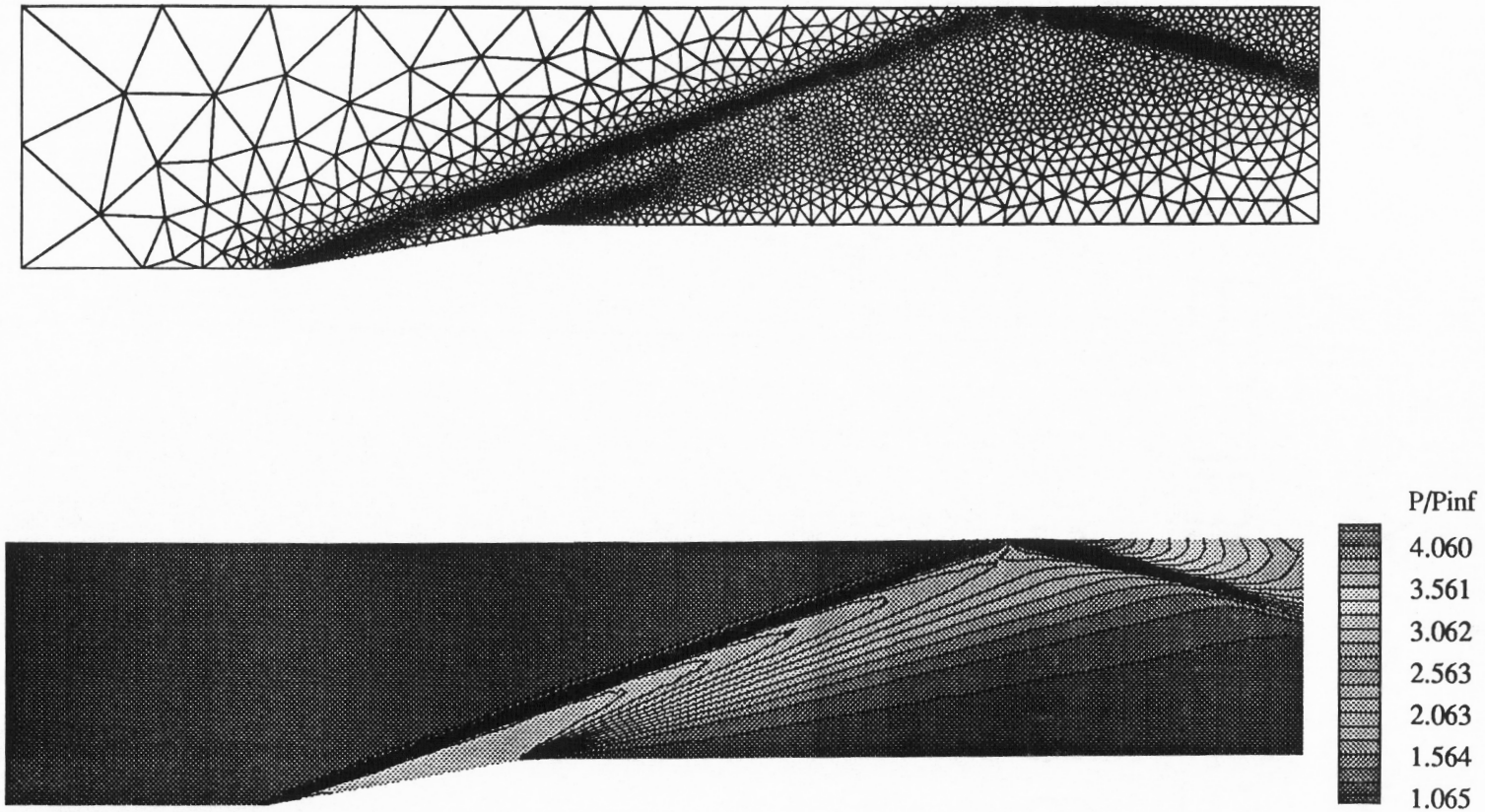
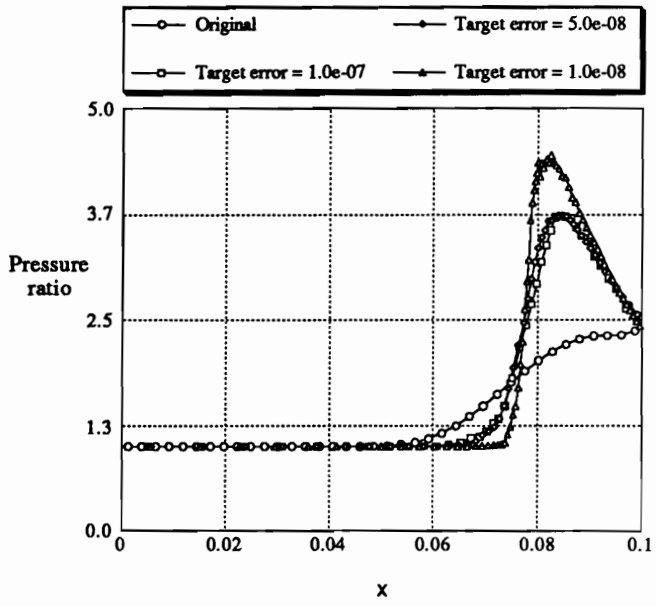
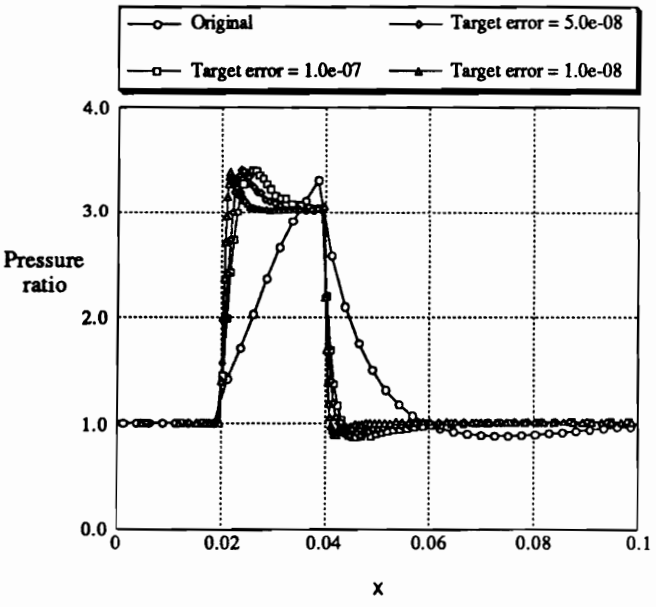


Fig. 8.4 – Adapted grid and pressure contours for 10 degree Ramp, $k = 0$, 6330 cells, Target error = $1.0e-08$ (Continued).



a – Top wall



b – Bottom wall

Fig. 8.5 – Wall pressure comparison for 10 degree Ramp, $k = 0$

Chapter 9

Concluding Remarks

A new multidimensional reconstruction algorithm has been successfully developed and validated for solving the Navier Stokes equations for arbitrary discretizations. The formulation is based on multidimensional reconstruction techniques designed for general grid topologies. The method is a direct extension and generalization of proven structured flow solver algorithms, whereby the pointwise spatial variation of the flow properties are reconstructed for each face in the domain using cell averages. The reconstruction accuracy was successfully assessed by evaluating the integral error norms for several test functions on triangular grid topologies. The results indicated that high accuracy is possible using the reconstruction procedure and that the accuracy for reconstructing polynomials of degree k is $0(k+1)$. The accuracy of the solution algorithm within a finite volume framework was assessed by comparing computed solutions with the analytical results for Ringleb's flow. The numerical experiments indicated that sufficiently accurate flux integration procedures are necessary in order to achieve the lowest levels of solution error, however the formal accuracy of the reconstruction step is not affected by this step. On the other hand, insufficiently accurate boundary condition treatment was shown to directly alter the global accuracy of the steady state solutions. These documented results represent the first analysis that formally quantifies the solution accuracy for the Euler equations on unstructured, two dimensional grids. In addition, a comparison of this unique approach with other linear reconstruction algorithms demonstrated the improved solution prediction capabilities.

The robustness of the method lies in the stencil selection flexibility. Several sten-

cil selection strategies have been introduced to demonstrate upwind and upwind-biased reconstruction capabilities. An upwinding algorithm was presented that was valid for general discretizations. Inviscid solutions for both smooth and discontinuous flows were shown. For flows with strong shock waves, a simple limiter was devised to control the oscillations. Results showed the effective control of the extrema as the overshoots were greatly reduced.

The primary concern for solving the complete Navier Stokes equations was the numerical approximation of the gradient expressions on highly distorted meshes. Researchers have claimed that there are geometrical limitations associated with triangles that have large interior angles, i.e. angles greater than 105°. In order to provide a viscous mesh capable of resolving the high gradients common in the boundary layer region subject to this cell shape restriction, the grid sizes would become uncontrollably large even in two dimensions. However, by using precise reconstruction and proper stencil selection, accurate viscous solutions on highly stretched two dimensional structured and unstructured meshes have been calculated with success. For the viscous test cases presented in this work, as many as 30% of the grid angles were greater that the presumed limitation of 105°, and 25% were greater than 150°. The gradient expressions present in the viscous flux vector, i.e. shear stress and heat flux terms, were successfully calculated using polynomial differentiation. The choice of a viscous stencil was based on a locality condition whereby the cell set is centered around the face in question.

Inviscid three-dimensional unstructured results using implicit time integration, domain decomposition, and a computationally efficient implementation of $k-exact$ reconstruction have been shown. Experimental data was included for comparison and revealed accurate prediction of the pressure coefficient for the ONERA M6 wing at various spanwise stations. The lack of agreement at 80% span is due to viscous effects not modeled in the Euler calculations.

The surface discretization procedure for a three dimensional advancing front grid generation algorithm was successfully modified and improved to allow bodies with high curvature. Iterative numerical procedures were designed and implemented to place the points on the body. These included a closest point algorithm for determining the values of the parametric variables u, v and a method that located the third point to form the base of a tetrahedron in unstretched coordinates. Surface refinement techniques including a spring analogy for moving points and a Delaunay criteria for switching faces resulted in improved grid distributions. Several examples demonstrating the effectiveness of the complete process were shown including a full aircraft configuration.

The role of reconstruction in adaptive remeshing algorithms was discussed and a first order error estimator was successfully developed and validated for discontinuous and shock-free flows. The fundamental idea of uniformly distributing the error within the domain was preserved. The development of the error estimator had both quantitative and qualitative foundation. Similar behavior in the estimated error and the exact error was observed for Ringleb's flow. These results represent the first analyses that verify the numerical consistencies between the approximate and exact error. As a result, the relative error levels common to shock waves and expansion fans were investigated and identified.

The writer believes that the results of this research indicate the need for continued research to improve the robustness and computational efficiency and memory requirements for both grid generation and flow solver algorithms for unstructured discretizations. The skepticism in the CFD community towards accurate viscous solutions on highly stretched unstructured grids should be somewhat softened as the results of this work indicate a capability in this area. In addition to the numerical efficiency and memory issues, the viscous capability should be extended and demonstrated in three dimensions. However, the extension should be straightforward, in that the reconstruction procedure has already

been applied to the three dimensional Euler equations. Thus, the only required action would involve a viscous stencil selection algorithm applicable in three dimensions. A formal extension of the structured ENO techniques to unstructured environments would be helpful. This task falls along the lines of new and improved moving stencil algorithms. Other items related to solution algorithms that warrant investigation include higher order boundary conditions using reconstruction and improved limiters for high speed flows.

Three dimensional advancing front grid generation research is critical in developing a realistic capability for analyzing the flow physics for complicated viscous flow fields. In particular, the most obvious weakness in the method lies in the inability to generate highly stretched three dimensional tetrahedral meshes for viscous applications. While the improvements in the background grid technology will certainly help to preserve grid smoothness and improve algorithm robustness, the performance of the algorithms that check for grid integrity during the generation process, i.e. face crossing, are susceptible to numerical roundoff errors for highly clustered cells. An alternative procedure for discretizing the viscous region which the writer feels warrants investigation is to discretize the dense boundary layer regions with prisms (semi-structured). One method of accomplishing this would be to simply project the triangular structured grid off the body and normal to the body an adequate distance to include the boundary layer height. Stretching functions can then distribute the triangular surfaces throughout this small volume. The interior grid could then be generated using the projected triangular surface as the initial boundary front. The benefits of this approach include a $\frac{2}{3}$ reduction in control volumes distributed though the boundary layer, since a prism can be divided into three tetrahedra. As a consequence, the grid sizes are greatly reduced.

This issue aside, there are still other areas that deserve continued attention. First of all, the surface grid improvement algorithms need to be applied to the interior mesh to improve the integrity of the interior domain. Due to the irregular geometrical patterns

that develop during the interior grid generation phase, visual inspection of grid quality is virtually impossible. Therefore, grid assessment models need to be established that can reveal weaknesses within the domain. The feasibility of implementing blocking strategies for generating grids conducive to space marching algorithms should also be investigated.

Continued research in the area of adaptive remeshing algorithms will become increasingly important as the grid generation capabilities mature. Moreover, it is felt that the source of the success in this area will be linked to numerically validating the performance of the error estimators by formally comparing the approximate error with an exact error. In addition, numerical procedures for accurately calculating the error terms become important for nonuniform meshes. Relative to this work, the first order estimator needs to be extended to include the effects of triangles which are not equilateral as well as being extended to three dimensions. In addition, a second order error estimator should be investigated that follows the same development. Finally, remeshing algorithms for viscous flows need to be studied.

REFERENCES

[1] Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors and Difference Schemes," *Journal of Computational Physics*, Vol. 43, pp. 357-372, 1981.

[2] Osher, S. and Solomon, F., "Upwind Differencing Schemes for Hyperbolic Conservation Laws," *Mathematics and Computation*, Vol. 38, pp. 339-374, 1982.

[3] Van Leer, B., "Flux-Vector Splitting for the Euler Equations," *Lecture Notes in Physics*, Vol. 170, pp. 507-512, 1982.

[4] Roe, P. L., "Characteristic Based Schemes for the Euler Equations," *Journal of Fluid Mechanics*, Vol. 18, pp. 337-365, 1986.

[5] Steger, J. L. and Warming, R. F., "Flux Vector Splitting of the Inviscid Gas-Dynamic Equations with Application to Finite-Difference Methods," *Journal of Computational Physics*, Vol. 40, pp. 263-293, 1986.

[6] Chakravarthy, S. R., "Some Aspects of Essentially Non-oscillatory (ENO) Formulations for the Euler Equations," *NASA Contractor Report 4285*, May, 1990.

[7] Harten, A. and Osher, S., "Uniformly High-Order Accurate Essentially Non-Oscillatory Schemes I.," *MRC Technical Summary Report No. 2823*, May, 1985.

[8] Harten, A., "On High-Order Accurate Interpolation for Non-Oscillatory Shock-Capturing Schemes," *MRC Technical Summary Report No. 2829*, June, 1985.

[9] Godfrey, A. G., Mitchell, C. R., and Walters, R. W., "Practical Aspects of Spatially High Accurate Methods," *AIAA 92-0054*, January, 1992.

[10] Pirzadeh, S., "Structured Background Grids for Generation of Unstructured Grids by Advancing Front Method," *AIAA 91-3233*, January, 1991.

[11] Pirzadeh, S., "Recent Progress n Unstructured Grid Generation," *AIAA 92-0445*, January, 1992.

[12] Barth, T. J., and Jespersen, D. C., "The Design and Application of Upwind Schemes on Unstructured Meshes," *AIAA 89-0366*, January, 1989.

[13] Barth, T. J., and Frederickson, P. O., "Higher Order Solution of the Euler Equations on Unstructured Grids Using Quadratic Reconstruction," *AIAA 90-0013*, January, 1990.

[14] Slack, D. C., Walters, R. W., and Lohner, R., "An interactive Adaptive Remeshing Algorithm for the Two-Dimensional Euler Equations," *AIAA 90-0331*, January, 1990.

[15] Whitaker, D. L., Slack, D. C., Walters, R. W., "Solution Algorithms for the Two-Dimensional Euler Equations on Unstructured Meshes," *AIAA 90-0697*, January, 1990.

[16] Jameson, A. and Mavriplis, D., "Finite Volume Solution of the Two-Dimensional Euler Equations on a Regular Triangular Mesh," *AIAA 85-0435*, January, 1985.

[17] Frink, N. T., Parikh, P., and Pirzadeh, S., "A Fast Upwind Solver for the Euler Equations on Three Dimensional Unstructured Meshes," *AIAA 91-0102*, January, 1991.

[18] Mavriplis, D. J.:, "Accurate Multigrid Solution of the Euler Equations on Unstructured and Adaptive Meshes.," AIAA Journal, Vol. 28, No. 2, Feb. 1990, pp. 213-221.

[19] Marcum, D. L., and Agarwal, R. K., "A Three Dimensional Finite Element Navier-Stokes Solver with k-e Turbulence Model for Unstructured Grids," *AIAA 90-1652*, January, 1990.

[20] Batina, J. T., "Three Dimensional Flux Split Euler Schemes Involving Unstructured Dynamic Meshes," *AIAA 90-1649*, January, 1990.

[21] Zhang, H., Trépanier, M., Reggio, M., and Camarero, R., "A Navier-Stokes Solver for Stretched Triangular Grids," *AIAA 92-0183*, January, 1992.

[22] Löhner, "FEM-FCT and Adaptive Refinement Schemes For Strongly Unsteady Flows," Proc. ASME Winter Annual Meeting, Anaheim, California, December 1986.

[23] Peeters, M. F., Habashi, W. G., Nguyen, B. Q., "Finite Element Solutions of the Navier-Stokes Equations for Compressible Internal Flows," *AIAA 90-0441*, January, 1990.

[24] Thomas, J. L., private communication.

[25] Godunov, S. K., "A Finite-Difference Method for Numerical Computation of Discontinuous Solutions of the Equations of Fluid Dynamics," *Matematicheskii Sbornik*, vol. 47 pp 271-306, 1959, (In Russian), (Translated by US Joint Publ. Res. Service as JPRS 7226, 1969).

[26] Peyret, R., Taylor, T. D., *Computational Methods for Fluid Flow*, pp. 112-115, Springer-Verlag, New York, 1983.

[27] Thomas, J. L., and Walters, R. W., "Upwind Relaxation Algorithms for the Navier Stokes Equations," *AIAA 85-1501*, July, 1985.

[28] Thomas, J. L., van Leer, B., and Walters, R. W., "Implicit Flux-Split Schemes for the Euler Equations," *AIAA 85-1680*, July, 1985.

[29] Jameson, A., Schmidt, W., and Turkel, E., "Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time Stepping Schemes," *AIAA 81-1259*, June, 1981.

[30] Harten, A., Engquist, B., Osher, S., and Chakravarthy, S. R., "Uniformly High-Order Accurate Essentially Non-Oscillatory Schemes III.," *ICASE report 86-22*, 1986.

[31] Casper, J., "An Extension of Essentially Non-Oscillatory Shock Capturing Schemes to MultiDimensional Systems of Conservation Laws," *PhD Dissertation*, Old Dominion University, 1990.

[32] Barth, T. J., "A 3-D Upwind Euler Solver for Unstructured Meshes," *AIAA 91-1548*, January, 1991.

[33] Börk, Å, "Least Squares Methods," to appear in *Handbook of Numerical Analysis*, vol. 1 Solutions of Equations in R^n, (Ed. Ciarlet, Lions), Elsevier.

[34] Dhatt, G., and Touzot, G., "The Finite Element Method Displayed,"John Wiley & Sons, 1984, pp.253-261.

[35] Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T., "Numerical Recipes in C, The Art of Scientific Computing" Cambridge University Press, 1988, pp. 28-84.

[36] Chiocchia, G., "Exact Solutions to Transonic and Supersonic Flows," AGARD Advisory Report AR-211, May 1985.

[37] Bowyer, A. "Computing Dirichlet Tessellations," *The Computer Journal*, Vol. 24, no. 2, pp. 162-166, 1981

[38] Gumbert, C., Löhner, R., Parikh, P., and Pirzadeh, S., "A Package for Unstructured Grid Generation and Finite Element Flow Solvers," *AIAA 89-2175*, 1989.

[39] Löhner, R. and Parikh, P. "Generation of Three-Dimensional Unstructured Grids by the Advancing Front Method," *International Journal of Numerical Methods Fluids*, Vol. 8, pp. 1135-1149, 1988.

[40] Baker, T. J., "Three Dimensional Mesh Generation by Triangulation of Arbitrary Point Sets," *AIAA 87-1124-CP*, 1987.

[41] McGrory W. D., "Generalized Spatial Discretization Techniques For Space-Marching Algorithms," *Ph.D. Dissertation*, January, 1991.

[42] McGrory, W. D., Aerosoft, Inc., private communication.

[43] Mortenson, M. E., "Geometric Modeling," John Wiley & Sons, 1985.

[44] White, F. M., "Viscous Fluid Flow,"McGraw-Hill Book Company, 1974, pp.112-114.

[45] Hakkinen, R. J., Greber, I., Trilling, L., and Arbarbanel, S. S., "The Interaction of an Oblique Shock Wave with a Laminar Boundary Layer,"NASA Memo-2-18-59W, March 1959.

[46] Schmitt, V., and Charpin, F., "Pressure Distributions on the ONERA M6-Wing at Transonic Mach Number," AGARD Advisory Report 138, May 1979.

[47] Thareja R. R., Prabhu, R. K., Morgan, K., Peraire, J., Piero, and Soltani, S. "Applications of an Adaptive Unstructured Solution Algorithm to the Analysis of High Speed Flows," *AIAA 90-0395*, January, 1990.

[48] Peraire J., Vahdati, M., Morgan, K., and Zienkiewicz, O. C., "Adaptive Remeshing for Compressible Flow Computations," Journal of Computational Physics, Vol. 72, No. 2, October 1987, pp. 449-466.

[49] Ramakrishnan, R., Bey, K. S., and Thornton, E. A., "Adaptive Quadrilateral and Triangular Finite-Element Scheme for Compressible Flows," AIAA Journal, Vol. 28, No. 1, Jan. 1990, pp. 51-59.

[50] Zeeuw, D. D., Powell, K. G., "Euler Calculations of Axisymmetric Under-Expanded Jets by an Adaptive-Refinement Method," *AIAA 92-0321*, January, 1992.

[51] Paillere, H., Powell, K. G., and Zeeuw, D. D., "A Wave-Model-Based Refinement Criterion for Adaptive-Grid Computation of Compressible Flows," *AIAA 92-0322*, January, 1992.

[52] Hetu, J. F., Pelletier, D. H., "A Fast Adaptive Finite Element Scheme for Viscous Incompressible Flows," *AIAA 91-1564*, January, 1991.

[53] Warren, G. P., Anderson, W. K., Thomas, J. L., and Krist, S. L., "Grid Convergence For Adaptive Methods," *AIAA 91-1592-CP*, 1991.

[54] Dadone, A. and Grossman, B., "Surface Boundary Conditions for the Numerical Solution of the Euler Equations," Virginia Tech ICAM Report *ICAM 92-10-04*, 1992.

Vitae

The author was born on May 12, 1964 in Manassas Virginia. He began his pre-engineering studies at James Madison University and transferred to the Mechanical Engineering program at Virginia Polytechnic Institute and State University. He received a Bachelor of Science degree graduating Magna Cum Laude in June, 1986. After working nine months for the Westinghouse Electric Corporation, the author returned to VPI&SU and earned a Masters of Science degree in Mechanical Engineering in May, 1988. For the next fifteen months the author was employed by the McDonnell Douglas Corporation before returning to VPI&SU in pursuit of a Ph.D. in Aerospace Engineering.