To the Graduate Council:

I am submitting herewith a dissertation written by James Downing Freels entitled "A Taylor Weak Statement Finite Element Algorithm for Real-Gas Compressible Navier-Stokes Simulation." I have examined the final copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirement for the degree of Doctor of Philosophy, with a major in Engineering Science.

Allen J. Baker, Major Professor

We have read this dissertation
and recommend its acceptance:

Accepted for the Council:

Associate Vice Chancellor
and Dean of The Graduate School

A TAYLOR WEAK STATEMENT FINITE ELEMENT ALGORITHM FOR REAL-GAS COMPRESSIBLE NAVIER-STOKES SIMULATION

A Dissertation

Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

by

James Downing Freels

May 1992

Copyright © James Downing Freels 1992
ALL RIGHTS RESERVED

This dissertation is dedicated to my parents, James Cleveland and Patricia Moore Freels of Morristown, Tennessee, to my wife, Elizabeth Gilmore Little Freels of Clinton, Tennessee, and to my sons, Matthew Webster Freels and Michael Little Freels of Clinton, Tennessee.

Acknowledgements

There are several groups which are vital contributors toward the completion of this research. Dr. A. J. Baker, as my major professor, has committed significant time and effort in support of my completion of the research. Our relationship has been extraordinarily good, with timely progress having been made. I am most grateful to have had Dr. Baker as major professor, and I do hope our working relationship may continue in the future. I wish to acknowledge my graduate committee for their advice and hope that knowledge gained during the course of this research would benefit their respective fields.

The Computational Fluid Dynamics (CFD) research group of the ESM department of UTK, advised by Dr. Baker, have contributed to this research. I especially acknowledge Drs. Jin Kim, Wilbert Noronha, and Joe Iannelli for their valuable insight and contributions. The value of a knowledgeable CFD working group is not measurable.

Oak Ridge National Laboratory (ORNL, operated by Martin Marietta Energy Systems), Science Applications International Corporation (SAIC), and Technology for Energy Corporation (TEC), as my employers, have contributed valuable resources toward this research in several ways including contracting my services for research when funds are available, use of corporate computing facilities (VAX-cluster, PCs, and peripherals), reimbursing tuition costs, and influential advice from several talented individuals. In particular I would especially

like to acknowledge Robert A. Hedrick for his many contributions over the years as both a supervisor and teacher.

The ESM department of UTK, through the Computational Fluid Dynamics Laboratory, and Computational Mechanics Corporation (COMCO) has supported me financially through contracts. This financial support has helped tremendously in allowing me more time to devote toward the research.

The University of Tennessee Computing Center (UTCC) and the Cornell National Supercomputer Facility (CNSF) have provided computer resource grants. Without this support, this research would not be possible and is greatly appreciated.

Finally, I acknowledge my family, Libby, Matthew, and Michael, for tolerating the many hours which I had to spend away from them during the course of this research. I hope that one day they will see the benefits of this investment of our time.

A TAYLOR WEAK STATEMENT FINITE ELEMENT ALGORITHM FOR REAL-GAS COMPRESSIBLE NAVIER-STOKES SIMULATION

James Downing Freels
The University of Tennessee

Abstract

A new finite element numerical computational fluid dynamics (CFD) algorithm has been developed for efficiently solving multi-dimensional real-gas compressible flow problems in generalized coordinates on modern parallel-vector computer systems. The algorithm employs a Taylor extension on the classical Galerkin weak statement formulation, a time-relaxed iteration procedure, and a tensor matrix product based factorization of the linear algebra jacobian under a generalized coordinate transformation. Allowing for a general conservation law system, the algorithm has been exercised for the two-dimensional Euler and the laminar and turbulent forms of the Navier-Stokes equations. Equilibrium real-gas air properties are admitted, and numerical results verify solution accuracy, consistency, convergence, efficiency, and stability over a range of test problem parameters. The algorithm is cast in a fully generalized form, such that extension to other flow problems, including three dimensions or two-phase thermal-hydraulics, is direct.

Table of Contents

1 Introduction **1**

1.1 Literature Review 3

1.2 Scope 10

1.3 Organization 13

2 The Compressible Navier-Stokes Problem Statement **15**

2.1 Governing Partial Differential Equation System 15

2.2 Constitutive Equations 18

2.3 Generalized Coordinate Transformation 20

3 Reduced Forms of the Governing Equations **23**

3.1 Parabolized Navier-Stokes 24

3.2 Quasi-One-Dimensional 32

4 Taylor Weak Statement **33**

4.1 Derivation 34

4.2 Inviscid Flux Vector Jacobian and Modified Forms 37

4.3 β Parameter Determination 52

5 Discrete Approximations **55**

5.1 Finite Element Semi-Discretization 55

5.2 Nomenclature 59

5.3 Fully-Discrete Approximation 62

5.4 Semi-Implicit Runge-Kutta Procedure 69
5.5 Incorporation of Closure Equations 72
5.6 Newton Iteration Jacobian Matrix and Approximations 73

6 Boundary and Initial Conditions **83**
6.1 Boundary Conditions 83
6.2 Initial Conditions 95

7 Theoretical Analyses/Expectations **101**
7.1 Convergence with Discretization Refinement 101
7.2 Quasi-Newton Iteration Convergence 104
7.3 Time-Step Selection 105
7.4 Evolution Stability 107

8 Algorithm Assessment **112**
8.1 Subsonic Converging Duct 113
8.1.1 Problem Statement 114
8.1.2 Geometry Specification 117
8.1.3 Quasi-One Dimensional Flow Results 122
8.1.4 Inviscid 2D Solution 129
8.1.5 Viscous Flow Verifications 139
8.2 Transonic Wind Tunnel with Shock-Boundary Layer Interaction . . 163
8.2.1 Problem Description 165
8.2.2 Discretization Specifications 176
8.2.3 Quasi-One Dimensional Flow Simulation 181
8.2.4 Inviscid Flow 2D Simulation 184
8.2.5 Viscous Flow Simulations 196
8.3 GAMM Double-Throat Nozzle 233
8.3.1 Problem Statement 233

8.3.2 Geometry Specification 236
8.3.3 Solution Procedure 237
8.3.4 Quasi-One Dimensional Flow 238
8.3.5 Inviscid 2D Flow Simulation 241
8.3.6 Viscous Flow Simulation 249

9 Supercomputer Utilization **267**
9.1 General Organization 267
9.2 Vectorization Techniques 272
9.3 Parallelization Techniques 284
9.4 Impact of Real-Gas Closure 296

10 Summary and Conclusions **298**

11 Recommendations for Future Research **301**

Bibliography **305**

Appendices **320**

A Derivation of the Governing Equations **321**
A.1 Introduction 321
A.2 Conservation of Mass 322
A.3 Conservation of Momentum 322
A.4 Conservation of Energy 324
A.5 Extension to Three Dimensions 326
A.6 Extension to Turbulent Form 327
A.7 Constitutive Equations 329
A.7.1 Equations of State 329
A.7.2 Turbulence Closure 330

B Transformation to Dimensionless Form **334**

B.1 Introduction 334

B.2 Conservation of Mass 335

B.3 Conservation of Momentum 336

B.4 Conservation of Energy 337

B.5 Constitutive Equations 339

B.5.1 Equations of State 339

B.5.2 Turbulence Closure 339

B.6 Extension to Three Dimensions 340

C Generalized Coordinate Transformation **341**

D The Pilot Code Input Description **345**

E Mini-Investigation of Band-Matrix Direct Solvers **353**

F A Digital Filter to Reduce Time-Step Oscillations Near Steady-State **357**

VITA **361**

List of Tables

3.1 Symbolic Space-Marching (PNS) Stability Analysis Results for the Reynolds-Averaged Navier-Stokes Equations and Reduced Forms . 28

6.1 Tests of Well-Posed Dirichlet Boundary Conditions for the Researched State Variable Set $\{\rho, m_1, m_2, g\}$ on the Euler equations . 89

8.1 Subsonic Converging Duct Problem Isentropic Flow Initial and Boundary Conditions 114

8.2 Constants Used to Define the Converging Inlet Wall Curvature for the Subsonic Converging Duct Flow Problem. 117

8.3 Comparison of the Computed Quasi-1D Euler Solution with the Theoretical Isentropic Solution for the Subsonic Converging Duct Problem 125

8.4 Quasi-Newton Iteration Convergence Rate for the Quasi-1D Subsonic Converging Duct Problem. 126

8.5 Comparison Quasi-Newton Iteration Convergence Rates at Midpoint in the Solution of the 1D Riemann Shock Tube Problem as a Function of System Iteration Jacobian Matrix Formulations 128

8.6 Quasi-Newton Iteration Convergence Rate Summary for the 2D Inviscid Subsonic Converging Duct Problem 137

8.7 Transonic Wind Tunnel Problem Isentropic Initial and Boundary Conditions for the Weak-Shock Case 175

8.8 Constants Used to Define the Geometry of the Transonic Wind Tunnel Problem 177

8.9 Representative Comparison of Calculated Results with Experimental Data for the Transonic Wind Tunnel Problem Top Wall (Surface 4) 224

8.10 Functional Relationship of the Wall Surface for the GAMM Nozzle 235

9.1 Comparative Percent CPU Utilization of the IBM-3090 Vector Facility on a Model Problem Utilizing the Pilot Code 281

9.2 Parallel Performance of the IBM-3090 Vector Facility on a Model Problem Utilizing the Pilot Code 293

E.1 Comparison of Alternative Banded-Matrix Direct Solvers 355

List of Figures

4.1 Distribution of β in Viscous Flow Reaches. 54

5.1 A Typical Bilinear Basis Function Finite Element and Coordinate Transformation. 58

8.1 Subsonic Converging Duct Problem Definition. 115

8.2 Coarse Mesh Grid for the 2D Inviscid Solution of the Converging Duct Problem 119

8.3 Grid Specification for the 2D Viscous Solution of the Converging Duct Problem 121

8.4 Quasi-1D Euler Solution of the Subsonic Converging Duct Problem Plotted in Physical Space 123

8.5 Quasi-1D Euler Solution of the Subsonic Converging Duct Problem Plotted in Node Space 124

8.6 Convergence Rate for the Error in Energy Norm for the Quasi-1D Finite Element Euler Solution of the Subsonic Converging Duct Problem 130

8.7 Inviscid Solution of the Subsonic Converging Duct Problem Shown as Iso Contours in Physical Space; (a) Density, (b) Principal Momentum, (c) Transverse Momentum, and (d) Volume-Specific Stagnation Enthalpy (vertical scale expanded for clarity). 131

8.8 Inviscid Solution of the Subsonic Converging Duct Problem Shown as Iso Contours in Physical Space; (e) Pressure, (f) Principal Velocity, (g) Transverse Velocity, and (h) Temperature (vertical scale expanded for clarity). 132

8.9 Inviscid Solution of the Subsonic Converging Duct Problem Plotted in Nodal Space as Viewed from the Top Exit. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 134

8.10 Inviscid Solution of the Subsonic Converging Duct Problem Plotted in Nodal Space as Viewed from the Bottom Entrance. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 135

8.11 Convergence Rate of the Error in H_0 Norm for the 2D Inviscid Finite Element Solution of the Subsonic Converging Duct Problem. 140

8.12 Laminar Viscous Solution of the Subsonic Converging Duct Problem Shown as Iso Contours in Physical Space; Free Stream Exit Mach Number 0.8, Reynolds Number 10^3; (a) Pressure, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature (vertical scale expanded for clarity). 142

8.13 Laminar Viscous Solution of the Subsonic Converging Duct Problem Plotted in Nodal Space as Viewed from the Top Exit; Free Stream Exit Mach Number 0.8, Reynolds Number 10^3; (a) Pressure, (b) Temperature, (c) Principal Velocity, (d) Transverse Velocity. 144

8.14 Laminar Viscous Solution of the Subsonic Converging Duct Problem Plotted in Nodal Space as Viewed from the Bottom Exit; Free Stream Exit Mach Number 0.8, Reynolds Number 10^3; (a) Pressure, (b) Temperature, (c) Principal Velocity, (d) Transverse Velocity. . 145

8.15 Boundary Layer Velocity Profile Comparison Between the Laminar Viscous Solution of the Subsonic Converging Duct Problem and the Flat Plate Incompressible Solution of Blasius; Free Stream Exit Mach Number 0.8, Reynolds Number 10^3; [δ_{exit} = .05 (top), δ_{exit} = .023 (bottom)]. 149

8.16 Convergence Rate of the Error in Energy Norm for the 1D Space Marching Finite Element Boundary Layer Solution of a Flat Plate Problem. 151

8.17 Turbulent Viscous Solution of the Subsonic Converging Duct Problem Shown as Iso Contours in Physical Space; Free Stream Exit Mach Number 0.8, Reynolds Number 10^5; (a) Pressure, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature (vertical scale expanded for clarity). 153

8.18 Turbulent Viscous Solution of the Subsonic Converging Duct Problem Plotted in Nodal Space as Viewed from the Top Exit; Free Stream Exit Mach Number 0.8, Reynolds Number 10^5; (a) Pressure, (b) Temperature, (c) Principal Velocity, (d) Transverse Velocity. . 154

8.19 Turbulent Viscous Solution of the Subsonic Converging Duct Problem Plotted in Nodal Space as Viewed from the Bottom Exit; Free Stream Exit Mach Number 0.8, Reynolds Number 10^5; (a) Pressure, (b) Temperature, (c) Principal Velocity, (d) Transverse Velocity. . 155

8.20 Turbulent Eddy Viscosity Produced by the Baldwin-Lomax Model for the Subsonic Converging Duct Problem Plotted in Nodal Space; Free Stream Exit Mach Number 0.8, Reynolds Number 10^5; (a) Initial Conditions from the Laminar Problem Viewed from Top Exit, (b) Initial Conditions from the Laminar Problem Viewed from Bottom Exit, (c) Final Conditions of the Turbulent Problem Viewed from Top Exit, (d) Final Conditions of the Turbulent Problem Viewed from Bottom Exit. 156

8.21 Boundary Layer Velocity Profile Comparison Between the Turbulent Viscous Solution of the Subsonic Converging Duct Problem and the Flat Plate Incompressible Solution of Coles; Free Stream Exit Mach Number 0.8, Reynolds Number 10^5. 159

8.22 Near-Wall Boundary Layer Temperature Profile Comparison Between Various Heat Transfer Coefficients for the Laminar Viscous Solution of the Subsonic Converging Duct Problem. Free Stream Exit Mach Number 0.8, Reynolds Number 10^3; [h = 0.0 (top), h = 1.0 (middle), h = 10.0 (bottom)]. 161

8.23 Transonic Wind Tunnel Problem Definition 166

8.24 Measured Pressure Distribution Along the Top Wall for the Transonic Wind Tunnel Problem, from Bogar et al.[129] 168

8.25 Measured Axial Velocity Profiles Downstream of the Shock for the Strong-Shock Transonic Wind Tunnel Problem, from Salmon et al.[128] 169

8.26 Calculated Pressure for the Inviscid, Laminar-Viscous, and Turbulent-Viscous Solution Phases of the Strong-Shock Transonic Wind Tunnel Problem (vertical scale expanded by ≈ 2 for clarity). 171

8.27 Calculated Transverse Momentum for the Inviscid, Laminar-Viscous, and Turbulent-Viscous Solution Phases of the Strong-Shock Transonic Wind Tunnel Problem (vertical scale expanded by ≈ 2 for clarity). 172

8.28 Calculated Mach Number for the Inviscid, Laminar-Viscous, and Turbulent-Viscous Solution Phases of the Strong-Shock Transonic Wind Tunnel Problem (vertical scale expanded by ≈ 2 for clarity). 174

8.29 Three Mesh Adaptations for the Solution Progression of the Strong-Shock Transonic Wind Tunnel Problem 178

8.30 Local Magnification of Inlet Transition Region Mesh for Inviscid Simulation of the Transonic Wind Tunnel Problem 179

8.31 Grid Specification for the Viscous Transonic Wind Tunnel Problem Plotted in Computational Space 180

8.32 Weak-Shock Transonic Wind Tunnel Problem Quasi-1D Solution; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Velocity. 182

8.33 Strong-Shock Transonic Wind Tunnel Problem Quasi-1D Solution; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Velocity. 183

8.34 Beta Sensitivity on Principal Momentum for the Weak-Shock Transonic Wind Tunnel Problem Quasi-1D Solution. 185

8.35 Beta Sensitivity on Mach Number for the Weak-Shock Transonic Wind Tunnel Problem Quasi-1D Solution. 186

8.36 Inviscid Solution of the Weak-shock Transonic Wind Tunnel Problem Shown as Iso-Value Contours in Physical Space; (a) Density, (b) Principal Momentum, (c) Transverse Momentum, and (d) Volume-Specific Stagnation Enthalpy (vertical scale expanded for clarity). . 187

8.37 Inviscid Solution of the Weak-Shock Transonic Wind Tunnel Problem Shown as Iso-Value Contours in Physical Space; (e) Pressure, (f) Principal Velocity, (g) Transverse Velocity, and (h) Mach Number (vertical scale expanded for clarity). 188

8.38 Inviscid Solution of the Weak-Shock Transonic Wind Tunnel Problem, Plotted in Nodal Space as Viewed from the Top Exit. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 190

8.39 Inviscid Solution of the Weak-Shock Transonic Wind Tunnel Problem, Plotted in Nodal Space as Viewed from the Bottom Entrance. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 191

8.40 Inviscid Solution of the Strong-shock Transonic Wind Tunnel Problem Shown as Iso-Value Contours in Physical Space; (a) Density, (b) Principal Momentum, (c) Transverse Momentum, and (d) Volume-Specific Stagnation Enthalpy (vertical scale expanded for clarity). . 194

8.41 Inviscid Solution of the Strong-Shock Transonic Wind Tunnel Problem Shown as Iso-Value Contours in Physical Space; (e) Pressure, (f) Principal Velocity, (g) Transverse Velocity, and (h) Mach Number (vertical scale expanded for clarity). 195

8.42 Inviscid Solution of the Strong-Shock Transonic Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Top Exit. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 197

8.43 Inviscid Solution of the Strong-Shock Transonic Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Bottom Entrance. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 198

8.44 Laminar-Viscous Solution of the Weak-Shock Transonic Wind Tunnel Problem Shown as Iso-Value Contours in Physical Space; Shock Mach Number $\approx$ 1.235, Reynolds Number 10^4; (a) Pressure, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature (vertical scale expanded for clarity). 202

8.45 Laminar-Viscous Solution of the Weak-Shock Transonic Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Top Exit; Shock Mach Number 1.235, Reynolds Number 10^4; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 203

8.46 Laminar-Viscous Solution of the Weak-Shock Transonic Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Bottom Entrance; Shock Mach Number 1.235, Reynolds Number 10^4; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 204

8.47 Laminar-Viscous Solution of the Strong-Shock Transonic Wind Tunnel Problem Shown as Iso-Value Contours in Physical Space; Shock Mach Number $\approx$ 1.353, Reynolds Number 10^3; (a) Pressure, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature (vertical scale expanded for clarity). 206

8.48 Laminar-Viscous Solution of the Strong-Shock Transonic Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Top Exit; Shock Mach Number 1.353, Reynolds Number 10^3; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 207

8.49 Laminar-Viscous Solution of the Strong-Shock Transonic Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Bottom Entrance; Shock Mach Number 1.353, Reynolds Number 10^3; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 208

8.50 Laminar-Viscous Solution of the Weak-Shock Transonic Wind Tunnel Problem Shown as Velocity Vectors in Physical Space; Shock Mach Number $\approx$ 1.235, Reynolds Number 10^3; Magnifications increase from top to bottom as defined by box. 210

8.51 Laminar-Viscous Solution of the Strong-Shock Transonic Wind Tunnel Problem Shown as Velocity Vectors in Physical Space; Shock Mach Number $\approx$ 1.353, Reynolds Number 10^3; Magnifications increase from top to bottom as defined by box. 211

8.52 Turbulent-Viscous Solution of the Weak-Shock Wind Tunnel Problem Shown as Iso-Value Contours in Physical Space; Shock Mach Number $\approx$ 1.235, Reynolds Number 10^5; (a) Pressure, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature (vertical scale expanded for clarity). 212

8.53 Turbulent-Viscous Solution of the Weak-Shock Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Top Exit; Shock Mach Number 1.235, Reynolds Number 10^5; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 214

8.54 Turbulent-Viscous Solution of the Weak-Shock Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Bottom Entrance; Shock Mach Number 1.235, Reynolds Number 10^5; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 215

8.55 Turbulent-Viscous Solution of the Strong-Shock Wind Tunnel Problem Shown as Iso-Value Contours in Physical Space; Shock Mach Number $\approx$ 1.353, Reynolds Number 10^5; (a) Pressure, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature (vertical scale expanded for clarity). 216

8.56 Turbulent-Viscous Solution of the Strong-Shock Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Top Exit; Shock Mach Number 1.353, Reynolds Number 10^5; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 219

8.57 Turbulent-Viscous Solution of the Strong-Shock Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Bottom Entrance; Shock Mach Number 1.353, Reynolds Number 10^5; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 220

8.58 Turbulent-Viscous Solution of the Weak-Shock Wind Tunnel Problem Shown as Velocity Vectors in Physical Space; Shock Mach Number $\approx$ 1.235, Reynolds Number 10^5; Magnifications increase from top to bottom as defined by box. 222

8.59 Turbulent-Viscous Solution of the Strong-Shock Wind Tunnel Problem Shown as Velocity Vectors in Physical Space; Shock Mach Number $\approx$ 1.353, Reynolds Number 10^5; Magnifications increase from top to bottom as defined by box. 223

8.60 Calculated Pressure Distribution Along the Top Wall (Surface 4) for the Turbulent Viscous Solution of the Strong-Shock Transonic Wind Tunnel Problem 226

8.61 Calculated Principal Velocity Profiles Downstream of the Shock for the Turbulent Viscous Solution of the Strong-Shock Transonic Wind Tunnel Problem 228

8.62 Calculated Principal Velocity Profiles Downstream of the Shock for the Laminar Viscous Solution of the Strong-Shock Transonic Wind Tunnel Problem 229

8.63 Comparison Between Real- and Ideal-Gas Equations of State for the Quasi-1D Solution of the Weak-Shock Transonic Wind Tunnel Problem 231

8.64 GAMM Double-Throat Nozzle Problem Definition 234

8.65 GAMM Double-Throat Nozzle Problem Ideal-Gas Quasi-1D Solution; Vanishing Derivative Exit Boundary Conditions; Variables Plotted in Physical Space; (a) Pressure, (b) Temperature, (c) Principal Velocity, (d) Mach Number. 239

8.66 GAMM Double-Throat Nozzle Problem Ideal-Gas Quasi-1D Solution; Non-vanishing Derivative Exit Boundary Conditions; Variables Plotted in Physical Space; (a) Pressure, (b) Temperature, (c) Principal Velocity, (d) Mach Number. 240

8.67 Coarse-Grid Ideal-Gas Inviscid Flow Solution of the GAMM Double-Throat Nozzle Problem Shown as Iso Contours in Physical Space; (a) Density, (b) Principal Momentum, (c) Transverse Momentum, and (d) Volume-Specific Stagnation Enthalpy ($\approx$ to scale). 242

8.68 Coarse-Grid Inviscid Temperature Solution of the GAMM Double-Throat Nozzle Problem Plotted in Nodal Space as Viewed from the Top Exit. Ideal-Gas (top), Real-Gas (bottom). 243

8.69 Mesh Density Refinement for the GAMM Double-Throat Nozzle Inviscid Problem. 244

8.70 Ideal-Gas Pressure Solutions of Respective Mesh Density Refinements for the GAMM Double-Throat Nozzle Inviscid Problem. . . 245

8.71 Fine-Grid Ideal-Gas Inviscid Solution of the GAMM Double-Throat Nozzle Problem Shown as Iso Contours in Physical Space; (a) Density, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature ($\approx$ to scale). 247

8.72 Fine-Grid Ideal-Gas Inviscid Flow Solution of the GAMM Double-Throat Nozzle Problem Plotted in Nodal Space as Viewed from the Top Exit. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 248

8.73 Fine-Grid Ideal-Gas Inviscid Flow Solution of the GAMM Double-Throat Nozzle Problem Plotted in Nodal Space as Viewed from the Bottom Entrance. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. . . 250

8.74 Laminar Ideal-Gas Viscous Solution of the GAMM Double-Throat Nozzle Problem Shown as Iso Contours in Physical Space; (a) Pressure, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature (approximately to scale). 252

8.75 Laminar Ideal-Gas Viscous Solution of the GAMM Double-Throat Nozzle Problem Plotted in Nodal Space as Viewed from the Top Exit. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 253

8.76 Laminar Ideal-Gas Viscous Solution of the GAMM Double-Throat Nozzle Problem Plotted in Nodal Space as Viewed from the Bottom Entrance. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 254

8.77 Turbulent Ideal-Gas Viscous Solution of the GAMM Double-Throat Nozzle Problem Shown as Iso Contours in Physical Space; (a) Pressure, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature (approximately to scale). 256

8.78 Turbulent Ideal-Gas Viscous Solution of the GAMM Double-Throat Nozzle Problem Plotted in Nodal Space as Viewed from the Top Exit. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 257

8.79 Turbulent Ideal-Gas Viscous Solution of the GAMM Double-Throat Nozzle Problem Plotted in Nodal Space as Viewed from the Bottom Entrance. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum. 258

8.80 Fine-Grid Temperature Solution of the GAMM Double-Throat Nozzle Problem Plotted in Nodal Space as Viewed from the Top Exit. (a) Inviscid, (b) Laminar Viscous, (c) Turbulent Viscous. 260

8.81 Pressure and Mach Number Iso Contours for Reynolds Number of 1600, Thomas and Walters, from GAMM Workshop Proceedings, Braunschweig, Wiesbaden, Bristeau, et al.[133] 261

8.82 Calculated Mach Number for the Ideal-Gas Invsicid Solution of the GAMM Double-Throat Nozzle Problem, Expanded Shock-Reflection Region Shown. ($\approx$ to scale). 263

8.83 Calculated Pressure for the Ideal-Gas Inviscid Solution of the GAMM Double-Throat Nozzle Problem ($\approx$ to scale). 264

8.84 Calculated Transverse Momentum for the Ideal-Gas Laminar Viscous Solution of the GAMM Double-Throat Nozzle Problem (≈ to scale). 265

9.1 Overall Organization of the Pilot Code 268

9.2 Organization of the Finite Element Assembly Loop of the Pilot Code 269

9.3 Organization of the Nonlinear Iterative Solution Procedure of the Pilot Code 271

9.4 Vectorization of a Simple Loop Within Subroutine `SWITCH` of the Pilot Code 274

9.5 Selected Portion of the Original `FQI` Subroutine Prior to Vectorization 275

9.6 Selected Portion of the `FQI` Subroutine After Rewriting and Vectorization 277

9.7 Array Equivalencing Used by the Vectorized `FQI` Subroutine . . . 278

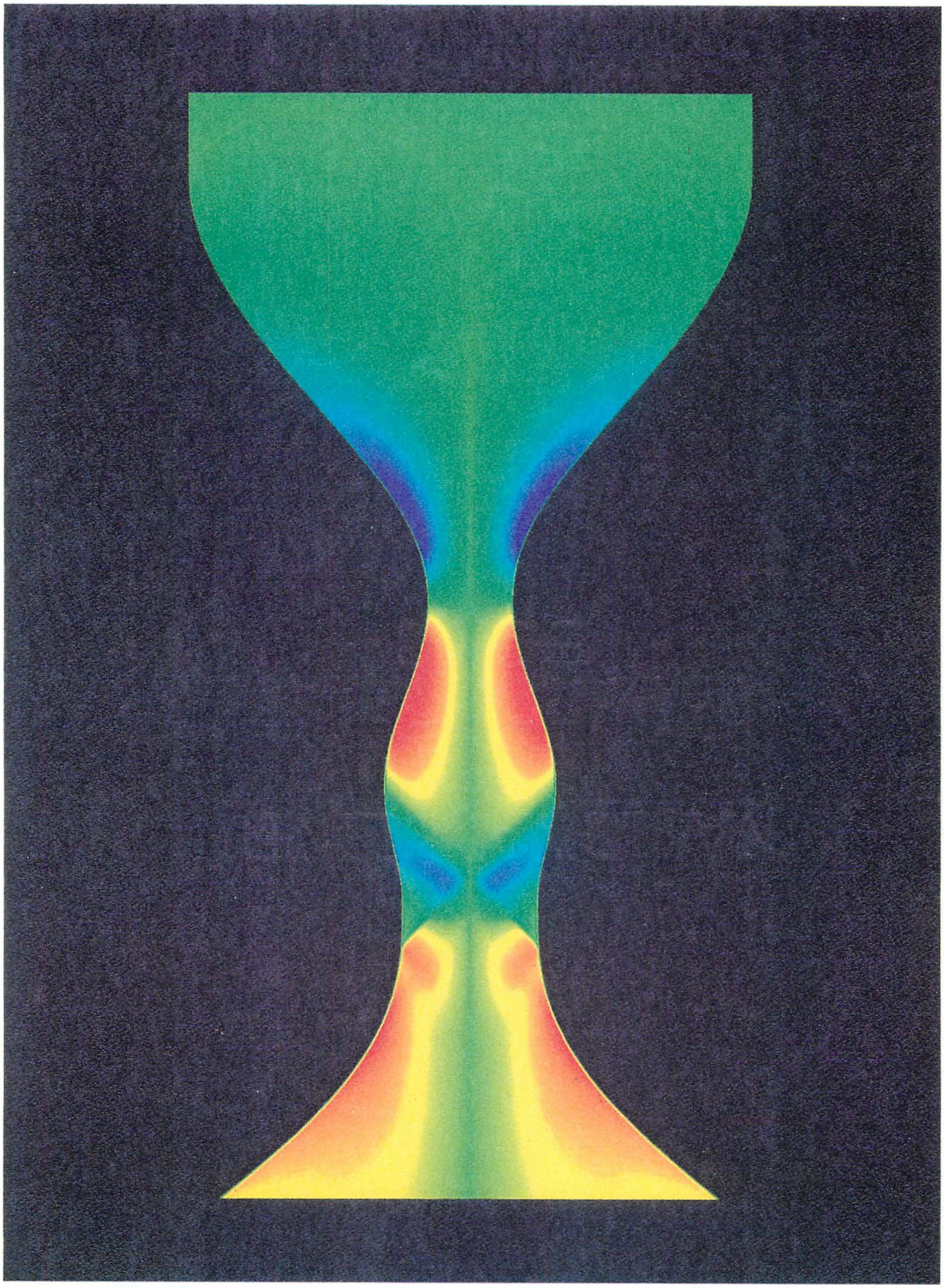
9.8 Subroutine `SMLOAD` as an Example of Code Which Cannot be Vectorized 279

9.9 Parallelization of Subroutine `SWITCH` of the Pilot Code 286

9.10 Parallelization of Subroutine `FQI` of the Pilot Code 288

9.11 Parallelization of Subroutine `SOLUTI` of the Pilot Code 289

F.1 Analog Circuit for a Low-Pass Filter. 358



Chapter 1

Introduction

Computational fluid dynamics, with CFD the acronym, has become synonymous with the most demanding calculations performed on modern computers. Indeed, the acronym CFD was not widely used before electronic computing existed. Many might think of CFD as the science of numerical solution of the Navier-Stokes equations or it's reduced forms. This is most certainly not a comprehensive definition since the true definition of the Navier-Stokes equations is the conservation of mass and momentum for an incompressible fluid.[1] Probably the best definition for CFD is "the numerical simulation of solutions to the conservation law statements of fluid mechanics."

This very broad definition allows for many complex problems to be considered a CFD problem. The author became acquainted with CFD as a profession through the use of large computer programs to simulate "transient thermal-hydraulics" (T-H) associated with the safety and operation of commercial nuclear power plants.[2–4] After working with these codes, users inevitably ask "isn't there a better way?" Although the results for essential physics simulation were usually adequate, there is ample room for improvement. For example, the efficient utilization of computer resources is not ideal due to the long execution time required for 1D unsteady calculations of "detailed" T-H models of less

than 1000 degrees of freedom. Recently, some of the codes have upgraded to implicit time integration.[4] However, time-accuracy is compromised because full implicitness, i.e., multiple iterations per time step, is required. Therefore, only one-dimensional (1D) T-H simulations are currently practiced. [a]

In the nuclear industry alone there are many complex problems and/or issues that could take advantage of modern computational methods (they currently do not). These include:

- Heat transfer by conduction, radiation, and convection,
- Nuclear power production by point kinetics or more accurate particle transport equations,
- Nonlinear structural dynamics (flow-induced vibration),
- Two-phase (liquid-vapor) flow,
- Metal melt propagation and distribution, and
- Metal-water (fuel-coolant) interaction.

Many problems require coupling of one or more (or all!) of the phenomena listed above and should be solved in a time-accurate and 3D mode to reliably simulate, hence understand the consequences.

At first glance, this extremely demanding problem set or definition might seem impossible to solve, even with today's computing capability. However, computer hardware and software technology are advancing at a phenomenal pace. Indeed, the CFD community is now solving 3D aerodynamic problems,

[a] The TRAC code, vis-a-vis Los Alamos National Laboratory, claims to have a 3D reactor vessel component model. Close examination reveals severe reductions to the full 3D equations. For example, it is not possible to produce a boundary layer profile in the solution!

with upwards of $10^6 \rightarrow 10^7$ nodes, and five or more degrees of freedom per node, that only a few years ago seemed too demanding. By far, the most advanced applications of CFD are presently in the aerospace industry.[5] Therefore, the most efficient way to advance the state of the art in the nuclear or other industry is by taking advantage of the experiences of the aerospace sciences and scientists. This is not to say that the evolution of the aerospace CFD sciences are complete. Quite the contrary, it is hoped that this dissertation might represent some advancement toward that end. Furthermore, the technology gained from this dissertation research might also be directly applied in the nuclear industry.

1.1 Literature Review

FE Compressible Navier-Stokes Algorithms

When this dissertation topic was proposed,[6] there was no question that such a FE algorithm had never been developed beforehand. The FE CFD community has accomplished significant research and development in recent years. Therefore, it is necessary to compare the present algorithm to those published in the field to assure uniqueness and originality. A comprehensive search of related work has been conducted to produce abstracts of all published papers from international journals and technical conferences. The key words for the search were "finite element(s)" or "Galerkin" and "fluids." The list was reduced by eliminated those papers that did not concern "Navier-Stokes" equations and by considering only the "compressible" form. The final set of papers represents all researchers who have proposed, attempted, or solved the compressible Navier-Stokes equations by a finite element method. Then the list was categorized into nine principal authors (those that had two or more papers) and a tenth category that includes all other researchers with only a single paper in this particular field; each

was compared in detail to the present algorithm. The comparison is made on the primary dissertation objectives, as listed, to determine whether an equivalent level of development has been achieved in the respective category. For example, the inclusion of turbulence in the viscous dissipation term (through an algebraic turbulence model or a more complex representation) is one test for comparison. If the author(s) presented their algorithm in sufficient detail for comparison, then they were included in this discussion.

The place to start is the present FE CFD research group at the University of Tennessee, Knoxville, lead by Baker.[7–20] Each researcher has certainly benefited from the knowledge gained from others in the group both previous and present. Among other accomplishments, Baker and Soliman[21–24] began the present FE algorithm nomenclature used throughout the group, demonstrated the tensor product factorization of the multi-dimensional problems, and demonstrated early shock-capturing ability of 1D/2D compressible Euler/Navier-Stokes problems. Baker and Moore[25] concentrated on the artificial dissipation mechanism of the penalty-Galerkin method and demonstrated a tuning procedure for several 1D Euler problems. Baker and Kim[26–28] then expanded on this concept to develop the Taylor Weak Statement (TWS) companion or modified conservation law system for the 1D and 2D Euler equation system. Thoroughly analyzed, the TWS was proven to embed a large number of presently-utilized dissipation algorithms, demonstrating the generality. The generality is verified by the TWS incorporation into the incompressible flow algorithm developed by Baker and Noronha[29] which also resulted in a unique and extremely beneficial linear algebra procedure by Baker, Wachspress, and Noronha.[30] The concept of stability within the TWS-Galerkin framework was thoroughly documented by Baker and Iannelli[31–35] with extension of the FE algorithm into the compressible, laminar Navier-Stokes problem. Furthermore, several milestones similar to those

developed here were accomplished. Finally, this author with Baker[36] has also helped to contribute many new ideas which have benefited the research group as documented by this dissertation.

Hughes et al.[37–42] have developed a "streamline-upwind/Petrov-Galerkin" (SUPG) algorithm and later with "discontinuity capturing" (DC) terms to solve advection-diffusion problems. The DC terms were added due to the inability of the initial SUPG method to produce non oscillatory solutions in internal and boundary layers. Even so, the cited papers do not indicate attempting a viscous flow solution. The approach is distinguished by the transformation of the original conservation law equations to an entropy-based form which possesses a positive-definite symmetric matrix system. Part of the trade-off is from a consistent to a lumped mass matrix, due to the method of characteristics employed. Careful attention is also paid to thermodynamics consistency due to the transformation of variables.

Oden et al.[43–47] have developed an "*h-p* method" for automated, adaptive mesh (*h*), and local degree of the polynomial shape function (*p*) selection for FE simulations of compressible flow. Both an upwind-diffusion Petrov-Galerkin method, based on a variation of the SUPG algorithm, and a Taylor-Galerkin/FE Lax-Wendroff method are utilized for needed artificial dissipation for problems with shocks. The price paid for the increased grid/interpolate accuracy is the programming tasks associated with the automation, and the restriction to explicit time integration methods, typical of nearly all unstructured grid methods.

Habashi et al.[48–50] have developed FE algorithms characterized by fully-coupled implicit methods for the solution of incompressible flows (NS). Similar to the present method, a true (full) Newton jacobian is typically employed yielding quadratic iterative convergence of the linear algebra statement. Unlike the

present research, Habashi has concentrated on iterative, sparse-matrix solvers, such as preconditioned conjugate-gradient schemes, to resolve the linear-algebra problem. The artificial dissipation mechanism utilized is similar to Petrov-Galerkin methods.

Pelletier[51, 52] has recently proposed and presented a FE adaptive mesh solution procedure for compressible NS (laminar); the unique feature of the algorithm is that an energy norm (Chapter 7) is used to directly determine the solution adaptive remeshing.

Zienkiewicz, Löhner, Morgan, et al.[53–56] have developed FE algorithms that are distinguished by automated or adaptive grids for compressible, high-speed flows. Other features unlike the present work include an empirical, and later a Taylor-Galerkin, artificial dissipation mechanism, lumped mass matrix in order to simplify the linear algebra problem, and inviscid problem statements. However, most recently Löhner et al. have extended their research to incorporate implicit methods, with application to laminar viscous problems,[57] and have realized the need for ADI or factorization type procedures with an interesting concept called "snakes and linelets."

A number of FE researchers have concentrated on least-squares (LS) methods. Bristeau and Periaux et al.[58–63] investigate several classes of FE methods, with LS being the favorite for the highly nonlinear problems. LS methods produce symmetric positive-definite iteration matrices allowing for use of known iterative methods for the linear algebra problem (typically conjugate gradient). Typically, the equations are written in non-conservative form. Self-adaptive mesh refinement techniques are also demonstrated.

Jiang and Povinelli[64] present a LS FE method, that claims a unified and comprehensive algorithm capable of solving a wide range of fluid problems.

Since it is a LS method, a symmetric, positive definite matrix is formed, allowing for an iterative method to be employed for the linear algebra procedure. The compressible flow examples are limited to the inviscid case. Even though the method is free of an explicit artificial viscosity term, the method implicitly incorporates numerical dissipation since a second order term is created by the product of the squared first order terms. Hence, a modified conservation law has been formed in the spirit of the TWS.

Glowinski et al.[65,66] also have developed LS FE algorithms for both incompressible and compressible NS problems. The solution typically employs conjugate gradient iterative methods with domain decomposition utilized to take advantage of modern vector/parallel computers.

Amestoy and Tilch[67] describe a FE solution to the compressible NS equations using a "multifrontal" method to solve the sparse matrix created by the FE assembly of the mass matrix. The scheme is presented as explicit predictor/corrector, but plans are cited to naturally extend the technique to an implicit time stepping. The problem formulation is based on a Taylor-Galerkin approach similar to that presented herein. A demonstration of external flow over a circular cylinder is also discussed, in a IBM-3090 VF parallel computing environment, similar to that discussed in Chapter 9 of this dissertation.

Cooke and Blanchard[68] presented a solution of 2D, compressible, constant total temperature, NS equations by a cubic FE interpolate. Although the solutions proved highly accurate, the authors concluded the higher order FE interpolates were not economical. The time discretization scheme was implicit, with the system matrices being nonsymmetric and banded, with a Cholesky decomposition utilized for the solution.

Fletcher[69] presents an alternating direction implicit (ADI) FE method for the compressible NS equations. The energy equation was removed by assuming an adiabatic equation of state to arrive at the fluid pressure. Found to be superior in accuracy to conventional ADI FD schemes on a test problem, the algorithm was successfully demonstrated on 2D flow past a rectangular body. A von Neumann stability analysis revealed stable solutions up to Reynolds number of 10^3 provided a sufficiently small time step is used.

Reddy and Reddy[70] propose and discuss a FE solver for 3D compressible NS flows. Based on a fully implicit Euler scheme, the need to solve a global matrix is removed by a locally implicit scheme based on a modified Gauss-Seidel method. An adaptive artificial dissipation term is included.

Spradley et al.[71] present an algorithm based on a general interpolants method (GIM) and a progressive assembly of generalized elements (PAGE). They report the algorithm uses the Petrov-Galerkin dissipation, incorporates a "consistent" lumped mass matrix, hence is explicit, and easily takes full advantage of vector and parallel computing. Furthermore, the algorithm is extended to parabolized NS applications described as a "quasi-parabolic."

FE Compressible Parabolized Navier-Stokes Algorithms

Many research groups have developed parabolized Navier-Stokes (PNS) algorithms and applied them to a wide range of problems ranging from incompressible subsonic flow (oceanic ship movement) to compressible transonic/supersonic flow (aerodynamics). In addition to handling of the streamwise pressure gradient, another distinction among the algorithms is the selected method of artificial dissipation. The search leading to a PNS research summary, although not com-

prehensive, provides some indication of the diversity of approaches towards a solution thereof.

Rubin et al.[72–74] pioneered early development of PNS algorithms. Among the unique developments were identification of pressure interaction, relaxation procedures for mildly separated flows, and a strongly implicit and pressure relaxation procedure for supersonic, viscous, external flows.

Tannehill et al.[75–77] extended PNS applications toward conical flows and included generalized coordinate transformation, more refined grids, and a more mathematically based pressure relaxation procedure.

Steger et al.[78,79] independently developed a PNS algorithm which later developed into a commercially-available PNS code.[80,81] Although certainly not and example of perfection, the AFWAL PNS code is perhaps in widest use for hypersonic external aerodynamic problems.

McDonald et al.[82,83] also independently developed PNS (and other) algorithms. The PNS algorithm later evolved into the PEPSIG and PEPSIS (and later improvements) codes which have been widely used for subsonic and supersonic internal flows respectively.

Walters et al.[84] have extended the compressible Euler flux-vector splitting concepts to PNS algorithms.

Baker et al.,[85–90] in addition to pioneering finite-element CFD algorithms/methods in general, also developed a robust finite-element based 3D PNS code applicable to subsonic, incompressible, external and internal flows. Fletcher et al.[91–93] have also developed a "finite-element based" finite-difference algorithm/procedure for subsonic flows. Using the pilot code developed for this dissertation as a basis, the author has conducted detailed convergence studies,

reported herein, and has contributed toward development of a new FE 3D PNS code for external hypersonics applications discussed in Chapter 3.

1.2 Scope

This dissertation documents development of an algorithm for solving the single-fluid (phase) compressible Navier-Stokes equations by a finite element (FE) methodology. The FE method is becoming increasingly more popular with CFD scientists and engineers as can be witnessed by simply counting the number of technical papers appearing in the proceedings of international CFD conferences each year.[5] Perhaps this is because it can be shown that the FE methodology embodies whole families of predecessor and current deterministic numerical methods as subsets including finite difference (FD) and finite volume (FV) methods. Because FE theory is integral-based, consistency with the original partial differential equation (PDE) is guaranteed.

Indeed, the origin of the FE method can be viewed as a variational principle approximation to obtaining the exact variational functional solution for problems formulated within the variational calculus branch of mathematics. That is, an exact variational functional is usually not possible to obtain, except in rare cases such as the heat conduction equation, so the solution must be approximated by a variational principle. These types of variational calculus approximations, such as FE methods, preserve consistency with the original conservation laws. Hence, a separate proof of consistency, typical of finite difference methods, is not necessary. This minimizes the need to validate arbitrary schemes that could lead to inconsistencies between the discrete formulation and the original PDE to be solved. The mathematics foundation not only provides for the richness

and versatility of the method, but also largely accounts for the difficulty in implementation for those not familiar with the theory.[94]

The algorithm essentials have been demonstrated[7] extensible to three dimensions (3D). However, only two dimensional (2D) solutions are the subject of this dissertation, as defined in the scope of this original research. The algorithm design is such that a steady viscous 2D compressible Navier-Stokes solution may be obtained in sequential steps, each step final solution used to initialize the initial conditions for the next step. The typical sequence constitutes a quasi-1D (where possible) inviscid, 2D inviscid, 2D laminar viscous, and finally a 2D turbulent viscous solution. Time-accurate solutions are obtainable within the nonlinear iterative procedure developed. However, only steady calculations are sought and employed for algorithm performance validations.

To summarize, the developed FE CFD algorithm is unique and has advanced the CFD field through the combination of the following essential ingredients:

- a compressible, 2D Navier-Stokes equation system with volume-specific stagnation enthalpy selected as the energy variable
  $\rho H = \rho(h + \boldsymbol{u} \cdot \boldsymbol{u}/2) = \rho(e + p/\rho + \boldsymbol{u} \cdot \boldsymbol{u}/2)$,
- a fully generalized coordinate transformation,
- ideal- and real-gas thermodynamic equation of state for air,
- laminar and an algebraic turbulent model for genuine (physical) dissipation,
- artificial (numerical) dissipation derived and implemented as theoretically predicted via a Taylor series extension on a weak statement, without infringing on the action of true physical dissipation in viscous flow regions,

- a finite element semi-discretization of the physical space,
- an elementary, but functional multi-grid and restart procedure,
- a nonlinear, iterative time advancement procedure utilizing and comparing a single-step, variably-implicit (sometimes called weighted approximation) and a semi-implicit Runge-Kutta time integration,
- a tensor-product matrix factorization quasi-Newton iteration for efficiently approximating the fully coupled Newton iteration jacobian matrix,
- analysis leading to reduced jacobian complexity, hence significantly less computer resource requirements,
- verified well-posed Dirichlet and Neumann boundary conditions with implementation of significant special cases,
- a unique approach to initial condition generation, based on straight-forward implementation of restart capability,
- confirmation of stability, convergence (in time and space), and consistency, and
- assessment of efficient utilization of supercomputer resources including both vector and parallel processing.

In addition, two reduced forms of the governing equations are derived and examined. A quasi-1D form has been implemented into the algorithm to establish initial conditions for the 2D problem. A detailed analysis of a parabolized Navier-Stokes (PNS) form of the equations has determined the stability envelope for space-marching (initial-value) applicability. Implementation and examination of the FE PNS algorithm was not a specific goal of this dissertation project; hence, only select pertinent PNS results are presented.[85,86]

Demonstration of the full Navier-Stokes algorithm is performed on three benchmark problems. Each problem is of increasing complexity allowing for independent confirmation of algorithm essentials. Comparisons with analytical, experimental, and independent calculations give strong credibility to both the algorithm theory and the numerical results.

1.3 Organization

This dissertation is composed of nine additional chapters. Chapter 2 presents the equation system solved, with derivation details in the appendices. Chapter 3 discusses reduced forms of the equations, including quasi-1D and PNS formulations. Chapter 4 derives a modified form of the governing equations, to arrive at the mechanism for designing artificial dissipation, via a Taylor weak statement. Chapter 5 discusses the discrete approximation to the modified equation system, including the finite element spatial semi-discretization and expansion to the fully discrete form using a implicit time integration. A detailed discussion of the quasi-Newton jacobian matrix, and approximations, is also presented. Chapter 6 discusses the verification of well posed boundary conditions (Dirichlet, Neumann, and mixed), and a unique approach to the establishment of initial conditions. Theoretical expectations for algorithm performance are detailed in Chapter 7, including semi-discrete asymptotic (FE) convergence, jacobian matrix quality, time-step selection, and stability of the fully-discrete iterative procedure.

Chapter 8 presents the results of the application of the derived algorithm on a succession of progressively more demanding problems. A confined duct flow with convergent entrance establishes the basic physics for validation of all boundary condition issues and turbulent simulation initiation. A transonic wind tunnel extends the difficulty to include an internal normal shock and a much

more complex wall geometry. The establishment of a controlled level of artificial dissipation, independent of the problem solved, is demonstrated on both the quasi-1D reduction, and extended to the 2D inviscid and viscous class to demonstrate the ability to capture shock-boundary layer interaction. The third problem is a double-throat nozzle, which extends algorithm application towards a design problem in a production computing environment, and further documents algorithm versatility for truly complex geometries, boundary conditions, thermodynamics, and flows.

Chapter 9 documents pilot computer program (re)design and (re)enhancement for utilization of supercomputer resources, including vector and parallel computing efficiencies and the impact of real-gas computations. Chapter 10 presents the summary and conclusions organized by project scope along with some recommendations for future research.

Appendices A, B, and C derive the governing equations, the transformation to dimensionless form, and the fully generalized coordinate transformation respectively. The pilot code input description is documented in Appendix D. A mini-investigation of efficiencies of several band-matrix direct equation solvers is referenced and presented in Appendix E. Application of a previously developed digital time-step filter is referenced, and details are presented in Appendix F.

Chapter 2

The Compressible Navier-Stokes Problem Statement

This chapter presents the governing partial differential equations, the constitutive equations used to close the equation system, and the transformation to a generalized coordinate system.

2.1 Governing Partial Differential Equation System

Appendix A summarizes derivation of the governing partial differential equations used in this research. Appendix B then defines how the governing equations are transformed to dimensionless form. The final equation system, written in tensor index notation is

$$\mathcal{L}(\rho)=\frac{\partial \rho}{\partial t}+\frac{\partial m_i}{\partial x_i}=0, \tag{2.1a}$$

$$\begin{aligned}\mathcal{L}(m_i)=&\frac{\partial m_i}{\partial t}+\frac{\partial (u_j m_i)}{\partial x_j}+\mathrm{Eu}\frac{\partial p}{\partial x_i}\\ &-\frac{\partial}{\partial x_j}\left(\frac{\mu_e}{\mathrm{Re}}\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right)-\frac{2}{3}\frac{\mu_e}{\mathrm{Re}}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right)=0,\end{aligned} \tag{2.1b}$$

$$\begin{aligned}\mathcal{L}(g)=&\frac{\partial g}{\partial t}+\frac{\partial (u_i g)}{\partial x_i}+2\mathrm{EcEu}\frac{\partial p}{\partial \rho}\frac{\partial m_i}{\partial x_i}-\frac{\partial}{\partial x_i}\left(\frac{k_e}{\mathrm{PrRe}}\frac{\partial T}{\partial x_i}\right)\\ &-\frac{2\mathrm{Ec}}{\mathrm{Re}}\frac{\partial}{\partial x_i}\left(u_j\mu_e\left[\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}-\frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right]\right)=0.\end{aligned} \tag{2.1c}$$

Equations 2.1a, 2.1b, and 2.1c are the conservation laws of mass, momentum, and energy. They are written in divergence, or conservative form, i.e., mass conservation is implicit within each subsequent equation, which results in the appearance of density as a direct multiple within the state variable (ρ, m_i, g). The choice of state variable is customarily the set operated on by the partial time derivative. The definition of each variable and parameter in Equation 2.1a through 2.1c is

$t \equiv$ time,

$x_i \equiv$ spatial coordinate system spanning physical space of dimension n, $1 \leq i \leq n$,

$\rho \equiv$ density,

$m_i \equiv$ momentum scalar component, $= \rho u_i$ in the x_i coordinate system,

$u_i \equiv$ velocity scalar component parallel to m_i,

$p \equiv$ pressure,

Eu $\equiv$ Euler number $= \frac{p_o}{\rho_o a_o^2}$,

$\mu_e \equiv \mu + \mu_T$, effective viscosity, sum of the laminar viscosity evaluated from the thermodynamic state, and the turbulent "eddy viscosity" as evaluated by a turbulence model,

Re $\equiv$ Reynolds number $= \frac{\rho_o a_o L}{\mu_o}$,

$g \equiv$ volume-specific stagnation enthalpy $= \rho H$,

$H \equiv$ mass-specific stagnation enthalpy $= h + \mathrm{Ec}u^2$,

$h \equiv$ enthalpy $= e + p/\rho$,

$e \equiv$ mass-specific internal energy,

Ec $\equiv$ Eckert number $= \frac{\rho_o a_o^2}{2g_o}$,

$k_e \equiv k + k_T$ effective thermal conductivity, sum of the laminar thermal conductivity, evaluated via the thermodynamic equation of state, and the turbulent "thermal conductivity" as evaluated by a turbulence model,

Pr $\equiv$ Prandtl number $= \frac{\mu_o C_{p_o}}{k_o}$, and

$T \equiv$ temperature.

In this research, the two dimensional ($n = 2$) subset of Equations 2.1a through 2.1c is considered. In expanded form, the 2D equation system is

$$\mathcal{L}(\rho)=\frac{\partial \rho}{\partial t}+\frac{\partial m_1}{\partial x_1}+\frac{\partial m_2}{\partial x_2}=0, \tag{2.2a}$$

$$\begin{aligned}\mathcal{L}(m_1)=&\frac{\partial m_1}{\partial t}+\frac{\partial (u_1 m_1)}{\partial x_1}+\frac{\partial (u_2 m_1)}{\partial x_2}+\mathrm{Eu}\frac{\partial p}{\partial x_1}-\frac{1}{\mathrm{Re}}\left[\frac{4}{3}\frac{\partial}{\partial x_1}\left(\mu_e\frac{\partial u_1}{\partial x_1}\right)\right.\\ &\left.-\frac{2}{3}\frac{\partial}{\partial x_1}\left(\mu_e\frac{\partial u_2}{\partial x_2}\right)+\frac{\partial}{\partial x_2}\left(\mu_e\frac{\partial u_1}{\partial x_2}\right)+\frac{\partial}{\partial x_2}\left(\mu_e\frac{\partial u_2}{\partial x_1}\right)\right]=0,\end{aligned} \tag{2.2b}$$

$$\begin{aligned}\mathcal{L}(m_2)=&\frac{\partial m_2}{\partial t}+\frac{\partial (u_1 m_2)}{\partial x_1}+\frac{\partial (u_2 m_2)}{\partial x_2}+\mathrm{Eu}\frac{\partial p}{\partial x_2}-\frac{1}{\mathrm{Re}}\left[\frac{4}{3}\frac{\partial}{\partial x_2}\left(\mu_e\frac{\partial u_2}{\partial x_2}\right)\right.\\ &\left.-\frac{2}{3}\frac{\partial}{\partial x_2}\left(\mu_e\frac{\partial u_1}{\partial x_1}\right)+\frac{\partial}{\partial x_1}\left(\mu_e\frac{\partial u_1}{\partial x_2}\right)+\frac{\partial}{\partial x_1}\left(\mu_e\frac{\partial u_2}{\partial x_1}\right)\right]=0,\end{aligned} \tag{2.2c}$$

$$\begin{aligned}\mathcal{L}(g)=&\frac{\partial g}{\partial t}+\frac{\partial (u_1 g)}{\partial x_1}+\frac{\partial (u_2 g)}{\partial x_2}+2\mathrm{EcEu}\frac{\partial p}{\partial \rho}\left(\frac{\partial m_1}{\partial x_1}+\frac{\partial m_2}{\partial x_2}\right)-\frac{2\mathrm{Ec}}{\mathrm{Re}}\left\{\right.\\ &\frac{\partial}{\partial x_1}\left[u_1\left(\frac{4\mu_e}{3}\frac{\partial u_1}{\partial x_1}-\frac{2\mu_e}{3}\frac{\partial u_2}{\partial x_2}\right)+u_2\mu_e\left(\frac{\partial u_1}{\partial x_2}+\frac{\partial u_2}{\partial x_1}\right)\right]\\ &\left.+\frac{\partial}{\partial x_2}\left[u_1\mu_e\left(\frac{\partial u_1}{\partial x_2}+\frac{\partial u_2}{\partial x_1}\right)+u_2\left(\frac{4\mu_e}{3}\frac{\partial u_2}{\partial x_2}-\frac{2\mu_e}{3}\frac{\partial u_1}{\partial x_1}\right)\right]\right\}\\ &-\frac{1}{\mathrm{RePr}}\left(\frac{\partial}{\partial x_1}\left(k_e\frac{\partial T}{\partial x_1}\right)+\frac{\partial}{\partial x_2}\left(k_e\frac{\partial T}{\partial x_2}\right)\right)=0.\end{aligned} \tag{2.2d}$$

2.2 Constitutive Equations

The state variable in Equations 2.2a through 2.2d is denoted $q = \{\rho, m_1, m_2, g\}^T$, i.e., those variables operated on by the time partial derivative. The velocity vector is thus a function of the state variable as $u_i = m_i/\rho$. The remaining variables, (p, μ_e, k_e, T) are also implicit functions of the state variable, as defined through constitutive equations.

Thermodynamic Equation of State

A general dimensionless thermodynamic equation of state (see Appendix B) is

$$p = z\rho T, \tag{2.3}$$

where z is a compressibility factor. The temperature definition is then

$$T = \frac{1}{\rho C_p}\left(g - \frac{\mathrm{Ec}(m_i m_i)}{\rho}\right), \tag{2.4}$$

and the heat capacity is

$$C_p = \frac{\gamma(\gamma_o - 1)}{\gamma_o(\gamma - 1)}, \tag{2.5}$$

where $\gamma = h/e$ and γ_o is the stagnation reference value of γ. For an ideal-gas, the compressibility factor is unity and $\gamma = 1.4$ for air.

For real-gas calculations, the compressibility factor and specific-heat ratio are not determined directly[a] but indirectly utilizing the TGAS[95] equilibrium air equation of state computer library package. The TGAS library returns equilibrium air properties as a function of density and internal energy $p = p(\rho, e)$ in Standard International (SI) units. Since the internal energy must be iteratively determined from the state variable selected in this research, i.e., $p = p(\rho, g)$, and since the

[a]If z and γ are really needed, they can be computed after the thermodynamic state is known from the definitions $z = p/(\rho T)$ and $\gamma = h/e$.

variable reference values are in U.S. customary units, additional computational work is required for a real-gas calculation. (This worst-case real-gas overhead requires approximately 10% additional work to perform a typical TGAS real-gas closure in 2D, see Chapter 9).

Unless specifically stated as an ideal-gas closure, all simulation results in this dissertation are performed with real-gas computations.

Closure for Laminar Flow

For the inviscid flow model, the laminar viscosity is assumed to vanish. The thermodynamic equation of state closes the equation system by providing pressure and $\frac{\partial p}{\partial \rho}$.

For a viscous flow case, the momentum and energy equations require closure for viscosity, thermal conductivity and temperature. In this research project, real-gas closure provided by the TGAS library are employed for most simulations. For an ideal-gas air simulation, only the temperature is directly available from Equation 2.4, for constant γ. Therefore, the ideal-gas fluid viscosity and thermal conductivity could be computed via Sutherland's law[96] for air viscosity as a function of temperature only. However, for those cases where ideal-gas simulation is employed, a unit value is used for the dimensionless viscosity and thermal conductivity as an approximation appropriate for transonic flows.

Closure for Turbulent Flow

The turbulent flow case requires a closure model for turbulent (eddy) viscosity μ_T and turbulent thermal conductivity k_T. Appendix A summarizes the definition of the aerodynamics-based Baldwin-Lomax[97] algebraic closure model. The earlier

and more fundamental Cebeci-Smith[98] algebraic model was also investigated. Either turbulence model is adequate to meet the research goal of verifying algorithm stability in the presence of mesh demands created by a turbulent flow simulation.

2.3 Generalized Coordinate Transformation

A generalized coordinate transformation (GCT), derived in Appendix C, transforms Equations 2.2a through 2.2d from the Cartesian system x_i ((x, y, z) or (x_1, x_2, x_3)) to an intrinsic curvilinear system η_i ((ξ, η, ζ) or (η_1, η_2, η_3)). Reduced to 2D for implementation herein, the transformation directly facilitates accurate semi-discretizations of domains bounded by arbitrary boundary shapes. Strongly coupled with the finite element semi discretization to be described, the drawback to the approach is that each spatial derivative in the governing equation system is expanded by a factor of two for each first-order spatial derivative (convection terms) and by four for each second-order spatial derivative (viscous terms).

The final 2D form for the governing system, Equations 2.2a through 2.2d, becomes

$$\mathcal{L}(\rho)=\frac{\partial \rho}{\partial t} + \left(\frac{\partial \eta_1}{\partial x_1}\frac{\partial}{\partial \eta_1} + \frac{\partial \eta_2}{\partial x_1}\frac{\partial}{\partial \eta_2}\right) m_1 + \left(\frac{\partial \eta_1}{\partial x_2}\frac{\partial}{\partial \eta_1} + \frac{\partial \eta_2}{\partial x_2}\frac{\partial}{\partial \eta_2}\right) m_2 = 0, \tag{2.6a}$$

$$\begin{aligned}\mathcal{L}(m_1)=&\frac{\partial m_1}{\partial t} + \mathrm{Eu}\left(\frac{\partial \eta_1}{\partial x_1}\frac{\partial}{\partial \eta_1} + \frac{\partial \eta_2}{\partial x_1}\frac{\partial}{\partial \eta_2}\right) p \\ &+ \left(\frac{\partial \eta_1}{\partial x_1}\frac{\partial}{\partial \eta_1} + \frac{\partial \eta_2}{\partial x_1}\frac{\partial}{\partial \eta_2}\right) u_1 m_1 + \left(\frac{\partial \eta_1}{\partial x_2}\frac{\partial}{\partial \eta_1} + \frac{\partial \eta_2}{\partial x_2}\frac{\partial}{\partial \eta_2}\right) u_2 m_1 \\ &- \frac{\mu_e}{\mathrm{Re}}\left\{\frac{4}{3}\left[\left(\frac{\partial \eta_1}{\partial x_1}\right)^2 \frac{\partial^2}{\partial \eta_1^2} + 2\left(\frac{\partial \eta_1}{\partial x_1}\frac{\partial \eta_2}{\partial x_1}\right)\frac{\partial^2}{\partial \eta_1 \eta_2} + \left(\frac{\partial \eta_2}{\partial x_1}\right)^2 \frac{\partial^2}{\partial \eta_2^2}\right] u_1 \right. \\ &+ \left[\left(\frac{\partial \eta_1}{\partial x_2}\right)^2 \frac{\partial^2}{\partial \eta_1^2} + 2\left(\frac{\partial \eta_1}{\partial x_2}\frac{\partial \eta_2}{\partial x_2}\right)\frac{\partial^2}{\partial \eta_1 \eta_2} + \left(\frac{\partial \eta_2}{\partial x_2}\right)^2 \frac{\partial^2}{\partial \eta_2^2}\right] u_1\end{aligned}$$

$$+\frac{1}{3}\left[\left(\frac{\partial\eta_1}{\partial x_1}\frac{\partial\eta_1}{\partial x_2}\right)\frac{\partial^2}{\partial\eta_1^2}+\left(\frac{\partial\eta_2}{\partial x_1}\frac{\partial\eta_2}{\partial x_2}\right)\frac{\partial^2}{\partial\eta_2^2}\right.$$
$$\left.+\left\{\left(\frac{\partial\eta_1}{\partial x_1}\frac{\partial\eta_2}{\partial x_2}\right)+\left(\frac{\partial\eta_2}{\partial x_1}\frac{\partial\eta_1}{\partial x_2}\right)\right\}\frac{\partial^2}{\partial\eta_1\eta_2}\right]u_2\Bigg\}=0, \tag{2.6b}$$

$$\mathcal{L}(m_2)=\frac{\partial m_2}{\partial t}+\mathrm{Eu}\left(\frac{\partial\eta_1}{\partial x_2}\frac{\partial}{\partial\eta_1}+\frac{\partial\eta_2}{\partial x_2}\frac{\partial}{\partial\eta_2}\right)p$$
$$+\left(\frac{\partial\eta_1}{\partial x_1}\frac{\partial}{\partial\eta_1}+\frac{\partial\eta_2}{\partial x_1}\frac{\partial}{\partial\eta_2}\right)u_1m_2+\left(\frac{\partial\eta_1}{\partial x_2}\frac{\partial}{\partial\eta_1}+\frac{\partial\eta_2}{\partial x_2}\frac{\partial}{\partial\eta_2}\right)u_2m_2$$
$$-\frac{\mu_e}{\mathrm{Re}}\left\{\frac{4}{3}\left[\left(\frac{\partial\eta_1}{\partial x_2}\right)^2\frac{\partial^2}{\partial\eta_1^2}+2\left(\frac{\partial\eta_1}{\partial x_2}\frac{\partial\eta_2}{\partial x_2}\right)\frac{\partial^2}{\partial\eta_1\eta_2}+\left(\frac{\partial\eta_2}{\partial x_2}\right)^2\frac{\partial^2}{\partial\eta_2^2}\right]u_2\right.$$
$$+\left[\left(\frac{\partial\eta_1}{\partial x_1}\right)^2\frac{\partial^2}{\partial\eta_1^2}+2\left(\frac{\partial\eta_1}{\partial x_1}\frac{\partial\eta_2}{\partial x_1}\right)\frac{\partial^2}{\partial\eta_1\eta_2}+\left(\frac{\partial\eta_2}{\partial x_1}\right)^2\frac{\partial^2}{\partial\eta_2^2}\right]u_2$$
$$+\frac{1}{3}\left[\left(\frac{\partial\eta_1}{\partial x_1}\frac{\partial\eta_1}{\partial x_2}\right)\frac{\partial^2}{\partial\eta_1^2}+\left(\frac{\partial\eta_2}{\partial x_1}\frac{\partial\eta_2}{\partial x_2}\right)\frac{\partial^2}{\partial\eta_2^2}\right.$$
$$\left.+\left\{\left(\frac{\partial\eta_1}{\partial x_1}\frac{\partial\eta_2}{\partial x_2}\right)+\left(\frac{\partial\eta_2}{\partial x_1}\frac{\partial\eta_1}{\partial x_2}\right)\right\}\frac{\partial^2}{\partial\eta_1\eta_2}\right]u_1\Bigg\}=0, \tag{2.6c}$$

$$\mathcal{L}(g)=\frac{\partial g}{\partial t}+\left(\frac{\partial\eta_1}{\partial x_1}\frac{\partial}{\partial\eta_1}+\frac{\partial\eta_2}{\partial x_1}\frac{\partial}{\partial\eta_2}\right)u_1g+\left(\frac{\partial\eta_1}{\partial x_2}\frac{\partial}{\partial\eta_1}+\frac{\partial\eta_2}{\partial x_2}\frac{\partial}{\partial\eta_2}\right)u_2g$$
$$+2\mathrm{EcEu}\frac{\partial p}{\partial\rho}\left(\left(\frac{\partial\eta_1}{\partial x_1}\frac{\partial}{\partial\eta_1}+\frac{\partial\eta_2}{\partial x_1}\frac{\partial}{\partial\eta_2}\right)m_1+\left(\frac{\partial\eta_1}{\partial x_2}\frac{\partial}{\partial\eta_1}+\frac{\partial\eta_2}{\partial x_2}\frac{\partial}{\partial\eta_2}\right)m_2\right)$$
$$-\frac{2\mathrm{Ec}\mu_e}{\mathrm{Re}}\left\{\left\{\frac{4u_1}{3}\left[\left(\frac{\partial\eta_1}{\partial x_1}\right)^2\frac{\partial^2}{\partial\eta_1^2}+2\left(\frac{\partial\eta_1}{\partial x_1}\frac{\partial\eta_2}{\partial x_1}\right)\frac{\partial^2}{\partial\eta_1\eta_2}+\left(\frac{\partial\eta_2}{\partial x_1}\right)^2\frac{\partial^2}{\partial\eta_2^2}\right]\right.\right.$$
$$+u_1\left[\left(\frac{\partial\eta_1}{\partial x_2}\right)^2\frac{\partial^2}{\partial\eta_1^2}+2\left(\frac{\partial\eta_1}{\partial x_2}\frac{\partial\eta_2}{\partial x_2}\right)\frac{\partial^2}{\partial\eta_1\eta_2}+\left(\frac{\partial\eta_2}{\partial x_2}\right)^2\frac{\partial^2}{\partial\eta_2^2}\right]$$
$$+\frac{u_2}{3}\left[\left(\frac{\partial\eta_1}{\partial x_1}\frac{\partial\eta_1}{\partial x_2}\right)\frac{\partial^2}{\partial\eta_1^2}+\left(\frac{\partial\eta_2}{\partial x_1}\frac{\partial\eta_2}{\partial x_2}\right)\frac{\partial^2}{\partial\eta_2^2}\right.$$
$$\left.\left.+\left\{\left(\frac{\partial\eta_1}{\partial x_1}\frac{\partial\eta_2}{\partial x_2}\right)+\left(\frac{\partial\eta_2}{\partial x_1}\frac{\partial\eta_1}{\partial x_2}\right)\right\}\frac{\partial^2}{\partial\eta_1\eta_2}\right]\right\}u_1$$
$$-\left\{\frac{4u_2}{3}\left[\left(\frac{\partial\eta_1}{\partial x_2}\right)^2\frac{\partial^2}{\partial\eta_1^2}+2\left(\frac{\partial\eta_1}{\partial x_2}\frac{\partial\eta_2}{\partial x_2}\right)\frac{\partial^2}{\partial\eta_1\eta_2}+\left(\frac{\partial\eta_2}{\partial x_2}\right)^2\frac{\partial^2}{\partial\eta_2^2}\right]\right.$$
$$+u_2\left[\left(\frac{\partial\eta_1}{\partial x_1}\right)^2\frac{\partial^2}{\partial\eta_1^2}+2\left(\frac{\partial\eta_1}{\partial x_1}\frac{\partial\eta_2}{\partial x_1}\right)\frac{\partial^2}{\partial\eta_1\eta_2}+\left(\frac{\partial\eta_2}{\partial x_1}\right)^2\frac{\partial^2}{\partial\eta_2^2}\right]$$
$$+\frac{u_1}{3}\left[\left(\frac{\partial\eta_1}{\partial x_1}\frac{\partial\eta_1}{\partial x_2}\right)\frac{\partial^2}{\partial\eta_1^2}+\left(\frac{\partial\eta_2}{\partial x_1}\frac{\partial\eta_2}{\partial x_2}\right)\frac{\partial^2}{\partial\eta_2^2}\right.$$

$$+\left\{\left(\frac{\partial\eta_1}{\partial x_1}\frac{\partial\eta_2}{\partial x_2}\right)+\left(\frac{\partial\eta_2}{\partial x_1}\frac{\partial\eta_1}{\partial x_2}\right)\right\}\frac{\partial^2}{\partial\eta_1\eta_2}\Bigg]\Bigg\}u_2\Bigg\}$$

$$-\frac{k_e}{\mathrm{RePr}}\Bigg\{\Bigg[\left(\frac{\partial\eta_1}{\partial x_1}\right)^2\frac{\partial^2}{\partial\eta_1^2}+2\left(\frac{\partial\eta_1}{\partial x_1}\frac{\partial\eta_2}{\partial x_1}\right)\frac{\partial^2}{\partial\eta_1\eta_2}+\left(\frac{\partial\eta_2}{\partial x_1}\right)^2\frac{\partial^2}{\partial\eta_2^2}\Bigg]$$

$$+\Bigg[\left(\frac{\partial\eta_1}{\partial x_2}\right)^2\frac{\partial^2}{\partial\eta_1^2}+2\left(\frac{\partial\eta_1}{\partial x_2}\frac{\partial\eta_2}{\partial x_2}\right)\frac{\partial^2}{\partial\eta_1\eta_2}+\left(\frac{\partial\eta_2}{\partial x_2}\right)^2\frac{\partial^2}{\partial\eta_2^2}\Bigg]\Bigg\}T=0. \qquad (2.6d)$$

For the sake of clarity (only), μ_e in the momentum Equations 2.6b and 2.6c, and μ_e, k_e, and u_i in the energy Equation 2.6d, have been extracted outside the higher-order derivative on all terms. In developing the FE algorithm (Chapter 5), use of a Green's theorem and integration by parts replaces this notational convenience with the proper representations.

The complex appearance of Equations 2.6a through 2.6d certainly could be reduced through utilization of index notation. However, the number of differential terms required for inclusion in the developed FE CFD code is apparent, even for this 2D reduction of the governing compressible Navier-Stokes equation system.

Chapter 3

Reduced Forms of the Governing Equations

Two reduced forms of the governing equations are considered. The steady-state space-marching form, traditionally called "parabolized Navier-Stokes (PNS)," allows for significantly reduced computer resources for solving a limited type of steady 2D and 3D viscous flows. The main feature of PNS is elimination of the outflow boundary condition. The disadvantage is that severely limited pressure variations, hence stability constraints, reduce applicability. The next section discusses the essential theoretical development of a finite-element based PNS algorithm and code.[85,86,90] Since the linear algebra statement for a 2D-NS time-marching problem is identical to 3D-PNS space-marching, the basis for the PNS option of the code system is essentially developed from this dissertation project.

A time-dependent, quasi-1D reduced form is also developed for including in the pilot code. Limited to internal flows with defined cross-sectional area, the fast and efficient quasi-1D solutions are extremely useful in establishing initial conditions for the 2D problem statements.

3.1 Parabolized Navier-Stokes

Introduction

The fundamental requirement for PNS space marching is that the flow is unidirectional (usually parallel to η_1). The predominate flow direction η_1 becomes the initial-value variable for space marching, hence utilizing a finite-difference scheme on $\Delta\eta_1$. The full discretization is completed by spanning the remaining coordinate variables (η_2 for 2D and η_2, η_3 for 3D) by a finite-element semi-discretization. Thus, the basic algorithm and code framework for a 2D, time-integration finite-element algorithm is very similar to that for a 3D space-marching or PNS algorithm/code.

A stability analysis of the governing equations defined by Equations 2.2a through 2.2d has been investigated with respect to PNS applicability.[89] In exactly the same manner as defined in Appendix A and B, the system of equations is reduced to 2D. The equation system is further reduced by assuming steady-state, thus eliminating the time dependence. A generalized coordinate transformation (Appendix C) was not included in the stability analysis. This leaves the state variable $q = \{\rho, m_1, m_2, g\}^T$ a function of the two independent variables x_1 and x_2. The stability analysis reveals that handling of the streamwise pressure gradient in subsonic regions can be responsible for unstable space marching, called the "departure solution." Thus, the essential distinction among the various PNS algorithms is the treatment of streamwise pressure gradient.

Symbolic Analysis

The original derivation of "symbolic analysis" is presented by Fletcher and others.[92] This research project sought to use this methodology to unify the theory

among the numerous published PNS methods. Denoting the Reynolds-Averaged Navier-Stokes state variable as $q = \{\rho, m_i, g\}^T$, the Fourier representation for the two dimensional form is

$$q=\frac{1}{4\pi^2}\sum_{k=-\infty}^{\infty}\sum_{j=-\infty}^{\infty}\hat{q}_{kj}\exp\{i\sigma_x jx\}\exp\{i\sigma_y ky\}, \tag{3.1a}$$

where i denotes $\sqrt{-1}$ and $\hat{q}$ denotes the Fourier transform of q. Upon linearization, only a single component of q in Equation 3.1a need be considered, hence,

$$q = \frac{1}{4\pi^2}\hat{q}\exp\{i\sigma_x x\}\exp\{i\sigma_y y\}. \tag{3.2}$$

Substitution of Equation 3.2 into all possible combinations of the two dimensional subset of the spatial partial derivatives from the governing equations results in the following set of equations

$$\frac{\partial q}{\partial x}=i\sigma_x q, \tag{3.3a}$$

$$\frac{\partial q}{\partial y}=i\sigma_y q, \tag{3.3b}$$

$$\frac{\partial^2 q}{\partial x^2}=-\sigma_x^2 q, \tag{3.3c}$$

$$\frac{\partial^2 q}{\partial y^2}=-\sigma_y^2 q, \text{ and} \tag{3.3d}$$

$$\frac{\partial^2 q}{\partial x\partial y}=-\sigma_x\sigma_y q. \tag{3.3e}$$

Substituting Equations 3.3a through 3.3e into the governing equations yields a system of equations of the form $[A]q = 0$, where the matrix $[A]$ is the "symbol" of the governing equation system. Stability information about the governing equation system can be immediately derived from the symbol. In particular, the roots of the determinant of the principal symbol (the diagonal real part) of $[A]$ reveals the system of governing equations to be

- elliptic, if no non-zero real roots exist ,

- hyperbolic, if the number of real roots equals the order of the symbol, or
- parabolic, if only one real root exits.

The symbolic analysis is expanded further by forming the complete determinant of $[A]$. Since the stability analysis of interest concerns a procedure for space-marching, and if it is assumed that the predominant direction is x, a polynomial is formed from the determinant of $[A]$ with σ_x as the independent variable. The roots to the characteristic polynomial in σ_x are then examined for exponential growth in x when substituted back into Equation 3.2. In particular, for a negative imaginary root of σ_x, exponential growth will occur and a space marching procedure will be unstable. The stability analysis of a space-marching procedure then reduces to determining what combination of state variable, fluid properties, Reynolds number, Prandtl number, and Euler number produces a negative, imaginary root for σ_x.

Several examples of the symbolic analysis procedure are given by Fletcher.[93] The remaining discussion concerns the application of the symbolic analysis to the governing Equations 2.2a through 2.2d.

Specific Reduced Forms

In addition to the complete equation system defined by Equations 2.2a through 2.2d, several reduced forms were examined for PNS applicability. These include

- Euler -- inviscid form defined by setting the fluid viscosity to zero,
- thin-layer Navier-Stokes -- viscous form defined by deleting all second order diffusion terms containing a partial derivative with respect to the predominant direction of flow,

- boundary-layer Form 1 -- viscous form defined by additionally deleting all convection terms in the transverse momentum equation, and the laminar portion of the viscous shear according to a Reynolds ordering, and
- boundary-layer Form 2 -- viscous form defined by deleting all terms in the transverse momentum equation except the pressure gradient.

Details, via equations, of the reduced forms is contained in Baker and Freels.[89]

Stability Analysis

The symbolic analysis has been applied to the specific reduced forms to determine admissibility to space-marching solutions. If a particular form is unstable, then the PNS stability enhancement techniques of Vigneron[75] and Fletcher[91] are also analyzed. Table 3.1 summarizes the results, which verifies that, except for the boundary layer form 2 equation system, all equation systems are unstable for space marching subsonic flows without augmentation of the equation system. Conversely, all supersonic flows within the equation systems analyzed are stable for space marching.

Applicability

Any space-marching solution for viscous flows will always have subsonic flow regions and therefore must be stabilized. The boundary-layer form 2 is rarely the equation system used, therefore some augmentation of the original equation system (Vigneron, Fletcher, or others) is required in order to solve reduced forms of the Navier-Stokes equations. Furthermore, the full Navier-Stokes equation system is impossible to solve via space marching. Thus, any space-marching solution is only valid when certain terms are correctly assumed of

Table 3.1: Symbolic Space-Marching (PNS) Stability Analysis Results for the Reynolds-Averaged Navier-Stokes Equations and Reduced Forms

Equation Form	Stability	Comment
RANS	unstable	always
TLNS	unstable	subsonic flow with no augmentation
TLNS	stable	all supersonic flow
TLNS	stable	by Vigneron's augmentation if $u < M_x$ for all sonic regions
TLNS	possibly stable	by Fletcher's augmentation if Reynolds number is large for all sonic regions
BL-1	unstable	subsonic flow with no augmentation
BL-1	stable	supersonic flow
BL-1	stable	by Vigneron's augmentation if $u < M_x$ for all sonic regions
BL-1	possibly stable	by Fletcher's augmentation if Reynolds number is large for all sonic regions
BL-2	stable	always
Euler	stable	supersonic flow
Euler	unstable	subsonic
Euler	stable	by Vigneron's augmentation if $u < M_x$ for all sonic regions
Euler	possibly stable	by Fletcher's augmentation for all sonic regions

little consequence in the conservation law equation system. The trade off then becomes: When is the PNS solution sufficiently close to the true NS solution such that it (PNS solution) is acceptable?

3D Finite Element Space Marching

Development of a finite-element based 3D PNS hypersonic flow solver has recently been completed,[90] as a direct extension and modification of the pilot code system developed for this dissertation. The governing equations for the FE PNS form are exactly as derived in Appendices A and B. Since a pressure-relaxed PNS solution is only concerned with the steady form of the equations, the time partial derivative terms are eliminated. In addition, some of the complexity inherent to the coordinate transformation of the convection terms was eliminated by the introduction of contravarient velocity and momentum.[a] The final form of the developed FE PNS equations is

$$\mathcal{L}(M1)=\frac{\partial M1}{\partial \eta_1}+\frac{\partial M2}{\partial \eta_2}+\frac{\partial M3}{\partial \eta_3}=0, \tag{3.4a}$$

$$\begin{aligned}\mathcal{L}(u_1 M1)=&\frac{\partial u_1 M1}{\partial \eta_1}+\frac{\partial u_1 M2}{\partial \eta_2}+\frac{\partial u_1 M3}{\partial \eta_3}\\ &+\mathrm{Eu}\left(\omega_o\overline{\eta_{11}}\frac{\partial}{\partial \eta_1}+(\overline{\eta_{21}}-\epsilon_o)\frac{\partial}{\partial \eta_2}+\overline{\eta_{31}}\frac{\partial}{\partial \eta_3}\right)p\\ &-\left(K_{11}\frac{\partial^2}{\partial \eta_2^2}+K_{12}\frac{\partial^2}{\partial \eta_3^2}+K_{13}\frac{\partial^2}{\partial \eta_2 \partial \eta_3}\right)u_1\\ &-\left(K_{21}\frac{\partial^2}{\partial \eta_2^2}+K_{22}\frac{\partial^2}{\partial \eta_3^2}+K_{23}\frac{\partial^2}{\partial \eta_2 \partial \eta_3}\right)u_2\\ &-\left(K_{31}\frac{\partial^2}{\partial \eta_2^2}+K_{32}\frac{\partial^2}{\partial \eta_3^2}+K_{33}\frac{\partial^2}{\partial \eta_2 \partial \eta_3}\right)u_3=0,\end{aligned} \tag{3.4b}$$

$$\mathcal{L}(u_2 M1)=\frac{\partial u_2 M1}{\partial \eta_1}+\frac{\partial u_2 M2}{\partial \eta_2}+\frac{\partial u_2 M3}{\partial \eta_3}$$

[a] In hindsight, it would have been better to utilize contravariant velocities in the NS form of the governing equations solved in this dissertation.

$$
\begin{aligned}
&+\mathrm{Eu}\left(\overline{\eta_{12}}\frac{\partial}{\partial\eta_1}+\overline{\eta_{22}}\frac{\partial}{\partial\eta_2}+\overline{\eta_{32}}\frac{\partial}{\partial\eta_3}\right)p\\
&-\left(L_{11}\frac{\partial^2}{\partial\eta_2^2}+L_{12}\frac{\partial^2}{\partial\eta_3^2}+L_{13}\frac{\partial^2}{\partial\eta_2\partial\eta_3}\right)u_1\\
&-\left(L_{21}\frac{\partial^2}{\partial\eta_2^2}+L_{22}\frac{\partial^2}{\partial\eta_3^2}+L_{23}\frac{\partial^2}{\partial\eta_2\partial\eta_3}\right)u_2\\
&-\left(L_{31}\frac{\partial^2}{\partial\eta_2^2}+L_{32}\frac{\partial^2}{\partial\eta_3^2}+L_{33}\frac{\partial^2}{\partial\eta_2\partial\eta_3}\right)u_3=0, \qquad (3.4c)
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{L}(u_3M1)=&\frac{\partial u_3M1}{\partial\eta_1}+\frac{\partial u_3M2}{\partial\eta_2}+\frac{\partial u_3M3}{\partial\eta_3}\\
&+\mathrm{Eu}\left(\overline{\eta_{13}}\frac{\partial}{\partial\eta_1}+\overline{\eta_{23}}\frac{\partial}{\partial\eta_2}+\overline{\eta_{33}}\frac{\partial}{\partial\eta_3}\right)p\\
&-\left(M_{11}\frac{\partial^2}{\partial\eta_2^2}+M_{12}\frac{\partial^2}{\partial\eta_3^2}+M_{13}\frac{\partial^2}{\partial\eta_2\partial\eta_3}\right)u_1\\
&-\left(M_{21}\frac{\partial^2}{\partial\eta_2^2}+M_{22}\frac{\partial^2}{\partial\eta_3^2}+M_{23}\frac{\partial^2}{\partial\eta_2\partial\eta_3}\right)u_2\\
&-\left(M_{31}\frac{\partial^2}{\partial\eta_2^2}+M_{32}\frac{\partial^2}{\partial\eta_3^2}+M_{33}\frac{\partial^2}{\partial\eta_2\partial\eta_3}\right)u_3=0, \qquad (3.4d)
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{L}(HM1)=&\frac{\partial HM1}{\partial\eta_1}+\frac{\partial HM2}{\partial\eta_2}+\frac{\partial HM3}{\partial\eta_3}\\
&-\left(N_{11}\frac{\partial^2}{\partial\eta_2^2}+N_{12}\frac{\partial^2}{\partial\eta_3^2}+N_{13}\frac{\partial^2}{\partial\eta_2\partial\eta_3}\right)u_1\\
&-\left(N_{21}\frac{\partial^2}{\partial\eta_2^2}+N_{22}\frac{\partial^2}{\partial\eta_3^2}+N_{23}\frac{\partial^2}{\partial\eta_2\partial\eta_3}\right)u_2\\
&-\left(N_{31}\frac{\partial^2}{\partial\eta_2^2}+N_{32}\frac{\partial^2}{\partial\eta_3^2}+N_{33}\frac{\partial^2}{\partial\eta_2\partial\eta_3}\right)u_3\\
&-\left(N_{41}\frac{\partial^2}{\partial\eta_2^2}+N_{42}\frac{\partial^2}{\partial\eta_3^2}+N_{43}\frac{\partial^2}{\partial\eta_2\partial\eta_3}\right)T=0, \qquad (3.4e)
\end{aligned}
$$

where

- the k index denotes a curvilinear coordinate direction,
- the j index denotes the Cartesian coordinate direction,

- M_i and U_i denote contravariant momentum and velocity respectively in the ith transformed direction, i.e., η_i direction,
- η_1 is the space marching direction of choice with the corresponding contravariant velocity scalar component $U1$,
- and the coefficients K_{jk}, L_{jk}, M_{jk}, and N_{jk} contain constants and metric data utilized in the viscous terms and are detailed in the cited PNS reports.

The PNS state variable appearing in the η_1 partial derivative terms yields $\{q\}_{\text{PNS}} = \{M1, u_1 M1, u_2 M1, u_3 M1, H M1\}^T$. It is very interesting that denoting the state variable $\{q\} = \{q_1, q_2, q_3, q_4, q_5\}^T$, an analog Navier-Stokes state variable is (see Chapter 5) $\{q\}_{\text{ns}} = \{\rho, m_1, m_2, m_3, g\}^T$. This allows an identical relationship to exist for the velocity vector and the stagnation enthalpy for both the PNS and NS form of the state variable

$$u_1 = \frac{q_2}{q_1}, \tag{3.5a}$$

$$u_2 = \frac{q_3}{q_1}, \tag{3.5b}$$

$$u_3 = \frac{q_4}{q_1}, \text{ and} \tag{3.5c}$$

$$H = \frac{q_5}{q_1}. \tag{3.5d}$$

The unique relationships defined by Equations 3.5a through 3.5d are among many that allow for a clean interface to exist between the NS and PNS variables. The next generation of code may then evolve as a smart algorithm to automatically switch from the NS to the PNS solution, and vice versa, based on the constraints of the PNS solution, with both algorithms residing within the same program.

3.2 Quasi-One-Dimensional

For internal flow problems, it is often possible and advantageous to solve a quasi-1D reduction of the problem to quickly establish initial conditions for the 2D problem. Following the equation derivation as presented by Moore,[25] the quasi-1D Euler equation system is

$$\mathcal{L}(\rho)=\frac{\partial \rho}{\partial t}+\frac{\partial m_1}{\partial x_1}+\left(\frac{m_1}{A}\right)\frac{\partial A}{\partial x_1}=0, \tag{3.6a}$$

$$\mathcal{L}(m_1)=\frac{\partial m_1}{\partial t}+\frac{\partial (u_1 m_1)}{\partial x_1}+\left(\frac{u_1 m_1}{A}\right)\frac{\partial A}{\partial x_1}+\mathrm{Eu}\frac{\partial p}{\partial x_1}=0, \tag{3.6b}$$

$$\mathcal{L}(m_2)=0, \tag{3.6c}$$

$$\mathcal{L}(g)=\frac{\partial g}{\partial t}+\frac{\partial (u_1 g)}{\partial x_1}+2\mathrm{EcEu}\frac{\partial p}{\partial \rho}\frac{\partial m_1}{\partial x_1}+\left(\frac{u_1 g+2\mathrm{EcEu}\frac{\partial p}{\partial \rho}m_1}{A}\right)\frac{\partial A}{\partial x_1}=0. \tag{3.6d}$$

The resulting system is reduced from four to three equations. Furthermore, because there is only a single spatial dimension, the solution complexity is significantly reduced. Each of the Equations 3.6a through 3.6d contains a time partial derivative, a convection spatial derivative on momentum, and a source with spatial derivative on cross-sectional area. The momentum Equation 3.6b additionally contains the pressure partial derivative as a source/coupling term. This reduced equation system provides a truly significant advantage in establishing initial conditions for internal, 2D inviscid flow problems.

Chapter 4

Taylor Weak Statement

There are phenomena in fluid mechanics and heat transfer, such as shocks, that are difficult if not impossible to simulate without numerical or artificial dissipation. Experience has shown that numerical algorithms or schemes that do not include sufficient artificial dissipation will not produce smooth, monotone, and stable solutions for aerodynamic problems which include shocks. Kim[28] documents this assertion by performing detailed analyses on a number of non-, anti-, and dissipative algorithms. Therefore, a basic necessary requirement for the algorithm is creation of "just adequate" dissipation, and to prevent its masking of the genuine viscous dissipation phenomena present in Navier-Stokes solutions.

Methods for the creation of artificial dissipation vary among researchers and analysts. Baker[27] and Kim[28] verify that about sixteen independently derived artificial dissipation algorithms are duplicated by appropriate choice of an initially arbitrary parameter set within the Taylor Weak Statement (TWS). Methods such as "flux vector splitting",[99] "Beam-Warming",[100] "Lax-Wendroff",[101] "Raymond-Garder"[102] and many others can all be rederived within the TWS. One of these methods, referred to as "dissipative-Galerkin" by Kim, and earlier as "Penalty-Galerkin" (PG) by Baker,[87] is an extension of the Raymond-Garder method.

The PG method had proven quite successful, but offers no real mathematical basis for assessing success other than the fact that it worked! Donea[103] derived the referenced "Taylor-Galerkin" method, which is also generalized within the foundation of TWS.[27]

4.1 Derivation

The TWS derivation first expresses the governing equation system in hyperbolic conservation law form

$$\mathcal{L}(q) = \frac{\partial q}{\partial t} + \nabla \cdot (\boldsymbol{f} - \boldsymbol{f}^v) - s = 0, \tag{4.1}$$

where $q = \{\rho, m_i, g\}^T$ is the state variable, $\boldsymbol{f}$ is the kinematic-kinetic flux vector, $\boldsymbol{f}^v$ is the dissipative flux vector arising from the viscous constitutive closure model, and s an the array of source terms if present. Artificial dissipation is sought (required) only in the inviscid flow regions where natural dissipation is not important (or present). Therefore, the inviscid form of Equation 4.1 is appropriate for the TWS derivation.

The source-free inviscid form of Equation 4.1 for TWS analysis is the Euler system

$$\mathcal{L}^E(q) = \frac{\partial q}{\partial t} + \frac{\partial f_i}{\partial x_i} = \frac{\partial q}{\partial t} + \frac{\partial f_i}{\partial q}\frac{\partial q}{\partial x_i} = \frac{\partial q}{\partial t} + A_i\frac{\partial q}{\partial x_i} = 0. \tag{4.2}$$

The flux vector jacobian $A_i = \frac{\partial f_i}{\partial q}$ is introduced, and the superscript "E" denotes the Euler reduction to Equation 4.1. The state variable q is semi-discretized in time yielding

$$q^{n+1} = q^n + \left(\frac{\partial q}{\partial t}\right)^n \Delta t + \left(\frac{\partial^2 q}{\partial t^2}\right)^n \frac{\Delta t^2}{2} + \left(\frac{\partial^3 q}{\partial t^3}\right)^n \frac{\Delta t^3}{6} + \mathcal{O}(\Delta t^4), \tag{4.3}$$

where $(\cdot)^n$ denotes time level and $\Delta t = t^{n+1} - t^n$. Equation 4.3, divided through by Δt, yields

$$\frac{q^{n+1} - q^n}{\Delta t} = \left(\frac{\partial q}{\partial t}\right)^n + \left(\frac{\partial^2 q}{\partial t^2}\right)^n \frac{\Delta t}{2} + \left(\frac{\partial^3 q}{\partial t^3}\right)^n \frac{\Delta t^2}{6} + \mathcal{O}(\Delta t^3), \tag{4.4}$$

which is a third order approximation for the state variable change over Δt.

Equation 4.2 allows interchange of temporal and spatial derivatives in Equation 4.4, as originally developed by Lax and Wendroff.[101] The first-order term substitution yields

$$\frac{\partial q}{\partial t} = -\frac{\partial f_i}{\partial x_i}. \tag{4.5}$$

The second-order-term substitution produces two forms, i.e.,

$$\frac{\partial^2 q}{\partial t^2} = \frac{\partial}{\partial t}\left(\frac{\partial q}{\partial t}\right) = -\frac{\partial}{\partial t}\left(\frac{\partial f_i}{\partial x_i}\right) = -\frac{\partial}{\partial x_i}\left(\frac{\partial f_i}{\partial t}\right) = \frac{\partial}{\partial x_i}\left(\frac{\partial f_i}{\partial q}\frac{\partial q}{\partial t}\right) = \frac{\partial}{\partial x_i}\left(A_i \frac{\partial q}{\partial t}\right), \tag{4.6}$$

and the convex combination is

$$\frac{\partial^2 q}{\partial t^2} = \frac{\partial}{\partial x_i}\left(A_i\left(\alpha\frac{\partial q}{\partial t} + \beta\frac{\partial f_j}{\partial x_j}\right)\right), \tag{4.7}$$

where α and β are arbitrary parameters to within the convex constraint $-\alpha + \beta = 1$. The third-order-term substitution yields, in a similar manner

$$\frac{\partial^3 q}{\partial t^3} = \frac{\partial}{\partial x_i}\left(A_i \frac{\partial}{\partial x_j} A_j \left(\gamma\frac{\partial q}{\partial t} + \mu\frac{\partial f_k}{\partial x_k}\right)\right), \tag{4.8}$$

subject to the constraint $-\gamma + \mu = 1$.

Equations 4.5, 4.7, and 4.8 substituted into Equation 4.4, and taking the limit $\Delta t \to 0$ and rearranging by partial derivative type yields the Taylor-series--modified or "companion" conservation law statement for Equation 4.1 as

$$\begin{aligned}\mathcal{L}^m(q) = &\left(1 - A_i\frac{\partial}{\partial x_i}\left(\alpha\frac{\Delta t}{2} + \gamma\frac{\Delta t^2}{6}A_j\frac{\partial}{\partial x_j}\right)\right)\frac{\partial q}{\partial t} - s \\ &+ \frac{\partial}{\partial x_i}\left(f_i - f_i^v - A_i\left(\beta\frac{\Delta t}{2}A_j\frac{\partial q}{\partial x_j} + \mu\frac{\Delta t^2}{6}A_j\frac{\partial}{\partial x_j}A_k\frac{\partial q}{\partial x_k}\right)\right) = 0.\end{aligned} \tag{4.9}$$

Several comments regarding Equation 4.9 are appropriate. The parent Navier-Stokes conservation law statement, Equation 4.1, is recovered for the TWS parameter definition $\alpha = \beta = \gamma = \mu = 0$. For this research project, only steady-state solutions are sought. Therefore, the TWS parameters operating on the time derivative are not pertinent, hence $\alpha = \gamma = 0$. Further, the μ term was not considered based on the research by Kim.[28] Therefore, the only nonzero TWS parameter pertinent to these dissertation results is the β coefficient (set) in Equation 4.9.[a] As will be validated, this selection was sufficient for stable solution of problems that include shock interaction with viscous flow regions.

The time step Δt appearing in Equation 4.9 can be eliminated by the introduction of a reference Courant number $\mathsf{C}_{\underline{j}}$

$$\mathsf{C}_{\underline{j}} = \frac{\mathsf{U}_{\underline{j}} \Delta t}{\mathsf{L}_{\underline{j}}}, \tag{4.10}$$

where the underbar denotes an "indicator" index not eligible for implied tensor notation summation, and $\mathsf{U}_{\underline{j}}$, and $\mathsf{L}_{\underline{j}}$ are reference speed and length scale respectively, to be precisely defined in Chapter 5. The only restriction is that they be positive. For the reference Courant number inclusion with β, the TWS-modified conservation law statement replacing Equation 4.1 is

$$\mathcal{L}^m(q) = \frac{\partial q}{\partial t} + \frac{\partial}{\partial x_i}\left(f_i - f_i^v - A_i\left(\beta \frac{\mathsf{L}_{\underline{j}}}{2\mathsf{U}_{\underline{j}}} A_j \frac{\partial q}{\partial x_j}\right)\right) - s = 0. \tag{4.11}$$

The reference length and velocity scales, the inviscid flux vector jacobian A_i and inner products, along with β remain to be quantified.

[a] All the TWS parameters were incorporated into the pilot code in a straight forward manner similar to the β term, investigated to some extent, and left for future research (see Chapter 5).

4.2 Inviscid Flux Vector Jacobian and Modified Forms

Expanding the conservation law statement, Equation 4.11, in two dimensional Cartesian coordinates yields

$$\begin{aligned}\mathcal{L}^m(q)=&\frac{\partial q}{\partial t}+\frac{\partial(f_x-f_x^v)}{\partial x}+\frac{\partial(f_y-f_y^v)}{\partial y}\\&-\frac{\partial}{\partial x}\left(\frac{A_1\beta}{2}\left(\frac{\mathsf{L}_1A_1}{\mathsf{U}_1}\frac{\partial q}{\partial x}+\frac{\mathsf{L}_2A_2}{\mathsf{U}_2}\frac{\partial q}{\partial y}\right)\right)\\&-\frac{\partial}{\partial y}\left(\frac{A_2\beta}{2}\left(\frac{\mathsf{L}_1A_1}{\mathsf{U}_1}\frac{\partial q}{\partial x}+\frac{\mathsf{L}_2A_2}{\mathsf{U}_2}\frac{\partial q}{\partial y}\right)\right)=0.\end{aligned}\tag{4.12}$$

This section derives the flux vector jacobian matrices A_1 and A_2, and the required matrix products A_1^2, A_2^2, A_1A_2, and A_2A_1.

Extracting from Equation 4.12 the partial derivative of the x-component of the inviscid flux vector with respect to the x direction and denoting the partial derivative by subscript x yields

$$\frac{\partial f_x}{\partial x}=\left\{\begin{array}{c}(m_1)_x\\(um_1)_x+\mathrm{Eu}p_x\\(um_2)_x\\(ug)_x+2\mathrm{EuEc}\frac{a^2}{\gamma}(m_1)_x\end{array}\right\}.\tag{4.13}$$

Similarly,

$$\frac{\partial f_y}{\partial y}=\left\{\begin{array}{c}(m_2)_y\\(vm_1)_y\\(vm_2)_y+\mathrm{Eu}p_y\\(vg)_y+2\mathrm{EuEc}\frac{a^2}{\gamma}(m_2)_y\end{array}\right\}.\tag{4.14}$$

The final form of this flux vector jacobian $\left(A_i=\frac{\partial f_{x_i}}{\partial q}\right)$ depends on the treatment of the pressure, and the relationship assumed between density and the state variable. Five forms have been evaluated. The first two adhere to a strict definition of the jacobian formulation by treating the state variable array as

fully independent (including density). The remaining forms constitute various approximations.

The first form treats pressure in the momentum equations strictly as a function of the state variable. Real-gas effects are explicitly accounted for via the compressibility factor and specific heat. This is referred to as the "form 1" inviscid flux vector jacobian (F1IFVJ).

The second form (F2IFVJ) treats pressure directly as a function of the equation of state. Because this results in the speed of sound and specific heat ratio being without a spatial derivative, the complexity of the F2IFVJ form is reduced from that of F1IFVJ.

The third and fourth forms (F3IFVJ and F4IFVJ respectively) are obtained by taking advantage of the relationship between the momentum and energy state variable members and the density. By expressing the state variable in this fashion, the first column of the jacobian matrices are modified. The distinction between the F3IFVJ and F4IFVJ forms is similar to that of distinguishing F1IFVJ and F2IFVJ, via treatment of pressure in the momentum equations.

The fifth form (F5IFVJ) is obtained as a simplification. The development (formulation of the inviscid flux vector jacobian) objective is to simplify the direct method for implementing numerical stability, via the TWS, to not overburden the algorithm with additional terms, hence calculational work.

Pressure Determined from State Variable

The thermodynamic equation of state with volume-specific stagnation enthalpy as the energy variable is

$$p = \frac{z}{C_p}\left[g - \mathrm{Ec}\left(\frac{m_1^2 + m_2^2}{\rho}\right)\right], \tag{4.15}$$

where $z = \left(\frac{p}{\rho T}\right)$. In general,[104] the pressure differential expanded as a function of the state variable is

$$dp = \frac{\partial p}{\partial \rho} d\rho + \frac{\partial p}{\partial m_1} dm_1 + \frac{\partial p}{\partial m_2} dm_2 + \frac{\partial p}{\partial g} dg. \tag{4.16}$$

In this algorithm, the pressure is determined from an equation of state as a function of density ρ and volume-specific stagnation enthalpy g. [b] Expanding again as a function of the thermodynamic independent variables, Equation 4.16 becomes

$$\begin{aligned} dp = & \left[\frac{\partial p}{\partial \rho}\frac{\partial \rho}{\partial \rho} + \frac{\partial p}{\partial g}\frac{\partial g}{\partial \rho}\right] d\rho + \left[\frac{\partial p}{\partial \rho}\frac{\partial \rho}{\partial m_1} + \frac{\partial p}{\partial g}\frac{\partial g}{\partial m_1}\right] dm_1 \\ & + \left[\frac{\partial p}{\partial \rho}\frac{\partial \rho}{\partial m_2} + \frac{\partial p}{\partial g}\frac{\partial g}{\partial m_2}\right] dm_2 + \left[\frac{\partial p}{\partial \rho}\frac{\partial \rho}{\partial g} + \frac{\partial p}{\partial g}\frac{\partial g}{\partial g}\right] dg. \end{aligned} \tag{4.17}$$

Realizing that $\left[\frac{\partial \rho}{\partial \rho} = \frac{\partial g}{\partial g} = 1\right]$, $\left[\frac{\partial \rho}{\partial m_1} = \frac{\partial \rho}{\partial m_2} = \frac{\partial \rho}{\partial g} = \frac{\partial g}{\partial \rho} = 0\right]$, utilizing Equation 4.15 to define p and g, and ignoring the thermodynamic variation of the compressibility factor z and heat capacity C_p yields

$$dp = z^* \left[\mathrm{Ec}\left\{\left(u^2 + v^2\right) d\rho + 2\left(u dm_1 + v dm_2\right)\right\} + dg\right], \tag{4.18}$$

where $z^* = \frac{z}{C_p}$.

Pressure Determined Directly from an Equation of State

A relationship often utilized for the thermodynamic state of a fluid[106] is

$$a^2 = g_c \frac{C_p}{C_v}\frac{\partial p}{\partial \rho} = g_c \left.\frac{\partial p}{\partial \rho}\right|_{\text{isentropic}}. \tag{4.19}$$

[b] The equation-of-state package[105] actually solves for the fluid thermodynamic state as a function of the density ρ and mass-specific internal energy e in SI engineering units of (kg/m^3) and (m^2/sec^2) respectively. The pilot code iterates with ρ and g as the independent variables, in US common engineering units of (lbm/ft^3) and (Btu/ft^3) respectively, until an equation of state solution is obtained with an acceptable level of error tolerance. The choice of solving pressure as a function of the density and volume-specific stagnation enthalpy $p(\rho, g)$ is a matter of convenience.

Equation 4.19 can be incorporated into the governing equations by utilizing

$$\mathrm{Eu}\partial p = \frac{a^2}{\gamma}\partial\rho, \tag{4.20}$$

where a^2 is made nondimensional by a_o^2 (see Appendix B). The specific heat ratio $\gamma = \frac{C_p}{C_v}$ is by default nondimensional, hence is <u>not</u> referenced to γ_o.

Equations 4.19 and 4.20 are easily verified. Assuming ideal-gas and isentropic flow for this demonstration, the relationships $\mathrm{Eu} = 1/\gamma_o$ and $a^2 = \gamma RT$ are true. Substituting into Equation 4.20, and referencing temperature to T_o yields the familiar ideal gas identity

$$\left.\frac{\partial p}{\partial \rho}\right|_{\text{ideal}} = T. \tag{4.21}$$

In general, fluids are not ideal nor are fluid processes typically isentropic. This requires a^2 and γ to be determined from real-gas relationships. Therefore, it is imperative that the right hand side of the discrete form of the governing equations retain pressure gradients as written directly on pressure. The pressure is expressed in terms of state variable only for evaluating partial derivatives in the inviscid flux vector jacobian and system iteration matrix jacobian of the fully discrete form (defined later). Equations 4.20 and 4.18 are both available for this purpose, with Equation 4.20 more desirable due to simplicity.

F1IFVJ

Utilizing pressure considered strictly as a function of the state variable, and defining $a^* = 2\mathrm{EuEc}\left(\frac{a^2}{\gamma}\right)$, the partial derivative of the x-component of the

inviscid flux vector with respect to the x direction becomes

$$\frac{\partial f_x}{\partial x} = \left\{ \begin{array}{c} (m_1)_x \\ \left(\frac{m_1^2}{\rho}\right)_x + z^* \left[\mathrm{Ec}\left\{\left(u^2+v^2\right)\rho_x + 2\left(u\,(m_1)_x + v\,(m_2)_x\right)\right\} + g_x\right] \\ \left(\frac{m_1 m_2}{\rho}\right)_x \\ \left(\frac{m_1 g}{\rho}\right)_x + a^*\,(m_1)_x \end{array} \right\}. \tag{4.22}$$

Similarly,

$$\frac{\partial f_y}{\partial y} = \left\{ \begin{array}{c} (m_2)_y \\ \left(\frac{m_1 m_2}{\rho}\right)_y \\ \left(\frac{m_2^2}{\rho}\right)_y + z^* \left[\mathrm{Ec}\left\{\left(u^2+v^2\right)\rho_y + 2\left(u\,(m_1)_y + v\,(m_2)_y\right)\right\} + g_y\right] \\ \left(\frac{m_2 g}{\rho}\right)_y + a^*\,(m_2)_y \end{array} \right\}. \tag{4.23}$$

Performing the required differentiation of each entry in the array of the x-component of the inviscid flux vector with respect to each differentiated state dependent variable yields the F1IFVJ form in the x direction

$$\frac{\partial f_x}{\partial q} = A_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ z^*\mathrm{Ec}\left(u^2+v^2\right) - u^2 & 2u\,(z^*\mathrm{Ec}+1) & 2vz^*\mathrm{Ec} & z^* \\ -uv & v & u & 0 \\ -uH & H + a^* & 0 & u \end{bmatrix}. \tag{4.24}$$

Hence, also

$$\frac{\partial f_y}{\partial q} = A_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ -uv & v & u & 0 \\ z^*\mathrm{Ec}\left(u^2+v^2\right) - v^2 & 2uz^*\mathrm{Ec} & 2v\,(z^*\mathrm{Ec}+1) & z^* \\ -vH & 0 & H + a^* & v \end{bmatrix}. \tag{4.25}$$

The information for the TWS quantities A_1^2, A_2^2, A_1A_2, and A_2A_1 is now available. For the F1IFVJ form, each resulting matrix is approximated by its

diagonal entries only. Hence,

$$A_1^2 = A_1 A_1 \cong \text{diag}\lceil A_1 A_1 \rfloor =$$

$$\begin{bmatrix} \left\{ \begin{matrix} z^*\gamma^* \left(u^2 + v^2\right) \\ -u^2 \end{matrix} \right\} & 0 & 0 & 0 \\ 0 & \left\{ \begin{matrix} u^2 \left(4\,(z^*\text{Ec})^2 \right. \\ +9z^*\text{Ec} + 3) \\ +v^2\,(3z^*\text{Ec}) \\ +(H + a^*)\,z^* \end{matrix} \right\} & 0 & 0 \\ 0 & 0 & \left\{ \begin{matrix} 2v^2 z^*\text{Ec} \\ +u^2 \end{matrix} \right\} & 0 \\ 0 & 0 & 0 & \left\{ \begin{matrix} z^*\,(H + a^*) \\ +u^2 \end{matrix} \right\} \end{bmatrix}, \tag{4.26a}$$

$$A_2^2 = A_2 A_2 \cong \text{diag}\lceil A_2 A_2 \rfloor =$$

$$\begin{bmatrix} \left\{ \begin{matrix} z^*\gamma^* \left(u^2 + v^2\right) \\ -v^2 \end{matrix} \right\} & 0 & 0 & 0 \\ 0 & \left\{ \begin{matrix} 2u^2 z^*\text{Ec} \\ +v^2 \end{matrix} \right\} & 0 & 0 \\ 0 & 0 & \left\{ \begin{matrix} v^2 \left(4\,(z^*\text{Ec})^2 \right. \\ +9z^*\text{Ec} + 3) \\ +u^2\,(3z^*\text{Ec}) \\ +z^*\,(H + a^*) \end{matrix} \right\} & 0 \\ 0 & 0 & 0 & \left\{ \begin{matrix} z^*\,(H + a^*) \\ +v^2 \end{matrix} \right\} \end{bmatrix}, \tag{4.26b}$$

and

$$A_1 A_2 = A_2 A_1 \cong \text{diag}\lceil A_2 A_1 \rfloor =$$

$$\begin{bmatrix} -uv & 0 & 0 & 0 \\ 0 & \left\{ \begin{matrix} 2uv\,(z^*\mathrm{Ec}+1) \\ +4uv\,(z^*\mathrm{Ec})^2 \end{matrix} \right\} & 0 & 0 \\ 0 & 0 & 2uv\,(z^*\mathrm{Ec}+1) & 0 \\ 0 & 0 & 0 & uv \end{bmatrix}. \tag{4.26c}$$

F2IFVJ

Utilizing pressure directly as a function of the equation of state yields

$$\frac{\partial f_x}{\partial x} = \begin{Bmatrix} (m_1)_x \\ \left(\frac{m_1^2}{\rho}\right)_x + \frac{a^2}{\gamma}\rho_x \\ \left(\frac{m_1 m_2}{\rho}\right)_x \\ \left(\frac{m_1 g}{\rho}\right)_x + a^*\,(m_1)_x \end{Bmatrix}. \tag{4.27}$$

A similar expression for the y direction is

$$\frac{\partial f_y}{\partial y} = \begin{Bmatrix} (m_2)_y \\ \left(\frac{m_1 m_2}{\rho}\right)_y \\ \left(\frac{m_2^2}{\rho}\right)_x + \frac{a^2}{\gamma}\rho_y \\ \left(\frac{m_2 g}{\rho}\right)_y + a^*\,(m_2)_y \end{Bmatrix}. \tag{4.28}$$

The inviscid flux vector jacobians are then

$$\frac{\partial f_x}{\partial q} = A_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \left(\frac{a^2}{\gamma} - u^2\right) & 2u & 0 & 0 \\ -uv & v & u & 0 \\ -uH & H + a^* & 0 & u \end{bmatrix}, \tag{4.29}$$

and

$$\frac{\partial f_y}{\partial q} = A_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ -uv & v & u & 0 \\ (\frac{a^2}{\gamma} - v^2) & 0 & 2v & 0 \\ -vH & 0 & H + a^* & v \end{bmatrix}. \tag{4.30}$$

For the earlier definition, the terms for F2IFVJ form are

$$A_1^2 = A_1 A_1 \cong \text{diag}\lceil A_1 A_1 \rfloor = \begin{bmatrix} \frac{a^2}{\gamma} - u^2 & 0 & 0 & 0 \\ 0 & \frac{a^2}{\gamma} + 3u^2 & 0 & 0 \\ 0 & 0 & u^2 & 0 \\ 0 & 0 & & u^2 \end{bmatrix}, \tag{4.31a}$$

$$A_2^2 = A_2 A_2 \cong \text{diag}\lceil A_2 A_2 \rfloor = \begin{bmatrix} \frac{a^2}{\gamma} - v^2 & 0 & 0 & 0 \\ 0 & v^2 & 0 & 0 \\ 0 & 0 & \frac{a^2}{\gamma} + 3v^2 & 0 \\ 0 & 0 & 0 & v^2 \end{bmatrix}, \tag{4.31b}$$

and (4.31c)

$$A_1 A_2 = A_2 A_1 \cong \text{diag}\lceil A_2 A_1 \rfloor = \begin{bmatrix} -uv & 0 & 0 & 0 \\ 0 & 2uv & 0 & 0 \\ 0 & 0 & 2uv & 0 \\ 0 & 0 & 0 & uv \end{bmatrix}. \tag{4.31d}$$

F3IFVJ

The results of the inviscid flux vector forms F1IFVJ and F2IFVJ present an interesting dilemma. From past experience with artificial dissipation mechanisms, it is necessary that the A_i matrix products on the diagonal have positive sign. It is on this premise that a modified definition of the inviscid flux vector jacobian is now considered. The expansion of the momentum and energy state variable $m_i = \rho u_i$ and $g = \rho H$ is utilized for each jacobian entry which requires a partial

derivative with respect to density $\left(\frac{\partial(\cdot)}{\partial \rho}\right)$. Thus, a form of the inviscid flux vector jacobian, in which only the first column is changed from the prior definition, is obtained.

This inviscid flux vector form is identical to Equation 4.22 for the x direction and 4.23 for the y direction. Performing the required differentiation of each entry in the state variable yields in the x direction

$$\frac{\partial f_x}{\partial q} = A_1 = \begin{bmatrix} u & 1 & 0 & 0 \\ \left\{ \begin{matrix} u^2(1+z^*(2+\mathrm{Ec})) \\ +v^2(z^*\mathrm{Ec}) \end{matrix} \right\} & 2u(z^*\mathrm{Ec}+1) & 2vz^*\mathrm{Ec} & z^* \\ 0 & v & u & 0 \\ u(H+a^*) & H+a^* & 0 & u \end{bmatrix}, \quad (4.32)$$

and in the y direction

$$\frac{\partial f_y}{\partial q} = A_2 = \begin{bmatrix} v & 0 & 1 & 0 \\ 0 & v & u & 0 \\ \left\{ \begin{matrix} v^2(1+z^*(2+\mathrm{Ec}))+ \\ u^2(z^*\mathrm{Ec}) \end{matrix} \right\} & 2uz^*\mathrm{Ec} & 2v(z^*\mathrm{Ec}+1) & z^* \\ v(H+a^*) & 0 & H+a^* & v \end{bmatrix}. \quad (4.33)$$

The F3IFVJ form is then

$A_1^2 = A_1 A_1 \cong \mathrm{diag}\lceil A_1 A_1 \rfloor =$

$$\begin{bmatrix} \left\{ \begin{matrix} u^2(2 \\ +z^*(2+\mathrm{Ec})) \\ +v^2(z^*\mathrm{Ec}) \end{matrix} \right\} & 0 & 0 & 0 \\ 0 & \left\{ \begin{matrix} u^2\left(4(z^*\mathrm{Ec})^2 \right. \\ +9z^*\mathrm{Ec}+5) \\ +2z^* \\ +v^2\left(4(z^*\mathrm{Ec})^2 \right. \\ +z^*\mathrm{Ec})+ \\ z^*(H+a^*) \end{matrix} \right\} & 0 & 0 \\ 0 & 0 & u^2+2v^2z^*\mathrm{Ec} & 0 \\ 0 & 0 & 0 & \left\{ \begin{matrix} z^*(H+a^*) \\ +u^2 \end{matrix} \right\} \end{bmatrix},$$

(4.34a)

$A_2^2 = A_2A_2 \cong \mathrm{diag}\lceil A_2A_2 \rfloor =$

$$\begin{bmatrix} \left\{ \begin{matrix} v^2(2 \\ +z^*(2+\mathrm{Ec})) \\ +u^2(z^*\mathrm{Ec}) \end{matrix} \right\} & 0 & 0 & 0 \\ 0 & v^2+2u^2z^*\mathrm{Ec} & 0 & 0 \\ 0 & 0 & \left\{ \begin{matrix} v^2\left(4(z^*\mathrm{Ec})^2 \right. \\ +9z^*\mathrm{Ec}+5) \\ +2z^* \\ +u^2\left(4(z^*\mathrm{Ec})^2 \right. \\ +z^*\mathrm{Ec}) \\ +z^*(H+a^*) \end{matrix} \right\} & 0 \\ 0 & 0 & 0 & \left\{ \begin{matrix} z^*(H+a^*) \\ +v^2 \end{matrix} \right\} \end{bmatrix},$$

and

(4.34b)

$$A_1 A_2 = A_2 A_1 \cong \text{diag}\lceil A_2 A_1 \rfloor =$$

$$\begin{bmatrix} uv & 0 & 0 & 0 \\ 0 & \left\{ \begin{matrix} 2uv\,(z^*\text{Ec} + 1) \\ +4uv\,(z^*\text{Ec})^2 \end{matrix} \right\} & 0 & 0 \\ 0 & 0 & uv\,(2z^*\text{Ec} + 3) & 0 \\ 0 & 0 & 0 & uv \end{bmatrix}. \tag{4.34c}$$

The matrices given by Equations 4.34a, 4.34b, and 4.34c have positive sign on all diagonal entries. Furthermore, the magnitude of the terms multiplied by z^* and Ec are less than unity.

F4IFVJ

The inviscid flux vectors for the F4IFVJ form are identical to Equation 4.27 in the x direction and 4.28 in the y direction. Performing the required differentiation of each entry in the array yields in the x direction

$$\frac{\partial f_x}{\partial q} = A_1 = \begin{bmatrix} u & 1 & 0 & 0 \\ (\frac{a^2}{\gamma} + u^2) & 2u & 0 & 0 \\ 0 & v & u & 0 \\ u(H + a^*) & H + a^* & 0 & u \end{bmatrix}, \tag{4.35}$$

and in the y direction

$$\frac{\partial f_y}{\partial q} = A_2 = \begin{bmatrix} v & 0 & 1 & 0 \\ 0 & v & u & 0 \\ (\frac{a^2}{\gamma} + v^2) & 0 & 2v & 0 \\ v(H + a^*) & 0 & H + a^* & v \end{bmatrix}. \tag{4.36}$$

The F4IFVJ form is then

$$A_1^2 = A_1 A_1 \cong \text{diag}\lceil A_1 A_1 \rfloor = \begin{bmatrix} \frac{a^2}{\gamma} + 2u^2 & 0 & 0 & 0 \\ 0 & \frac{a^2}{\gamma} + 5u^2 & 0 & 0 \\ 0 & 0 & u^2 & 0 \\ 0 & 0 & 0 & u^2 \end{bmatrix}, \quad (4.37a)$$

$$A_2^2 = A_2 A_2 \cong \text{diag}\lceil A_2 A_2 \rfloor = \begin{bmatrix} \frac{a^2}{\gamma} + 2v^2 & 0 & 0 & 0 \\ 0 & v^2 & 0 & 0 \\ 0 & 0 & \frac{a^2}{\gamma} + 5v^2 & 0 \\ 0 & 0 & 0 & v^2 \end{bmatrix}, \quad (4.37b)$$

and (4.37c)

$$A_1 A_2 = A_2 A_1 \cong \text{diag}\lceil A_2 A_1 \rfloor = \begin{bmatrix} uv & 0 & 0 & 0 \\ 0 & 2uv & 0 & 0 \\ 0 & 0 & 2uv & 0 \\ 0 & 0 & 0 & uv \end{bmatrix}. \quad (4.37d)$$

F5IFVJ

Irrespective of the form of the inviscid flux vector jacobian or approximation, the TWS will enhance numerical stability in a nonlinear fashion through variations in velocity. Furthermore, directional dependence is also incorporated via length and speed scales. From past experience, it can be assumed that the fluid velocity variation is the dominant requirement for proper numerical dissipation control. This leads to the cited reductions of the inviscid flux vector jacobian complexity by eliminating variations in the non-velocity variables. This is done by arriving at the F5IFVJ form through a rational process.

First, the F1IFVJ and F2IFVJ forms are no longer considered because the resulting A_i matrices are not positive definite. The F3IFVJ and F4IFVJ forms

are not ideal because the magnitude of the elements of the resulting A_i matrices do not tend to zero as the velocity tends to zero. The feature of having a direct multiplication of the velocity upon each matrix element is desired because in regions of low velocity (such as within a boundary layer), numerical dissipation must not overwhelm the physical dissipation (via viscous effect). Therefore, the presented forms for F3IFVJ and F4IFVJ possess a numerical dissipation bias.

Consider eliminating the off-diagonal elements of the A_i matrices prior to the matrix multiplication process. This simplification yields matrix elements which are directly proportional to the velocity without dissipation bias. Furthermore, with the exception of a factor of $(z^*\mathrm{Ec} + 1)$, the F3IFVJ and F4IFVJ simplified forms are identical. Furthermore, the approximate order of magnitude of the dimensionless quantity $(z^*\mathrm{Ec} + 1)$ can be obtained by assuming that z and C_p (which are variables) have a value close to unity and that $\widehat{\gamma_o}$ (which is a constant) has a value of ≈ 1.4 for air. This will yield

$$z^*\mathrm{Ec} \approx \frac{1}{1} \times \frac{1.4 - 1}{2} = \frac{1}{5}.$$

Since this quantity is nearly an order of magnitude smaller than unity, it can be assumed small by comparison. The ramifications of this assumption can be verified by examining the steady-solution distribution of $z^*\mathrm{Ec}$.

With these two assumptions, the F5IFVJ form is

$$A_1 \cong \begin{bmatrix} u & 0 & 0 & 0 \\ 0 & 2u & 0 & 0 \\ 0 & 0 & u & 0 \\ 0 & 0 & 0 & u \end{bmatrix}, \tag{4.38a}$$

$$A_2 \cong \begin{bmatrix} v & 0 & 0 & 0 \\ 0 & v & 0 & 0 \\ 0 & 0 & 2v & 0 \\ 0 & 0 & 0 & v \end{bmatrix}, \tag{4.38b}$$

$$A_1^2 = A_1 A_1 \cong \begin{bmatrix} u^2 & 0 & 0 & 0 \\ 0 & 4u^2 & 0 & 0 \\ 0 & 0 & u^2 & 0 \\ 0 & 0 & 0 & u^2 \end{bmatrix}, \tag{4.38c}$$

$$A_2^2 = A_2 A_2 \cong \begin{bmatrix} v^2 & 0 & 0 & 0 \\ 0 & v^2 & 0 & 0 \\ 0 & 0 & 4v^2 & 0 \\ 0 & 0 & 0 & v^2 \end{bmatrix}, \tag{4.38d}$$

and (4.38e)

$$A_1 A_2 = A_2 A_1 \cong \begin{bmatrix} uv & 0 & 0 & 0 \\ 0 & 2uv & 0 & 0 \\ 0 & 0 & 2uv & 0 \\ 0 & 0 & 0 & uv \end{bmatrix}. \tag{4.38f}$$

which is quite simple. Though zero throughout the validation results shown in this dissertation, the TWS α term is implemented by a direct multiplication of u from A_1, or x-direction terms and of v from A_2, or y-direction terms. The momentum terms in the respective directions are weighted by an additional factor of two.

The TWS γ terms, although zero for all validation results, is implemented in a manner similar to the TWS β terms by a direct multiplication of u^2 and v^2 from the A_1^2 and A_2^2 or x- and y-directions terms respectively. The cross terms created by the $A_1 A_2$ and $A_2 A_1$ multiplications contain direct multiplication by uv. Again, a weighting toward the momentum equations exists by additional

integer multiplications of 4 and 2 on the A_i^2 and A_iA_j terms respectively. These additional weightings on the β term are consistent with previous work.[25]

Additional simplification to the F5IFVJ form could be made by eliminating the integer weighting coefficients entirely and incorporating the weighting into the input values of α,β, and γ. On the other hand, it is possible, that upon determination of an optimum set of TWS parameters, a unified distribution of TWS parameters are obtained across the entire state variable array. In other words, a single user-input value for the TWS parameters α, β, and γ exist while, transparent to the user, a weighting of coefficients exists within the algorithm.

Conclusions

The F5IFVJ form is computationally preferred, and derived as a straight-forward approximation. Because of a requirement of matrix positivity for positive velocity, the partial derivatives with respect to density in the inviscid flux vector jacobians is relaxed by taking advantage of the density dependence within the state variable. The simplification of eliminating off-diagonals in the jacobian prior to matrix multiplication, and an order-of-magnitude assumption, yields consistency with both interpretations of the pressure expansion. The validity of the order-of-magnitude assumption of the z^*Ec parameter has been verified (not shown) by examining steady-state solution distributions, such as those to be shown in Chapter 8, and found to validate the assumption across the entire solution domain (variations of less than .14%). This assumption has proven to be acceptable, thus a simple inviscid flux vector jacobian has been obtained that retains all the desired computational features for numerical dissipation.

4.3 β Parameter Determination

The TWS parameter β magnitude appropriate for transonic shock capturing, was determined via benchmark experiments. For inviscid flows, a constant value is utilized throughout the solution domain remaining consistent with previous research related to the TWS,[25, 28, 29, 35] i.e., the TWS theory derivation holds for different forms of the conservation law equations. The value chosen for this research project is

$$\frac{\beta}{2} = \frac{\beta_o}{2\sqrt{15}}, \tag{4.39}$$

where β_o is input data. The integer 2 is retained for derivational consistency, and $\sqrt{15}$ results from a Fourier stability analysis on the earlier Penalty-Galerkin method in 1D.[87] Numerical experimentation (Chapter 8) indicated that β_o in the range $3.0 \rightarrow 4.0$ yielded the best results. Unless otherwise noted, all computations reported in this dissertation use $\beta_o = 3$; hence $\frac{\beta}{2} \approx .387$. The analysis of Iannelli[35] indicates that $\beta/2 \leq 0.5$ is required for stability; hence this experimental determination is consistent with theory. Further, β_o is appropriate for multidimensional flow, as attributed to the intrinsic generation of cross-term dissipation developed by the TWS β term in transformed curvilinear space (Chapter 5).

For viscous-dominated flow reaches, β must be modulated to ensure that artificial dissipation does not compromise physical (fluid viscosity) dissipation. This need is realized by comparing representative artificial and natural dissipation terms, e.g.,

$$\frac{\mu}{\mathrm{Re}}\frac{\partial^2 u}{\partial y^2} \quad \text{as compared to} \quad \left(\frac{\beta_o}{2\sqrt{15}}\right) A_j \frac{\underline{\mathsf{L}_i}}{\underline{\mathsf{U}_i}} A_i \frac{\partial^2 u}{\partial y^2},$$

where underbar denotes an "indicator" <u>not</u> eligible for implied tensor summation rule. The inviscid flux vector jacobian product yields terms on the order u^2;

eliminating like terms then yields the comparison

$$\frac{\mu}{\mathrm{Re}} \quad \text{as compared to} \quad \frac{\beta}{2}\mathsf{L}u.$$

Assuming the length scale $\mathsf{L}/2$ and (non-dimensional) viscosity μ are of order unity, for natural dissipation to dominate in a viscous layer region requires

$$\beta u \ll \mathrm{Re}^{-1}. \tag{4.40}$$

The velocity magnitude tends to zero as the wall is approached. However, for typical Reynolds numbers ($10^3 \rightarrow 10^5$), extremely small velocities would be required to remove the action of artificial dissipation from the viscous flow region.

Hence, a damping function was derived and applied to β in all viscous flow regions for problems simulated in this dissertation. Following considerable computational experimentation, smooth transition between the inviscid and viscous flow region accrues to

$$\beta_f = 1 - k1 \exp[-\{(u - k2)/u_{ref}\}^{k3}]. \tag{4.41}$$

Here, u is the local (nodal) value of velocity magnitude, u_{ref} is a reference velocity, and $k1, k2$, and $k3$ are scaling parameters. Figure 4.1 graphs Equation 4.41 for $k1 = 1, k2 = 0$ and various values of $k3$.

Substituting Equation 4.41 modifies β_o for viscous problem regions to the form

$$\beta_o^v = \beta_f \beta_o. \tag{4.42}$$

For all viscous (laminar and turbulent) simulations reported herein, the parameter set $(k1, k2, k3)$ was set to $(1, 0, 20)$, and the reference velocity was either the centerline or freestream velocity. This choice assures that no artificial dissipation exists in subsonic flow reaches anywhere in the solution domain, especially in the viscous boundary layer (with separation) regions.

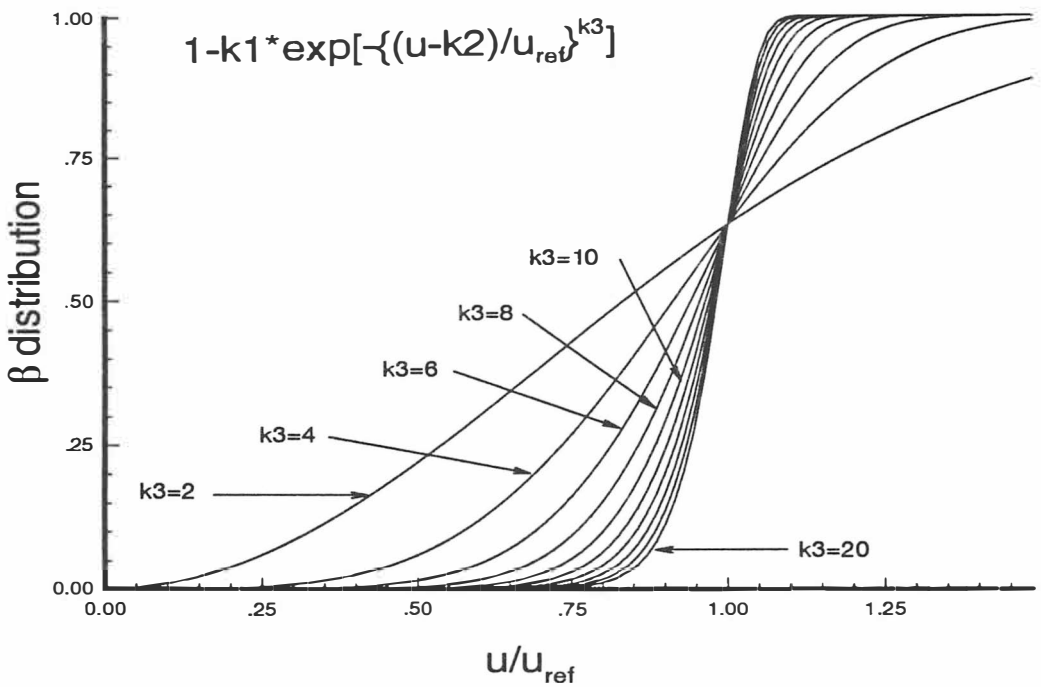


Figure 4.1: Distribution of β in Viscous Flow Reaches.

Chapter 5

Discrete Approximations

This chapter presents the discrete approximation of the derived companion conservation law systems. The development of the spatial semi-discretization is based on a finite element (FE) method. Previous CFD research in FE methods has developed a streamlined notation for this discrete form of the equation system[24,87] which is expanded further here. Two finite-difference time integration schemes are then selected to develop the fully discretized algorithm; advantages and disadvantages of both schemes are discussed. Incorporation of the constitutive equations is presented. Details of the time-relaxed iteration procedure is discussed, with particular attention to the selection of integration time step. Finally, the quasi-Newton jacobian matrix is derived here.

5.1 Finite Element Semi-Discretization

There is no exact solution to the full compressible NS equations. Furthermore, in the variational calculus branch of mathematics, it has been proven that it is impossible to exactly minimize the functional formed by the compressible Navier-Stokes equations.[107] Therefore, the solution is approximated, i.e., a weak statement formulated, through direct variational methods. Among several direct

methods, also termed variational principles, that have been examined over the years,[108] one is the method of weighted residuals[87] (MWR).

FE methods have arisen as an implementation of the general MWR procedure. The MWR specifies the semi-discrete approximation of the TWS of the conservation law system, as presented herein, as

$$\mathrm{TWS}^N(q(t)) \equiv \int_\Omega \Phi_j(\boldsymbol{x})\mathcal{L}^m(q(t))d\boldsymbol{x} = 0, \forall \Phi_j, \tag{5.1}$$

where $\Phi_j(\boldsymbol{x})$ is an arbitrary weighting function. The foundation for the MWR, hence a FE method, suggests the exact solution q lies on a finite-dimensional subspace S^N (i.e., Euclidean space) of an infinite-dimensional Hilbert space H^N. Then, an approximate solution q^N can be constructed as

$$q(\boldsymbol{x},t) \approx q^N(\boldsymbol{x},t) = \sum_{k=1}^{N} \Psi_k(\boldsymbol{x})Q_k(t), \tag{5.2}$$

where $\Psi_k(\boldsymbol{x})$ is a trial function lying in S^N which the approximation coefficient set $Q_k(t)$ interpolates via time-dependant nodal values. Equation 5.2 implicitly assumes a separation of space and time via the selected trial function support.

The FE method constructs Equation 5.2 as the union over finite element domains, within which the $\Psi_k(\boldsymbol{x})$ are approximated by (Lagrange) polynomial expansions, i.e.,

$$q(\boldsymbol{x},t) \approx q^N(\boldsymbol{x},t) = \sum_{k=1}^{N} \Psi_k(\boldsymbol{x})Q_k(t) \equiv \bigcup_e \{N_k(\boldsymbol{x})\}^T\{Q(t)\}_e, \tag{5.3}$$

where $\bigcup_e$ denotes the union over all finite element domains contained in the solution domain semi-discretization, Ω^h.

The Galerkin constraint for MWR requires the weighting function set in Equation 5.1 be identical to the corresponding trial function, i.e., $\Phi_k(\boldsymbol{x}) = \Psi_k(\boldsymbol{x})$. Hence, from Equation 5.3

$$\mathrm{TWS}^h(\{Q(t)\}) = \int_\Omega^h \{N_k(\boldsymbol{x})\}\mathcal{L}^m(\{N_k(\boldsymbol{x})\}^T\{Q(t)\})d\boldsymbol{x} = \{0\}. \tag{5.4}$$

The Galerkin MWR FE formulation is the significant distinction between FE methods and other semi-discrete approximations. It can be shown that selective restriction for trial and weighting functions can reproduce many finite-difference and finite-volume methods.[108,109]

One advantage of the selected FE method is the treatment of metric data. Because the trial space member, hence FE basis function, is selectable, higher order accuracy is possible by increasing the degree of the interpolation polynomials in the basis. Further, the choice in FE basis function also determines the structure of the assembled quasi-Newton jacobian matrix. The second-order accurate basis function utilized throughout this dissertation is a bilinear tensor-product Lagrange basis function[87]

$$q_e(x_i,t) \equiv \{N_1^+(\eta_i)\}^T\{Q(t)\}_e, \tag{5.5}$$

where the subscript notation $(\cdot)_e$ denotes restricted to a basis finite element domain depicted in Figure 5.1. The bilinear basis function coordinate system η_i also forms the basis of the generalized coordinate transformation derived in Appendix C which yields

$$\{N_1^+(\eta_i)\} = \frac{1}{4}\begin{Bmatrix} (1-\eta_1)(1-\eta_2) \\ (1+\eta_1)(1-\eta_2) \\ (1+\eta_1)(1+\eta_2) \\ (1-\eta_1)(1+\eta_2) \end{Bmatrix}. \tag{5.6}$$

Each entry in the $\{N_1^+(\eta_i)\}$ array multiplies respective finite element nodal values in the order given by subscripts $1 \rightarrow 4$ of Figure 5.1. Also shown is the rectangular Cartesian shape of the finite element domain in the intrinsic (curvilinear) coordinate system η_i. Substituting the coordinate transformation in the Galerkin statement for the TWS Equation 5.4 yields

$$\mathrm{TWS}^h(\{Q(t)\}) = \int_\Omega \{N_k(\boldsymbol{\eta})\}\mathcal{L}^m(\{N_k(\boldsymbol{\eta})\}^T\{Q(t)\})detJ\,d\boldsymbol{\eta} = \{0\}. \tag{5.7}$$

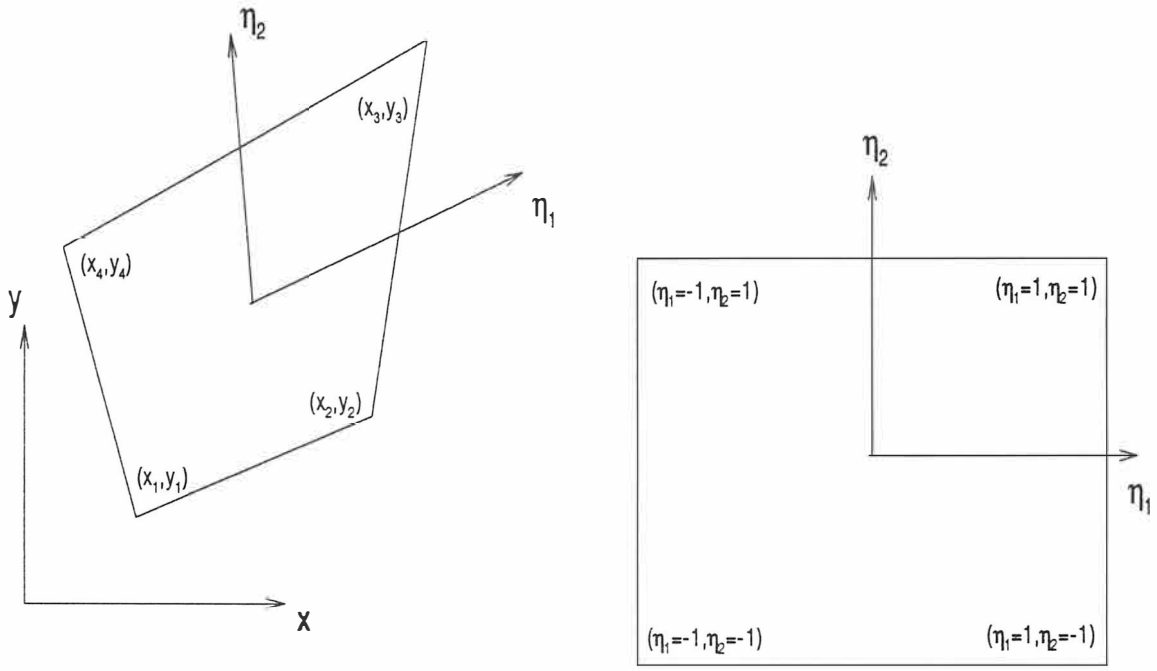


Figure 5.1: A Typical Bilinear Basis Function Finite Element and Coordinate Transformation.

It is advantageous (and necessary for the selected basis $\{N_1^+\}$ applied to Navier-Stokes equations) to utilize Green-Gauss theorem and integrate Equation 5.7 by parts resulting in placing a partial derivative on the weighting function and creating appropriate surface integral terms

$$\begin{aligned}
\mathrm{TWS}^h(Q(t)) = & \int_{\Omega^h} \frac{\partial(\{N_k(\eta)\}^T\{Q(t)\})}{\partial t} detJ\, d\eta \\
& - \int_{\Omega^h} \frac{\partial\{N_k(\eta)\}}{\partial \eta_i} \left(f_i(\{N_k(\eta)\}^T\{Q(t)\}) - f_i^v(\{N_k(\eta)\}^T\{Q(t)\}) \right. \\
& \left. - A_i \left(\beta \frac{\mathsf{L}_j}{2\mathsf{U}_j} A_j \frac{\partial(\{N_k(\eta)\}^T\{Q(t)\})}{\partial x_j} \right) \right) detJ\, d\eta \\
& + \int_{\partial\Omega^h} \{N_k(\eta)\} \left(f_i(\{N_k(\eta)\}^T\{Q(t)\}) - f_i^v(\{N_k(\eta)\}^T\{Q(t)\}) \right. \\
& \left. - A_i \left(\beta \frac{\mathsf{L}_j}{2\mathsf{U}_j} A_j \frac{\partial(\{N_k(\eta)\}^T\{Q(t)\})}{\partial x_j} \right) \right) ds = 0, \qquad (5.8)
\end{aligned}$$

where $\partial\Omega$ denotes the surface bounding the solution domain semi-discretization Ω^h. The next step in the algorithm development is to substitute each term of the conservation law equation system into Equation 5.8 and perform the defined spatial integrations.

5.2 Nomenclature

A large number of terms become created upon implementing Equation 5.8 for the governing Equations 2.6a through 2.6d. Even though each term is distinct, the integrations on Ω_e in η space can be quite routine. In fact, if the grid remains stationary, these integrations are only required once for the selection of $\{N_1^+\}$.

Soliman[24] and Baker[87] devised a nomenclature for organizing the detail of this process which is best conveyed by example. Consider the semi-discretization

of a representative term from the momentum equation

$$-\frac{\mu}{\mathrm{Re}}\frac{\partial^2 u}{\partial y^2}. \tag{5.9}$$

The generalized coordinate transformation (Appendix C) applied to the term of Equation 5.9 yields

$$-\frac{\mu}{\mathrm{Re}}\left((\xi_y)^2\frac{\partial^2}{\partial\xi^2}+(\eta_y)^2\frac{\partial^2}{\partial\eta^2}+(\xi_y\eta_y)\frac{\partial^2}{\partial\xi\partial\eta}+(\eta_y\xi_y)\frac{\partial^2}{\partial\eta\partial\xi}\right)u. \tag{5.10}$$

The semi-discrete form is written via Galerkin to yield

$$\begin{aligned}-\frac{1}{\mathrm{Re}}\int_{\Omega_e}\{\mu\}^T\{N_1^+\}\{N_1^+\}\Big(&(\{N_1^+\}^T\{\xi_y\})^2\frac{\partial^2}{\partial\xi^2}+(\{N_1^+\}^T\{\eta_y\})^2\frac{\partial^2}{\partial\eta^2}\\&+(\{N_1^+\}^T\{\xi_y\}\{N_1^+\}\{\eta_y\})\frac{\partial^2}{\partial\xi\partial\eta}+(\{N_1^+\}^T\{\eta_y\}\{N_1^+\}\{\xi_y\})\frac{\partial^2}{\partial\eta\partial\xi}\Big)\\&\{N_1^+\}^T\{u\}\{N_1^+\}^T\{detJ\}d\boldsymbol{\eta}.\end{aligned} \tag{5.11}$$

Equation 5.11 is too burdensome with five interpolation basis functions $\{N_1^+\}$ to integrate. The metric data dependence can be efficiently approximated by evaluating the partial derivatives ξ_y, η_y and the transformation determinant ($detJ$ Appendix C) at the finite element centroid $\xi = \eta = 0$. Denoted with an over line, integration by parts and Green's theorem then yields

$$\begin{aligned}\frac{1}{\mathrm{Re}}\int_{\Omega_e}\{\mu\}^T\{N_1^+\}\Big(&\overline{\xi_y}^2\frac{\partial\{N_1^+\}}{\partial\xi}\frac{\partial\{N_1^+\}^T\{u\}}{\partial\xi}+\overline{\eta_y}^2\frac{\partial\{N_1^+\}}{\partial\eta}\frac{\partial\{N_1^+\}^T\{u\}}{\partial\eta}\\&+\overline{\xi_y\eta_y}\frac{\partial\{N_1^+\}}{\partial\eta}\frac{\partial\{N_1^+\}^T\{u\}}{\partial\xi}+\overline{\eta_y\xi_y}\frac{\partial\{N_1^+\}}{\partial\xi}\frac{\partial\{N_1^+\}^T\{u\}}{\partial\eta}\Big)\overline{detJ_e}d\xi d\eta\\-\frac{1}{\mathrm{Re}}\int_{\partial\Omega_e}\{\mu\}^T\{N_1\}\{N_1\}\Big(&\overline{\xi_y}^2\frac{\partial\{N_1\}\{u\}}{\partial\xi}+\overline{\eta_y}^2\frac{\partial\{N_1\}^T\{u\}}{\partial\eta}\\&+\overline{\xi_y\eta_y}\frac{\partial\{N_1\}^T\{u\}}{\partial\xi}+\overline{\eta_y\xi_y}\frac{\partial\{N_1\}^T\{u\}}{\partial\eta}\Big)ds,\end{aligned} \tag{5.12}$$

where ds is the differential length in physical space that bounds the finite element at the respective surface, and the basis function in the surface integral becomes $\{N_1\}$ to denote one dimensional interpolation. Such surface terms are ignored for brevity in this example.

Terms in Equation 5.12 require the integration of three occurrences of the basis function $\{N_1^+\}$. One goal of the algorithm/code development was to limit the finite element assembly matrix size to a 4×4, i.e., only allow two basis functions $\{N_1^+\}$ in the integrand. In order to accomplish this goal, accuracy of the element-interpolated fluid viscosity is compromised by assuming an element-average. The same assumption is used throughout this dissertation on all the viscous terms in the momentum and energy equations. Equation 5.12 becomes

$$\frac{\overline{\mu}}{\mathrm{Re}} \int_{\Omega_e} \left(\overline{\xi_y}^2 \frac{\partial\{N_1^+\}}{\partial\xi} \frac{\partial\{N_1^+\}^T\{u\}}{\partial\xi} + \overline{\eta_y}^2 \frac{\partial\{N_1^+\}}{\partial\eta} \frac{\partial\{N_1^+\}^T\{u\}}{\partial\eta} \right.$$
$$\left. + \overline{\xi_y \eta_y} \frac{\partial\{N_1^+\}}{\partial\eta} \frac{\partial\{N_1^+\}^T\{u\}}{\partial\xi} + \overline{\eta_y \xi_y} \frac{\partial\{N_1^+\}}{\partial\xi} \frac{\partial\{N_1^+\}^T\{u\}}{\partial\eta} \right) \overline{detJ_e} d\xi d\eta.$$

The integration of Equation 5.13, and other terms similar in structure, has been performed by gaussian quadrature.[87] The author independently computed and stored the integrated matrices of all 1D, linear and quadratic, and 2D, bilinear and biquadratic basis forms. The details of this procedure are not included here. Since the results of this integration are always the same, they are performed only once and stored as "data", and labeled according to the nomenclature of Baker and Soliman,[87] for example upon Equation 5.13

$$\frac{\overline{\mu}}{\mathrm{Re}} \left(\overline{\xi_y}^2 [B211]\{u\} + \overline{\eta_y}^2 [B222]\{u\} + \overline{\xi_y \eta_y} [B221]\{u\} + \overline{\eta_y \xi_y} [B212]\{u\} \right) 4\overline{detJ_e}, \tag{5.13}$$

where the 4×4 matrices $[B2ij]$ denote a 2D form, whereas the letters A and C would represent a 1D and 3D form respectively. The integer 2 shown in the second position of $[B2ij]$ denotes that two $\{N_1^+\}$ interpolation functions are integrated. The remaining indices in the matrix notation denote the η_i direction of embedded partial derivatives, with zero indicating no derivative. A typical

resulting standard matrix is the integer array

$$[B212] = \frac{1}{4}\begin{bmatrix} 1 & 1 & -1 & -1 \\ -1 & -1 & 1 & 1 \\ -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 \end{bmatrix}. \tag{5.14}$$

5.3 Fully-Discrete Approximation

The fully-discrete approximation is obtained by replacing the time derivative in TWS^h, with a finite difference scheme. For example, the variably-implicit (Θ) (other names often given are "Θ-implicit", trapezoidal rule, weighted approximation, or Crank-Nicholson ($\Theta = \frac{1}{2}$) is

$$\{F(Q)\} = [M]\{\Delta Q^*\} + \Delta t\,(\Theta\{R(Q)\}_{n+1} + (1-\Theta)\{R(Q)\}_n) = \{0\}, \tag{5.15}$$

where $(\cdot)_n$ denotes evaluation at time-step n, $\Delta t = t_{n+1} - t_n$, and $\{\Delta Q_i^*\} = \{(Q_{n+1} - Q_n)DETJ\}$. $DETJ$ is the array of nodal values of $detJ_e$ and $Q_i^* = Q_i DETJ_i$. The implicitness factor Θ, is limited by stability constraints to values of 0.0 for fully explicit, or $0.5 \rightarrow 1.0$ for implicit;[110] $\Theta = 0.5$ is often called trapezoidal rule (or Crank-Nicholson) and $\Theta = 1.0$ indicates fully implicit or backwards Euler. The residual array $\{R(Q)\}$ is formed by the FE assembly of the semi-discretization of all terms in Equation 5.8 except the time derivative term. The mass matrix $[M]$ is formed by the FE assembly of time derivative $\bigcup_e [B200]_e$. The superior accuracy of the mass matrix is another distinguishing characteristic of the FE method. If nonzero values of the TWS parameters (α, γ) are utilized, these terms would also be included within the mass matrix term.

Some further nomenclature definition is required before completing the fully-discrete statements. The following 2D FE matrices are defined from the standard FE matrix set

$$[XB210] \equiv \left(\overline{\frac{\partial \eta_1}{\partial x_1}} [B210] + \overline{\frac{\partial \eta_2}{\partial x_1}} [B220] \right) \overline{detJ_e} \tag{5.16a}$$

$$[XB201] \equiv \left(\overline{\frac{\partial \eta_1}{\partial x_1}} [B201] + \overline{\frac{\partial \eta_2}{\partial x_1}} [B202] \right) \overline{detJ_e} \tag{5.16b}$$

$$[XB220] \equiv \left(\overline{\frac{\partial \eta_1}{\partial x_2}} [B210] + \overline{\frac{\partial \eta_2}{\partial x_2}} [B220] \right) \overline{detJ_e} \tag{5.16c}$$

$$[XB202] \equiv \left(\overline{\frac{\partial \eta_1}{\partial x_2}} [B201] + \overline{\frac{\partial \eta_2}{\partial x_2}} [B202] \right) \overline{detJ_e} \tag{5.16d}$$

$$[XB211] \equiv \left(\left(\overline{\frac{\partial \eta_1}{\partial x_1}} \right)^2 [B211] + \left(\overline{\frac{\partial \eta_2}{\partial x_1}} \right)^2 [B222] + \left| \overline{\frac{\partial \eta_1}{\partial x_1} \frac{\partial \eta_2}{\partial x_1}} \right| [B212] + \left| \overline{\frac{\partial \eta_2}{\partial x_1} \frac{\partial \eta_1}{\partial x_1}} \right| [B221] \right) \overline{detJ_e} \tag{5.16e}$$

$$[XB222] \equiv \left(\left(\overline{\frac{\partial \eta_1}{\partial x_2}} \right)^2 [B211] + \left(\overline{\frac{\partial \eta_2}{\partial x_2}} \right)^2 [B222] + \left| \overline{\frac{\partial \eta_1}{\partial x_2} \frac{\partial \eta_2}{\partial x_2}} \right| [B212] + \left| \overline{\frac{\partial \eta_2}{\partial x_2} \frac{\partial \eta_1}{\partial x_2}} \right| [B221] \right) \overline{detJ_e} \tag{5.16f}$$

$$[XB20N] \equiv \left(\left| \overline{\frac{\partial \eta_1}{\partial x_1} \frac{\partial \eta_1}{\partial x_2}} \right| [B211] + \left| \overline{\frac{\partial \eta_2}{\partial x_1} \frac{\partial \eta_2}{\partial x_2}} \right| [B222] + \left| \overline{\frac{\partial \eta_1}{\partial x_1} \frac{\partial \eta_2}{\partial x_2}} \right| [B212] + \left| \overline{\frac{\partial \eta_2}{\partial x_1} \frac{\partial \eta_1}{\partial x_2}} \right| [B221] \right) \overline{detJ_e} \tag{5.16g}$$

$$[XB210D] \equiv ([B210] + [B220]) \overline{detJ_e} \tag{5.16h}$$

$$[XB220D] \equiv ([B210] + [B220]) \overline{detJ_e} \tag{5.16i}$$

$$[XB211D] \equiv \left(\left| \overline{\frac{\partial \eta_1}{\partial x_1}} \right| [B211] + \left| \overline{\frac{\partial \eta_2}{\partial x_1}} \right| [B222] + \sqrt{\left| \overline{\frac{\partial \eta_1}{\partial x_1} \frac{\partial \eta_2}{\partial x_1}} \right|} [B212] + \sqrt{\left| \overline{\frac{\partial \eta_2}{\partial x_1} \frac{\partial \eta_1}{\partial x_1}} \right|} [B221] \right) \overline{detJ_e} \tag{5.16j}$$

$$[XB222D] \equiv \left(\left| \overline{\frac{\partial \eta_1}{\partial x_2}} \right| [B211] + \left| \overline{\frac{\partial \eta_2}{\partial x_2}} \right| [B222] \right.$$

$$+\sqrt{\left|\frac{\partial\eta_1}{\partial x_2}\frac{\partial\eta_2}{\partial x_2}\right|}[B212]+\sqrt{\left|\frac{\partial\eta_2}{\partial x_2}\frac{\partial\eta_1}{\partial x_2}\right|}[B221]\Bigg)\overline{detJ_e} \quad (5.16k)$$

$$[XB20ND]\equiv\Bigg(\sqrt{\left|\frac{\partial\eta_1}{\partial x_1}\frac{\partial\eta_1}{\partial x_2}\right|}[B211]+\sqrt{\left|\frac{\partial\eta_2}{\partial x_1}\frac{\partial\eta_2}{\partial x_2}\right|}[B222]$$

$$+\sqrt{\left|\frac{\partial\eta_1}{\partial x_1}\frac{\partial\eta_2}{\partial x_2}\right|}[B212]+\sqrt{\left|\frac{\partial\eta_2}{\partial x_1}\frac{\partial\eta_1}{\partial x_2}\right|}[B221]\Bigg)\overline{detJ_e} \quad (5.16l)$$

$$[XB211E]\equiv([B211]+[B222]+[B212]+[B221])\overline{detJ_e} \quad (5.16m)$$

$$[XB222E]\equiv[XB211E] \quad (5.16n)$$

$$[XB20NE]\equiv[XB211E] \quad (5.16o)$$

Following Iannelli,[35] the matrices $[XB211D],[XB222D],[XB20ND]$ incorporate the length scale incorporated in the TWS β term $\underline{L}_j$ via the coordinate transformation metrics $\overline{\frac{\partial\eta_k}{\partial x_j}}$. Extended in a similar manner, the α ($[XB210D]$ and $[XB220D]$) and γ ($[XB211E],[XB222E]$, and $[XB20NE]$) TWS term matrices become free of metric data.

With these additional matrix definitions, the definition of the TWS terms from Chapter 4, and the preferred form of the inviscid flux vector jacobian (F5IFVJ), the following matrix definitions complete the definition of all TWS terms

$$[T1L]=[XB200]+\frac{\alpha_o}{2\sqrt{15}}(\text{sign}(\overline{u})[XB220D]+\text{sign}(\overline{v})[XB210D])$$

$$+\frac{\gamma_o}{6\sqrt{15}}([XB211E]+2[XB20NE]+[XB222E]) \quad (5.17a)$$

$$[T1R]=\frac{\beta_f\beta_o}{2\sqrt{15}}\left(|\overline{u}|[XB211D]+2\sqrt{|\overline{u}|\,|\overline{v}|}[XB20ND]+|\overline{v}|[XB222D]\right) \quad (5.17b)$$

$$[T2L]=[XB200]+\frac{\alpha_o}{2\sqrt{15}}(2\text{sign}(\overline{u})[XB220D]+\text{sign}(\overline{v})[XB210D])$$

$$+\frac{\gamma_o}{6\sqrt{15}}(4[XB211E]+4[XB20NE]+[XB222E]) \quad (5.17c)$$

$$[T2R]=\frac{\beta_f\beta_o}{2\sqrt{15}}\left(4|\overline{u}|[XB211D]+4\sqrt{|\overline{u}|\,|\overline{v}|}[XB20ND]+|\overline{v}|[XB222D]\right) \quad (5.17d)$$

$$[T3L]=[XB200]+\frac{\alpha_o}{2\sqrt{15}}(\text{sign}(\overline{u})[XB210D]+2\text{sign}(\overline{v})[XB220D])$$
$$+\frac{\gamma_o}{6\sqrt{15}}([XB211E]+4[XB20NE]+4[XB222E]) \quad (5.17e)$$

$$[T3R]=\frac{\beta_f\beta_o}{2\sqrt{15}}\left(|\overline{u}|[XB211D]+4\sqrt{|\overline{u}||\overline{v}|}[XB20ND]+4|\overline{v}|[XB222D]\right) \quad (5.17f)$$

$$[T4L]=[XB200]+\frac{\alpha_o}{2\sqrt{15}}(\text{sign}(\overline{u})[XB210D]+\text{sign}(\overline{v})[XB220D])$$
$$+\frac{\gamma_o}{6\sqrt{15}}([XB211E]+2[XB20NE]+[XB222E]) \quad (5.17g)$$

$$[T4R]=\frac{\beta_f\beta_o}{2\sqrt{15}}\left(|\overline{u}|[XB211D]+2\sqrt{|\overline{u}||\overline{v}|}[XB20ND]+|\overline{v}|[XB222D]\right) \quad (5.17h)$$

In a similar manner, the following 1D FE matrices are also defined for utilization in the tensor-product factorization of the 2D matrices

$$[XA200]=2[A200] \quad (5.18a)$$
$$[XA210]=[A210]\Theta\Delta t \quad (5.18b)$$
$$[XA211]=\frac{1}{2}[A211]\Theta\Delta t \quad (5.18c)$$
$$[YA200]=2[A200] \quad (5.18d)$$
$$[YA220]=[A210]\Theta\Delta t \quad (5.18e)$$
$$[YA222]=\frac{1}{2}[A211]\Theta\Delta t, \quad (5.18f)$$

since the following identities are true for the tensor-product bilinear matrices

$$[B200]=2[A200]\otimes 2[A200] \quad (5.19a)$$
$$[B211]=2[A200]\otimes\frac{1}{2}[A211] \quad (5.19b)$$
$$[B201]=2[A200]\otimes[A201] \quad (5.19c)$$
$$[B210]=2[A200]\otimes[A210], \quad (5.19d)$$

where $\otimes$ denotes an outer product. The cross-derivative 2D matrices cannot be factored.

All the information is now defined to complete the expansion of Equation 5.15 as

$$\{F(\rho)\}=[T1L]\{\Delta\rho\}+\Theta\Delta t\Big[[T1R]\{\rho\}$$
$$-[XB210]\{m_1\}-[XB220]\{m_2\}$$
$$+\Delta L[XA200]\{\boldsymbol{m}\cdot\hat{\boldsymbol{n}}\}\Big]^{n+1}+(1-\Theta)\Delta t\Big[\cdot\Big]^{n}=\{0\},\qquad(5.20a)$$

$$\{F(m_1)\}=[T2L]\{\Delta m_1\}+\Theta\Delta t\Big[[T2R]\{m_1\}$$
$$-[XB210]\{um_1\}-[XB220]\{um_2\}$$
$$+\frac{\overline{\mu_e}}{\mathrm{Re}}\left(\frac{4}{3}[XB211]\{u\}+[XB222]\{u\}+\frac{1}{3}[XB20N]\{v\}\right)$$
$$-\mathrm{Eu}[XB210]\{p\}+\Delta L[XA200]\{\{u(\boldsymbol{m}\cdot\hat{\boldsymbol{n}})\}+\mathrm{Eu}\{p\hat{n}_x\}\}\Big]^{n+1}$$
$$+(1-\Theta)\Delta t\Big[\cdot\Big]^{n}=\{0\},\qquad(5.20b)$$

$$\{F(m_2)\}=[T3L]\{\Delta m_2\}+\Theta\Delta t\Big[[T3R]\{m_2\}$$
$$-[XB210]\{vm_1\}-[XB220]\{vm_2\}$$
$$+\frac{\overline{\mu_e}}{\mathrm{Re}}\left(\frac{4}{3}[XB222]\{v\}+[XB211]\{v\}+\frac{1}{3}[XB20N]\{u\}\right)$$
$$-\mathrm{Eu}[XB220]\{p\}+\Delta L[XA200]\{\{v(\boldsymbol{m}\cdot\hat{\boldsymbol{n}})\}+\mathrm{Eu}\{p\hat{n}_y\}\}\Big]^{n+1}$$
$$+(1-\Theta)\Delta t\Big[\cdot\Big]^{n}=\{0\},\qquad(5.20c)$$

and

$$\{F(g)\}=[T4L]\{\Delta g\}+\Theta\Delta t\Big[[T4R]\{g\}$$
$$-[XB210]\{ug\}-[XB220]\{vg\}$$
$$-a^*([XB210]\{m_1\}+[XB220]\{m_2\})$$
$$+\frac{\overline{k_e}}{\mathrm{Re}\,\mathrm{Pr}}([XB211]+[XB222])\{T\}$$
$$+2\mathrm{Ec}\frac{\overline{\mu_e}}{\mathrm{Re}}\left\{\left(\overline{u}\left(\frac{4}{3}[XB211]+[XB222]\right)+\frac{\overline{v}}{3}[XB20N]\right)\{u\}\right.$$
$$\left.+\left(\overline{v}\left(\frac{4}{3}[XB222]+[XB211]\right)+\frac{\overline{u}}{3}[XB2ON]\right)\{v\}\right\}\Big]^{n+1}$$

$$+\Delta L[XA200]\{(H+a^*)(\boldsymbol{m}\cdot\hat{\boldsymbol{n}})\}$$
$$+(1-\Theta)\Delta t\left[\cdot\right]^n=\{0\}. \tag{5.20d}$$

Equations 5.20a through 5.20d include the approximation of element-averaged fluid viscosity μ, thermal conductivity k, and u and v in the energy equation viscous terms. The approximation results in reduced accuracy from the maximum possible for the $\{N_1^+\}$ full interpolation. The loss of accuracy can be quantized directly by examining a representative viscous dissipation term in the semi-discretization formed as a result of integration by parts

$$\frac{1}{\mathrm{Re}}\{k_e\}^T[B3011]\{u\}\approx\frac{\overline{k_e}}{\mathrm{Re}}[B211]\{u\}. \tag{5.21}$$

Additional notation introduced in Equations 5.20a through 5.20d are the surface area ΔL (length in 2D) computed in the original physical coordinates, and the flow tangency term $(\boldsymbol{m}\cdot\hat{\boldsymbol{n}})$ resulting from the integration by parts of the convection terms. Simply not including these surface terms allows flow tangency to be naturally enforced whereas inclusion allows for a momentum flux to be imposed at a given boundary. A complete discussion of boundary conditions is presented in Chapter 6.

Equation 5.15 is solved iteratively via a Newton iteration procedure

$$\left[[M]+\Theta\Delta t\left[\frac{\partial R(Q)}{\partial Q^*}\right]^p_{n+1}\right]\{\delta Q^*\}^{p+1}_{n+1}= \tag{5.22a}$$
$$-\left[[M]\{\Delta Q^*\}^p_{n+1}+\Delta t\left(\Theta\{R(Q)\}^p_{n+1}+(1-\Theta)\{R(Q)\}_n\right)\right],$$
$$\{Q\}^{p+1}_{n+1}=\{Q\}^p_{n+1}+\left\{\delta Q^*\cdot(DETJ)^{-1}\right\}^{p+1}_{n+1}, \tag{5.22b}$$

where $(\cdot)_n$ denotes the time-step index, $\{\Delta Q^*\}^p_{n+1}=\{Q^*\}^p_{n+1}-\{Q^*\}_n$, and the iteration index ^{p} is repeated until $\{\delta Q^*\}^{p+1}_{n+1}$ is small. The direct solution to Equation 5.22a is approximated by guassian elimination via the jacobian matrix

$$[J]=\left[[M]+\Theta\Delta t\left[\frac{\partial R(Q)}{\partial Q^*}\right]^p_{n+1}\right]. \tag{5.23}$$

For 2D and 3D, the jacobian $[J]$ has a large bandwidth and it is not reasonable to evaluate directly. Even for 1D the bandwidth is equal to 15 for the four-equation system solved implicitly. Therefore, the jacobian is approximated by a tensor-product-based matrix factorization. Symbolically, the 2D matrix factorization of the selected quasi-Newton iteration procedure is denoted

$$[J] \approx \left[[M] + \Theta \Delta t \left[\frac{\partial R(Q)}{\partial Q^*}\right]^p_{n+1}\right]_{\eta_1} \bigotimes \left[[M] + \Theta \Delta t \left[\frac{\partial R(Q)}{\partial Q^*}\right]^p_{n+1}\right]_{\eta_2} . \qquad (5.24)$$

The factorization is based on the identities given by Equations 5.19a through 5.19d details of which are shown in a later section. With the matrix factorization defined, the sequence of calculations to obtain the quasi-Newton iteration solution becomes the approximate solution to

$$\left[[M] + \Theta \Delta t \left[\frac{\partial R(Q)}{\partial Q^*}\right]^p_{n+1}\right]_{\eta_1} \{\delta P^*\}^{p+1}_{n+1} = \qquad (5.25a)$$

$$- \left[[M]\{\Delta Q^*\}^p_{n+1} + \Delta t \left(\Theta\{R(Q)\}^p_{n+1} + (1-\Theta)\{R(Q)\}_n\right)\right],$$

$$\left[[M] + \Theta \Delta t \left[\frac{\partial R(Q)}{\partial Q^*}\right]^p_{n+1}\right]_{\eta_2} \{\delta Q^*\}^{p+1}_{n+1} = \{\delta P^*\}^{p+1}_{n+1} \qquad (5.25b)$$

$$\{Q\}^{p+1}_{n+1} = \{Q\}^p_{n+1} + \left\{\delta Q^* \cdot (DETJ)^{-1}\right\}^{p+1}_{n+1}, \qquad (5.25c)$$

where $\{Q\}^{p=0}_{n+1} = \{Q\}_n$, and the initial state variable state $\{Q\}^{p=0}_{n=0}$ is provided as input data.

The research has confirmed that the fully-implicit (Θ = 1.0) form of the Θ-implicit method is best suited for the early part of the calculated approach to steady-state. As the calculation nears steady-state, the preferred choice has been to switch over to the trapezoidal rule method (Θ = .5) to offer a smoother approach to a steady condition. Furthermore, if time accuracy were important, the trapezoidal rule method would certainly be preferred (required) throughout the entire calculation. The only time when the fully explicit method (Θ = 0.0) is preferred is if the quasi-Newton jacobian matrix is not known or desired (which

is not the case in this research). The time-step size is severely limited for an explicit choice.

The quality of the quasi-Newton jacobian matrix can be evaluated by examining successive iterations of the Newton iteration procedure. In particular, for linear and near-linear systems, the convergence rate of the Newton iteration is quadratic. In addition, the first iteration per time step can be used to help control the time step size. Details of these and other theoretical expectations are discussed in a later chapter.

5.4 Semi-Implicit Runge-Kutta Procedure

Although the fully-discrete form obtained by the Θ-implicit algorithm combined with a quasi-Newton iteration procedure is certainly sufficient to obtain accurate and stable solutions, it is not the best available for all applications. A more general class of ordinary differential equation (ODE) solvers is the family of Runge-Kutta (RK) methods.[110] RK solvers provide, among other things, superior stability, accuracy, and step size characteristics over Θ-implicit solvers. In fact, it can be shown that the (Θ = 1)-implicit (backwards Euler) is a special RK method. Perhaps the most popular of RK schemes is the RK-Fehlberg (RKF) (4/5 fourth order accurate/fifth order error estimate) commonly found in many solver packages (or more recently, Fehlberg derived a RKF 7/9 scheme).

The number of possible RK schemes is only limited by imagination and patience in deriving the RK coefficients.[111,112] One such RK scheme is second-order accurate and has shown to be a nonlinearly B-stable semi-implicit time integration.[113,114] Iannelli[35] has proposed a stability-improved semi-implicit scheme and in addition a new and similar fully-implicit RK scheme.

For the semi-discrete form of Equation 5.8,

$$\{TWS^N(Q(t))\} = [M]\frac{d\{Q^*\}}{dt} + \{R(Q)\}, \tag{5.26}$$

The Iannelli two-stage semi-implicit Runge-Kutta used to approximate the fully-discrete solution of Equation 5.26 is

$$\left[[M] + \alpha_1 \Delta t \left[\frac{\partial\{R(Q)\}}{\partial\{Q^*\}}\right]_n\right]\{K1\} = -\Delta t\,\{R(Q_n)\} \tag{5.27a}$$

$$\left[[M] + \alpha_1 \Delta t \left[\frac{\partial\{R(Q)\}}{\partial\{Q^*\}}\right]_n\right]\{K2\} = -\Delta t\,\{R(Q_n + \beta_{21}K1)\} \tag{5.27b}$$

$$\{Q\}_{n+1} = \{Q\}_n + (C_1\{K1\} + C_2\{K2\}) \cdot (DETJ) \tag{5.27c}$$

where the SIRK coefficients are[35]

$\alpha_1 = 0.292893219,$

$\beta_{21} = 0.485281374,$

$C_1 = 0.426776696,$ and

$C_2 = 0.573223304.$

Equations 5.27a through 5.27c constitute a two-stage SIRK scheme of second-order accuracy. The scheme is semi-implicit as opposed to fully-implicit, since there is no provision to iterate at each time step until an acceptable level of per-time-step accuracy is obtained. Of course this is not necessary since only a steady-state is sought. Iannelli[35] improved further upon this scheme and developed a fully-implicit extension to this SIRK scheme.

As in the Θ-implicit scheme, the exact solution to Equations 5.27a through 5.27c requires the equation solver to incorporate a prohibitively large banded matrix to be computed for multi-dimensional problems. Therefore, as before, the

Newton jacobian matrix is approximated by a tensor-product matrix factorization to yield the following computation sequence for the SIRK solution

$$\left[[M] + \alpha_1 \Delta t \left[\frac{\partial\{R(Q)\}}{\partial\{Q^*\}}\right]_n\right]_{\eta_1} \{P1\} = -\Delta t \{R(Q_n)\} \tag{5.28a}$$

$$\left[[M] + \alpha_1 \Delta t \left[\frac{\partial\{R(Q)\}}{\partial\{Q^*\}}\right]_n\right]_{\eta_2} \{K1\} = \{P1\} \tag{5.28b}$$

$$\left[[M] + \alpha_1 \Delta t \left[\frac{\partial\{R(Q)\}}{\partial\{Q^*\}}\right]_n\right]_{\eta_1} \{P2\} = -\Delta t \{R(Q_n + \beta_{21} K1)\} \tag{5.28c}$$

$$\left[[M] + \alpha_1 \Delta t \left[\frac{\partial\{R(Q)\}}{\partial\{Q^*\}}\right]_n\right]_{\eta_2} \{K2\} = \{P2\} \tag{5.28d}$$

$$\{Q\}_{n+1} = \{Q\}_n + (C_1\{K1\} + C_2\{K2\}) \cdot (DET.J)^{-1} \tag{5.28e}$$

As in the tensor-product matrix factorization of the Θ-implicit fully discrete form, the quasi-Newton 1D matrices in Equations 5.28a through 5.28d created by sweeps along the η_1 and η_2 grid lines is certainly more efficient than the exact 2D matrices. However, for the SIRK scheme, because it is a two-stage method, requires twice as many sweeps and subsequent quasi-Newton jacobian matrices as the Θ-implicit per iteration. Furthermore, the SIRK also requires an additional evaluation of the right-hand-side residual array $\{R(Q + \beta_{21} K1)\}$. It turns out that the additional work required for the SIRK scheme is well worth it. Because it is semi-implicit, smaller time steps are required for the SIRK scheme during the early part of the solution. Thus there is a trade off during the early time steps for the work required between the Θ-implicit (multiple iterations per step) and SIRK (two-stage). However, as the solution progresses, the SIRK scheme not only provides superior stability margins over the Θ-implicit, but at much larger time steps. This allows the net work required for a steady solution to be less for the SIRK scheme than for the Θ-implicit. Therefore, the SIRK scheme is used to obtain the solutions, and the Θ-implicit scheme is used to build, test, and evaluate the quality of the quasi-Newton jacobian matrix.

5.5 Incorporation of Closure Equations

The thermodynamic state of the fluid, the assumption of a Newtonian fluid, and the algebraic turbulence model comprise the constitutive relationships utilized to close the equation system. These closure models produce from the state variable array (ρ, m_1, m_2, g) the remaining parameters of pressure p, temperature T, dynamic viscosity μ, turbulent eddy viscosity μ_T, thermal conductivity k, and turbulent eddy thermal conductivity k_T. For ideal-gas closure, the equation of state is utilized to obtain pressure and temperature and the thermal conductivity and dynamic viscosity are both assumed equal to a nondimensional value of unity. For real-gas closure, the TGAS subroutine library[95] is utilized to obtain all thermodynamic properties for equilibrium air. For both ideal- and real-gas closure, the turbulent eddy viscosity and thermal conductivity is optionally determined from either the Baldwin-Lomax[97] or Cebeci-Smith[98] algebraic models. Details of the specific forms of the closure equations are included in Appendixes A and B.

The variables computed by the closure equations lag the state variable by a single time step iteration. The closure variables are behind by a single iteration index ^{p} for the Θ-implicit discrete form, whereas they are behind by a single time step index $_{n}$ for the SIRK discrete form. In an attempt to reduce computational work, increased lag times were experimented with. However, it was found that increasing the lag time beyond a single index in time contributed unacceptable solution behavior and in some cases unstable solutions.

5.6 Newton Iteration Jacobian Matrix and Approximations

For either the Θ-implicit or the SIRK fully discrete forms, the Newton jacobian matrix, containing partial derivatives of the residual with respect to the state variable, is required. This section presents the jacobian matrix and its approximation using a tensor-product matrix factorization.

State variable

Equations 5.20a through 5.20d present the fully-discrete form of the modified conservation law system of equations to be solved. Prior to determined the jacobian matrix, it is necessary to recast these equations as a function of the state variable only. Furthermore, only those terms which appear at the current time step index are entered into the jacobian formulation.

$$\{F_1\}=[T1L]\{\Delta\rho\} + \Theta\Delta t\Big[[T1R]\{\rho\} - [XB210]\{m_1\} - [XB220]\{m_2\} + 2\Delta L[A200]\{\boldsymbol{m}\cdot\hat{\boldsymbol{n}}\}\Big], \tag{5.29a}$$

$$\{F_2\}=[T2L]\{\Delta m_1\} + \Theta\Delta t\Bigg[[T2R]\{m_1\} - [XB210]\left\{\frac{m_1^2}{\rho}\right\} - [XB220]\left\{\frac{m_1 m_2}{\rho}\right\} + \frac{\overline{\mu_e}}{Re}\left(\frac{4}{3}[XB211]\left\{\frac{m_1}{\rho}\right\} + [XB222]\left\{\frac{m_1}{\rho}\right\} + \frac{1}{3}[XB20N]\left\{\frac{m_2}{\rho}\right\}\right) - \overline{\left(\frac{a^2}{\gamma}\right)}[XB210]\{\rho\} + 2\Delta L[A200]\left\{(\frac{m_1}{\rho})(\boldsymbol{m}\cdot\hat{\boldsymbol{n}}) + \overline{\left(\frac{a^2}{\gamma}\right)}\{\rho\hat{n}_x\}\right\}\Bigg] \tag{5.29b}$$

$$\{F_3\}=[T3L]\{\Delta m_2\} + \Theta\Delta t\Bigg[[T3R]\{m_2\} - [XB210]\left\{\frac{m_2 m_1}{\rho}\right\} - [XB220]\left\{\frac{m_2^2}{\rho}\right\} + \frac{\overline{\mu_e}}{Re}\left(\frac{4}{3}[XB222]\left\{\frac{m_2}{\rho}\right\} + [XB211]\left\{\frac{m_2}{\rho}\right\} + \frac{1}{3}[XB20N]\left\{\frac{m_1}{\rho}\right\}\right) - \overline{\left(\frac{a^2}{\gamma}\right)}[XB220]\{\rho\} + 2\Delta L[A200]\left\{(\frac{m_2}{\rho})(\boldsymbol{m}\cdot\hat{\boldsymbol{n}}) + \overline{\left(\frac{a^2}{\gamma}\right)}\{\rho\hat{n}_y\}\right\}\Bigg] \tag{5.29c}$$

and

$$\{F_4\}=[T4L]\{\Delta g\}+\Theta\Delta t\Bigg[[T4R]\{g\}-[XB210]\left\{\frac{m_1 g}{\rho}\right\}-[XB220]\left\{\frac{m_2 g}{\rho}\right\}$$
$$-a^*\left([XB210]\{m_1\}+[XB220]\{m_2\}\right)$$
$$+\frac{\overline{k_e}}{RePr}\left([XB211]+[XB222]\right)\left\{\frac{1}{\rho C_p}\left[g-\frac{Ec}{2}\left(\frac{m_1^2+m_2^2}{\rho}\right)\right]\right\}$$
$$+2Ec\frac{\overline{\mu_e}}{Re}\Bigg\{\left(\overline{u}\left(\frac{4}{3}[XB211]+[XB222]\right)+\frac{\overline{v}}{3}[XB20N]\right)\left\{\frac{m_1}{\rho}\right\}$$
$$+\left(\overline{v}\left(\frac{4}{3}[XB222]+[XB211]\right)+\frac{\overline{u}}{3}[XB2ON]\right)\left\{\frac{m_2}{\rho}\right\}\Bigg\}\Bigg]$$
$$+2\Delta L[A200]\left\{\left(\frac{g}{\rho}+a^*\right)(\boldsymbol{m}\cdot\hat{\boldsymbol{n}})\right\} \tag{5.29d}$$

Θ in equations 5.29a through 5.29d is replaced by α_1 for the SIRK scheme.

Newton Iteration Jacobian Matrix

The Newton jacobian matrix is defined on each finite element as

$$\left[\frac{\partial\{F_i\}}{\partial Q_j}\right]_e=\begin{bmatrix}\left[\frac{\partial F_1}{\partial\rho}\right]_e & \left[\frac{\partial F_1}{\partial m_1}\right]_e & \left[\frac{\partial F_1}{\partial m_2}\right]_e & \left[\frac{\partial F_1}{\partial g}\right]_e\\ \left[\frac{\partial F_2}{\partial\rho}\right]_e & \left[\frac{\partial F_2}{\partial m_1}\right]_e & \left[\frac{\partial F_2}{\partial m_2}\right]_e & \left[\frac{\partial F_2}{\partial g}\right]_e\\ \left[\frac{\partial F_3}{\partial\rho}\right]_e & \left[\frac{\partial F_3}{\partial m_1}\right]_e & \left[\frac{\partial F_3}{\partial m_2}\right]_e & \left[\frac{\partial F_3}{\partial g}\right]_e\\ \left[\frac{\partial F_4}{\partial\rho}\right]_e & \left[\frac{\partial F_4}{\partial m_1}\right]_e & \left[\frac{\partial F_4}{\partial m_2}\right]_e & \left[\frac{\partial F_4}{\partial g}\right]_e\end{bmatrix}. \tag{5.30}$$

Each element in the Newton jacobian matrix is

$$\left[\frac{\partial F_1}{\partial\rho}\right]_e=[T1L]+\Theta\Delta t[T1R] \tag{5.31a}$$
$$\left[\frac{\partial F_1}{\partial m_1}\right]_e=\Theta\Delta t\left(-[XB210]+2\Delta L[A200]\{\hat{n}_x\}\right), \tag{5.31b}$$
$$\left[\frac{\partial F_1}{\partial m_2}\right]_e=\Theta\Delta t\left(-[XB220]+2\Delta L[A200]\{\hat{n}_y\}\right), \tag{5.31c}$$

$$\left[\frac{\partial F_1}{\partial g}\right]_e = 0, \tag{5.31d}$$

$$\begin{aligned}\left[\frac{\partial F_2}{\partial \rho}\right]_e =& \Theta\Delta t\left(\left(\overline{u}^2 - \overline{\left(\frac{a^2}{\gamma}\right)}\right)[XB210] + \overline{uv}[XB220]\right.\\ &-\frac{\overline{\mu_e}}{Re\overline{\rho}}\left(\frac{4\overline{u}}{3}[XB211] + \overline{u}[XB222] + \frac{\overline{v}}{3}[XB20N]\right)\\ &\left.-2\Delta L[A200]\left\{(\boldsymbol{u}\cdot\hat{n})u - \overline{\left(\frac{a^2}{\gamma}\right)}\hat{n}_x\right\}\right),\end{aligned} \tag{5.31e}$$

$$\begin{aligned}\left[\frac{\partial F_2}{\partial m_1}\right]_e =& [T2L] + \Theta\Delta t\left([T2R] - 2\overline{u}[XB210] - \overline{v}[XB220]\right.\\ &+\frac{\overline{\mu_e}}{Re\overline{\rho}}\left(\frac{4}{3}[XB211] + [XB222]\right)\\ &\left.+2\Delta L[A200]\{(\boldsymbol{u}\cdot\hat{n}) + u\hat{n}_x\}\right),\end{aligned} \tag{5.31f}$$

$$\left[\frac{\partial F_2}{\partial m_2}\right]_e = \Theta\Delta t\left(-\overline{u}[XB220] + \frac{\overline{\mu_e}}{3Re\overline{\rho}}[XB20N] + 2\Delta L[A200]\{u\hat{n}_y\}\right), \tag{5.31g}$$

$$\left[\frac{\partial F_2}{\partial g}\right]_e = 0, \tag{5.31h}$$

$$\begin{aligned}\left[\frac{\partial F_3}{\partial \rho}\right]_e =& \Theta\Delta t\left(\overline{uv}[XB210] + \left(\overline{v}^2 - \overline{\left(\frac{a^2}{\gamma}\right)}\right)[XB220]\right.\\ &-\frac{\overline{\mu_e}}{Re\overline{\rho}}\left(\frac{4\overline{v}}{3}[XB222] + \overline{v}[XB211] + \frac{\overline{u}}{3}[XB20N]\right)\\ &\left.-2\Delta L[A200]\left\{(\boldsymbol{u}\cdot\hat{n})v - \overline{\left(\frac{a^2}{\gamma}\right)}\hat{n}_y\right\}\right),\end{aligned} \tag{5.31i}$$

$$\left[\frac{\partial F_3}{\partial m_1}\right]_e = \Theta\Delta t\left(-\overline{v}[XB210] + \frac{\overline{\mu_e}}{3Re\overline{\rho}}[XB20N] + 2\Delta L[A200]\{v\hat{n}_x\}\right), \tag{5.31j}$$

$$\begin{aligned}\left[\frac{\partial F_3}{\partial m_2}\right]_e =& [T3L] + \Theta\Delta t\left([T3R] - \overline{u}[XB210] - 2\overline{v}[XB220]\right.\\ &+\frac{\overline{\mu_e}}{Re\overline{\rho}}\left(\frac{4}{3}[XB222] + [XB211]\right)\\ &\left.+2\Delta L[A200]\{(\boldsymbol{u}\cdot\hat{n}) + v\hat{n}_y\}\right),\end{aligned} \tag{5.31k}$$

$$\left[\frac{\partial F_3}{\partial g}\right]_e = 0, \tag{5.31l}$$

$$\begin{aligned}\left[\frac{\partial F_4}{\partial \rho}\right]_e =& \Theta\Delta t\Big(\overline{uH}[XB210] + \overline{vH}[XB220] \\ &-k^*\left[\overline{H} - Ec\overline{(u^2+v^2)}\right]([XB211]+[XB222]) \\ &-2Ec\frac{\overline{\mu_e}}{Re\,\overline{\rho}}\Big\{\overline{u}^2\left(\frac{4}{3}[XB211]+[XB222]\right) + \overline{uv}[XB20N] \\ &+\overline{v}^2\left(\frac{4}{3}[XB222]+[XB211]\right) + \overline{uv}[XB20N]\Big\} \\ &-2\Delta L[A200]\left\{(\boldsymbol{u}\cdot\hat{n})\overline{H}\right\}\Big),\end{aligned} \tag{5.31m}$$

$$\begin{aligned}\left[\frac{\partial F_4}{\partial m_1}\right]_e =& \Theta\Delta t\Big(-\left(\overline{H}+a^*\right)[XB210] \\ &-k^*Ec\overline{u}\,([XB211]+[XB222]) \\ &+2Ec\frac{\overline{\mu_e}}{Re\,\overline{\rho}}\Big\{\overline{u}\left(\frac{4}{3}[XB211]+[XB222]\right) + \overline{u}[XB20N]\Big\} \\ &+2\Delta L[A200]\left\{(H+a^*)\hat{n}_x\right\}\Big),\end{aligned} \tag{5.31n}$$

$$\begin{aligned}\left[\frac{\partial F_4}{\partial m_2}\right]_e =& \Theta\Delta t\Big(-\left(\overline{H}+a^*\right)[XB220] \\ &-k^*Ec\overline{v}\,([XB211]+[XB222]) \\ &+2Ec\frac{\overline{\mu_e}}{Re\,\overline{\rho}}\Big\{\overline{v}\left(\frac{4}{3}[XB222]+[XB211]\right) + \overline{u}[XB20N]\Big\} \\ &+2\Delta L[A200]\left\{(H+a^*)\hat{n}_y\right\}\Big),\end{aligned} \tag{5.31o}$$

and

$$\begin{aligned}\left[\frac{\partial F_4}{\partial g}\right]_e =& [T4L] + \Theta\Delta t\Big([T4R] - \overline{u}[XB210] - \overline{v}[XB220] \\ &+\frac{k^*}{\overline{\rho}}([XB211]+[XB222]) \\ &+2\Delta L[A200]\left\{\boldsymbol{u}\cdot\hat{n}\right\}\Big).\end{aligned} \tag{5.31p}$$

Quasi-Newton Iteration Jacobian Matrix

To reduce the bandwidth of the Newton iteration jacobian matrix, the multidimensional element matrix is approximated by factoring the full system iteration jacobian matrix, as defined in the previous section, into one-dimensional element matrix outer products. The factorization process is

$$\left[\frac{\partial\{F_i\}}{\partial Q_j}\right]_e \cong \left[\frac{\partial\{F_i\}}{\partial Q_j}\right]^e_{\eta_1} \otimes \left[\frac{\partial\{F_i\}}{\partial Q_j}\right]^e_{\eta_2}, \tag{5.32}$$

where $\otimes$ denotes an matrix outer product operation.[87,115] A symbolic notation for this process which follows the computer coding is

$$\left[\frac{\partial\{F_i\}}{\partial Q_j}\right]^e_{\eta_1} = \begin{bmatrix} [K11] & [K12] & [K13] & [K14] \\ [K21] & [K22] & [K23] & [K24] \\ [K31] & [K32] & [K33] & [K34] \\ [K41] & [K42] & [K43] & [K44] \end{bmatrix}^e_{\eta_1}, \text{ and} \tag{5.33a}$$

$$\left[\frac{\partial\{F_i\}}{\partial Q_j}\right]^e_{\eta_2} = \begin{bmatrix} [L11] & [L12] & [L13] & [L14] \\ [L21] & [L22] & [L23] & [L24] \\ [L31] & [L32] & [L33] & [L34] \\ [L41] & [L42] & [L43] & [L44] \end{bmatrix}^e_{\eta_2}. \tag{5.33b}$$

In order to minimize clutter, the following notation is adopted

$$\overline{\left(\frac{\partial\eta_1}{\partial x_1}\right)} = \xi_x \tag{5.34a}$$

$$\overline{\left(\frac{\partial\eta_1}{\partial x_2}\right)} = \xi_y \tag{5.34b}$$

$$\overline{\left(\frac{\partial\eta_2}{\partial x_1}\right)} = \eta_x \tag{5.34c}$$

$$\overline{\left(\frac{\partial\eta_2}{\partial x_2}\right)} = \eta_y. \tag{5.34d}$$

Each submatrix of the η_1 contribution to the approximate system iteration jacobian matrix is

$$[K11]=[XA200]+\alpha_1 U1[XA210]+(\beta_1+\gamma_1)\,U2[XA211D], \tag{5.35a}$$

$$[K12]=-\xi_x[XA210] + 2\Delta L\Theta\Delta t[A200]\{\hat{n}_x\}|_\xi\,, \tag{5.35b}$$

$$[K13]=-\xi_y[XA210] + 2\Delta L\Theta\Delta t[A200]\{\hat{n}_y\}|_\xi\,, \tag{5.35c}$$

$$[K14]=0, \tag{5.35d}$$

$$[K21]=\left(\left(\overline{u}^2-\overline{\left(\frac{a^2}{\gamma}\right)}\right)\xi_x+\overline{uv}\xi_y\right)[XA210] -\frac{\overline{\mu_e u}}{\mathrm{Re}\overline{\rho}}\left(\frac{4}{3}\xi_x^2+\xi_y^2\right)[XA211] -2\Delta L\Theta\Delta t[A200]\left\{(\boldsymbol{u}\cdot\hat{n})u-\overline{\left(\frac{a^2}{\gamma}\right)}\hat{n}_x\right\}\Big|_\xi\,, \tag{5.35e}$$

$$[K22]=[XA200]+\alpha_2 U1[XA210]+(\beta_2+\gamma_2)\,U2[XA211D] -(2\overline{u}\xi_x+\overline{v}\xi_y)[XA210] +\frac{\overline{\mu_e}}{\mathrm{Re}\overline{\rho}}\left(\frac{4}{3}\xi_x^2+\xi_y^2\right)[XA211] +2\Delta L\Theta\Delta t[A200]\{(\boldsymbol{u}\cdot\hat{n})+u\hat{n}_x\}|_\xi\,, \tag{5.35f}$$

$$[K23]=-\overline{u}\xi_y[XA210] + 2\Delta L\Theta\Delta t[A200]\{u\hat{n}_y\}|_\xi\,, \tag{5.35g}$$

$$[K24]=0, \tag{5.35h}$$

$$[K31]=\left(\overline{uv}\xi_x+\left(\overline{v}^2-\overline{\left(\frac{a^2}{\gamma}\right)}\right)\xi_y\right)[XA210] -\frac{\overline{\mu_e v}}{\mathrm{Re}\overline{\rho}}\left(\xi_x^2+\frac{4}{3}\xi_y^2\right)[XA211] -2\Delta L\Theta\Delta t[A200]\left\{(\boldsymbol{u}\cdot\hat{n})v-\overline{\left(\frac{a^2}{\gamma}\right)}\hat{n}_y\right\}\Big|_\xi\,, \tag{5.35i}$$

$$[K32]=-\overline{v}\xi_x[XA210]$$
$$+2\Delta L\Theta\Delta t[A200]\{v\hat{n}_x\}|_\xi\,, \tag{5.35j}$$

$$[K33]=[XA200]+\alpha_3 U1[XA210]+(\beta_3+\gamma_3)U2[XA211D]$$
$$-(\overline{u}\xi_x+2\overline{v}\xi_y)[XA210]$$
$$+\frac{\overline{\mu_e}}{\mathrm{Re}\overline{\rho}}\left(\xi_x^2+\frac{4}{3}\xi_y^2\right)[XA211]$$
$$+2\Delta L\Theta\Delta t[A200]\{(\boldsymbol{u}\cdot\hat{n})+v\hat{n}_y\}|_\xi\,, \tag{5.35k}$$

$$[K34]=0, \tag{5.35l}$$

$$[K41]=\left(\overline{uH}\xi_x+\overline{vH}\xi_y\right)[XA210]$$
$$-k^*\left[\overline{H}-\mathrm{Ec}(\overline{u^2+v^2})\right]\left(\xi_x^2+\xi_y^2\right)[XA211]$$
$$-\frac{2\mathrm{Ec}\overline{\mu_e}}{\mathrm{Re}\overline{\rho}}\left(\left(\frac{4\overline{u}^2}{3}+\overline{v}^2\right)\xi_x^2+\left(\frac{4\overline{v}^2}{3}+\overline{u}^2\right)\xi_y^2\right)[XA211]$$
$$-2\Delta L\Theta\Delta t[A200]\left\{(\boldsymbol{u}\cdot\hat{n})\overline{H}\right\}\Big|_\xi\,, \tag{5.35m}$$

$$[K42]=-\left(\overline{H}+a^*\right)\xi_x[XA210]$$
$$-\mathrm{Ec}\overline{u}\left(k^*-\frac{8\overline{\mu_e}}{3\mathrm{Re}\overline{\rho}}\right)\left(\xi_x^2+\xi_y^2\right)[XA211]$$
$$+2\Delta L\Theta\Delta t[A200]\{(H+a^*)\hat{n}_x\}|_\xi\,, \tag{5.35n}$$

$$[K43]=-\left(\overline{H}+a^*\right)\xi_y[XA210]$$
$$-\mathrm{Ec}\overline{v}\left(k^*-\frac{8\overline{\mu_e}}{3\mathrm{Re}\overline{\rho}}\right)\left(\xi_x^2+\xi_y^2\right)[XA211]$$
$$+2\Delta L\Theta\Delta t[A200]\{(H+a^*)\hat{n}_y\}|_\xi\,, \tag{5.35o}$$

and

$$[K44]=[XA200]+\alpha_4 U1[XA210]+(\beta_4+\gamma_4)U2[XA211D]$$
$$-(\overline{u}\xi_x+\overline{v}\xi_y)[XA210]$$
$$+\frac{k^*}{\overline{\rho}}\left(\xi_x^2+\xi_y^2\right)[XA211]$$
$$+2\Delta L\Theta\Delta t[A200]\{\boldsymbol{u}\cdot\hat{n}\}|_\xi\,. \tag{5.35p}$$

Each submatrix of the η_2 contribution to the approximate system iteration jacobian matrix is

$$[L11]=[YA200]+\alpha_1 V1[YA220]+(\beta_1+\gamma_1)V2[YA222D], \tag{5.36a}$$

$$[L12]=-\eta_x[YA220]$$
$$+2\Delta L\Theta\Delta t[A200]\{\hat{n}_x\}|_\eta, \tag{5.36b}$$

$$[L13]=-\eta_y[YA220]$$
$$+2\Delta L\Theta\Delta t[A200]\{\hat{n}_y\}|_\eta, \tag{5.36c}$$

$$[L14]=0, \tag{5.36d}$$

$$[L21]=\left(\left(\overline{u}^2-\overline{\left(\frac{a^2}{\gamma}\right)}\right)\eta_x+\overline{uv}\eta_y\right)[YA220]$$
$$-\frac{\overline{\mu_e u}}{\mathrm{Re}\overline{\rho}}\left(\frac{4}{3}\eta_x^2+\eta_y^2\right)[YA222]$$
$$-2\Delta L\Theta\Delta t[A200]\left\{(\boldsymbol{u}\cdot\hat{n})u-\overline{\left(\frac{a^2}{\gamma}\right)}\hat{n}_x\right\}\Big|_\eta, \tag{5.36e}$$

$$[L22]=[YA200]+\alpha_2 V1[YA220]+(\beta_2+\gamma_2)V2[YA222D]$$
$$-(2\overline{u}\eta_x+\overline{v}\eta_y)[YA220]$$
$$+\frac{\overline{\mu_e}}{\mathrm{Re}\overline{\rho}}\left(\frac{4}{3}\eta_x^2+\eta_y^2\right)[YA222]$$
$$+2\Delta L\Theta\Delta t[A200]\{(\boldsymbol{u}\cdot\hat{n})+u\hat{n}_x\}|_\eta, \tag{5.36f}$$

$$[L23]=-\overline{u}\eta_y[YA220]$$
$$+2\Delta L\Theta\Delta t[A200]\{u\hat{n}_y\}|_\eta, \tag{5.36g}$$

$$[L24]=0, \tag{5.36h}$$

$$[L31]=\left(\overline{uv}\eta_x+\left(\overline{v}^2-\overline{\left(\frac{a^2}{\gamma}\right)}\right)\eta_y\right)[YA220]$$
$$-\frac{\overline{\mu_e v}}{\mathrm{Re}\overline{\rho}}\left(\eta_x^2+\frac{4}{3}\eta_y^2\right)[YA222]$$
$$-2\Delta L\Theta\Delta t[A200]\left\{(\boldsymbol{u}\cdot\hat{n})v-\overline{\left(\frac{a^2}{\gamma}\right)}\hat{n}_y\right\}\Big|_\eta, \tag{5.36i}$$

$$[L32] = -\overline{v}\eta_x[YA220]$$
$$+ 2\Delta L\Theta\Delta t[A200]\{v\hat{n}_x\}|_\eta , \tag{5.36j}$$

$$[L33] = [YA200] + \alpha_3 V1[YA220] + (\beta_3 + \gamma_3) V2[YA222D]$$
$$-(\overline{u}\eta_x + 2\overline{v}\eta_y)[YA220]$$
$$+ \frac{\overline{\mu_e}}{\mathrm{Re}\overline{\rho}}\left(\eta_x^2 + \frac{4}{3}\eta_y^2\right)[YA222]$$
$$+ 2\Delta L\Theta\Delta t[A200]\{(\boldsymbol{u}\cdot\hat{n}) + v\hat{n}_y\}|_\eta , \tag{5.36k}$$

$$[L34] = 0, \tag{5.36l}$$

$$[L41] = \left(\overline{uH}\eta_x + \overline{vH}\eta_y\right)[YA220]$$
$$-k^*\left[\overline{H} - \mathrm{Ec}(\overline{u^2 + v^2})\right]\left(\eta_x^2 + \eta_y^2\right)[YA222]$$
$$-\frac{2\mathrm{Ec}\overline{\mu_e}}{\mathrm{Re}\overline{\rho}}\left(\left(\frac{4\overline{u}^2}{3} + \overline{v}^2\right)\eta_x^2 + \left(\frac{4\overline{v}^2}{3} + \overline{u}^2\right)\eta_y^2\right)[YA222]$$
$$-2\Delta L\Theta\Delta t[A200]\left\{(\boldsymbol{u}\cdot\hat{n})\overline{H}\right\}\Big|_\eta , \tag{5.36m}$$

$$[L42] = -\left(\overline{H} + a^*\right)\eta_x[YA220]$$
$$-\mathrm{Ec}\overline{u}\left(k^* - \frac{8\overline{\mu_e}}{3\mathrm{Re}\overline{\rho}}\right)\left(\eta_x^2 + \eta_y^2\right)[YA222]$$
$$+ 2\Delta L\Theta\Delta t[A200]\{(H + a^*)\hat{n}_x\}|_\eta , \tag{5.36n}$$

$$[L43] = -\left(\overline{H} + a^*\right)\eta_y[YA220]$$
$$-\mathrm{Ec}\overline{v}\left(k^* - \frac{8\overline{\mu_e}}{3\mathrm{Re}\overline{\rho}}\right)\left(\eta_x^2 + \eta_y^2\right)[YA222],$$
$$+ 2\Delta L\Theta\Delta t[A200]\{(H + a^*)\hat{n}_y\}|_\eta , \tag{5.36o}$$

and

$$[L44] = [YA200] + \alpha_4 V1[YA220] + (\beta_4 + \gamma_4) V2[YA222D]$$
$$-(\overline{u}\eta_x + \overline{v}\eta_y)[YA220]$$
$$+ \frac{k^*}{\overline{\rho}}\left(\eta_x^2 + \eta_y^2\right)[YA222]$$
$$+ 2\Delta L\Theta\Delta t[A200]\{\boldsymbol{u}\cdot\hat{n}\}|_\eta . \tag{5.36p}$$

where

$$k^* \equiv \overline{\left(\frac{ka^2}{\rho C_p \gamma}\right)} \left(\frac{\gamma_o - 1}{\gamma_o a^*}\right) \left(\frac{1}{\mathrm{RePr}}\right)$$

$$U1 \equiv u\xi_x + v\xi_y,$$

$$U2 \equiv (u\xi_x)^2 + (v\xi_y)^2,$$

$$V1 \equiv u\eta_x + v\eta_y,$$

$$V2 \equiv (u\eta_x)^2 + (v\eta_y)^2,$$

and the symbols $|_\xi$ and $|_\eta$ denote that this quantity is assembled only on a surface bounded by a $\xi(\eta_1)$ and $\eta(\eta_2)$ line respectively.

Two forms of the quasi-Newton matrix approximation have been incorporated into the pilot code. The first form, specified by setting the code input variable `soltyp`=0, inputs all the off-diagonal submatrices to zero, reorders the solution array q, and computes the solution using a banded tridiagonal jacobian matrix. This first form offers significantly increased computational resource utilization. The second form, specified by setting the code input variable `soltyp`=2, inputs the full form of Equations 5.33a and 5.33b with all off-diagonal matrices allowed to take on a non-zero value. The preferred choice of both forms is discussed in a later chapter.

Chapter 6

Boundary and Initial Conditions

Boundary and initial conditions are categorized into well-posed Dirichlet and/or Neumann constraints, and implementation of flow tangency. The algorithm allows for all parts of these boundary conditions to be imposed. The issue of initial conditions is centered about a "restart" concept, with the root initialization being the estimated isentropic 1D flowfield and each major flow complexity (2D inviscid, laminar viscous) is continued from the previous solution until the turbulent viscous case is simulated.

6.1 Boundary Conditions

Classically,[109] boundary conditions are categorized as Dirichlet, Neumann, and/or mixed (Robin). Dirichlet and Neumann boundary conditions respectively constitute zeroth and first order normal partial derivatives of a state variable member on a given boundary segment. Enforcement is conceptually straight forward; however, in practice there are significant and necessary details for consistent implementation, as documented herein.

Well-Posed Dirichlet Constraints

The maximum number and location (inlet or outlet) of Dirichlet boundary conditions that can be imposed for an Euler/Navier-Stokes equation system, Strickwerda[116] et al., for 2-D is:

Euler Equations

3 conditions for subsonic inflow

1 condition for subsonic outflow

4 conditions for supersonic inflow

0 conditions for supersonic outflow

Navier-Stokes Equations

(for all sonic ranges)

4 conditions for inflow

3 conditions for outflow

Strickwerda's analysis only determines the number of inflow/outflow Dirichlet constraints that are well-posed. This does not imply however, that the Dirichlet state variable members can be arbitrarily be assigned at inflow/outflow boundaries. Chakravarthy[117] documents the mathematical derivation for determining the well-posedness of a particular set of Dirichlet variables for the inviscid (Euler) problem. The additional Dirichlet constraints needed for the viscous (Navier-Stokes) case are via velocity and temperature variables operated on by the viscous-shear (elliptic) terms. This subsection employs the method of Chakravarthy for well-posedness determination for the particular Euler equation system utilized in this research.

The well-posed test procedure of Chakravarthy constitutes:

- cast the equation system in a hyperbolic conservation-law form, as in Equation 4.1,.
- determine inviscid flux vector jacobian matrices A_j (Section 4.2.),
- determine eigenvalue arrays Λ and the corresponding eigenvectors X of the inviscid flux vector jacobian matrix A_j,
- confirm the transformation of the inviscid flux vector jacobian matrix $A_j = [\mathrm{X}\Lambda\mathrm{X}]^{-1}$ such that the inviscid portion of Equation 4.1 may be rewritten as

$$\mathcal{L}_{\mathrm{inv}}(q) = \frac{\partial q}{\partial t} + [\mathrm{X}\Lambda\mathrm{X}]^{-1}\frac{\partial q}{\partial x_i} = 0, \tag{6.1}$$

  and further modified to yield

$$\mathcal{L}_{\mathrm{inv}}(q) = [\mathrm{X}]^{-1}\frac{\partial q}{\partial t} + [\Lambda\mathrm{X}^{-1}]\frac{\partial q}{\partial x_i} = 0, \tag{6.2}$$

- hypothesize a set of inlet and/or outlet boundary conditions based on the theory established by Strickwerda,
- substitute the proposed boundary conditions to create a Dirichlet-constrained state-variable array q_{Dir},
- derive the jacobian matrix of q_{Dir} with respect to the state variable $C = \left[\frac{\partial q_{\mathrm{Dir}}}{\partial q}\right]$,
- form a test matrix by substituting the rows of the matrix C that correspond to the row number of Dirichlet constraint on the state variable order into the inverse matrix of eigenvectors $[\mathrm{X}]^{-1}$,
- evaluate the determinant of the test matrix C to determine if the inverse matrix C^{-1} is nonsingular, and

- if the test matrix is singular, i.e., the determinant is zero, the proposed boundary condition is not well posed, hence should not be utilized.

The key to understanding how the Chakravarthy procedure works is realized by examining Equation 6.2. The inverse eigenvector matrix $[X]^{-1}$ contains the flux-vector jacobian information operating on the state variable array. The equivalent type of information is then superimposed through the creation of the test matrix in an manner analogous to imposing the actual Dirichlet data upon the state variable array. Because the test matrix must be inverted to allow for the state variable time partial derivative to solely exist on the left-hand side of the equation, the evaluation of the test matrix determinant then becomes the test for well-posed Dirichlet boundary conditions.

The selected equation system state variable array is $q = \{\rho, m_1, m_2, g\}^T$. This choice differs from the usual situation because of the choice of the volume-specific stagnation enthalpy g as the energy variable. First attempts at a numerical solution did not benefit from the Chakravarthy well-posedness test, hence were arbitrary "guesses." The subsequent undesirable results confirmed that the wrong choice was made for the Dirichlet boundary conditions. The resultant search uncovered the Chakravarthy approach, hence the detailed well-posedness analysis now presented.

In Chapter 4, several variations in the form of the inviscid flux-vector jacobian matrix were considered. The F2IFVJ matrix treats the pressure strictly as a function of the equation of state, and is utilized for this analysis. The corresponding jacobian matrices $\frac{\partial f_x}{\partial q}$ and $\frac{\partial f_y}{\partial q}$ are given by Equations 4.29 and 4.30 respectively. Hence solving the equations

$$\left|\frac{\partial f_x}{\partial q} - \lambda I\right| = 0, \tag{6.3a}$$

$$\left| \frac{\partial f_y}{\partial q} - \lambda I \right| = 0, \tag{6.3b}$$

yields the corresponding sets of eigenvalues, for the x and y directions, respectively as

$$\lambda_x = \{u, u, u + a, u - a\}^T, \tag{6.4a}$$

$$\lambda_y = \{v, v, v + a, v - a\}^T. \tag{6.4b}$$

These are typical eigenvalues obtained for the 2D Euler equations,[117] and the corresponding matrix of eigenvectors is obtained via

$$X_x = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & u+a & u-a \\ 0 & 1 & v & v \\ 1 & 0 & H^+ & H^- \end{bmatrix}, \text{and} \tag{6.5a}$$

$$X_y = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & u & ua \\ 0 & 0 & v+a & v-a \\ 0 & 1 & H^+ & H^- \end{bmatrix}, \tag{6.5b}$$

where $H^+ = H + (\gamma_o - 1)(u + a)a$ and $H^- = H - (\gamma_o - 1)(u + a)a$. Then, the inverse of the matrix of eigenvectors is classical manner

$$[X_x]^{-1} = \frac{1}{2a} \begin{bmatrix} u\Delta H - a(H^+ + H^-) & -\Delta H & 0 & 2a \\ -2va & 0 & 2a & 0 \\ -(u-a) & 1 & 0 & 0 \\ (u+a) & -1 & 0 & 0 \end{bmatrix}, \text{and} \tag{6.6a}$$

$$[X_y]^{-1} = \frac{1}{2a} \begin{bmatrix} -2ua & 2a & 0 & 0 \\ v\Delta H - a(H^+ + H^-) & 0 & -\Delta H & 2a \\ -(v-a) & 0 & 1 & 0 \\ (v+a) & 0 & -1 & 0 \end{bmatrix}, \tag{6.6b}$$

where $\Delta H = H^+ - H^- = 2ua(\gamma_o - 1)$. Equations 6.6a and 6.6b are the two test matrices in the x and y directions respectively. The accuracy of these matrices is checked to confirm that

$$\frac{\partial f_x}{\partial q} = [\mathrm{X}_x \Lambda \mathrm{X}_x^{-1}], \tag{6.7a}$$

$$\frac{\partial f_y}{\partial q} = [\mathrm{X}_y \Lambda \mathrm{X}_y^{-1}], \tag{6.7b}$$

$$[I] = [\mathrm{X}_y \mathrm{X}_y^{-1}], \text{ and} \tag{6.7c}$$

$$[I] = [\mathrm{X}_y \mathrm{X}_y^{-1}]. \tag{6.7d}$$

As an example of a boundary condition that is not well-posed is $q_{\mathrm{Dir}} = \{\rho, m_1, 0, 0\}^T$ on either an inlet or exit. The jacobian matrix of q_{Dir} is then

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{6.8}$$

Superimposing non-zero rows of Equation 6.8 into Equations 6.6a and 6.6b yields two matrix determinants to evaluate

$$|\mathrm{X}_x|_{\mathrm{test}} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -(u-a) & 1 & 0 & 0 \\ (u+a) & -1 & 0 & 0 \end{vmatrix}, \tag{6.9a}$$

$$|\mathrm{X}_y|_{\mathrm{test}} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -(v-a) & 0 & 1 & 0 \\ (v+a) & 0 & -1 & 0 \end{vmatrix}. \tag{6.9b}$$

Both Equations 6.9a and 6.9b are zero. Therefore, density and principal momentum (and not any other data) as Dirichlet constraints on either an inlet or

outlet boundary is not well-posed. The additional constraint of volume-specific stagnation enthalpy g on the boundary will cause the last row in Equations 6.9a and 6.9b to become $(0, 0, 0, 1)$, hence the determinant is non-zero, and the boundary condition is well-posed in the y direction only.

Table 6.1 summarizes additional boundary condition well-posedness tests

Table 6.1: Tests of Well-Posed Dirichlet Boundary Conditions for the Researched State Variable Set $\{\rho, m_1, m_2, g\}$ on the Euler equations

Test Number	Constraint Array	Result	Comment
1	$\{0, 0, 0, p\}^T$	well-posed	under all conditions where p is imposed via g
2	$\{0, 0, 0, g\}^T$	well-posed	under all conditions
3	$\{\rho, 0, 0, g\}^T$	well-posed	only if $\frac{\partial}{\partial y} = 0$ along the boundary
4	$\{p, 0, 0, 0\}^T$	well-posed	under all conditions where p is imposed via ρ
5	$\{\rho, 0, 0, 0\}^T$	not-well-posed	under all conditions
6	$\{\rho, m_1, 0, 0\}^T$	not-well-posed	under all conditions
7	$\{\rho, 0, m_2, g\}^T$	not-well-posed	under all conditions
8	$\{\rho, m_1, 0, g\}^T$	well-posed	only if $\frac{\partial}{\partial x} = 0$ along the boundary
9	$\{\rho, m_1, m_2, g\}^T$	well-posed	under all conditions
10	$\{0, 0, 0, 0\}^T$	well-posed	under all conditions

and their results. The result of test numbers 9 and 10 are understood by realizing that the matrices considered are the identity matrix and the original (unaltered) matrices given by Equations 6.6a and 6.6b respectively. Tests 9 and 10 correspond to the supersonic outflow and inflow boundaries respectively. The inviscid outflow boundary condition finally utilized, which also satisfies the constraint of Strickwerda via tests 1,4 and 10 respectively, are constant pressure for subsonic flow and no constraints for supersonic flow. As was confirmed by computational experiments and analytically by test numbers 2 and 5, a constant density or volume-specific stagnation enthalpy alone are ill-posed inflow/outflow boundary conditions.

From Table 6.1, further conclusions regarding Dirichlet inviscid inflow boundary conditions include:

- that density and volume-specific stagnation enthalpy (3) is well posed only if a zero gradient of the state variable exists in the y direction,
- unlike the more traditional state variable $q = \{\rho, m_1, m_2, e\}^T$, in the present case, specification of transverse momentum m_2 (such as holding to zero) is not well-posed (7),
- the imposition of the principal momentum m_1 is well-posed, but only if all gradients in the x direction are zero at the boundary (8), finally
- holding only density and principal momentum constant is not well-posed (6).

Therefore, the Strickwerda criteria is satisfied for subsonic flows if Dirichlet inflow boundary conditions constrain density ρ and volume-specific stagnation enthalpy g, while a supersonic in flow requires that all variables be specified on inflow boundaries (9) and no Dirichlet constraints at the outflow (10). It is

certainly possible to validate additional boundary conditions with Chakravarthy's technique.

Neumann Constraints

From Chapter 5, there is no explicit expression in the fully-discrete approximation for the Neumann constraints that arise from the semi-discretization of the TWS modified conservation law equations. As a result of integration by parts of the artificial and viscous dissipation terms, the following surface terms must be accounted for at each boundary surface

- from TWS dissipation (β) terms
  - -- $\mathcal{L}(\rho) : \frac{\partial \rho}{\partial \eta_1}, \frac{\partial \rho}{\partial \eta_2}$
  - -- $\mathcal{L}(m_1) : \frac{\partial m_1}{\partial \eta_1}, \frac{\partial m_1}{\partial \eta_2}$
  - -- $\mathcal{L}(m_2) : \frac{\partial m_2}{\partial \eta_1}, \frac{\partial m_2}{\partial \eta_2}$
  - -- $\mathcal{L}(g) : \frac{\partial g}{\partial \eta_1}, \frac{\partial g}{\partial \eta_2}$
- from natural viscous dissipation terms
  - -- $\mathcal{L}(\rho)$: none
  - -- $\mathcal{L}(m_1) : \frac{\partial u_1}{\partial \eta_1}, \frac{\partial u_1}{\partial \eta_2}, \frac{\partial u_2}{\partial \eta_1}, \frac{\partial u_2}{\partial \eta_2}$
  - -- $\mathcal{L}(m_2) : \frac{\partial u_1}{\partial \eta_1}, \frac{\partial u_1}{\partial \eta_2}, \frac{\partial u_2}{\partial \eta_1}, \frac{\partial u_2}{\partial \eta_2}$
  - -- $\mathcal{L}(g) : \frac{\partial u_1}{\partial \eta_1}, \frac{\partial u_1}{\partial \eta_2}, \frac{\partial u_2}{\partial \eta_1}, \frac{\partial u_2}{\partial \eta_2}, \frac{\partial T}{\partial \eta_1}, \frac{\partial T}{\partial \eta_2}$

Three of four choices for each partial derivative have been validated for the treatment of these surface partial derivatives and encompasses most of the possible Neumann constraints:

1. if a Dirichlet constraint exists on a state variable member, either directly or indirectly[a], then no explicit Neumann constraint of that particular variable is necessary, nor effective, since the corresponding node is independent,

2. if a vanishing normal derivative of a state variable member is required, then no explicit Neumann constraint is required, i.e., a vanishing derivative is obtained by doing nothing!

3. if a non vanishing normal derivative of a state variable member is required, such as an outflow boundary condition, then the Neumann constraint is treated by one of two possibilities

   (a) if data within the simulated body specifies the normal gradient, then a special treatment, to be described, is needed,

   (b) if known data outside of the simulated body specifies the normal gradient, then this data is assembled into the linear algebra statement in a manner similar to that of Dirichlet data.

Implementation of items 1 and 2 above is straight forward while item 3 requires further clarification. The special case 3b, has not been validated in this research. It is anticipated that this special case could also be treated in a similar manner to the outflow boundary case. The best way to explain the special treatment of outflow boundaries, 3a, is by example.

[a]for example a no-slip wall viscous constraint on the velocity also constrains the momentum to a no-slip condition

Consider the equivalent finite-difference expression for the finite-element assembly of a representative dissipation term in 1D for the boundary node $i+1$

$$
\begin{aligned}
&\overbrace{-\int_{i+1}\{N_k\}\frac{\partial^2\{N_k\}^T\{Q\}}{\partial\eta^2}d\Omega_i}^{\text{original term}} \\
&= \overbrace{\int_{i+1}\frac{\partial\{N_k\}}{\partial\eta}\frac{\partial\{N_k\}^T\{Q\}}{\partial\eta}d\Omega_i}^{\text{volume term}} - \overbrace{\int_{\partial_{i+1}}\{N_k\}\frac{\partial\{N_k\}^T\{Q\}}{\partial\eta}dL_{i+1}}^{\text{surface term}} \\
&= \frac{1}{(\Delta L_i)^2}[A211]\{Q\}\Delta L_i - \int_{\partial_{i+1}}\{N_k\}\frac{\partial\{N_k\}^T\{Q\}}{\partial\eta}dL_{i+1} \\
&= \frac{1}{(\Delta L_i)^2}(q_{i+1}-q_i)\Delta L_i - \frac{\partial q}{\partial\eta}|_{i+1},
\end{aligned}
\tag{6.10}
$$

where the distance ΔL_i is the element measure of the boundary finite element measured normal to the boundary. If the boundary condition is a vanishing derivative, then $\frac{\partial q}{\partial\eta}|_{i+1}$ is zero, and by not assembling the surface term, the usual assembly of the volume terms will yield (with proper discretization) the desired result of a vanishing derivative at the boundary.

But, if the desired boundary gradient is non-zero, as is the case for some outflow boundaries, then the surface term must also be assembled. For the special case of an outflow boundary where conditions are determined completely by flow inside, the gradient at the boundary can be approximated, given a sufficiently refined grid, by assuming the gradient generated at the finite element adjacent to the boundary is equal to the gradient at the boundary. This assumption has the affect of causing (for the 1D case) Equation 6.10 to be zero. In practice, this is implemented by resetting the row of the assembled dissipation matrix corresponding to the boundary node $(i+1)$ equal to zero. In 2D, there are two assembly matrix rows to reset to zero corresponding to the two nodes of a finite element at the boundary. Similarly, if bilinear basis function is also used, there would be 4 rows to reset to zero in the assembly matrix in 3D. The

code is built to optionally control this type of outflow boundary for all surfaces and the seven possible dissipation variables.

To complete this discussion, a specific example of the implementation of non-vanishing outflow Neumann boundary conditions follows. In Chapter 5, a typical 2D finite element utilizing the bilinear basis function typical in this research is shown in Figure 5.1. Assume, for example, that a nonzero gradient on the principal velocity component u_1 exists in the η_1 direction between and on the nodes 2 and 3, i.e., $\frac{\partial u_1}{\partial \eta_1}\Big|_{2-3} \neq 0$. Based on the discussion above, this type of boundary condition may be implemented by the finite-element assembly (i.e., not a process performed after the assembly) of the following equation

$$\frac{1}{(\Delta\eta_1)^2}[B211]_{2-3}\{u_1\}\det J_e \Delta\eta_1 \Delta\eta_2$$

$$= \frac{1}{6(\Delta\eta_1)^2}\begin{bmatrix} 2 & -2 & -1 & 1 \\ -2 & 2 & 1 & -1 \\ -1 & 1 & 2 & -2 \\ 1 & -1 & -2 & 2 \end{bmatrix}_{2-3} \{u_1\}\det J_e \Delta\eta_1 \Delta\eta_2 \qquad (6.11a)$$

$$= \frac{1}{6(\Delta\eta_1)^2}\begin{bmatrix} 2 & -2 & -1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & -1 & -2 & 2 \end{bmatrix} \{u_1\}\det J_e \Delta\eta_1 \Delta\eta_2 \qquad (6.11b)$$

Flow Tangency, Blowing, and Momentum Flux

Boundary conditions involving flow tangency, "blowing", and nonzero momentum flux are all implemented by the proper treatment of the surface terms arising from the weak statement integration by parts of the convection terms in each of the conservation law equations. The finite element semi-discretization allows for the assembly process to naturally incorporate these types of boundary conditions.

For example, the semi-discretization of the continuity equation includes the surface finite element assembly of a momentum flux term $[A200]\{\boldsymbol{m}\cdot\hat{\boldsymbol{n}}\}\Delta L$, where $\boldsymbol{m}$ is the momentum vector at the surface, $\hat{\boldsymbol{n}}$ is a unit vector normal to the surface, and ΔL is the length (area in 3D) of the surface defined by the finite element being assembled. Similar momentum flux terms exist on all the remaining conservation law equations, Chapter 5. It is not necessary to explicitly assemble the momentum flux terms for all the internal finite elements that are bounded by another finite element since the assembly process would result in a net cancellation. However, those finite elements which include a boundary must also account for the assembly of these momentum flux terms.

The key to implementing these types of boundary conditions is the treatment of the function $(\boldsymbol{m}\cdot\hat{\boldsymbol{n}})$. For flow tangency boundaries, such as inviscid flow along a wall, $(\boldsymbol{m}\cdot\hat{\boldsymbol{n}}) = 0$, this is implemented by simply not assembling the momentum flux terms. Thus, flow tangency comes for "free" by doing nothing. On the other hand, a nonzero momentum flux boundary conditions, such as an inflow or an outflow boundary, require the assembly of the $(\boldsymbol{m}\cdot\hat{\boldsymbol{n}})$ terms.

A blowing boundary condition is a special case of a nonzero momentum flux imposed at a wall segment. In practice, such a boundary condition might provide cooling for a surface experiencing hypersonic flow kinetic energy deposition. This type of boundary condition is directly implemented as a Dirichlet constraint on the appropriate momentum flux although no validation is shown herein.

6.2 Initial Conditions

The code has been designed to minimize the amount of manual manipulation of data for problem initialization. There are only three points in the solution process

where the state variable array must be (re)initialized: (1) the quasi-1D isentropic problem ($q_{n=0}$), (2) the restarting of any solution from a previous solution "checkpoint," and (3) the setting of the boundary condition constraints for the viscous problem. This minimal amount of manual initialization is accomplished by a very flexible restart/interpolation procedure. The key is that any initialization must be consistent with the new problem statement boundary conditions. This section summarizes the initialization/restart process.

Isentropic 1D Initialization

All validation problems solved for this dissertation have a suitable quasi-1D simplification. The generation of quasi-1D initial conditions from isentropic flow theory depends on the problem geometry and utilizes the following relationships:

- mass conservation

$$\rho A v = \text{constant.}$$

- energy conservation

$$h + \frac{v^2}{2 g_c J} = \text{constant.}$$

- isentropic flow

$$\frac{T_o}{T} = \left(\frac{p_o}{p}\right)^{\frac{\gamma-1}{\gamma}} = \left(\frac{\rho_o}{\rho}\right)^{\gamma-1} = \text{constant.}$$

- equation of state

$$p = z \rho R T$$

- Mach number

$$M = \frac{v}{a} = \frac{v}{\sqrt{\gamma R T g_c}}$$

The analytical isentropic and normal shock solutions are also available, and tabulated[118] as a function of Mach number.

The entrance and exit boundary conditions are input to the code, which then linearly interpolates between the endpoints for most variables but does allow for mass conservation through area variation. This minimal amount of input is then passed to the quasi-1D solver to obtain the steady-state solution while maintaining the desired boundary conditions. Thus, the algorithm/code can handle quite complex quasi-1D problems without a strict nodal definition of the initial conditions.

Restart and Interpolation Procedures

The solution procedure to a final turbulent viscous solution then follows a systematic progression. The code input deck requires minimal changes between each restart, details are in Appendix E. At the end of each solution phase, including the quasi-1D solution, a restart/plot data file is stored which contains all the current state variable, geometry, and reference data associated with the solution. These data are then available, along with a modified input deck, for restart information to the next sequence of the solution.

First, the solution of the quasi-1D equation system is quickly established to machine accuracy. As will be seen, essentially the only difference between the analytical isentropic and numerical quasi-1D solutions is that the numerical stagnation enthalpy is not constant. Then the following sequences are performed to reach the final solution.

Quasi-1D to Inviscid-2D

The generated quasi-1D solution establishes initial conditions for the 2D inviscid problem. The two main restart features are: (1) the utilization of the finite element basis function to interpolate the 2D inviscid initial conditions from the 1D solution in computational space, and (2) the splitting of the 1D total velocity into contravariant scalar components parallel with the grid lines, and then from the geometry, Cartesian components (u, v).

The 2D inviscid problem is then iterated to a pre-determined level of accuracy with the residuals generally less than 10^{-6} on both the iterate (left-hand side,$\{\delta Q\}$) and extremum (right-hand side, $\{R(Q^{n+1})\}$) of the fully-discrete system of equations (see Chapter 5). It is possible for the residuals of the 2D inviscid problem to be reduced to machine accuracy, since very little error is introduced by the matrix factorization bias. Normally, the calculation is stopped when the Euclidean norm of the extremum, also important for stability monitoring and discussed in a later chapter, is less than 10^{-6}. Often the 2D inviscid problem is restarted from a coarse to a refined grid, to more accurately capture exposed flow details. Such grid refinement is particularly important around curved boundaries and embedded shocks. The number of nodes in each direction on the fine mesh is normally on the order of $100 \rightarrow 200$.

For both the quasi-1D and 2D inviscid solutions, the narrow bandwidth version of the quasi-Newton jacobian matrix (`soltyp=0`) can be utilized to yield essentially the same convergence rate as the full bandwidth jacobian (`soltyp=2`). The minimum bandwidth, i.e., tridiagonal, is structured by not assembling the off-block-diagonal entries and reordering the solution array. However the reduced jacobian bandwidth is not recommended for the viscous problem (discussed next), since it severely degrades the quasi-Newton matrix iteration convergence rate.

Inviscid-2D to Laminar-Viscous

The restart procedure is utilized again, without velocity redistribution along grid lines, to establish the laminar viscous initial conditions from the 2D inviscid solution. The no-slip wall velocity along the wall is set to zero via Dirichlet constraint. The heat transfer coefficient along the same wall is set to a pre-determined value, usually zero corresponding to an adiabatic wall. Non-zero heat transfer is also validated via progressive increases in the heat transfer coefficient until a sufficiently large value yields essentially perfect heat transfer, i.e., the wall node temperature equals the wall reference temperature.

The Reynolds number of the viscous problem statement is intrinsically determined from the fluid thermodynamic properties at stagnation reference conditions and an appropriate length scale. This computed Reynolds number may be manually overridden by specifying another length scale, e.g., the boundary layer thickness. The Reynolds number predicates the design of the viscous grid near walls. It has been determined that for higher Reynolds numbers ($Re > 10^4$), a three macro-element nest is needed for a suitable grid specification in the wall-normal direction. The first macro element, which nominally contains the laminar sub-layer, is very thin, contains 10-20 finite elements, has a unit mesh progression ratio, and the first finite element span is on the order of Re^{-1}. The second macro element has a progression ratio greater than unity, and is sufficiently thick to capture the entire transition from viscous to inviscid flow. The third macro element spans the essentially inviscid flow region and may also have a progression ratio greater than unity. For symmetric problems with smaller Reynolds numbers (on the order of 10^3), all the inviscid and viscous flow details may be captured with a single macro element mesh wall-normal provided

a sufficient number of elements are used, and an aggressive progression ratio is utilized (on the order of 10% $\rightarrow$ 20% change between adjacent elements).

Laminar-Viscous to Turbulent-Viscous

After obtaining a laminar solution, the turbulent simulation is restarted by simply "turning on" the turbulence model within the solution process. It may be necessary to restructure the viscous region grid, depending on the progression of the turbulent solution. Furthermore, because the algebraic turbulence model requires a smooth, well established velocity profile in order to function properly, it may be necessary to utilize a smaller Reynolds number to start the turbulent solution. After getting the turbulent solution started, the Reynolds number is then returned to the genuine value. Since there is no artificial dissipation in the viscous boundary layer region, as the Reynolds number is increased, the laminar sub-layer region of the turbulent simulation contains less viscous dissipation and hence is less stable. Therefore, the grid may again have to be redefined, as the laminar portion of boundary layer becomes thinner.

Chapter 7

Theoretical Analyses/Expectations

The compressible NS equations, as well as reduced forms such as thin-layered NS, Euler, quasi-1D, etc., are truly nonlinear. As with any solution approximation of a set of nonlinear partial differential equations, an iterative linearized algorithm/procedure is employed in this research. During the development of this algorithm, theoretical issues, both linearized and nonlinear, were identified for later confirmation via computational experiments. These details are itemized in this chapter in preparation for validation results in Chapter 8.

7.1 Convergence with Discretization Refinement

The semi-discretization error of the state variable approximate solution is

$$\varepsilon^h \equiv q - q^h, \tag{7.1}$$

where q is the true solution and q^h is the semi-discrete approximation. The function ε^h is an array of functions which resist a single measurement of the error. Therefore, the mathematicians have developed scalar measures called "norms" for the purpose of accuracy and convergence assessments. For example, the L_2

inner product (or norm) is defined as[108]

$$L_2(q,q) = \|q\|_{L_2} \equiv \int_\Omega q^T q \; d\tau. \tag{7.2}$$

The general H^r norm[87] is

$$H^r(q,q) = \|q\|_{H^r} \equiv \left[\sum_{i=0}^{r}\sum_{j=0}^{r}\int_\Omega \frac{\partial^i q}{\partial x_k^i}\frac{\partial^j q}{\partial x_k^j} d\tau\right]^{1/2}, \quad 1 \le k \le n. \tag{7.3}$$

The order of the H^r norm is "r," equal to the highest derivative of the argument which must be square-integrable on Ω. The H^0 norm is,

$$H^o(q,q) = \|q\|_{H^0} \equiv \left[\int_\Omega q^2 d\tau\right]^{1/2}, \tag{7.4}$$

and is identical to $\sqrt{L_2}$.

The H^r norm for $r = 1$ plays a special role. Since the TWS is also of order one in partial derivative, i.e., a "weakened" derivative, the norm

$$H^1(q,q) = \|q\|_{H^1} \equiv \left[\int_\Omega \left(q + \frac{\partial q}{\partial x_k}\right)\cdot\left(q + \frac{\partial q}{\partial x_k}\right) d\tau\right]^{1/2}, \tag{7.5}$$

and can be used to examine convergence rates.

For finite element analyses in linear problems, the energy norm is defined as[87]

$$E(q) = \|q\|_E \equiv \frac{1}{2}\int_\Omega (\nabla q \cdot \nabla q)\, d\tau, \tag{7.6}$$

which is related to a portion of the H^1 norm. Such norms may be used to determine the convergence of the finite element semi-discrete solution.

The energy norm of the error is sought, hence from Equation 7.1 and Equation 7.6

$$E(\varepsilon^h) = \|\varepsilon^h\|_E \equiv \int_\Omega \left(\nabla(q - q^h)\cdot\nabla(q - q^h)\right) d\tau. \tag{7.7}$$

Equation 7.7 is not directly evaluated, since the true solution q is never known (if it were, a discrete approximation would not be needed!). For a linear elliptic problem with smooth data, available finite element theory[87,119] predicts

$$\|\varepsilon^h\|_E = E(q - q^h) = E(q) - E(q^h) \equiv \int_\Omega (\nabla q \cdot \nabla q)\, d\tau - \int_\Omega \left(\nabla q^h \cdot \nabla q^h\right) d\tau. \quad (7.8)$$

Further, ignoring time discretization errors, the energy norm of the error is proportional to an element measure raised to an exponent[87,108]

$$E(q - q^h) \propto C_1 \Delta_e^{2(k+1-m)} \|\text{data}\|_{H^0}, \quad (7.9)$$

where the quantity $\|\text{data}\|$ is a constant in L_2. The integer parameters k and m denote the finite element basis function degree and PDE order respectively. To facilitate convergence estimation, taking $\log_{10}$ of both sides, and realizing that (for problems in this dissertation) $k = 1 = m$, yields

$$\log_{10}(E(q - q^h)) \cong 2k \log_{10}(\Delta_m) + \log_{10}(C_1) + \log_{10}(\| \, data \, \|). \quad (7.10)$$

Equation 7.10 is linear of the form $y = bx + a$, hence can be used to estimate the convergence rate with discretization refinement.

As stated earlier, the true solution q is not known but can be approximated as the best discrete solution q^b, i.e., that solution with the greatest refinement in the mesh and the least value of the energy norm. Furthermore, prior research[24] has demonstrated that for grids with variations in finite element size, the element measure Δ_e should be the maximum element measure of the mesh. This allows the energy norm error (or error in energy norm) to be estimated from the difference in

- successively refined grid solutions as a function of the maximum fractional (i.e., normalize the total area (2D) or volume (3D) to unity) element measure, and

- the best discrete solution q^b.

By examining the resulting slopes of $E(q^b - q^h)$ data versus Δ_e, on a $\log_{10} - \log_{10}$ graph, an estimate of the finite element semi-discretization convergence rate is possible. If the slope is equal to two for $k = m = 1$, then the nonlinear problem convergence rate follows the theory for the linear problem sub-class.

In practice, one complication of this approach is the corruption of the measurement from roundoff error. Round-off error underlies all computations in genuine multi-dimensional solutions, since a highly-refined grid is necessary for stability of NS systems. Hence, it becomes nearly impossible to firmly quantize energy norm convergence rates.

One alternative is to examine the energy-norm convergence rate of a less complex approximate solution to the form 2 boundary layer equations of Chapter 3. The space-marching solution to this problem indeed follows the energy-norm theory as will be seen later.

A higher order accurate alternative is to examine convergence in the H^0 norm which is a "natural" also for the problem class. Since the norm involves a square root radical, a slope of one for the convergence curve correlates with linear theory.

7.2 Quasi-Newton Iteration Convergence

The Newton iterative approximate solution for the Θ-implicit discretization procedure is presented in Chapter 5 as

$$\left[[M] + \Theta\Delta t \left[\frac{\partial R(Q)}{\partial Q}\right]^p_{n+1}\right] \{\delta Q^*\}^{p+1}_{n+1} = \tag{7.11a}$$
$$- \left[[M]\{\Delta Q^*\}^p_{n+1} + \Delta t \left(\Theta\{R(Q)\}^p_{n+1} + (1-\Theta)\{R(Q)\}_n\right)\right],$$

$$\{Q\}_{n+1}^{p+1} = \{Q\}_{n+1}^{p} + \left\{\delta Q^{*} \cdot (DETJ)^{-1}\right\}_{n+1}^{p+1}. \qquad (7.11b)$$

The solution to Equations 7.11a and 7.11a is approximated further by a tensor product matrix factorization of the Newton jacobian matrix. If the problem solution procedure is sufficiently linear between time steps, then a direct test of the quality of the quasi-Newton jacobian matrix is possible by examining the convergence rate of the iteration procedure. The absolute value of the base-10 logarithm [abs($\log_{10}$)] of the maximum fractional change in the solution field [$\left(\frac{\delta Q}{Q_o}\right)_{max}$] is chosen as the quantity to measure. After initial perturbations are allowed to settle through the solution field, the quasi-Newton iteration convergence rate is measured. The theoretical convergence rate of a full-Newton iteration should show a doubling of abs $\left(\log_{10}\left(\left(\frac{\delta Q}{Q_o}\right)_{max}\right)\right)$, i.e., for a nonlinear system of equations, a Newton iteration should yield a quadratic convergence rate between iterations. Since a quasi-Newton procedure is actually implemented, a degradation in convergence rate is expected, depending on the error in the matrix factorization and associated roundoff error. Since the steady residual tends to zero, as the solution approaches steady-state, the time step is usually limited by a maximum input value or a maximum input Courant number rather than a maximum residual change. These and other complications must be accounted for when attempting a measurement of this type.

7.3 Time-Step Selection

The estimation of admissible time step for all the simulations is tracked in terms of the extremum $\{R(Q^{n+1})\}$ and iterate $\{\delta Q^{n+1}\}$. Except for the initial stages when the residuals run up to a peak value, the residuals should monotonically decrease to a stable, steady-state. For the 1D and 2D inviscid problems, the stability margin (convergence interval) is relatively large. This allows the time

step to be automatically determined in a straight-forward manner. In most cases, the final steady-state time step is determined by specifying a maximum Courant number in the flow field.

However for viscous/turbulent flows, the stability margin is severely reduced by comparison, particularly for higher Reynolds number simulations ($Re > 10^4$), the primary cause due to the laminar wall sublayer. Although automatic time step selection criteria via maximum iterate residuals ($\delta Q/Q_o$) will essentially guarantee stable iterations, it has been found that this criteria places severe restrictions on the size of the time step and hence increases the length of the computational time. It has been found that this restriction can be overcome by manually selecting a minimum time step such that the stability-monitoring extremum residual norm (presented later) continues to decrease. By carefully continuing to manually increase the time step, an acceptable solution is obtained in (typically) half a dozen or less viscous restarts. Thus, the time step of the final viscous steady state has usually been manually determined rather than automatically via a extremum Courant number input definition.

Nearly all problems presented in this dissertation were setup such that the velocity components were essentially of the same order of magnitude throughout; typically at an angle of 45° from the horizontal. In this manner, the stability enhancements brought on by the TWS β term is maximized and the effects of matrix factorization and grid sweeping, which tend to reduce the convergence rate, are minimized. This is because the TWS β terms are directly proportional to the respective velocity measures (chapter 4). For problems where a predominate direction of flow exists, which was the case for all problems presented in Chapter 8, the transverse momentum equation TWS β will not be as effective as the streamwise direction, unless the angle of rotation is changed (typically 45°).

The time step is typically controlled by the quasi-Newton procedure only during the first few steps (in all cases less than 100) because beyond this point it tends to become equal to an input maximum; usually imposed by a Courant number limit. At the maximum time step, the fractional iterate residual at the first iteration is small and a quasi-Newton iteration convergence rate is no longer meaningful or does not exist if the initial iterate residual is less than the input minimum sought.

7.4 Evolution Stability

Predicting stability of an iterative algorithm for discrete approximations to nonlinear ordinary differential equation (ODE) is a demanding task. One method often utilized in examining the stability of nonlinear systems is that of Lyapunov.[120] For a system of nonlinear ODE equations in the form,

$$\frac{dq}{dt} = f(q,t), \tag{7.12}$$

if a positive definite functional $V(q)$ can be derived such that $\frac{dV}{dt} < 0$, for all $q, t > t_o$, then the system of Equation 7.12 is asymptotically stable. Most recently, Iannelli[35] has proposed an extension to Lyapunov's method, based on the criteria set forth by Lambert[113] and others,[114] in which the continuous form the conservation law equations are examined and analytically determined to be stable. This stability analysis and resulting convergence rate estimates are applicable to the present equation system. However, unlike an analytical method, it is desirable to determine and maintain the algorithm stability on line, i.e., as the NS approximate solution is progressing.

Two classically utilized linear analysis techniques were examined for applicability. Von Neumann's linearized analysis[110] determines an amplification factor $\} \equiv \frac{q^{n+1}}{q^n}$, where n is the time step index, after a Fourier transform into

the frequency domain. A necessary condition for stability is that no positive real exponential functions exist. This method was not considered further, since the resulting linearized amplification factor for this nonlinear set of equations would be marginally useful.

Another stability analysis technique, called the "matrix method",[110,109] requires that the fully-discrete form of the algorithm ultimately be written in the form $Q^{n+1} = [A]Q^n$. A necessary condition for a stable algorithm is that the real parts of the eigenvalues of the matrix $[A]$ be less than -1. It was originally envisioned, in the dissertation proposal, that the matrix method could be utilized to directly determine algorithm stability as the solution progressed. During the course of the research, it was found that this was not possible. The reason for this is that it is difficult, if not impossible, to derive the matrix $[A]$ for the present algorithm. Furthermore, even if $[A]$ can be derived, due to the iterative procedure embedded in the nonlinear solution algorithm, the best that can be said about the algorithm solution is that it is bounded rather than stable. Recent findings[111,112] have shown that further research is needed to directly determine stability of a generalized equation system solution algorithm such as is in the present dissertation.

As a consequence of these analyses, it appears that an "on-line" stability monitor is achieved by applying a Lyapunov stability criterion for the functional

$$V(Q^{n+1}) = \frac{1}{N}\sqrt{\sum_{i}^{N}\{R(Q^{n+1}\}_i \cdot \{R(Q^{n+1}\}_i}, \tag{7.13}$$

where $\{R\}$ is the algorithm extremum residual evaluated at the most current time station $n + 1$, and N is the rank of the quasi-Newton algorithm. Since $V(Q)$ in Equation 7.13 is positive definite, it is an eligible Lyapunov functional. The requirement is then to monitor the time rate of change in V as the solution approaches steady state, hence verify that it is maintained uniformly less than

zero, i.e., the Euclidean norm of the residual is always decreasing. This stability criteria has been verified as a sufficient condition for the large majority of the achieved evolutions to steady state. One exception is during the early transient portion of the solution, where the time step is increasing rapidly and the Lyapunov criteria alone was not sufficient to predict stability exists. This is quite understandable since the temporal partial derivative, also a decreasing function during the approach to steady state, is not present in the Lyapunov function defined by Equation 7.13, i.e., the time derivative is not included in the Euclidean norm of the extremum.

Several other aids to estimating online stability have been developed during the course of this research. For a given sufficient mesh, the fundamental issue is to determine the maximum time step. The linearization required for the quasi-Newton iteration procedure can be utilized to confirm the boundedness of the residual at each time step. This is analogous to examining the boundedness of the residual at each time step by a matrix method, in the equation form

$$\Delta t^{n+1} = \frac{\Delta t^n \left[\epsilon_{\text{set}} \pm \delta\epsilon\right]}{\left(\frac{\delta Q}{Q_o}\right)_{\max}}, \tag{7.14}$$

where $\left(\frac{\delta Q}{Q_o}\right)_{\max}$ is a maximum normalized residual defined earlier (when examining the linear algebra procedure convergence rate), $[\epsilon_{\text{set}}]$ is a fractional residual input set point, and n is a time step index. The input parameter $\delta\epsilon$ allows a window to exist about the set point to minimize the possibility of "cycling" while searching for a new time step. Equation 7.14 tends to decrease the time step when the residual becomes too large. This allows the time step to follow an anticipated course toward a maximum value, further limited by an input minimum and maximum allowed value.

Another parameter utilized to estimate and control the time step is a Courant (or CFL) number. Definitions used for this research include

$$\mathcal{C} \equiv \max\left(\mathcal{C}_{x_{(i,j)}}, \mathcal{C}_{y_{(i,j)}}\right), \tag{7.15a}$$

$$\mathcal{C}_{x_{(i,j)}} \equiv \frac{\Delta t\left(u_{(i+1,j)} + u_{(i-1,j)}\right)}{x_{(i+1,j)} - x_{(i-1,j)}} \tag{7.15b}$$

$$\mathcal{C}_{y_{(i,j)}} \equiv \frac{\Delta t\left(v_{(i,j+1)} + v_{(i,j-1)}\right)}{y_{(i,j+1)} - y_{(i,j-1)}} \tag{7.15c}$$

where $(\cdot)_x$ and $(\cdot)_y$ denote Cartesian coordinates in the direction of velocities u and v respectively, and the indices (i, j) reference the node numbers (in the transform space). Typically, the Courant number is utilized to limit the time step after the quasi-Newton iterate residual δQ is less than the set point ϵ_{set}. In this context, the Courant number is used as a non-dimensional measurement of how large the time step can be and still yield convergent solutions. During the course of this research it has been found that rather large Courant numbers ($> 10^2$) could be achieved without diverging. However, as the solution progressed it was found that a limit cycle was generated for the larger time steps causing the solution to not completely converge. Utilizing the Euclidean norm of the right-hand side residual as the Lyapunov functional, it was later realized that the Lyapunov stability criteria could nearly always be satisfied if the time step were limited by a Courant number of four ($\mathcal{C} = 4$). However, this is not proposed to be completely general since stable solutions were achieved using larger (quasi-1D) and smaller (viscous 2D) corresponding time steps.

Finally an additional stability enhancement, called a "digital filter", has been incorporated into the algorithm. The idea is to impose a filter on the solution of the system Q such that the frequency of the time step oscillations imposed by the discrete forcing function (Δt) can be eliminated. The derivation is given in Appendix F. The net result is that the solution array is altered by

the following equation

$$Q^{n+1} = \frac{Q^n + K_\omega \Delta t \omega_b Q^{n+1}}{1 + K_\omega \Delta t \omega_b}, \tag{7.16}$$

where ω_b is a desired (input) "break" frequency, and K_ω is an input constant. Since in reality time is continuous, the imposition of a fully discrete solution procedure via the time step Δt imposes an artificial time-step frequency onto the solution. It is desired to reduce, if not eliminate, this frequency from the discrete solution. If the break frequency is exactly equal to the inverse of the time step, then the digital filter can optimally remove the time-step frequency by the proper choice of K_ω. K_ω equal to unity guarantees elimination of all frequencies above the time-step frequency and results in the following equation

$$Q_i^{n+1} = \frac{1}{2}(Q_i^n + Q_i^{n+1}). \tag{7.17}$$

Originally implemented by Penland and Hedrick,[121] this digital filter has been extremely effective in accelerating the approach to steady-state convergence, especially for the larger time steps near the steady-state condition. Equation 7.17 significantly alters the time accuracy of the solution, which was not of impact since the goal of this research is strictly convergence at steady state.

Chapter 8

Algorithm Assessment

This chapter presents assessment of the developed algorithm for critical criteria including accuracy, consistency, stability, and convergence. Three model problems are chosen, such that each of these criteria may be individually and collectively examined. During the process, most aspects of algorithm performance and capabilities of the implemented pilot code are exercised and discussed.

Accuracy of the algorithm is determined by comparing the calculated results to theory and data. During pilot code development, the algorithm verification involved comparison with simple problems such as a converging subsonic nozzle. Since one objective is to verify a real-gas capability, the more difficult and rewarding challenge comparisons are shown with experimental data for a transonic wind tunnel with shock-boundary layer interactions. Finally, the algorithm is applied to the challenging benchmark problem of a GAMM workshop double throat nozzle with an oblique shock and supersonic wave compression/reflection.

8.1 Subsonic Converging Duct

The subsonic converging duct is an internal-flow, viscous problem, with minimal nonlinear contributions, that allows for basic verification while meeting specific solution goals including:

- confirm that mixed Dirichlet and Neumann boundary conditions are well-posed,
- confirm correct determination of laminar and turbulent boundary layers via combined viscous and modulated artificial dissipation,
- validate the convergence rate of the quasi-Newton iteration procedure and time-step sensitivity,
- examine the finite element discretization convergence rate with respect to mesh refinement and tensor-product factorization errors, and
- test the finite-element grid versatility via the generalized coordinate transformation.

Of paramount importance is the effect of the TWS artificial dissipation (β term) upon the true solution. Even though this problem has no embedded shock, the artificial dissipation is exercised to determine effects on the natural dissipation. By assuming similarity, the viscous numerical solution is directly compared to the classical flat plate analytical solutions for laminar (Blausius[122]) and turbulent (Coles[123]) boundary layers.

The simplest flat-plate boundary layer problem did not meet validation criteria since it would not generate sufficient density change to adequately enforce continuity within the formulated (subsonic) compressible Navier-Stokes equations.

By specifying a straight duct to begin downstream of a 10% area change, enough compressibility (hence, change in density) occurred such that the continuity equation was computationally sensitive for the fully-discrete density variable.

8.1.1 Problem Statement

The subsonic converging duct geometry is shown in Figure 8.1. The duct boundary surfaces are numbered $1 \rightarrow 4$ representing the left, right, bottom, and top respectively. Isentropic flow is assumed in establishing initial and boundary conditions shown in Table 8.1. Quantities shown with an asterisk designate input while the remainder are calculated. The stagnation values shown in the third column indicate a reference value used to transform the dependent variables to dimensionless form. The reference values are printed at solution initialization and serve as the first level of theory verification.

Table 8.1: Subsonic Converging Duct Problem Isentropic Flow Initial and Boundary Conditions

Quantity	Inlet	Outlet	Stagnation
$T(^o\mathrm{R})$	530.0*	507.39	572.29
A (ft^2)	.5*	.45*	**
ρ ($\mathrm{lbm/ft}^3$)	.0747	.06699	.0905
u (ft/sec)	713.23	883.73	**
a (ft/sec)	1129.0	1104.66	1173.19
p ($\mathrm{lbf/ft}^2$)	2112.48*	1813.28	2763.32
M (none)	.6317	.8*	**

$*$ input quantity, ** not meaningful for a stagnation variable

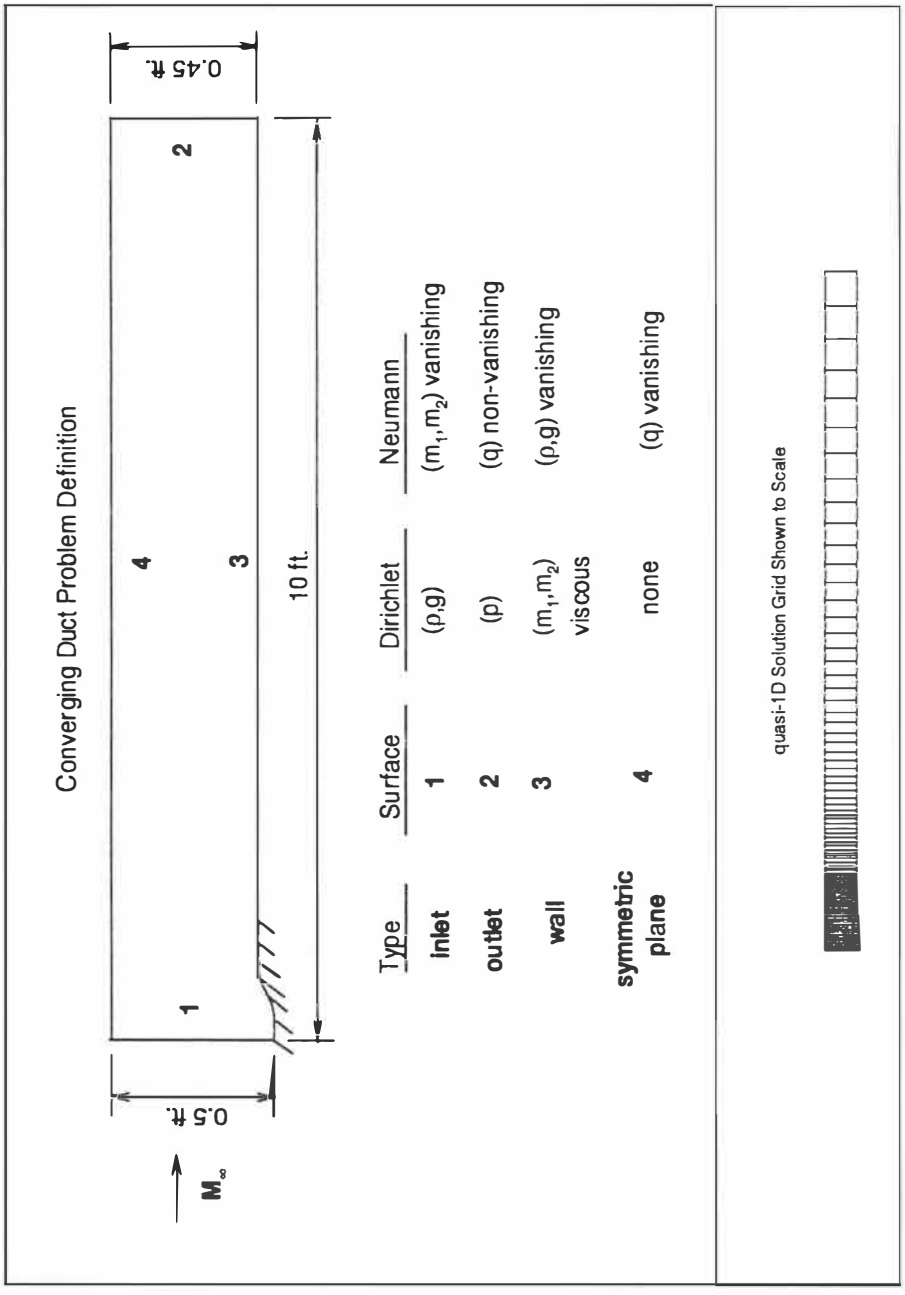


Figure 8.1: Subsonic Converging Duct Problem Definition.

The assumption of isentropic flow is essentially valid here for the inviscid solutions (quasi-1D or 2D), hence they should match the input and output quantities shown in Table 8.1. However, a key assumption in arriving at an isentropic solution is constant mass specific stagnation enthalpy (H). Since volume specific stagnation enthalpy ($g = \rho H$) is the computed energy equation state variable, a non-constant H will most likely result. However, as will be seen later, the solution does yield a nearly constant H and therefore matches the isentropic solution quite well.

Figure 8.1 defines the mathematically-determined well-posed boundary conditions, as described in Chapter 6. Dirichlet boundary data are held at the inlet (surface 1) for density and stagnation enthalpy (ρ, g) and at the outlet (surface 2) for pressure (p). Flow tangency ($\boldsymbol{m} \cdot \hat{\boldsymbol{n}} = 0$) is naturally enforced at surfaces 3 and 4 by not assembling the convection surface terms that arise from the finite element weak statement discretization (Chapter 6). However, these terms must be evaluated and included at the inlet and outlet ($\boldsymbol{m} \cdot \hat{\boldsymbol{n}} \neq 0$) on surfaces 1 and 2. For variables that do not have Dirichlet data imposed at a surface, vanishing derivatives are enforced due to the assumed zero surface terms arising from the integration by parts of the TWS dissipation (β) term ($(\hat{\boldsymbol{n}} \cdot \boldsymbol{\nabla})q = 0$). When solving the viscous extension to this problem, a no slip wall is imposed by specifying zero momentum ($\boldsymbol{m} = \mathbf{0}$) at the wall on surface 3, just downstream of the area change (approximately one foot from the inlet). The viscous dissipation terms also enforce outflow boundary non-vanishing derivatives on surface 2 as described earlier in Chapter 6. For the laminar viscous case, a Reynolds number of 10^3 defines the length scale. Therefore, for the flow to be fully developed, the duct would have to be 80 feet long for the velocity profile to become parabolic at the exit plane.[122] Therefore, the velocity distribution is expected to appear essentially inviscid along the centerline and adjacent regions.

8.1.2 Geometry Specification

A quadratic algebraic equation was derived to smoothly describe the wall curvature of surface 3, such that the wall tangent plane is continuously turning between segments. The selected form is

$$y = a + bx + cx^2. \tag{8.1}$$

The origin of the (x, y) system is at the intersection of surfaces 1 and 3. The constants in Equation 8.1 are given in Table 8.2.

The geometry is defined by inputting the eight Cartesian (x, y) coordinate point as the nodes of a biquadratic-basis finite element for a selected number of macro elements.[87] These data are keyboard input or generated by an auxiliary program. The input allows for a variable number of macro elements discretizing the two directions. Each macro element is thereafter allowed to be refined into a specified number of finite elements of non-uniform measure distribution via a variable progression ratio.[87] The net result is a two dimensional grid of highly structured coordinate lines in each direction. This approach allows the grid to have regions of variable refinement, enabling the algorithm solution to better

Table 8.2: Constants Used to Define the Converging Inlet Wall Curvature for the Subsonic Converging Duct Flow Problem.

a	b	c	location
0.0	0.0	0.0	$0.0 \leq x \leq 0.2$
0.011	-0.111	0.277	$0.2 \leq x \leq 0.5$
-0.1277	0.444	-0.277	$0.5 \leq x \leq 0.8$
0.05	0.0	0.0	$0.8 \leq x \leq 10.0$

resolve detailed phenomena caused by steep fluid property gradients such as shocks or rapid geometry variations. The geometry remains fixed during the solution process. A natural extension to the present algorithm would therefore be an "adaptive" grid procedure, in which the grid would automatically be reconfigured toward a determined optimum condition as the solution progresses.

For a typical viscous problem with shocks, there are at least four grid input specifications corresponding to separate problems being solved: (1) 1D inviscid with uniform grid; (2) 2D inviscid with uniform grid; (3) 2D inviscid with nonuniform grid to refine the solution around non-smooth and/or rapidly changing regions; and (4) 2D laminar and/or turbulent viscous with nonuniform grid to refine the mesh in the viscous boundary layer region. An additional viscous grid specification may be required for turbulent flows. The distinction between these grid specifications is not always straight forward. For example, it may be obvious from the 1D solution that a nonuniform grid is required and the second, uniform grid specification might be eliminated altogether.

All grids utilized to solve this subsonic converging duct problem have the same specification in the principal (η_1) direction. Figure 8.1 includes the grid used for the quasi-1D problem showing only one finite element in the transverse direction. A single element in the η_2 direction is a special indication to the pilot code to solve the quasi-1D form of the governing equations. Figure 8.2 shows the grid used for the 2D inviscid coarse mesh showing additional mesh solution detail in the transverse (η_2) direction yielding 3168 total nodes for the 2D inviscid solution.

A discussion in Chapter 6 states that it is often appropriate to restart and again solve the inviscid problem with a more refined grid specification. This level of grid refinement is not as critical for this subsonic converging

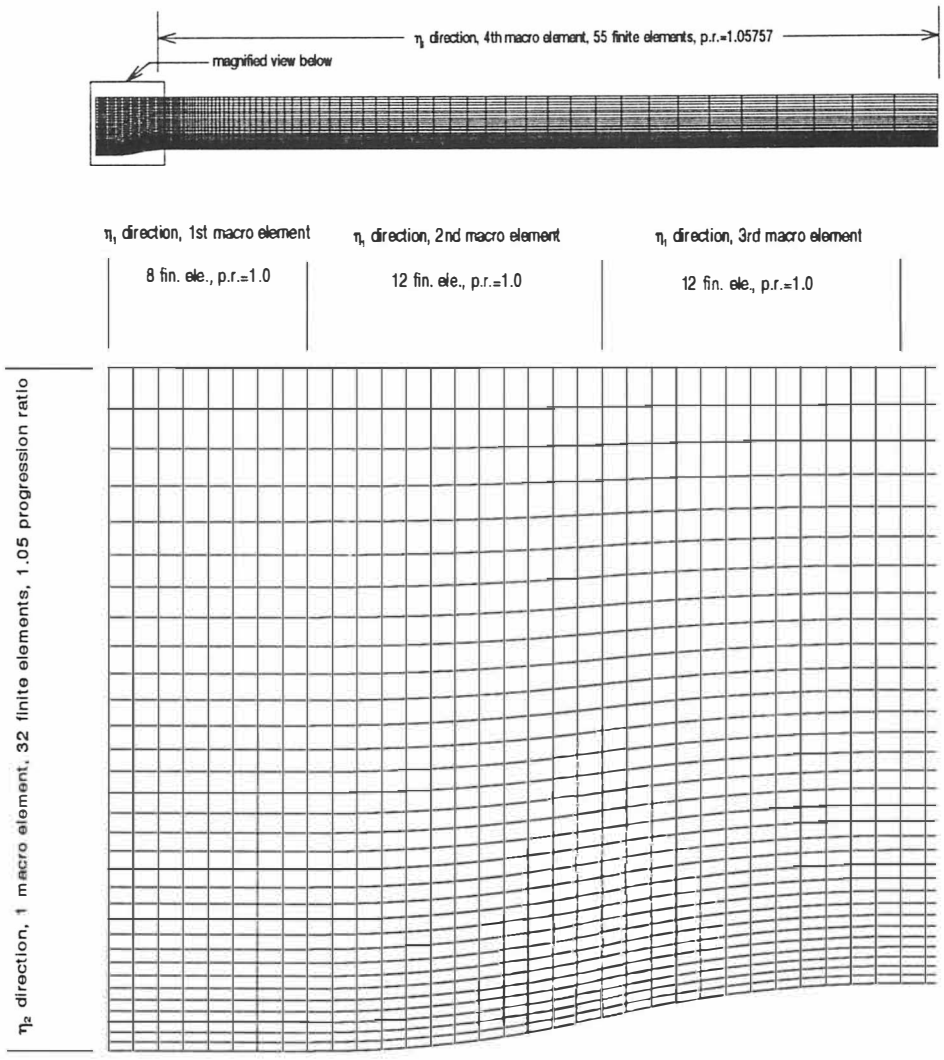


Figure 8.2: Coarse Mesh Grid for the 2D Inviscid Solution of the Converging Duct Problem

duct problem as others.[a] It is known from past research that there is an upper limit of grid refinement above which additional refinement does not offer additional accuracy;[24] i.e., roundoff error will eventually dominate a norm of the difference between the approximate solution and the "true" solution. Therefore, no additional mesh refinement was incorporated into the 2D inviscid grid for this problem.

The viscous grid must incorporate mesh refinements in the direction normal to non-slip walls; typically three macro elements are utilized for this purpose. Figure 8.3 shows a mesh for the viscous solution with the macro-element refinement near the wall in order to capture the laminar and turbulent boundary layer growth. The first macro element nearest the wall is the smallest, with the first grid spacing typically in the range ($L/Rey \rightarrow L/\sqrt{Rey}$, and captures the laminar region caused by the approximately linear velocity profiles. The second macro element from the wall has a progression ratio greater than one and captures the transition region from laminar to inviscid fluid. For turbulent flows, this region also captures the effect of the algebraic turbulence model. Finally the third macro element lies predominantly in the inviscid flow region. In this particular case, the mesh is the same for both the laminar and turbulent grids. Furthermore, the grid in the η_1 direction is the same as in the inviscid cases described above. Figure 8.3 gives the details showing a total of 5376 nodes for this viscous grid. As mentioned in Chapter 6, it is often advantageous to specify separate laminar and turbulent grids in order to properly capture all flow details, but this was not necessary for this problem.

[a] e.g., it is advantageous to refine the grid around a shock.

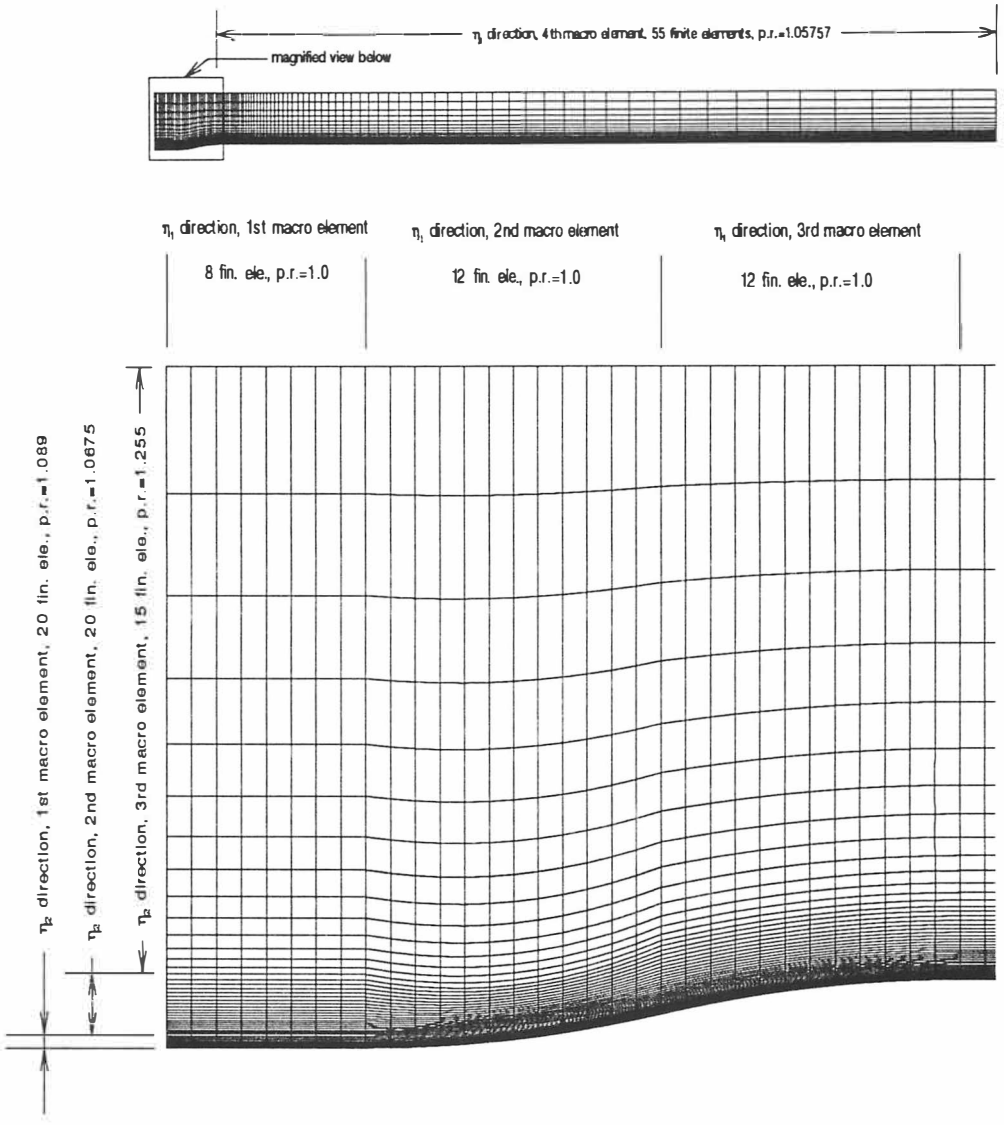


Figure 8.3: Grid Specification for the 2D Viscous Solution of the Converging Duct Problem

8.1.3 Quasi-One Dimensional Flow Results

Accuracy, Consistency

The quasi-1D state variable steady solution for the subsonic converging duct is shown in Figure 8.4. The four members (ρ, m_1, m_2, g) are shown in physical space with the region of change appearing in the very small and highly refined grid about the area change. Figure 8.5 shows the same solution graphed in nodal space, with smooth, vanishing gradients for all variables at the inlet and outlet. The only change occurs as expected across the converging inlet region.

The density (ρ) and volume-specific stagnation enthalpy (g) have essentially the same appearance. This is because the mass-specific stagnation enthalpy is computed as essentially constant ($\approx$ 1% variation) as expected for an isentropic solution. The transverse momentum is zero as expected for the quasi-1D solution. Therefore, this solution can be compared to the 1D isentropic analytical solution that was used to compute the initial and boundary conditions given by Table 8.1. Table 8.3 shows the calculated and analytical results for the state and other variables. The tabulated quantities are given in non-dimensional form utilizing the stagnation reference conditions from Table 8.1. The computed results show excellent agreement with the theory as shown to four significant digits. The relative difference is no greater than 2.5% for all variables shown.

Convergence of the Newton Iteration

From Chapter 7, a quadratic convergence rate is expected for a Newton iteration procedure if the step size is sufficiently small. For the quasi-1D problem, a full Newton iteration is utilized since the jacobian matrix is not factored, i.e., gridline sweeps are performed in only one direction. Therefore, quadratic convergence is

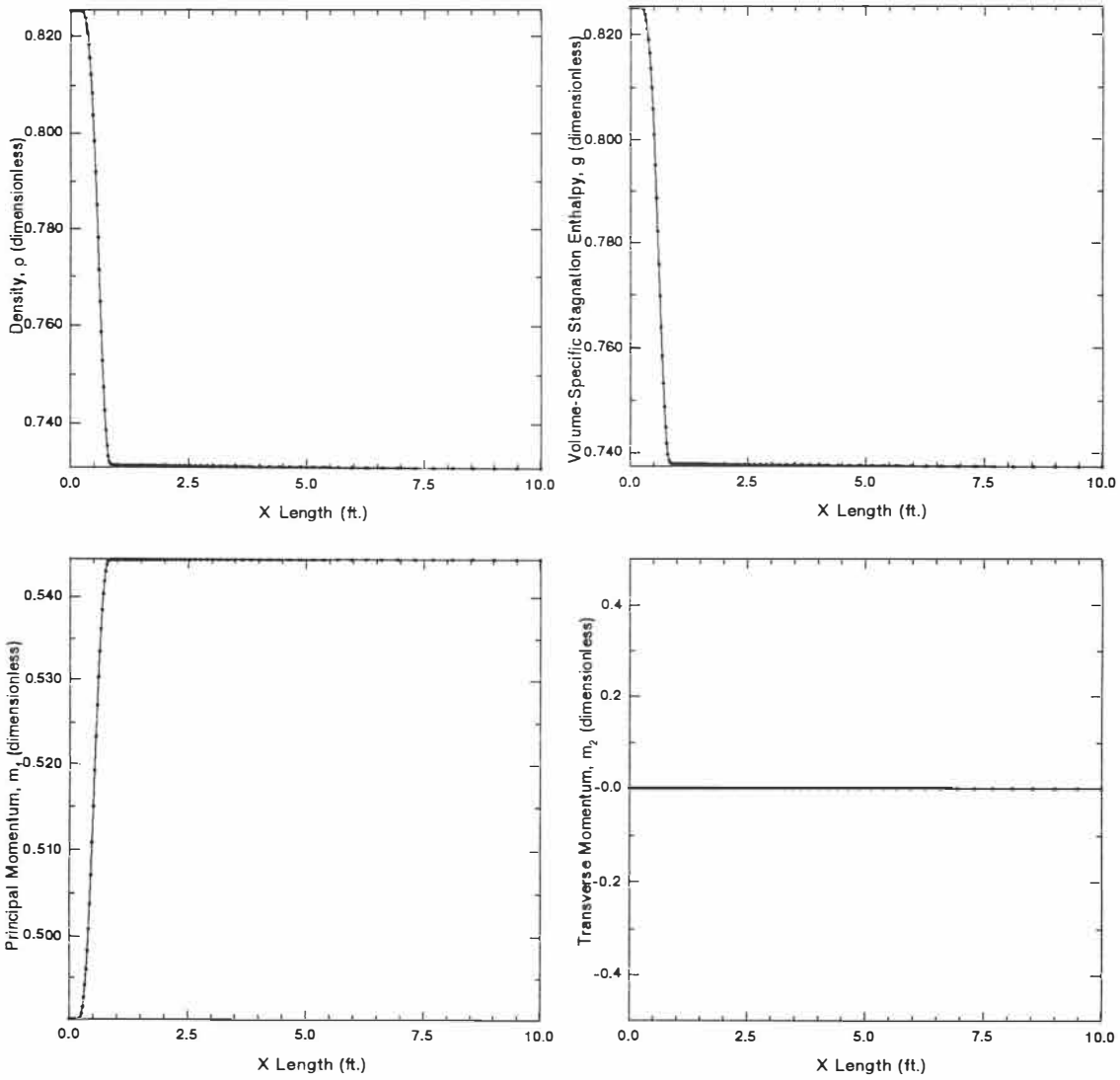


Figure 8.4: Quasi-1D Euler Solution of the Subsonic Converging Duct Problem Plotted in Physical Space

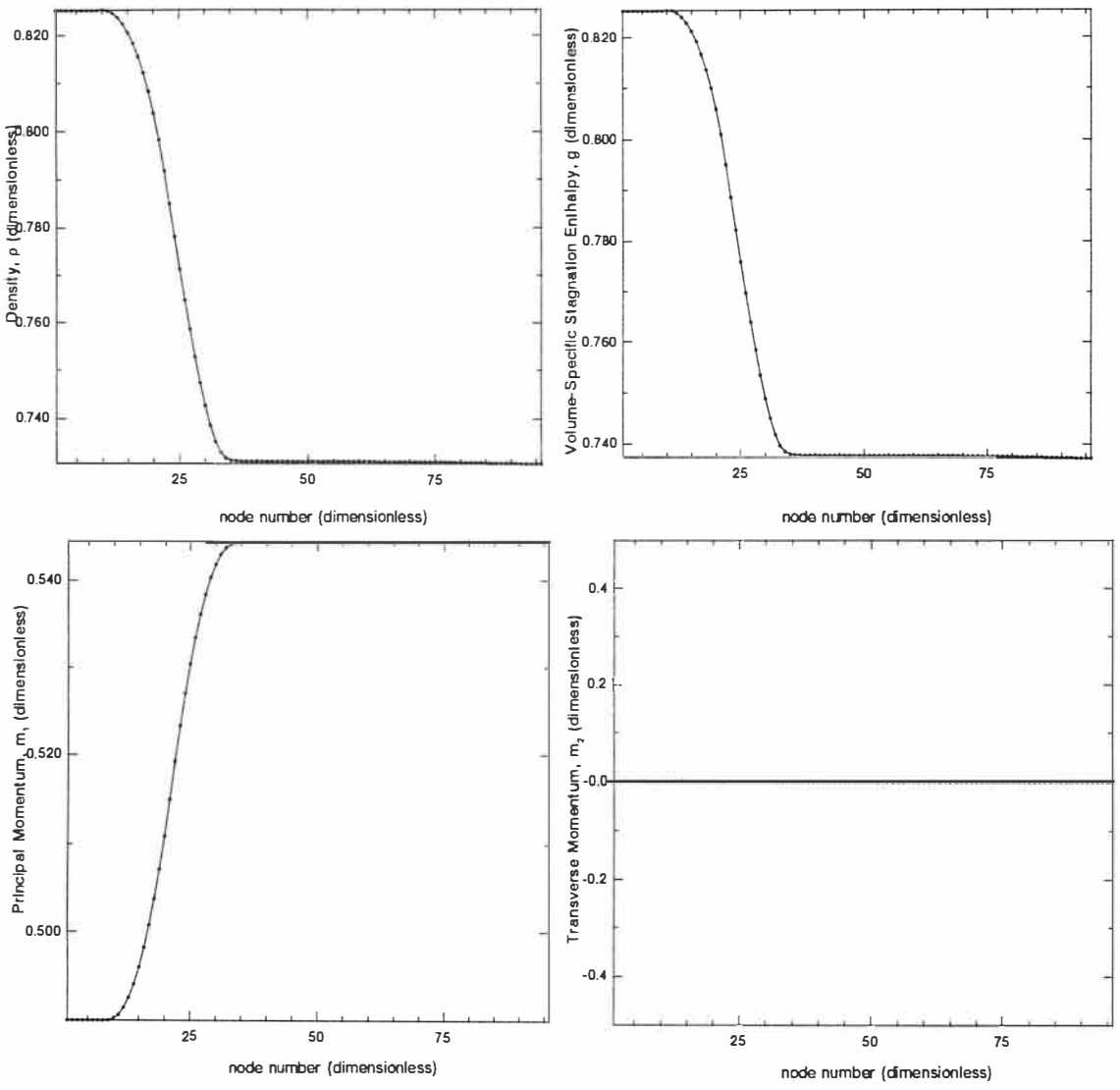


Figure 8.5: Quasi-1D Euler Solution of the Subsonic Converging Duct Problem Plotted in Node Space

Table 8.3: Comparison of the Computed Quasi-1D Euler Solution with the Theoretical Isentropic Solution for the Subsonic Converging Duct Problem

Quantity	Inlet		Outlet	
	Analytical	Computed	Analytical	Computed
ρ	.8254	.8252	.7402	.7308
m_1	.5018	.4900	.5576	.5445
g	.8254	.8253	.7402	.7373
u	.6079	.5938	.7533	.7451
T	.9261	.9276	.8866	.8962
p	.7645	.7657	.6562	.6550
M	.6171	.6317	.8000	.7877
H	1.00000	1.00001	1.0000	1.00902

expected if the Newton iteration jacobian matrix is derived and coded correctly. Thus, a measure of how far off from the theoretical goal (quadratic) the actual convergence rate is also measures the jacobian matrix quality. In Chapter 7, $\left(\frac{\delta Q}{Q_o}\right)_{max}$ is given as the quantity to measure the quasi-Newton convergence rate. This quantity is expected to double for each successive iteration if the convergence rate is exactly quadratic. In other words, if the quasi-Newton convergence rate could achieve that of the full Newton quadratic convergence rate, then the jacobian is correct. For example, if $\left(\frac{\delta Q}{Q_o}\right)_{max}$ is unity on the first iteration, then successive iterations should (theoretically) yield $2, 4, 8, \ldots$

Table 8.4 lists representative measured convergence rates for various initial $\left(\frac{\delta Q}{Q_o}\right)_{max}$ values. As $\left(\frac{\delta Q}{Q_o}\right)_{max}$ gets below about 0.1% in all columns, the convergence rate has a change per iteration of about 0.3 indicating linear convergence. The match with the theoretical expectation for a Newton iteration (quadratic convergence rate) occurs only at about a 10% residual change and then

Table 8.4: Quasi-Newton Iteration Convergence Rate for the Quasi-1D Subsonic Converging Duct Problem.

	Initial Normalized Change					
Iteration	.1%	1%	2%	5%	10%	20%
1	3.0	2.0	1.7	1.3	1.0	0.7
2	3.3	2.5	2.3	2.1	1.9	2.0
3	3.6	3.0	2.8	2.4	2.4	2.3
4	3.9	3.3	3.1	2.7	2.7	2.6

Quadratic convergence is indicated by
doubling of the tabulated values
between successive iterations.

only in the first two iterations. Typically, a maximum of a 5% initial normalized change has been the limit for most solutions discussed in this dissertation.

One cause for the degradation in theoretical convergence rate given by Table 8.4 is that the Newton iteration jacobian (matrix) does not include a correct contribution from the source term arising from the area change in the quasi-1D equation system. Because this information is not needed in the 2D case, it was not entered for the 1d case. Therefore, except in a very narrow range of fractional residual change, the Newton quadratic convergence rate will not occur. Later in the 2D results (which do not have a source term) it will be shown that the quasi-Newton iteration convergence rate follows the theoretical expectations more closely.

To further confirm this assertion, the Newton convergence rate was examined for a 1D problem without a source term. The Riemann shock tube problem is well documented[25] and is commonly solved as a benchmark for CFD solution performance. Table 8.5 lists comparison Newton iteration convergence rates at midpoint in the time-dependent solution of the 1D Riemann shock tube problem. The convergence rate of this approximate problem solution iteration follows the theoretical linear expectations over the portion where the residual is still relatively large. Below a residual of about 10^{-3}, roundoff error and time step contributions adversely affect the convergence rate.

Also included in Table 8.5 is the effect of different forms of Newton iteration jacobian matrix. Recalling Chapters 5 and 6, the pilot code variable `SOLTYP` controls the form of the jacobian matrix. Jacobian specifications `SOLTYP`=2,1, and 0, represents the full, reduced stored in full-band, and reduced stored in tridiagonal-band respectively. The `SOLTYP`=0 convergence rate performs very well remaining essentially one iteration behind the `SOLTYP`=2 for the entire

Table 8.5: Comparison Quasi-Newton Iteration Convergence Rates at Midpoint in the Solution of the 1D Riemann Shock Tube Problem as a Function of System Iteration Jacobian Matrix Formulations

	$-\{\log_{10}\{\left(\frac{\delta Q}{Q_o}\right)_{max}\}\}$		
Iteration	SOLTYP=2	SOLTYP=1	SOLTYP=0
1	-.15	-.16	-.16
2	.90	0.28	0.29
3	1.22	1.01	1.11
4	2.43	1.93	2.07
5	3.48	2.79	3.02
6	4.48	3.67	3.98
7	5.47	4.55	4.95
8	6.47	5.43	5.90
9		6.31	6.86

Quadratic convergence is indicated by doubling of the tabulated values between successive iterations.

duration of the iteration. Furthermore, as would be expected, the effect of reordering the solution array to allow a tridiagonal jacobian matrix does not alter the solution nor the convergence rate significantly; i.e., the SOLTYP=1 data are essentially the same as SOLTYP=0. As will be discussed later, the reduced jacobian performance is worthwhile for the 2D inviscid problem, but convergence degrades significantly for the 2D viscous problem due to matrix factorization approximations.

Convergence with Discretization Refinement

In Chapter 7, a linearized theoretical analysis for convergence rate of the finite element discretization refinement was presented. The determined convergence rate $\| e^h \|_E$ for the steady state finite element quasi-1D solution for subsonic converging duct is shown in Figure 8.6 for variables (ρ, m_1, g, u) in comparison to the theory. Theoretical convergence lines are shown with slope 2 for direct visual comparison which is very good. These results were obtained for a grid range of about $10 \rightarrow 200$ finite elements. For grids outside this range (below or above), other error mechanisms exist to compromise the definitive experiments, e.g., round off, time step, and initial data.

8.1.4 Inviscid 2D Solution

The steady-state quasi-1D solution initializes the 2D inviscid simulation following the described procedure. Figure 8.7 shows iso-contours for the 2D steady-state inviscid solution of the subsonic converging duct. The plot vertical scale has been expanded by a factor of approximately ten for clarity. Figure 8.8 shows the companion iso-contours of pressure, principal velocity, transverse velocity, and temperature, as determined from the state variable solution. For the Mach=0.8

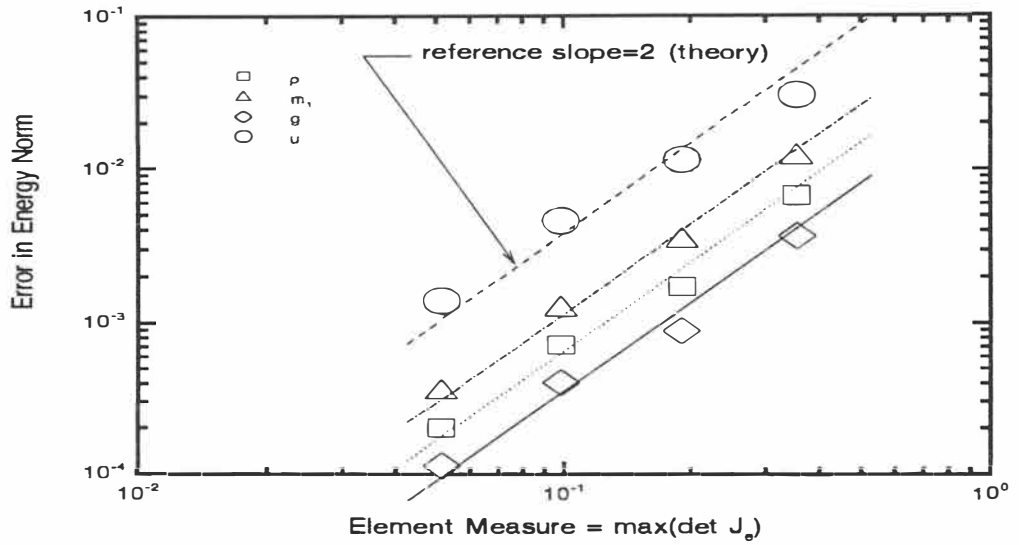


Figure 8.6: Convergence Rate for the Error in Energy Norm for the Quasi-1D Finite Element Euler Solution of the Subsonic Converging Duct Problem

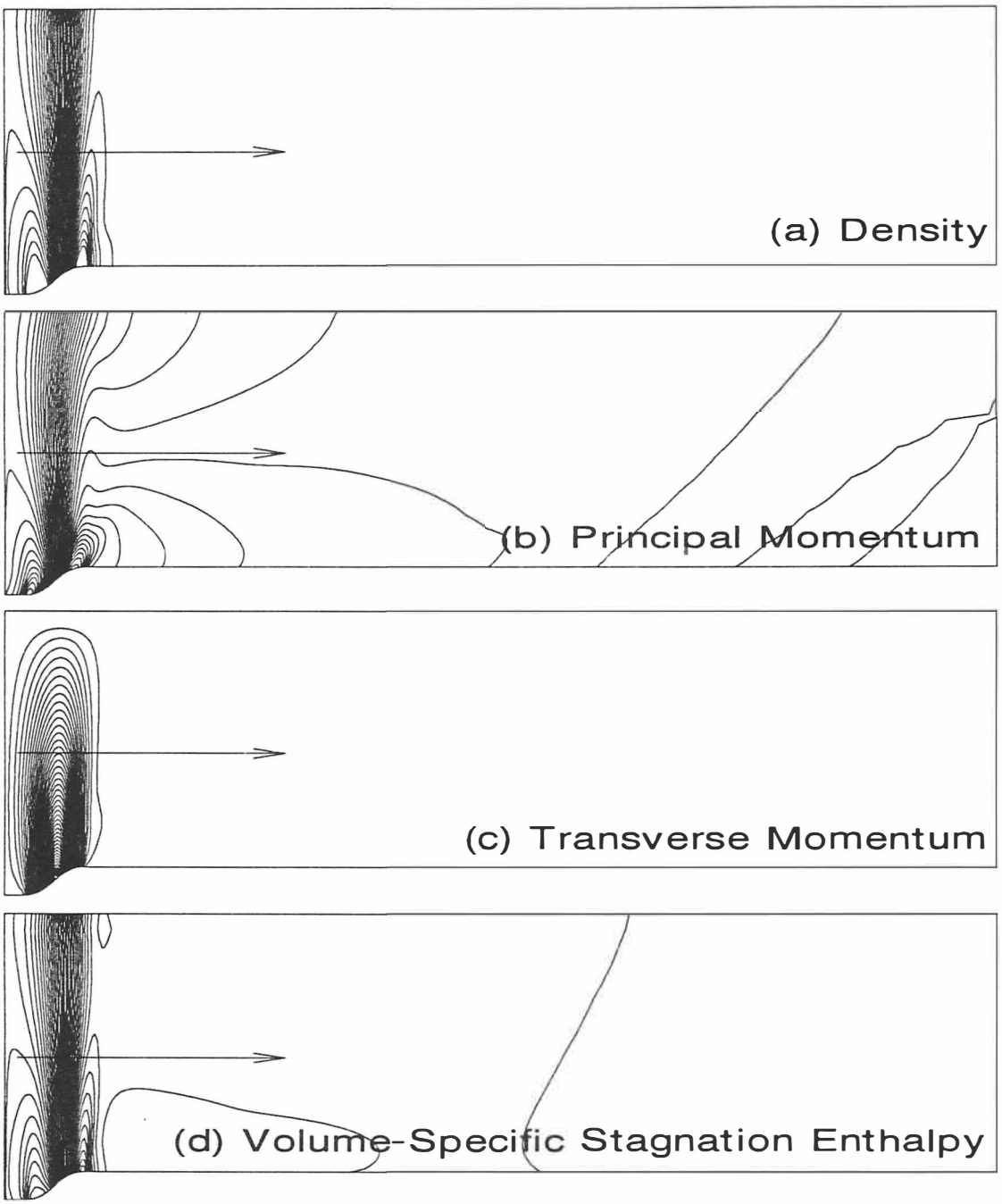


Figure 8.7: Inviscid Solution of the Subsonic Converging Duct Problem Shown as Iso Contours in Physical Space; (a) Density, (b) Principal Momentum, (c) Transverse Momentum, and (d) Volume-Specific Stagnation Enthalpy (vertical scale expanded for clarity).

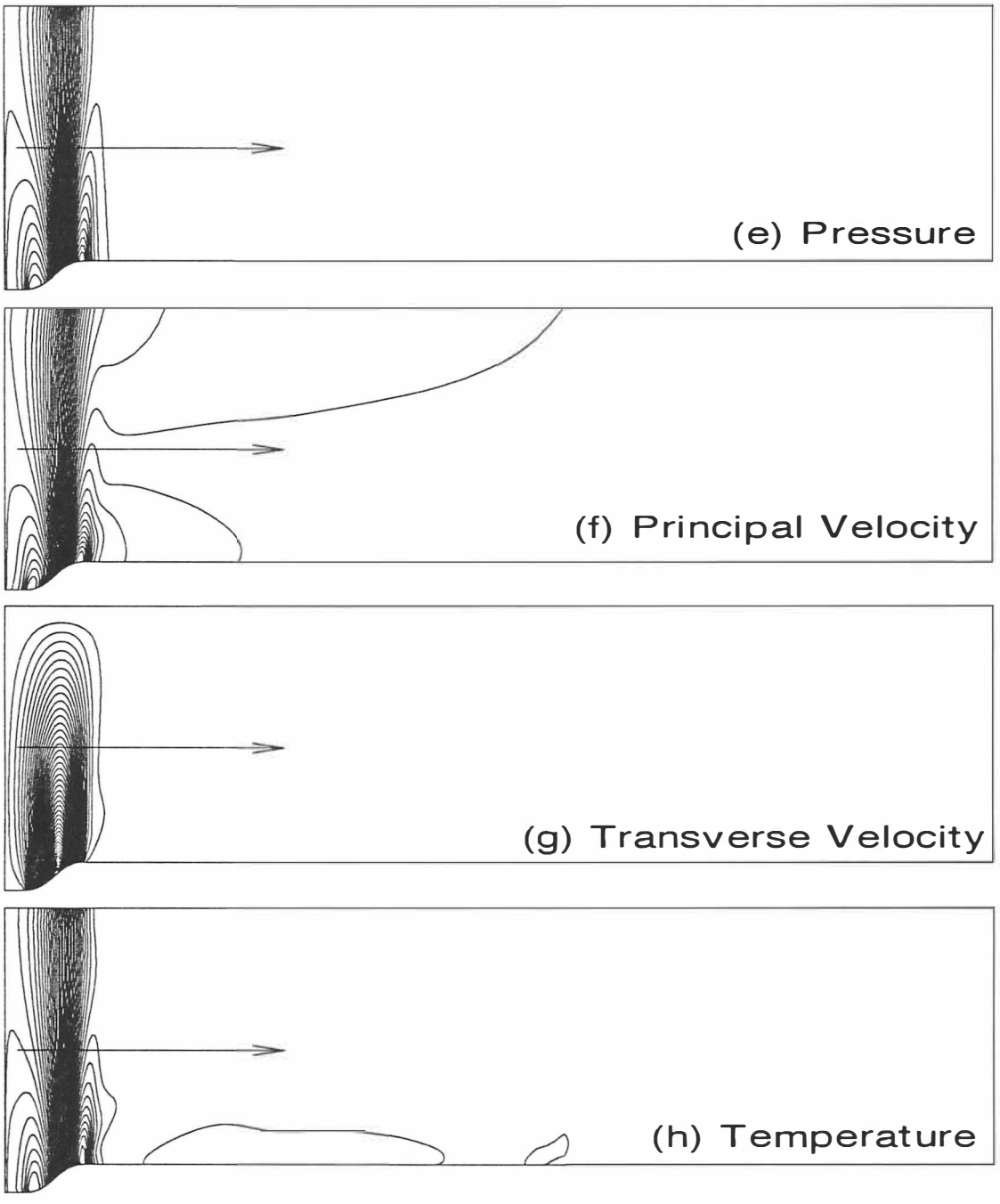


Figure 8.8: Inviscid Solution of the Subsonic Converging Duct Problem Shown as Iso Contours in Physical Space; (e) Pressure, (f) Principal Velocity, (g) Transverse Velocity, and (h) Temperature (vertical scale expanded for clarity).

outlet flow, the solution is indeed essentially incompressible, as verified by the iso-contours of momentum and velocity being nearly the same shape. The solution is also essentially isentropic, since the density and volume-specific stagnation enthalpy iso-contours have similarities, i.e., the mass-specific stagnation enthalpy ($H = g/\rho$),is nearly constant. Although not apparent from the contour map, only a 4% variation in temperature is observed; therefore the density and pressure contours are also of similar shape.

A key validation is that the solution is free of oscillation, which is easily assessed by viewing carpet plots in a computational nodal space, i.e., wherein the abscissa and ordinates are node numbers. This presentation spreads regions of highly refined mesh to provide an improved perspective especially for assessing essentially non-oscillatory (ENO) character of the solution. Figure 8.9 shows these data looking upstream into the duct, (from the top exit), while the data presentation of Figure 8.10 is viewed looking downstream from the flow entrance (opposite corner of the previous view). Because the grid is sufficiently detailed, all variables are smooth in the region of change in duct area near the wall. Indeed, for this inviscid problem, that is the only region of interest as seen by the high concentration of iso-contours for all variables shown in Figures 8.7 and 8.8. The long region of essentially constant solution (all variables beyond node $\xi = 42$) is present to later verify the viscous boundary layer development. Dirichlet boundary conditions cause a slight jump in the solution surface of ρ^h and g^h adjacent to the duct entrance plane, Figure 8.10. Vanishing normal derivatives are evidenced at all other boundaries, from the natural boundary conditions as a result of the TWS β term.

An ENO solution is indeed verified throughout, except for a modest transverse wave in each variable near the duct exit plane. The wave is visibly present in part b, principal momentum m_1 iso-contour, of Figure 8.7 and part

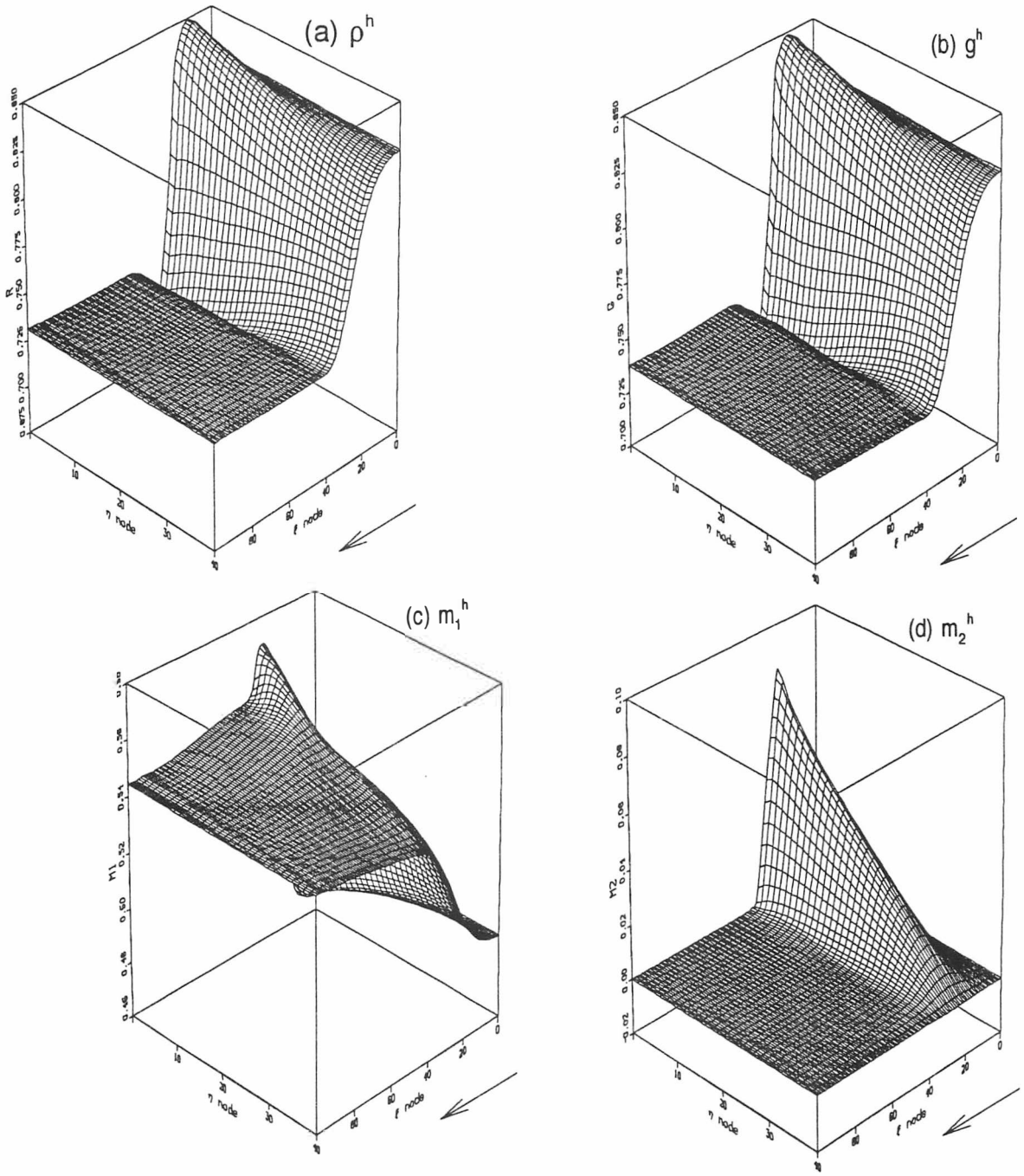


Figure 8.9: Inviscid Solution of the Subsonic Converging Duct Problem Plotted in Nodal Space as Viewed from the Top Exit. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

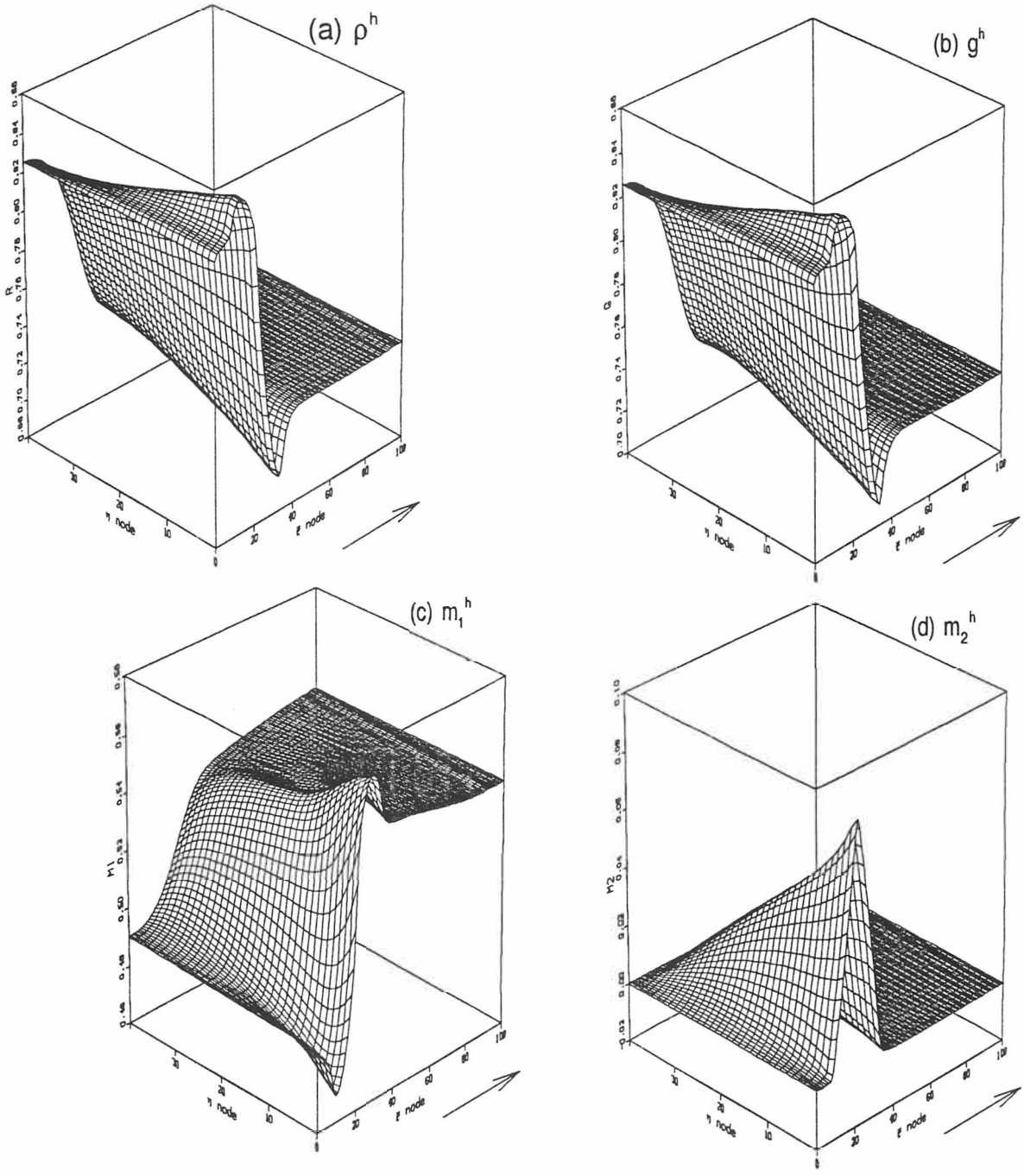


Figure 8.10: Inviscid Solution of the Subsonic Converging Duct Problem Plotted in Nodal Space as Viewed from the Bottom Entrance. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

c, the computational space carpet plot of m_1, in Figures 8.9 and 8.10. This wave is present because the solution was not driven to complete computational steady-state, i.e., the extremum residual $\{R(Q^{n+1})\}$ (Chapter 7) was $\approx 10^{-4}$ rather than say $\approx 10^{-6}$. This is not of consequence, since the viscous problem of real interest restarts from the inviscid solution and removes this wave as the extremum residual is driven to typically $\approx 10^{-6}$. Thus, it is not necessary to obtain a completely steady-state solution for each stage in the solution process in proceeding to the final turbulent flow solution.

Convergence of the quasi-Newton Procedure

An assessment of the quasi-Newton iteration linearization, i.e., a convergence rate test of the system iteration jacobian matrix, was also performed for the 2D inviscid problem. As before, the output variable measured is the base-10 logarithm ($\log_{10}$) of the maximum normalized iterate in the solution field $\left(\frac{\delta Q}{Q_o}\right)_{max}$.

The iterative convergence rate was verified using two forms of the quasi-Newton jacobian matrix. The first is the full jacobian with all off-diagonal submatrices included. The second is a reduced matrix with zero off-diagonal submatrices yielding a self-coupling approximation of the jacobian. This reduced form offers significantly reduced computational resource (CPU and memory) demands over the full jacobian, at a convergence rate penalty.

Representative convergence rates for various initial $\left(\frac{\delta Q}{Q_o}\right)_{max}$ values are listed in Table 8.6. For a full-Newton system (without matrix factorization), adherence to the theoretical convergence rate should show a doubling of $\text{abs}\left(\log_{10}\left(\left(\frac{\delta Q}{Q_o}\right)_{max}\right)\right)$ indicating quadratic convergence. The full-jacobian quasi-Newton data in Table 8.6 includes a range of initial normalized iterate residual

Table 8.6: Quasi-Newton Iteration Convergence Rate Summary for the 2D Inviscid Subsonic Converging Duct Problem

	Initial Normalized Change							
Iteration	.1%		1%		5%		10%	
	F	R	F	R	F	R	F	R
1	3.0	n/a	2.0	2.0	1.3	1.3	1.0	1.0
2	3.2	n/a	2.8	2.2	1.5	1.7	1.2	1.3
3	3.3	n/a	3.0	2.3	2.1	2.2	1.4	1.6
4	3.4	n/a	3.2	2.5	2.4	2.3	2.1	2.0
5	3.5	n/a	3.5	2.8	2.7	2.5	2.6	2.5
6	3.6	n/a	3.9	3.2	3.0	2.7	3.1	2.7

"F" denotes full jacobian and "R" denotes reduced jacobian

Quadratic convergence is indicated by
doubling of the tabulated values
between successive iterations.

changes from .1% to 10%. The reduced-jacobian quasi-Newton data includes the same entries with the exception of the .1% column. In the .1% column entry of the full jacobian, six iterations only reduces the residual from 3.0 to 3.6. Therefore, even with the full jacobian, for small initial normalized residuals below about .1%, reducing the residual at each time step is not cost effective. Initial residuals above 10% were not included in the table since, at that level or above, the automatic time step selection was typically above the imposed maximum input value. Therefore, a time step could not be selected to yield this large of a solution change and still be convergent; i.e., the convergence interval was exceeded.

The normalized change in iterate is above 1.0 throughout the entire range of Table 8.6, and often above 3.0. As expected, in all cases using the full jacobian, the residual decreased further than the reduced-jacobian residual over the full six iterations. The superior convergence rate of the full jacobian is small relative to the reduced form. In fact, the convergence rate of the first few iterations in the 5% and 10% columns is actually greater for the reduced jacobian! This suggests that using the reduced jacobian evaluations might be more cost effective for 2D inviscid problems during the early part of the approach to steady state. It was found that it becomes cost-effective to utilize the full jacobian to reduce the extremum residual from the 10^{-4} or 10^{-5} level to the final level of typically 10^{-6}. As will be documented later, this trend does not hold for the viscous problem since the full jacobian is always needed.

Convergence with Mesh Refinement

Previous research[24] has shown that linear theoretical expectations for mesh refinement convergence in the energy norm should hold for 2D simulations. However, these energy-norm convergence rate data were computed using a full-

dimensional Newton jacobian matrix, i.e. without a tensor-product factorization. Furthermore, these energy-norm data were computed on grids with relatively large mesh measures. In contrast, the energy-norm convergence rates for a quasi-Newton solution, i.e., a tensor-product factored system-iteration jacobian matrix, with relatively small mesh sizes were analyzed in this research. Compared to the full-Newton iteration with coarse grids, the energy-norm results were not meaningful. This is attributed to the tensor-product factorization error of order $(\Delta t)^2$, and to round-off error associated with data acquisition as the element measure is refined.

A more successful convergence rate analysis for this problem was performed using the H_0 norm, recall Chapter 7. Figure 8.11 summarizes the H_0 norm data for the steady solution in comparison to theoretical slopes of unity and two. The x axis is the $\log_{10}$ of the element measure, which is chosen as the maximum value of $detJ_e$ (proportional to the finite element area in 2D, volume in 3D) in the solution domain. The y axis is the H_0 norm of the respective variable analyzed, indicated by the legend on the $\log - \log$ plot. The results show a trend of matching a slope of unity generally, with a quadratic slope occurring only in a very narrow range of large element measure. The element measure range is well below that for the 1D analyses in the energy norm, thus confirming the hypothesis that at least round off error is immediately present in the 2D experiments as well.

8.1.5 Viscous Flow Verifications

The near-steady inviscid solution was used to initialize the viscous solution. The final turbulent solution was then obtained by computing three phases of problem definition. The first phase was computed by setting the non-slip wall Dirichlet data on the momentum variable nodes, and allowing for laminar viscous

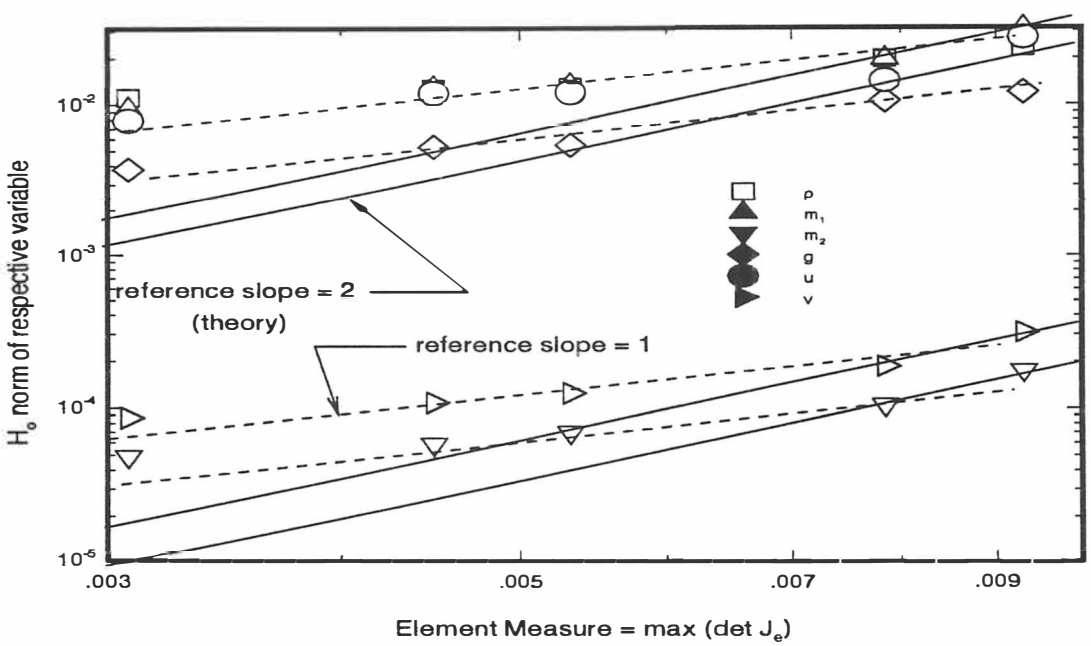


Figure 8.11: Convergence Rate of the Error in H_0 Norm for the 2D Inviscid Finite Element Solution of the Subsonic Converging Duct Problem.

dissipation to be imposed on the domain. The second phase restarts from the laminar viscous solution with algebraic turbulence model enabled to self compute turbulent eddy viscosity initiation to influence the flow field. Finally, a non-zero heat flux is imposed at the wall, via a convection coefficient, to examine the ability of the algorithm to simulate heat transfer at the wall.

One goal of the viscous verifications is to compare to flat-plate relationships of Blasius (laminar) and Coles (turbulent). Recall that the density and stagnation enthalpy profile are not known in the inflow boundary layer. Therefore, an inviscid inlet region is required for all problems in this dissertation. In addition, it is desired to compared boundary layer growth for this problem with known flat-plate (incompressible) data. Therefore, to emulate the flat plate, the no-slip wall Dirichlet boundary condition on the momentum variables is imposed downstream of the area change, i.e., at the first node of the straight section ($\xi = 42$) of the duct and thereafter along the wall. In this manner, the duct entrance and area-change regions ($\xi \leq 41$) do not influence the boundary layer and the solution will emulate a flat plate. This approach provides all the desired features for comparison except for some very important deficiencies: (1) compressibility effects are small but not perfectly incompressible like the relationships of Blasius and Coles, (2) the flow is internal and thus will produce a zero transverse velocity at the duct centerline unlike the boundary-layer correlations of comparison, and (3) a streamwise pressure gradient is present in this internal flow problem that is not present in the flat-plate problem. However, these deficiencies do not limit the analysis and the results of viscous boundary-layer growth in the algorithm is verified accurate.

Laminar Flow

Figure 8.12 shows the laminar viscous solution of the subsonic converging duct

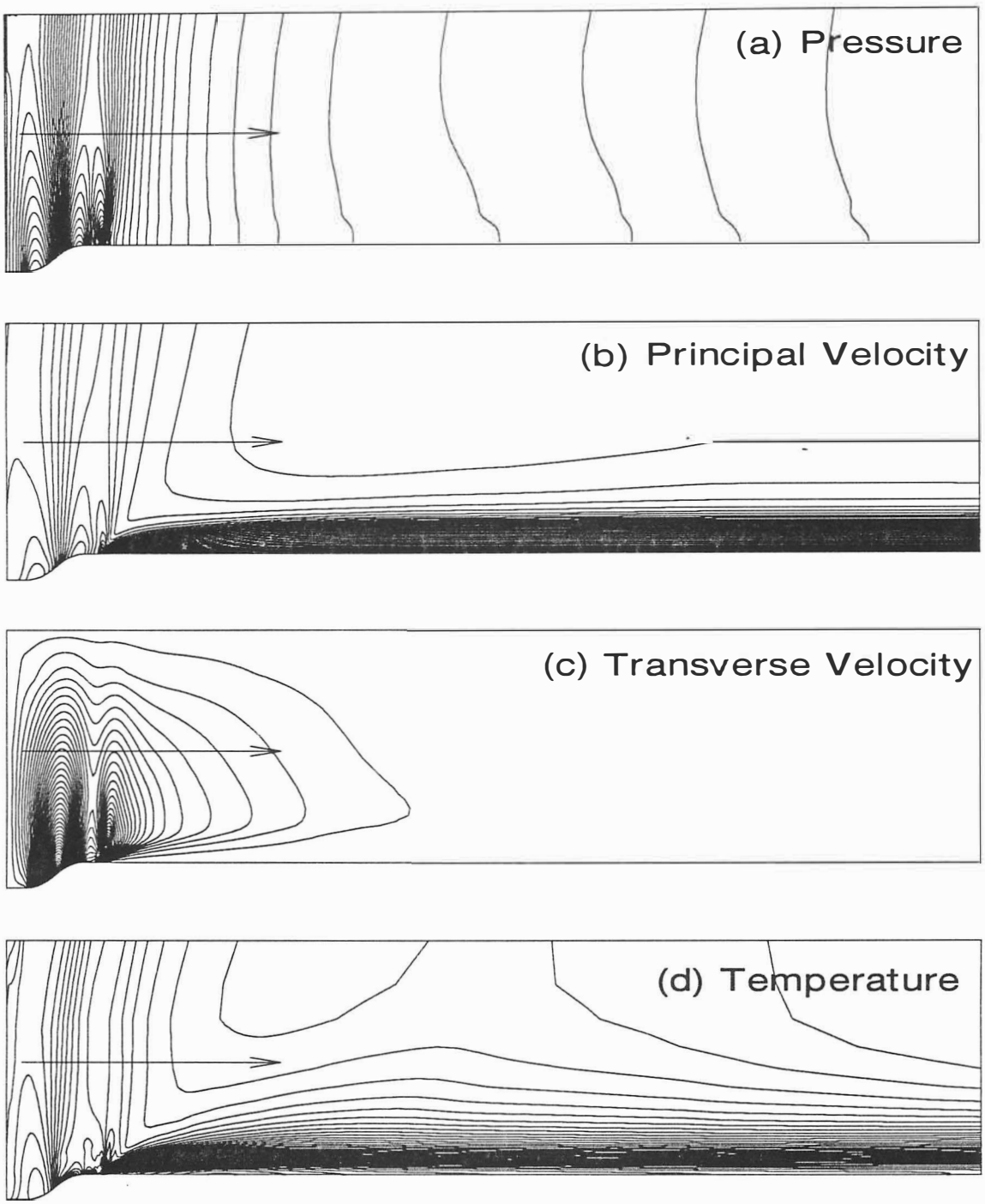


Figure 8.12: Laminar Viscous Solution of the Subsonic Converging Duct Problem Shown as Iso Contours in Physical Space; Free Stream Exit Mach Number 0.8, Reynolds Number 10^3; (a) Pressure, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature (vertical scale expanded for clarity).

problem as iso-contours of pressure p, principal velocity u, transverse velocity v, and temperature T. The vertical scale has been expanded by a factor of approximately ten for visual clarity. The problem statement is specifically designed to produce a thick boundary layer (approximately one third of the vertical span at the exit) via free stream exit Mach = 0.8 and Reynolds number = 10^3 based on arbitrary length scale. Unlike the inviscid problem, the state variable now extend throughout the length of the duct to the exit plane.

A non-zero axial pressure gradient is induced down the entire duct length by the viscous shear losses. A small but nonvanishing variation in pressure also exists across the boundary layer. The principal velocity appears to follow the classical shape expected for boundary layer growth, but does not become fully developed since the duct length is not sufficient. From Schlichting,[122] a channel flow velocity profile is parabolic (hence, fully developed) for Reynolds number 10^3 after approximately 80 feet of development. The computation confirms that rapid growth of the boundary layer thickness is complete by ten feet in duct length, as also confirmed by the small transverse velocity at the duct exit. Finally, because the wall is adiabatic, the temperature increases across the boundary layer.

The ENO quality of the solution is assessed by carpet plot data presentations, Figures 8.13 and 8.14. Shown viewed in nodal space from the top and bottom exits of the duct respectively, the combination of views allows for all solution surfaces of the solution to be seen. The top corner of the duct entrance ($\xi = 0, \eta = 56$) is the only part of the entire solution which exhibits a hint of oscillation in the solution. This inviscid corner has Dirichlet boundary conditions imposed on (ρ, g) and vanishing derivatives arising naturally from the TWS on (m_1, m_2). Experience has found that the corner inflow boundary condition is perhaps the least stable with respect to time step size of all the domain. It

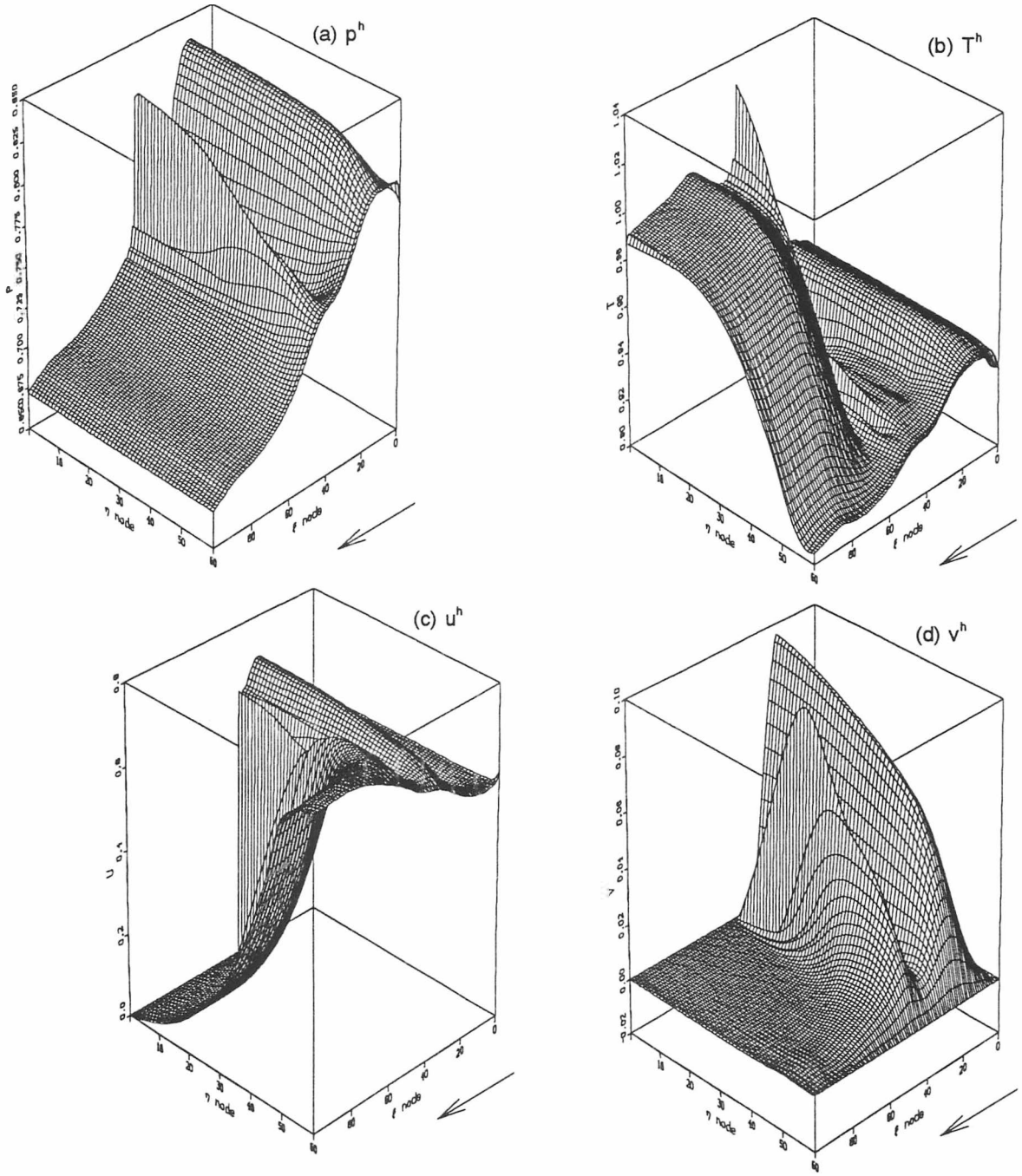


Figure 8.13: Laminar Viscous Solution of the Subsonic Converging Duct Problem Plotted in Nodal Space as Viewed from the Top Exit; Free Stream Exit Mach Number 0.8, Reynolds Number 10^3; (a) Pressure, (b) Temperature, (c) Principal Velocity, (d) Transverse Velocity.

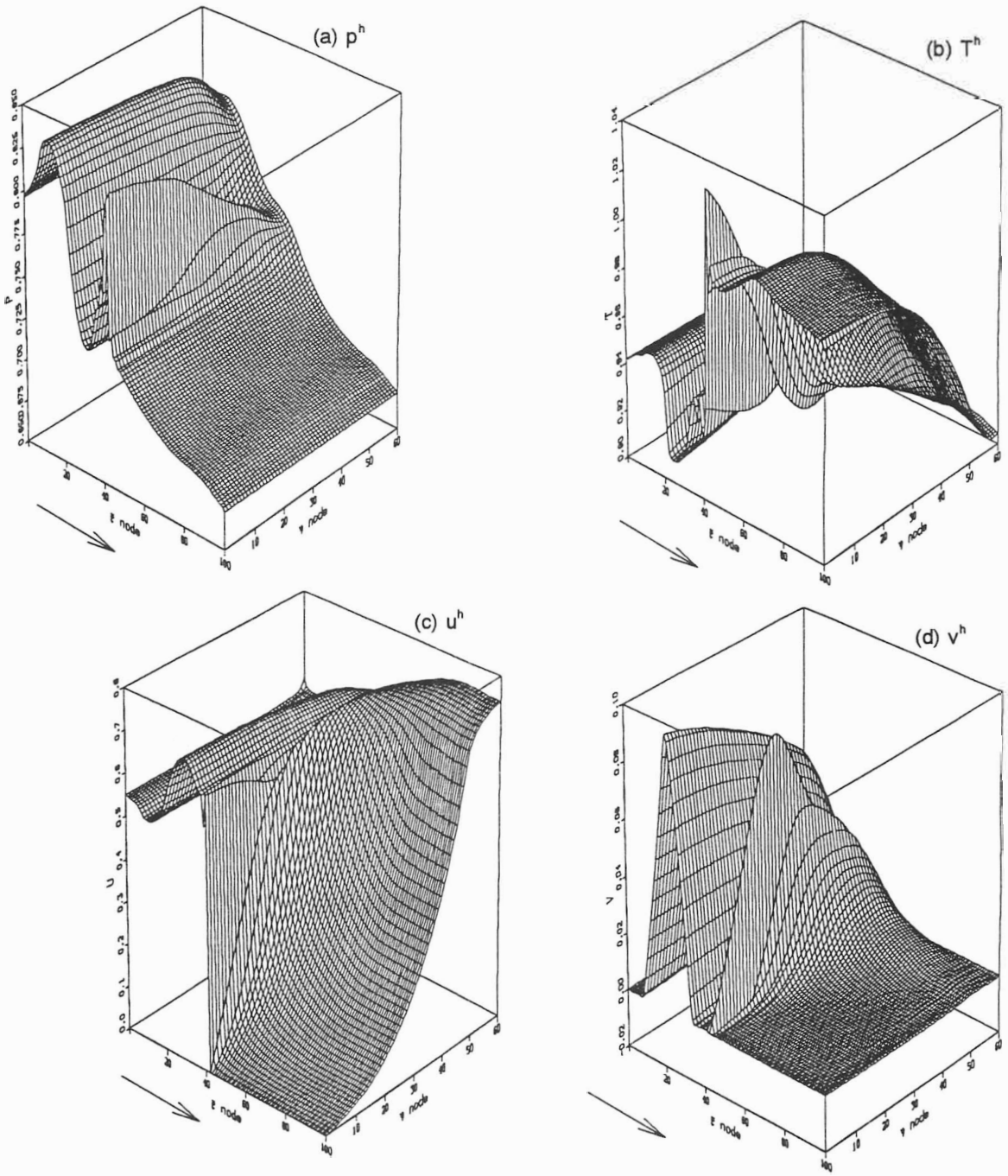


Figure 8.14: Laminar Viscous Solution of the Subsonic Converging Duct Problem Plotted in Nodal Space as Viewed from the Bottom Exit; Free Stream Exit Mach Number 0.8, Reynolds Number 10^3; (a) Pressure, (b) Temperature, (c) Principal Velocity, (d) Transverse Velocity.

is not important however in this case since the inviscid portion of the solution ($\xi \leq 40$) is essentially identical to the 2D inviscid solution otherwise.

The viscous portion of the solution starts at the straight section of the duct ($\xi \geq 42$). The transition from inviscid to viscous is very apparent at the wall, resembling a discontinuity in appearance. The temperature experiences a nominal overall change of only 10%, yet appears to have the greatest variation near the wall. This will always be the case for viscous solutions, since the only energy dissipation is that arising from the fluid viscosity, hence causing a temperature increase. This also explains why the temperature at the last exit plane node ($\xi = 96$) experiences a sudden drop in the viscous flow region where modulated $\beta = 0$. Even with a Reynolds number of 10^3, the fluid viscosity is not enough dissipation to enforce a smoothly varying temperature derivative at the outflow boundary. Yet in the inviscid region, where β has an influence on the state variable, the temperature exits the duct smoothly.

There are several other interesting aspects of the solution which can be observed by examining the nodal space carpet plots. The pressure satisfies both the equation of state and the Dirichlet constraint at the duct exit. Furthermore, the pressure experiences a non-zero gradient in the streamwise direction and nearly zero gradient in the wall-normal direction. Because the solution is steady, and the velocity streamwise gradients are nearly zero at the duct exit, as described by Schlichting,[122] the exit flow is in a transition between the flat-plate boundary layer model problem and fully developed flow in a duct or pipe. The temperature also experiences a slight streamwise gradient that is consistent with the problem definition boundary conditions. Because the transverse velocity is small at the duct exit, the continuity equation allows for only a small velocity streamwise gradient to exist. The transverse velocity experiences a double hump due to the area change in the inviscid region and the transition from inviscid to viscous

flow at node $\xi = 42$. Because the flow is internal, the transverse velocity is zero at the duct centerline.

An important development in this dissertation is establishment of well-posed mixed (Dirichlet and Neumann) boundary conditions, discussed in Chapter 6. The inlet Dirichlet constraint is on density and volume-specific stagnation enthalpy (ρ, g), while the momentum variables (m_1, m_2) have imposed a vanishing-derivative (Neumann) boundary condition. From the integration by parts of the convection terms, a non-zero momentum flux surface term evaluation is specifically required $(\boldsymbol{m} \cdot \hat{\boldsymbol{n}} \neq 0)$ on the inlet surface 1. The inlet momentum flux is not computed as Dirichlet data but rather an integral part of the iterative solution process, i.e., as a function of the solution. When the problem is solved at a non-zero angle with respect to the duct centerline (typically 45°), the momentum flux term is split between the momentum equations by following the direction-cosine relationship defined by the geometry and conserving the total momentum produced by the solution. No other boundary conditions exist on the inlet plane since this portion of the problem is totally inviscid.

The most interesting development regarding the boundary conditions occurs at the duct exit plane. The sole Dirichlet constraint is pressure variable held constant over the entire gridline at the duct exit plane as verified in Figures 8.13 and 8.14. The pressure experiences a non-zero axial gradient at the duct exit, which results from application of non-zero Neumann constraints on <u>all</u> variables with second order partial derivatives in the equation system, i.e., $(\hat{\boldsymbol{n}} \cdot \boldsymbol{\nabla})s \neq 0$, where $s = \{q, u_1, u_2, T\} = \{\rho, m_1, m_2, g, u_1, u_2, T\}$. Thus, except for the pressure Dirichlet data, all variables at the outflow boundary are influenced by internal nodes upstream of the flow exit due to the non-vanishing boundary condition (Chapter 6). This flow feature is most visible in the pressure surface plots (part (a)) of Figures 8.13 and 8.14, but is also present in the other variables shown.

The level of dissipation in the viscous flow region near the wall and exit is very small because of the relatively large Reynolds number (10^3) and distance between streamwise grid lines is relatively large (Figure 8.3). This explains why the last grid line of the temperature surface is shown with a non-smooth character near the wall. This situation is shown improved later in the turbulent results of the next section. Both velocity components show a smooth solution throughout with a slight non-zero gradient at the outflow plane.

No exact analytical solution to this problem exists.[122] However, because of problem similarity to subsonic incompressible flat plate flow, the principal velocity profile should be comparable to the analytical boundary layer solution of Blasius.[122] The main difference in the two solutions is no pressure gradient in the Blasius solution, and the transverse velocity remains non-zero in the free stream for the flat plate flow. The Blasius solution has two free parameters: (1) the free-stream principal velocity, and (2) the boundary layer displacement thickness. At any duct streamwise station, the free stream principal velocity was extracted from the calculated solution. The boundary layer displacement thickness was input as assumed existing at the duct exit. From the Blasius relationship between the displacement thickness and the streamwise direction ($\delta \propto \sqrt{x}$), and the known starting point of the viscous solution ($\xi = 42$), the required Blasius displacement thickness is interpolated at each gridline in the streamwise direction. Thus, from a single input value, the displacement thickness at the duct exit, the Blasius analytical solution may be compared to the calculated value of principal velocity.

Figure 8.15 shows the solution comparison to Blasius for two values of the duct exit plane displacement thickness. The top part of Figure 8.15 compares three streamwise velocity profiles extracted from wall-normal (η_2) gridlines with a displacement thickness to match the early part of the boundary layer growth

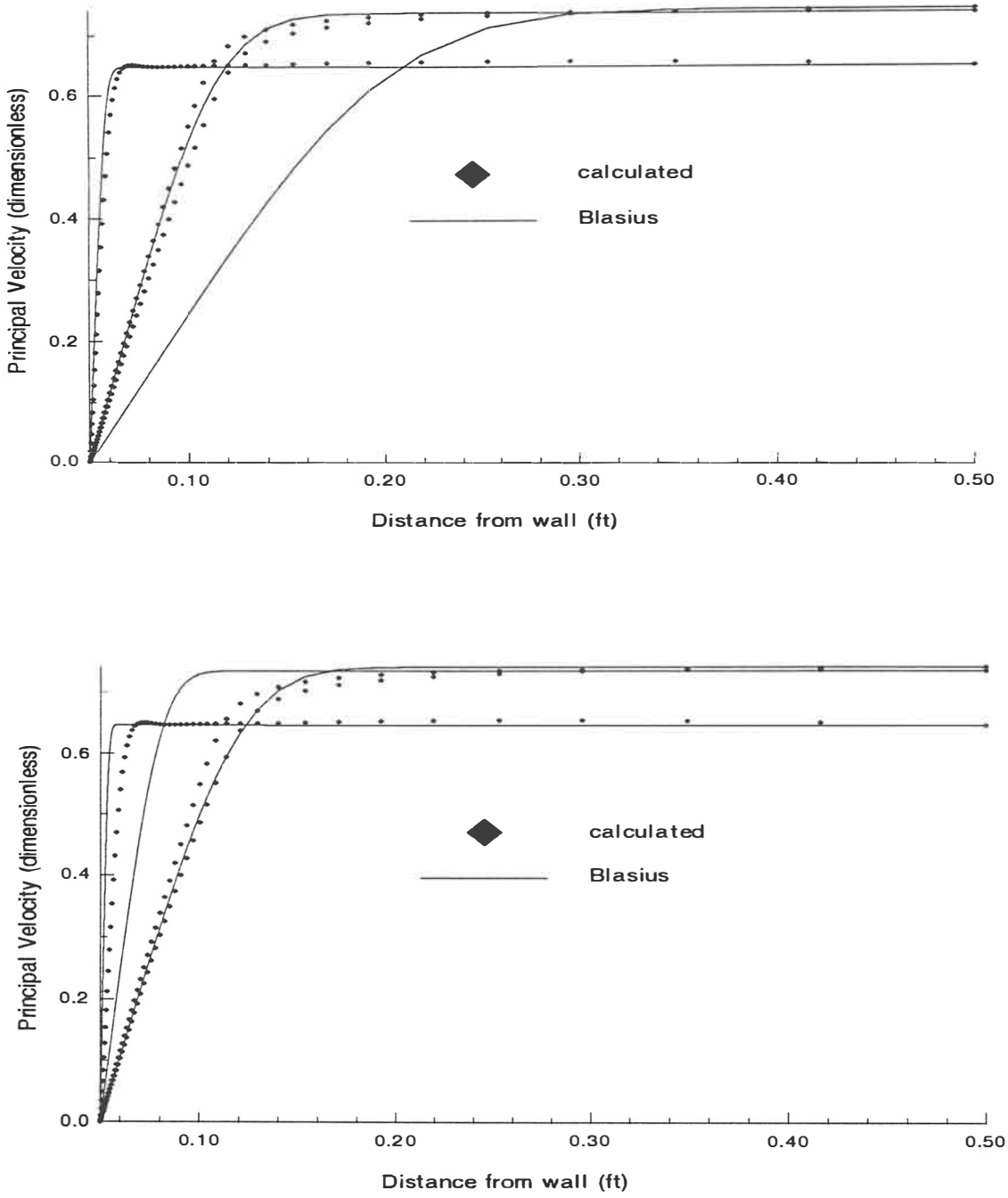


Figure 8.15: Boundary Layer Velocity Profile Comparison Between the Laminar Viscous Solution of the Subsonic Converging Duct Problem and the Flat Plate Incompressible Solution of Blasius; Free Stream Exit Mach Number 0.8, Reynolds Number 10^3; [δ_{exit} = .05 (top), δ_{exit} = .023 (bottom)].

(δ_{exit} = .05). The bottom part of Figure 8.15 shows a similar comparison with a displacement thickness to match the later part of the boundary layer growth (δ_{exit} = .023). The three streamwise grid lines selected correspond to the beginning (ξ = 42), middle (ξ = 69), and exit (ξ = 96) of the duct. Therefore, the Blasius model can be made to match the calculated value by the appropriate value of the displacement thickness. The displacement thickness required cannot be predicted since pressure gradients do not exist in the model. In contrast, the comparison displacement thickness is predicted by the calculation. However, since the shape and growth are consistent with Blasius, the Figure 8.15 comparison is sufficient to show that the laminar boundary layer profile is being calculated correctly. Most importantly, the boundary layer is being generated by the true physical viscosity and not by the artificial viscosity of the TWS β term.

An interesting detail in the velocity profiles of Figure 8.15, along the beginning wall-normal gridline (ξ = 42), is the slight overshoot between the no-slip wall (zero velocity) and free stream value. This overshoot is present in the simulation because of a peak in transverse velocity and coupled pressure gradient that is not accounted for in the Blasius model.

Convergence with Mesh Refinement

The 2D inviscid mesh refinement study confirmed the difficulty in obtaining accurate energy- and H_o-norm results with significant mesh refinement. For the viscous problem extension, the mesh is even more refined, hence the difficulties of meaningful norm calculations are compounded. The form-2 boundary layer equations presented in Chapter 3, were solved by a PNS (space marching) method early in this dissertation research[13, 124] and energy norms were computed from several solution variables. As expected from the theory discussed in Chapter 7, a slope of two and four resulted for the linear and quadratic basis functions

respectively. Shown in Figure 8.16, only a few points lie outside the theoretical lines. These can be attributed to round-off and step-size convergence errors, outside the narrow range of the element measure ($.01 \rightarrow .1$) mesh refinement.

Turbulent Flow

The two turbulent models selected for a simulation are: (1) Cebeci-Smith,[98] and (2) Baldwin-Lomax.[97] The Cebeci-Smith model predicts a larger turbulent length scale than does the Baldwin-Lomax model.[125] For this duct problem, the Baldwin-Lomax model yielded a turbulent length scale equal to approximately one third the vertical span of the duct, for a Reynolds number of 10^5 and matched reference data of Coles[123] reasonably well. Furthermore, because the Cebeci-Smith model requires a separate calculation to determine the displacement

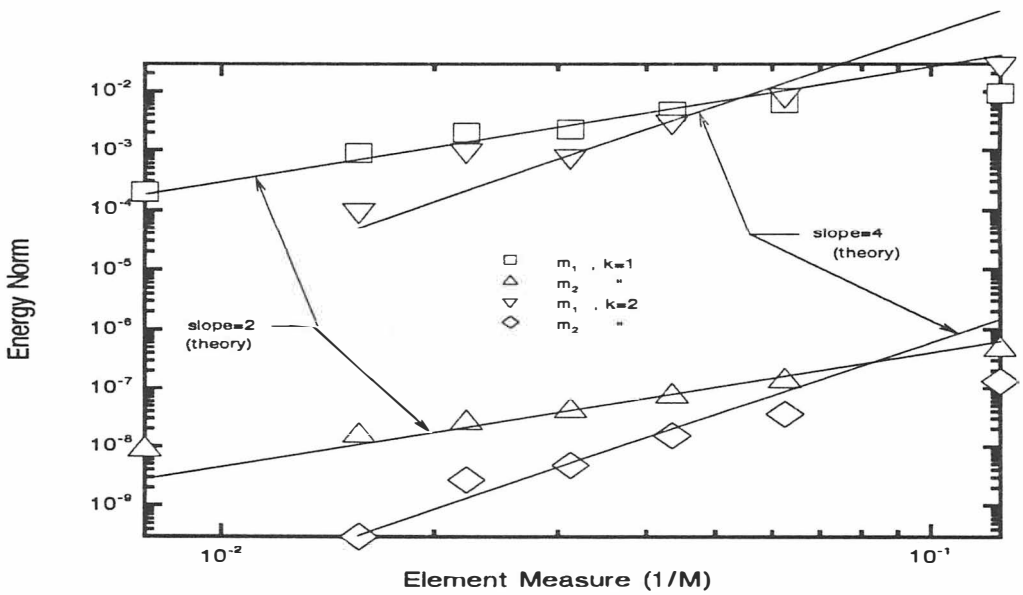


Figure 8.16: Convergence Rate of the Error in Energy Norm for the 1D Space Marching Finite Element Boundary Layer Solution of a Flat Plate Problem.

thickness of the boundary layer, then Baldwin-Lomax is more economical to evaluate. Therefore, the Baldwin-Lomax model is utilized for the turbulent solution verification herein.

The turbulent viscous solution field for the subsonic converging duct as iso-contours of pressure, principal velocity, transverse velocity, and temperature is shown by Figure 8.17. Two observations, comparing the turbulent results with the laminar, are (1) the turbulent boundary layer thickness at the exit plane is slightly greater, and (2) the transverse velocity variation extends the full length of the duct. Thus, because of increased viscous effects everywhere, the turbulent pressure contour shows a peak inside the boundary layer. The temperature shows small variations well into the free stream.

Examining the corresponding nodal-space solution surface graphics, Figures 8.18 and 8.19, reveals some additional differences between the laminar and turbulent solutions. The oscillation noted in the top entrance corner has been removed probably due to use of a reduced time step in the turbulent solution. The time step was reduced because of a decreased stability margin in the turbulent case. In part, the required smaller time step may have been because the solution is no longer smooth throughout the domain but experiences a longitudinal line-by-line fluctuation. This is traced directly to use of an algebraic turbulence model which calculates the eddy viscosity on an isolated gridline basis. One attempt at remedying this problem, i.e., smoothing the eddy viscosity between gridlines, involved computing nodal element-average quantities, as opposed to nodal data. This did not completely eliminate the computation oscillation between grid lines.

The problem of turbulence model oscillation between grid lines is better visualized by examining Figure 8.20. The top part of Figure 8.20 shows the eddy

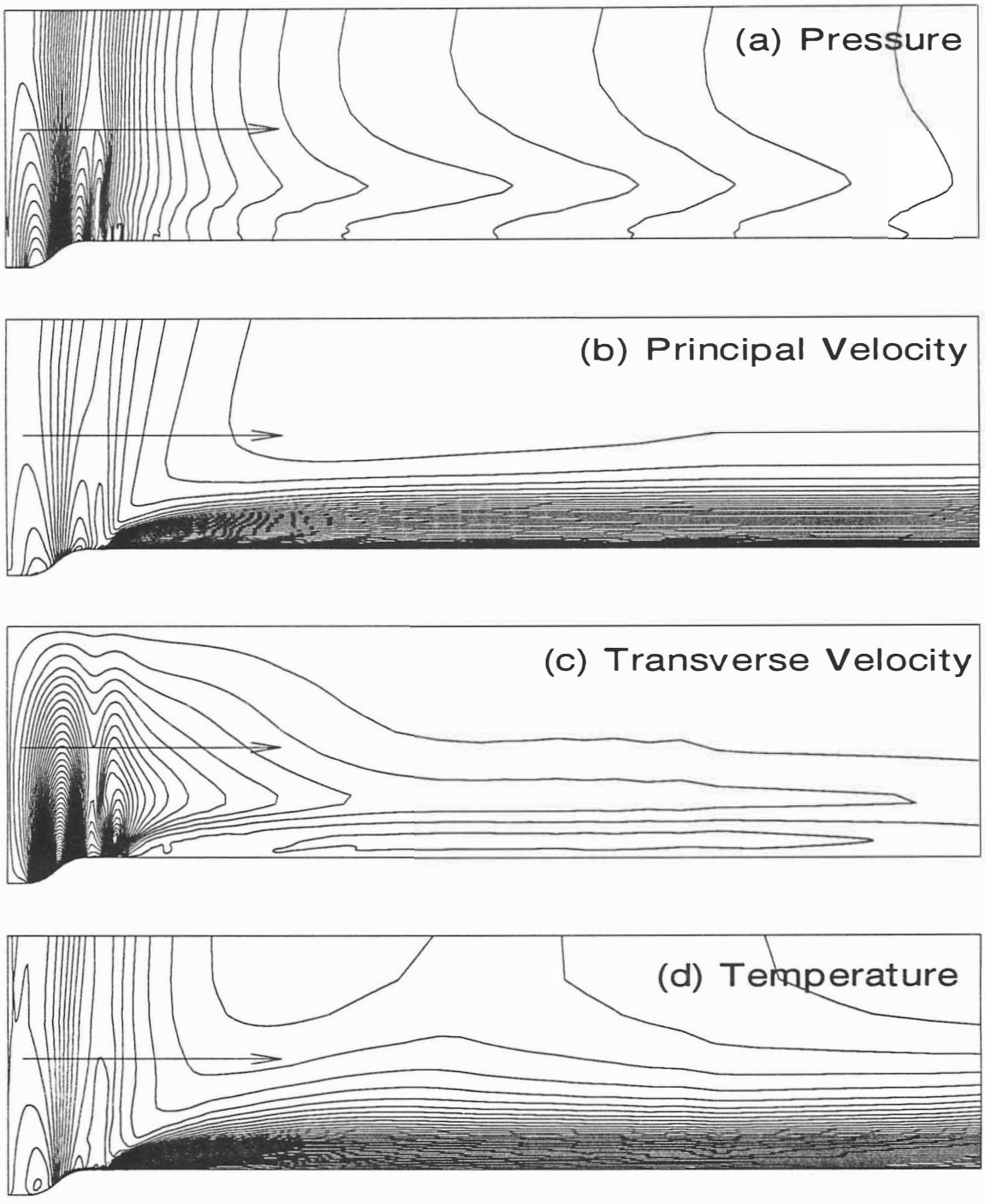


Figure 8.17: Turbulent Viscous Solution of the Subsonic Converging Duct Problem Shown as Iso Contours in Physical Space; Free Stream Exit Mach Number 0.8, Reynolds Number 10^5; (a) Pressure, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature (vertical scale expanded for clarity).

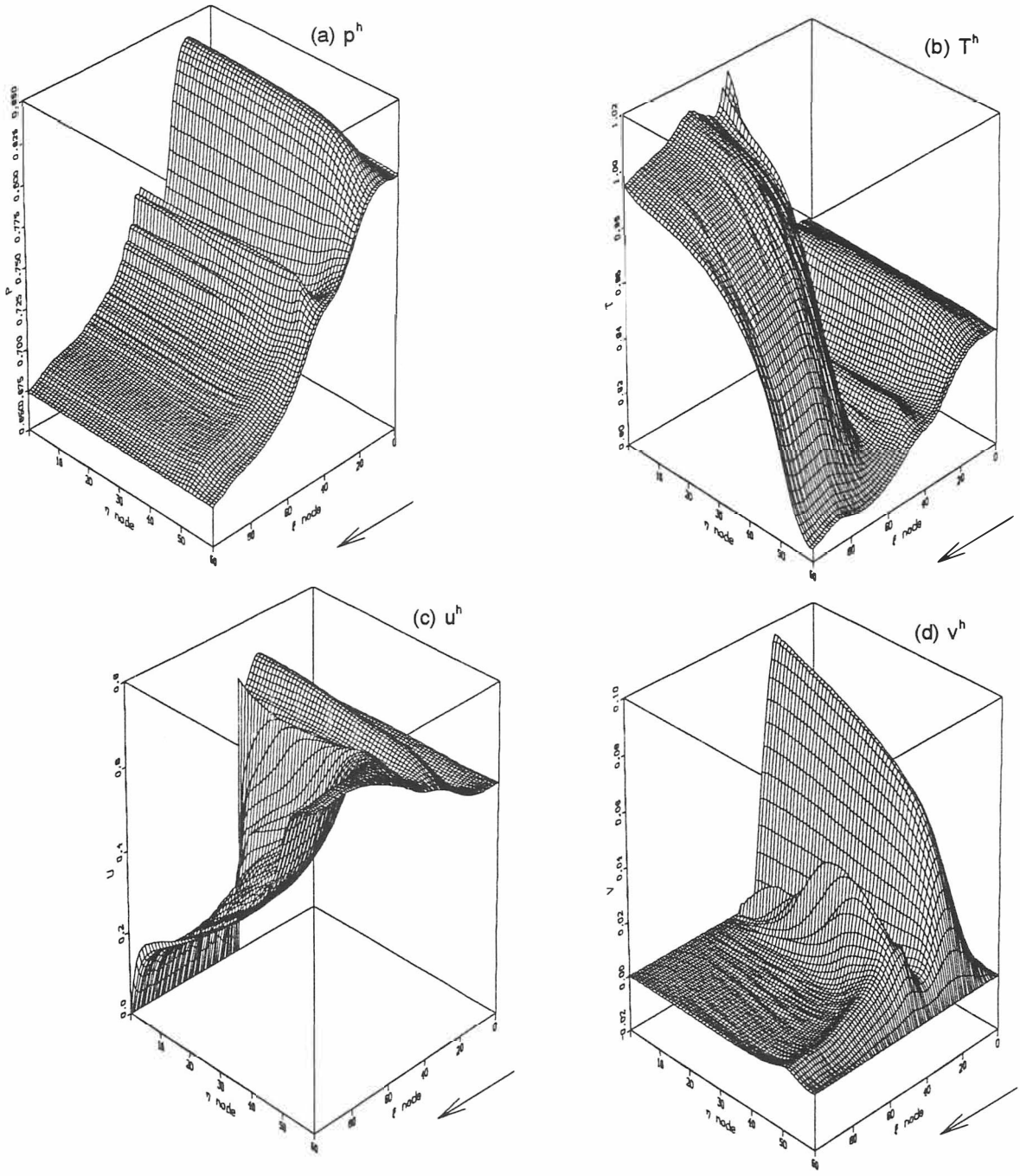


Figure 8.18: Turbulent Viscous Solution of the Subsonic Converging Duct Problem Plotted in Nodal Space as Viewed from the Top Exit; Free Stream Exit Mach Number 0.8, Reynolds Number 10^5; (a) Pressure, (b) Temperature, (c) Principal Velocity, (d) Transverse Velocity.

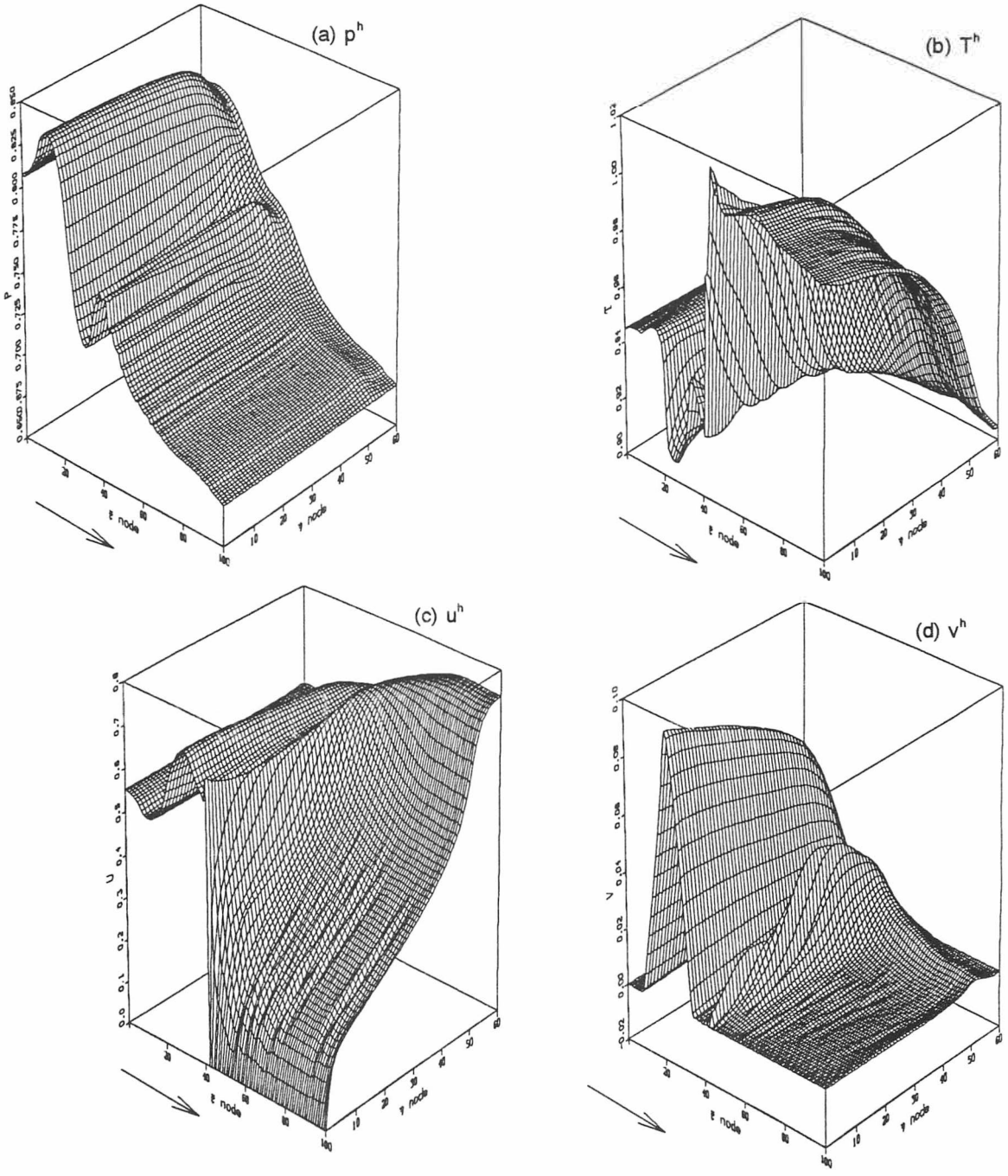


Figure 8.19: Turbulent Viscous Solution of the Subsonic Converging Duct Problem Plotted in Nodal Space as Viewed from the Bottom Exit; Free Stream Exit Mach Number 0.8, Reynolds Number 10^5; (a) Pressure, (b) Temperature, (c) Principal Velocity, (d) Transverse Velocity.

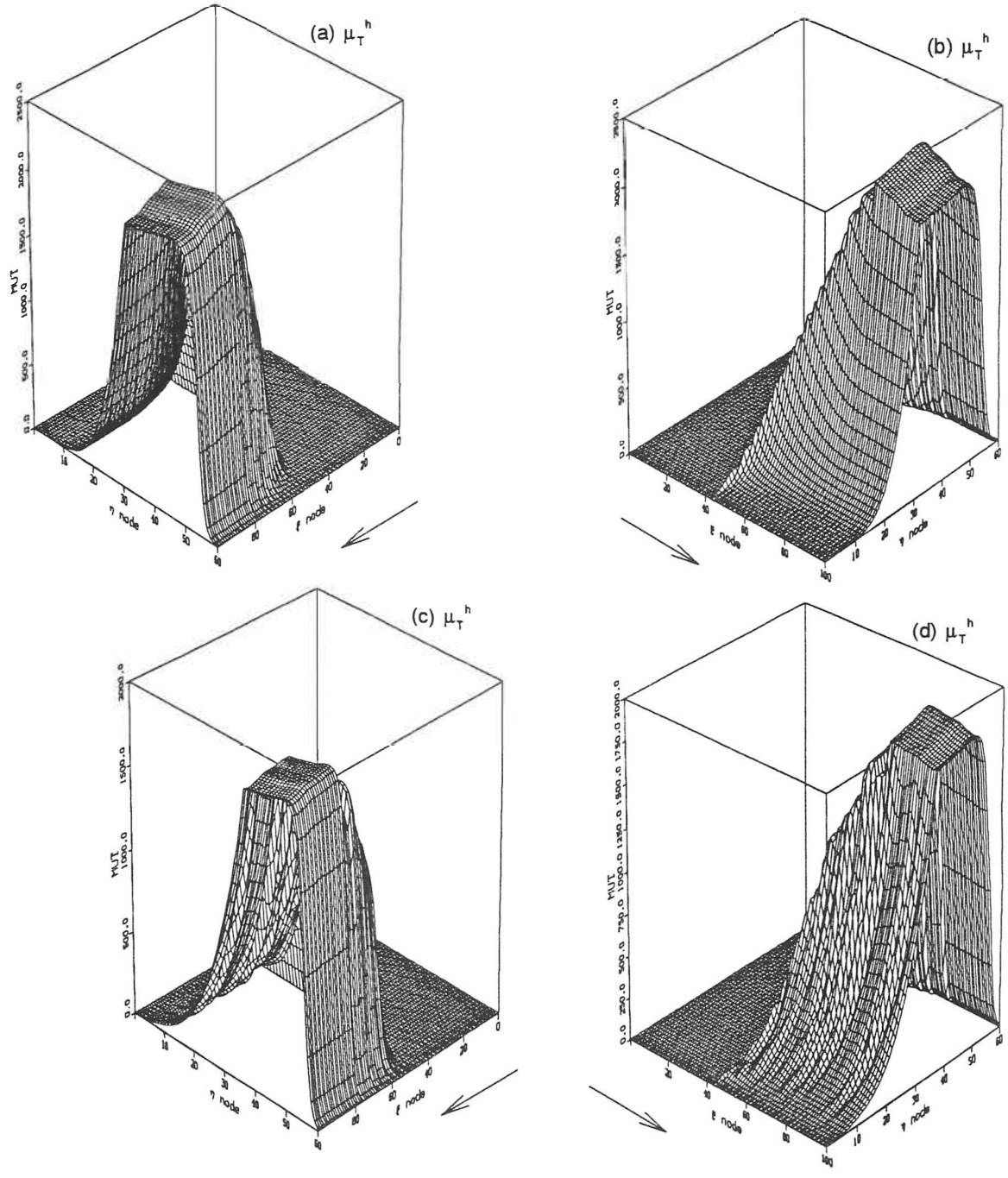


Figure 8.20: Turbulent Eddy Viscosity Produced by the Baldwin-Lomax Model for the Subsonic Converging Duct Problem Plotted in Nodal Space; Free Stream Exit Mach Number 0.8, Reynolds Number 10^5; (a) Initial Conditions from the Laminar Problem Viewed from Top Exit, (b) Initial Conditions from the Laminar Problem Viewed from Bottom Exit, (c) Final Conditions of the Turbulent Problem Viewed from Top Exit, (d) Final Conditions of the Turbulent Problem Viewed from Bottom Exit.

viscosity produced by the Baldwin-Lomax model, given the steady-state velocity profiles of the laminar viscous solution, i.e., the initial conditions for the turbulent problem. The bottom part of Figure 8.20 shows the eddy viscosity produced by the Baldwin-Lomax model at the steady-state achieved for the turbulent simulation. Because a distinct eddy viscosity is calculated for each wall-normal gridline, i.e., independently, the algebraic turbulence model receives no direct information from adjacent streamwise stations down the length of duct. The turbulent boundary layer dissipation is then non-smooth with respect to each station, and the resulting solution will likewise be lightly oscillatory between wall-normal gridlines.

Returning to Figures 8.18 and 8.19, the pressure and temperature surfaces are similar to the laminar case. However, there is obviously more dissipation present near the wall at the initial transition from inviscid to viscous flow, visually verified by the reduced peaks of all variables in that region. The velocity boundary layer profiles take on a different shape in the turbulent problem. In particular the principal velocity experiences the familiar two-slope turbulent profile. Because of the resulting turbulent pressure distribution and due to the continuity equation, the transverse velocity remains non-zero throughout the duct length.

As in the laminar case, the existence of well-posed mixed boundary conditions is also verified for this turbulent case. Figures 8.18 and 8.19 show a pressure surface plot (part (a)) similar to that of the laminar case (Figures 8.13 and 8.14). All variable gradients parallel to the outflow direction are steeper for the turbulent case. The temperature experiences a smooth outflow-direction gradient, perhaps due to the increased dissipation provided by the simulated turbulence. The principal velocity u_1 shows a slightly oscillatory feature near the wall and duct exit (part (c)), attributed to the nature of the turbulence model

computations as discussed. The boundary layer growth is further down the duct length in the turbulent case than for the laminar case also attributed to the eddy dissipation. This in turn causes the transverse velocity outflow non-zero gradient to appear more pronounced in the turbulent case (part (d) of the figures).

As in the laminar case, no analytical solution exist for this turbulent problem. However, because of the similarity to a subsonic incompressible flat plate flow, the principal velocity profile can be compared to the correlation of Coles.[123] The Coles model predicts turbulent principal velocity profiles given viscosity, density, normal distance from the wall, displacement thickness, and wall shear stress as computed from the velocity normal gradient. As with the Blasius solution, the Coles model does not account for any pressure gradients nor the interaction with the transverse velocity. The viscosity, density, and normal distance from the wall were obtained from the calculated solution. As in the laminar case, the displacement thickness was input to match the computational data. Given the free-stream principal velocity from the calculation, the wall shear stress required to satisfy Coles law curve fit, evaluated at the free stream, was determined. Thus, from the local displacement thickness, the Coles law solution may be compared to the calculated value of principal velocity.

Figure 8.21 shows comparisons to the Coles law model and the calculated principal velocity profiles for four axial locations, down the duct, corresponding to four values of duct exit plane displacement thickness. The computed velocity profiles match Coles model quite well with some deviations. In particular, the appearance of an overshoot on all calculated profiles is due to the transverse velocity returning to zero on the duct centerline. The slight oscillation in Coles law profile is due to the sinusoidal term in the curve fit being close to the end point of it's application. The calculated profile matches Coles law much better as the flow tends toward full development near the duct exit plane.

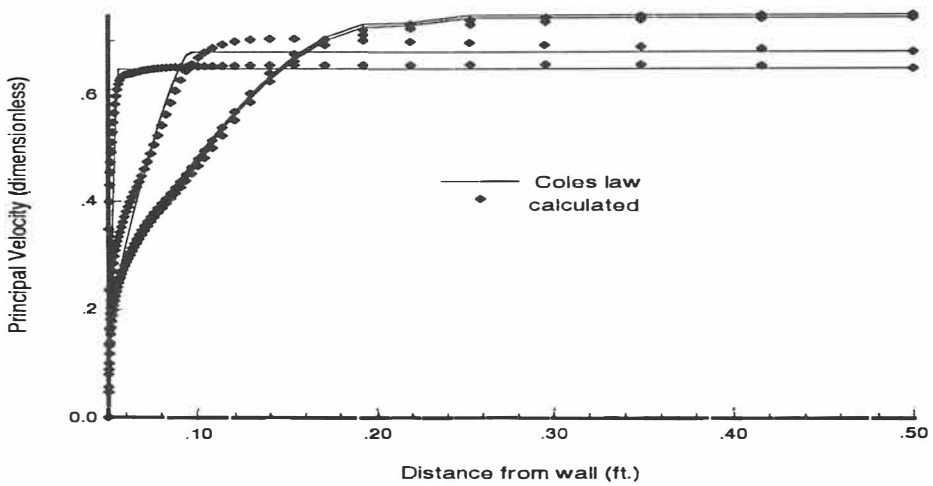


Figure 8.21: Boundary Layer Velocity Profile Comparison Between the Turbulent Viscous Solution of the Subsonic Converging Duct Problem and the Flat Plate Incompressible Solution of Coles; Free Stream Exit Mach Number 0.8, Reynolds Number 10^5.

This comparison is sufficient to verify that the turbulent boundary layer profile is indeed being calculated by the pilot code in a correct manner. Furthermore, the boundary layer is confirmed generated by the true laminar and turbulent eddy viscosity effects, not by artificial viscosity. This verified the deliberate elimination of the artificial viscosity in the viscous region, by the exponential filter of the TWS β term, Chapter 4.

Cold-Wall Results

A final case study verified the algorithm for simulating heat transfer to a wall. All cases previous to this assumed adiabatic heat transfer. For this special case, a cold wall temperature of $0^o F(460^o R)$ is assumed and perfect heat transfer yields a dimensionless cold wall temperature of 0.804.

The algorithm has been derived to incorporate wall heat transfer by allowing for a non-zero heat transfer coefficient (Chapter 6) to exist on any surface. The heat transfer coefficient is incorporated by the convection equation[126]

$$-k\frac{\partial T}{\partial \eta} = hA(T_{\text{wall}} - T), \tag{8.2}$$

where the convection coefficient h is typically an empirical function of the local fluid state. The steady-state, adiabatic, laminar viscous solution was restarted with $h = 1$ and $h = 10$, made nondimensional by $h_o = k_o/L_o \approx 10^{-5}$. It was not necessary to examine additional increases in h since $h = 10$ yielded a wall temperature very near the perfect heat transfer value, i.e., the fluid temperature approaches the wall temperature T_{wall} for increasing h.

Figure 8.22 shows the near-wall boundary layer temperature profile comparison for the three cases examined. The top, middle, and bottom sub figures show the temperature profiles for $h = 0, h = 1$, and $h = 10$ respectively. The profiles are obtained by extracting every 10th wall normal gridline starting at the

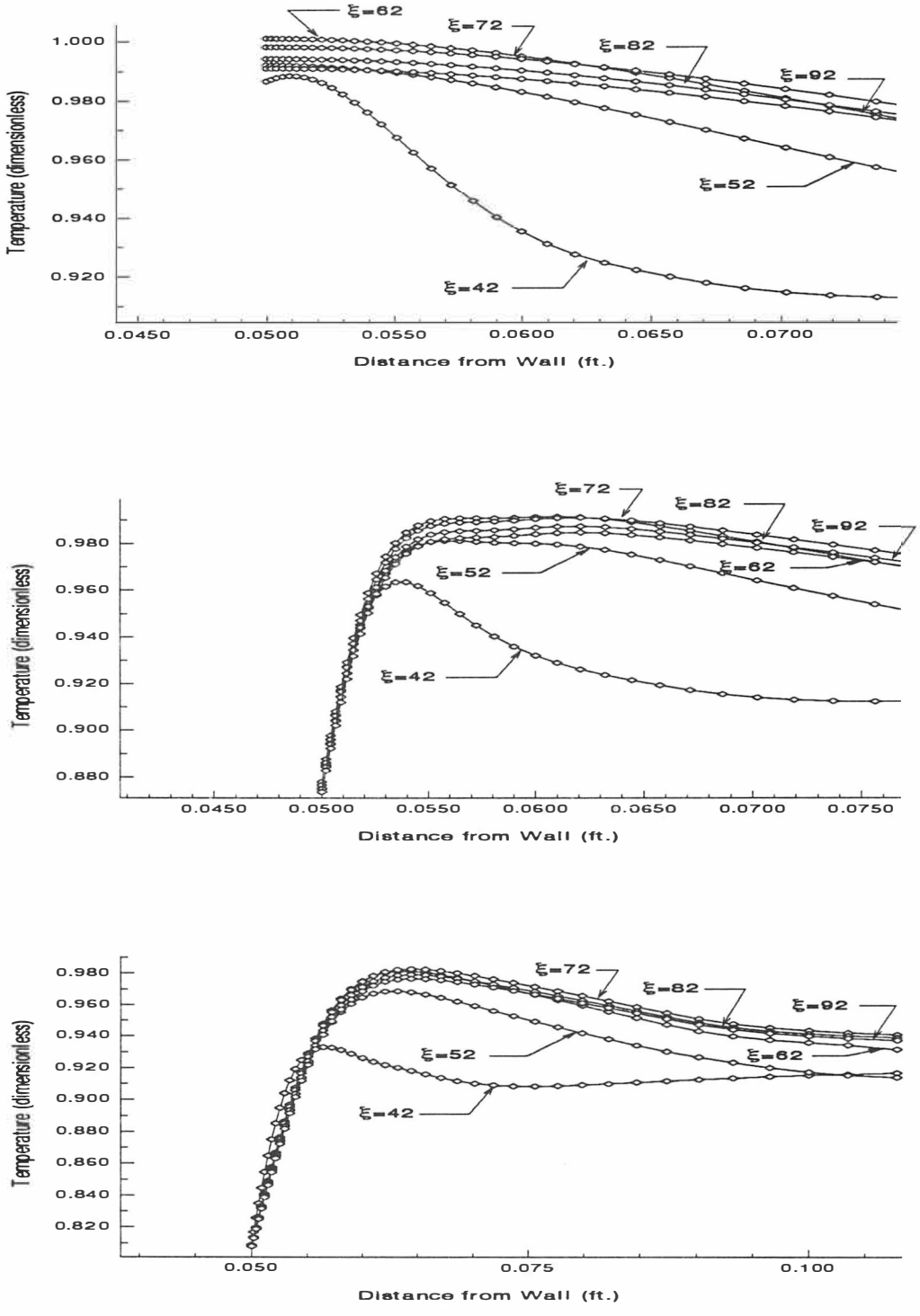


Figure 8.22: Near-Wall Boundary Layer Temperature Profile Comparison Between Various Heat Transfer Coefficients for the Laminar Viscous Solution of the Subsonic Converging Duct Problem. Free Stream Exit Mach Number 0.8, Reynolds Number 10^3; [h = 0.0 (top), h = 1.0 (middle), h = 10.0 (bottom)].

transition from inviscid to viscous at $\xi = 42$. The wall temperature drop is clearly present when comparing the three sub figures. Furthermore, the temperature boundary layer growth occurs quite early yielding almost identical wall-normal temperature gradients as the profiles are examined down the duct length. From Equation 8.2, the slope of the wall-temperature profile is directly proportional to the heat flux.

8.2 Transonic Wind Tunnel with Shock-Boundary Layer Interaction

The fundamental assessments of the developed finite element algorithm are established. Hence, algorithm application now moves to prediction of more realistic problem statements. One is a transonic wind tunnel, characterized by shock-turbulent-boundary layer interaction for which comparison experimental data are available.[127–130] Paraphrasing from these references, they describe an experimental program to examine "unsteady, (essentially) transonic flows in a 2D diffuser. The flows exhibit features similar to those found in supersonic inlets of air-breathing propulsion systems of aircraft and missiles."

The unsteady aspects of the experimental project were not pertinent to this project, since that was not an objective of the dissertation research (but not a limitation of the algorithm). Continuing, "Several configurations have been investigated and reported with exit-to-throat ratios ranging from 1.84 to 2.37. Regardless of area ratio, below shock Mach numbers of approximately 1.3, the flow displayed a pressure-gradient-induced separation, while at higher shock strengths, the separation was induced by the shock. The two types of flows are drastically different in terms of most measured steady and unsteady properties."

The CFD simulations presented and discussed herein also confirmed characteristics of two essential types of flow, which are labeled, as in the references, "weak- and strong-shock" cases respectively. The papers "provide a detailed description of experiments with a 1.52 area-ratio diffuser (wind tunnel) and a correspondingly mild streamwise rate of area change. The flow was fully attached up to a shock Mach number of 1.27. Beyond this Mach number, shock-induced separation occurred." These CFD results will also reveal and

quantize differences between the shock-boundary layer interaction characteristics of the weak- and strong-shock cases.

Two statements on the transonic wind tunnel problem are analyzed, with the labeled "weak-" and "strong-" shock cases corresponding to $M_s \approx 1.23$ and $M_s \approx 1.35$ respectively. The specific dissertation goal is to accurately simulate these flows, with the ultimate accuracy limitations being (only) the algebraic turbulence model. To achieve this goal, the ingredients of the CFD simulation must include resolution of:

- algorithm shock capturing via the TWS theory, without numerical diffusion pollution of the boundary-layer interaction,
- compare and analyze the numerical solution with the experimental data,
- assess the stability of the nonlinear iterative procedure on a more demanding problem,
- compare use of ideal- and real-gas simulation, and their impact on computational efficiency and accuracy,
- determine the impact of turbulent length scale upon the ability to match experimental data, and
- quantize limitations associated with the selected algebraic turbulence model.

The principal new validation is shock capturing via the artificial dissipation mechanism as developed by the TWS β term. The required CFD experiments are conducted on the quasi-1D simplification, as utilized previously, to determine a "proper" level of numerical dissipation. By "proper", the requirement is smoothness and monotonicity of the shock region, minimal spreading of the shock across the physical distance perpendicular to the shock curve (1-3 nodes),

sufficient mesh refinement so as to simulate the shock discontinuity, and consistency of the Rankine-Hugoniot condition of constant momentum across the shock.

Attendant computational well-posedness issues are:

- At the inflow plane, permit holding of vanishing derivatives without imposing additional Dirichlet data, by again defining the first few nodes of the inlet region mesh boundary to remain inviscid including the viscous problem. Dirichlet data for the inlet density ρ and volume-specific stagnation enthalpy g are set to determined appropriate values.
- The TWS parameter β is adjusted to achieve an accurate Rankine-Hugoniot approximation at the shock, and then held uniform, and
- Dirichlet data imposed on the outlet boundary remains solely on pressure, such that the normal shock Mach number matches the experimental data for both cases.

This validation procedure is iterative, since the value of β affects the shock Mach number. The results are slightly different from analytically-derived normal-shock tables.[118] Finally, the flow state experimental conditions will be slightly different than the CFD specification, due to 3D effects, and boundary layer bleed distributions along the side and bottom walls of the wind tunnel. Nevertheless, as will be seen, the CFD results are meaningful, provide very significant flowfield characterization (much more than is achievable via physical experiments in the wind tunnel), and are numerically accurate to within turbulence model limitations.

8.2.1 Problem Description

Figure 8.23 shows the cross-sectional geometry and certain related data for this

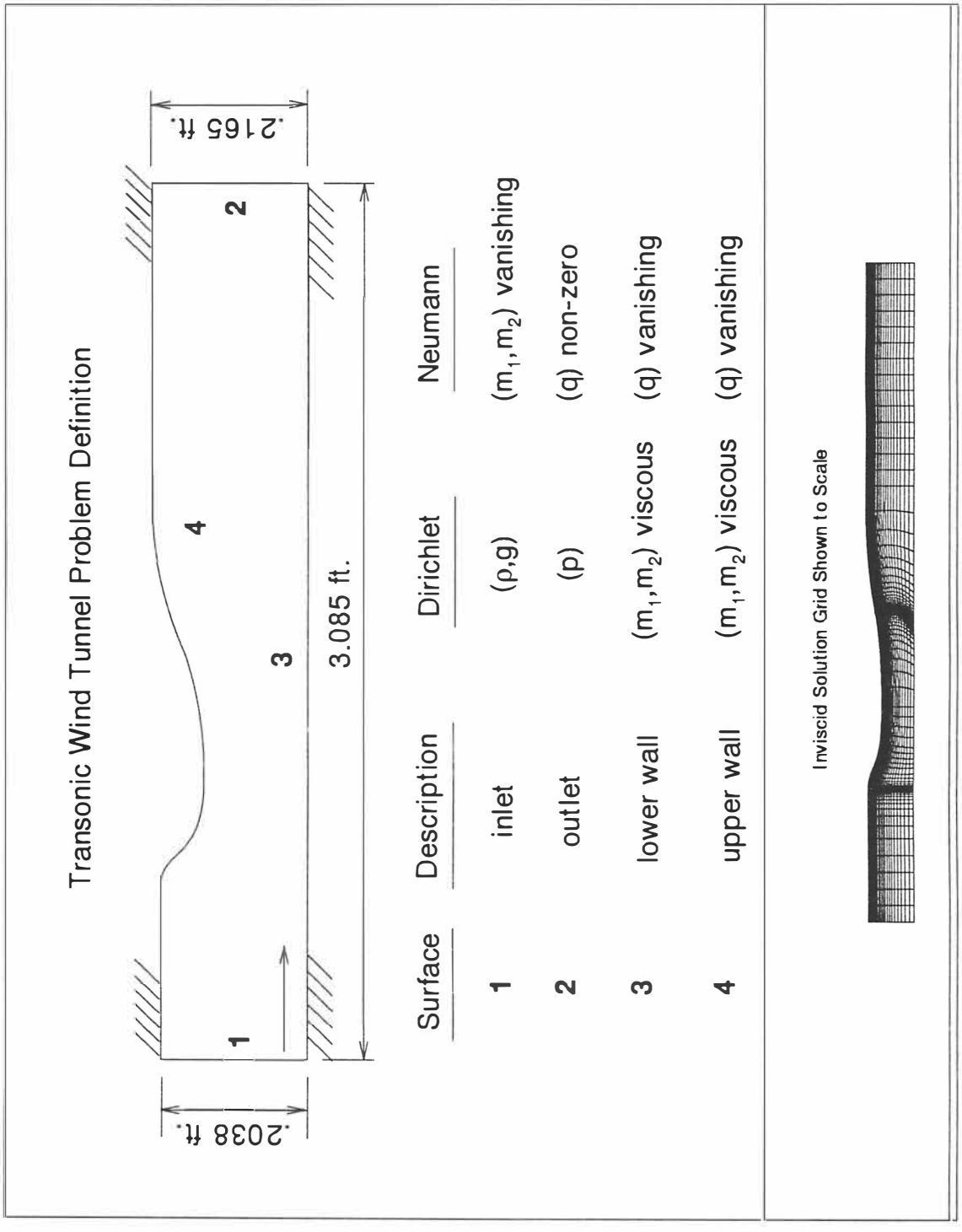


Figure 8.23: Transonic Wind Tunnel Problem Definition

problem. Flow enters from the left, exits to the right, and typical operating conditions are such that an essentially normal shock appears downstream of the throat in the diverging section. The real geometry is three dimensional, with walls on each side of the cross-section shown. During experimental operation, flow is extracted via boundary-layer bleed at three distinct locations along the side-walls, and along the bottom wall. The intent was to encourage the flow to remain laminar along the side walls and bottom. Then one set of experiments was designed to study the turbulence and flow separation that occurs along the curved wall (top) as a function of the strength of the normal shock. The tunnel wall geometry is certainly two dimensional, and the flow is predominantly unidirectional with designed-in minimal axial vortex component near the vertical centerplane of the wind tunnel.

The main data acquisition quantities of interest were pressure along the top wall of the nozzle, and the principal velocity distribution at various centerplane streamwise stations down the length of the nozzle. Figure 8.24 summarizes measured top-wall pressure distribution, normalized to the total inlet pressure, as a function of shock Mach number. The abscissa is referenced to zero at the nozzle throat, and is non-dimensionalized by the throat height. The wall pressure is flat at the inlet region, experiences a slight increase as the flow enters the nozzle; undergoes a significant decrease up through the minimum area (throat) to the shock location. (A modest plateau is present in the nozzle throat region.) Beyond the shock, the wall pressure distribution is highly dependent on the shock strength, i.e., as the shock strength increases, the minimum decreases and the post shock recovery is sharper. This is in agreement with classical 1D normal-shock theory. However, specific wall-pressure features such as the peak at the nozzle entrance, and the plateau at the throat are real geometry-induced phenomena, not predicted by the simple quasi-1D theory.

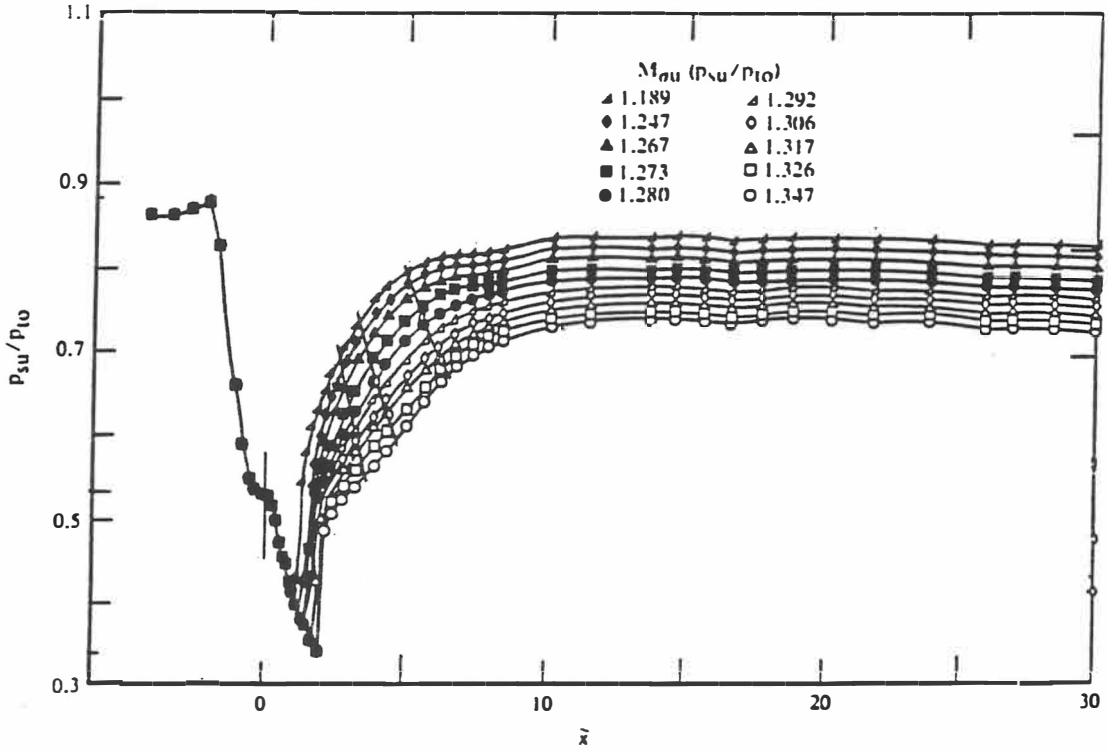


Figure 8.24: Measured Pressure Distribution Along the Top Wall for the Transonic Wind Tunnel Problem, from Bogar et al.[129]

Figure 8.25, from Salmon et al.,[127] shows measured axial velocity profiles at six streamwise stations for the strong-shock case. The stations start downstream of the throat at the approximate shock position ($\bar{x}$ = 2.88) and extend to just beyond the region of nozzle area variation, ($\bar{x}$ = 8.36). The right side of the graph is the bottom wall, while the left profiled ordinate is the top wall. Starting at the first station the profile sequence focusing on the top wall shows recirculation (indicated by negative velocity) caused by shock separation and/or pressure changes, subsequently recovery, and eventually a laminar profile formation in the region closest to the top wall. Flow bled from the bottom wall was designed to reduce turbulent effects. However, a turbulent character is obviously present in the velocity profiles, in the double slopes present. The

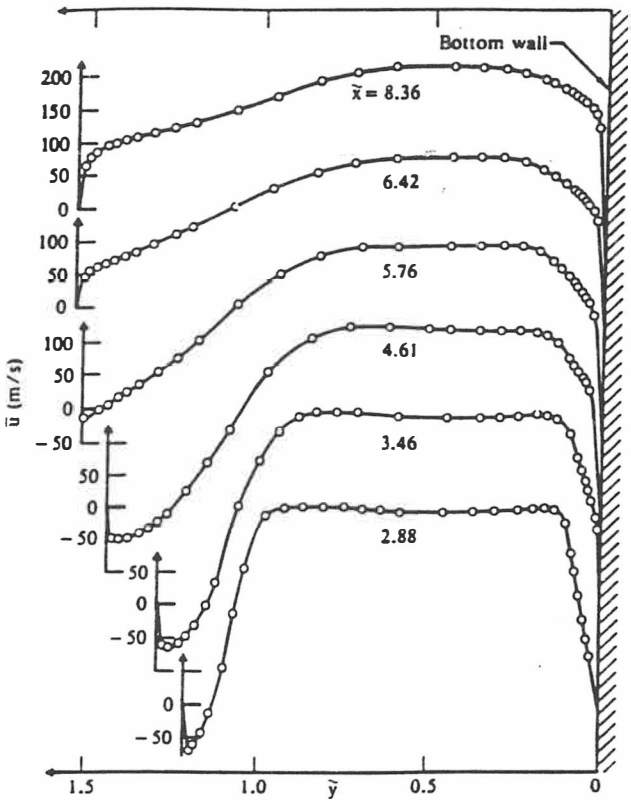


Figure 8.25: Measured Axial Velocity Profiles Downstream of the Shock for the Strong-Shock Transonic Wind Tunnel Problem, from Salmon et al.[128]

boundary layer is thicker than one might expect for a unit Reynolds number in excess of 10^5, which implies that the turbulent length scale is much smaller than the reference throat height. On the top wall, existence of turbulent flow is much more dramatic, with the boundary layer thickness eventually becoming greater than half of the total nozzle cross section.

The results of the present CFD experiments, for the turbulent strong-shock case, are summarized by color iso contours of key variables to assist in data total assimilation. Figure 8.26 shows, from top to bottom, the final steady-state solution for the inviscid, laminar-viscous, and turbulent-viscous flow simulation predictions for pressure. Throughout, red indicates the maximum and blue the minimum level. The pressure in the viscous cases are shown red at the flow inlet to orange at the flow outlet, indicating an overall pressure loss. The iso bars are nominally vertical, independent of closure model, except for curvature present at the blue shock region and essentially normal to the upper wall. The shock position moves upstream towards the throat, as additional pressure loss occurs in the transition to laminar and then turbulent flow.

Figure 8.27 summarizes the flow distinctions in terms of transverse momentum iso contours. Zero transverse velocity is green, minimum values are blue, and maximum values are red. The approach to the throat is clearly visible by the blue region. The viscous cases show a green region between the wall and the negative transverse velocity (blue) indicating the boundary layer. The shock is clearly visible by the yellow, orange, and red regions. The transverse momentum character, hence shock profile, is significantly altered by the viscous and turbulence model effects. Rather thick boundary layers are visible adjacent to the shock regions at the upper and lower walls. A slightly negative transverse momentum is visible just upstream of the shock in the laminar case.

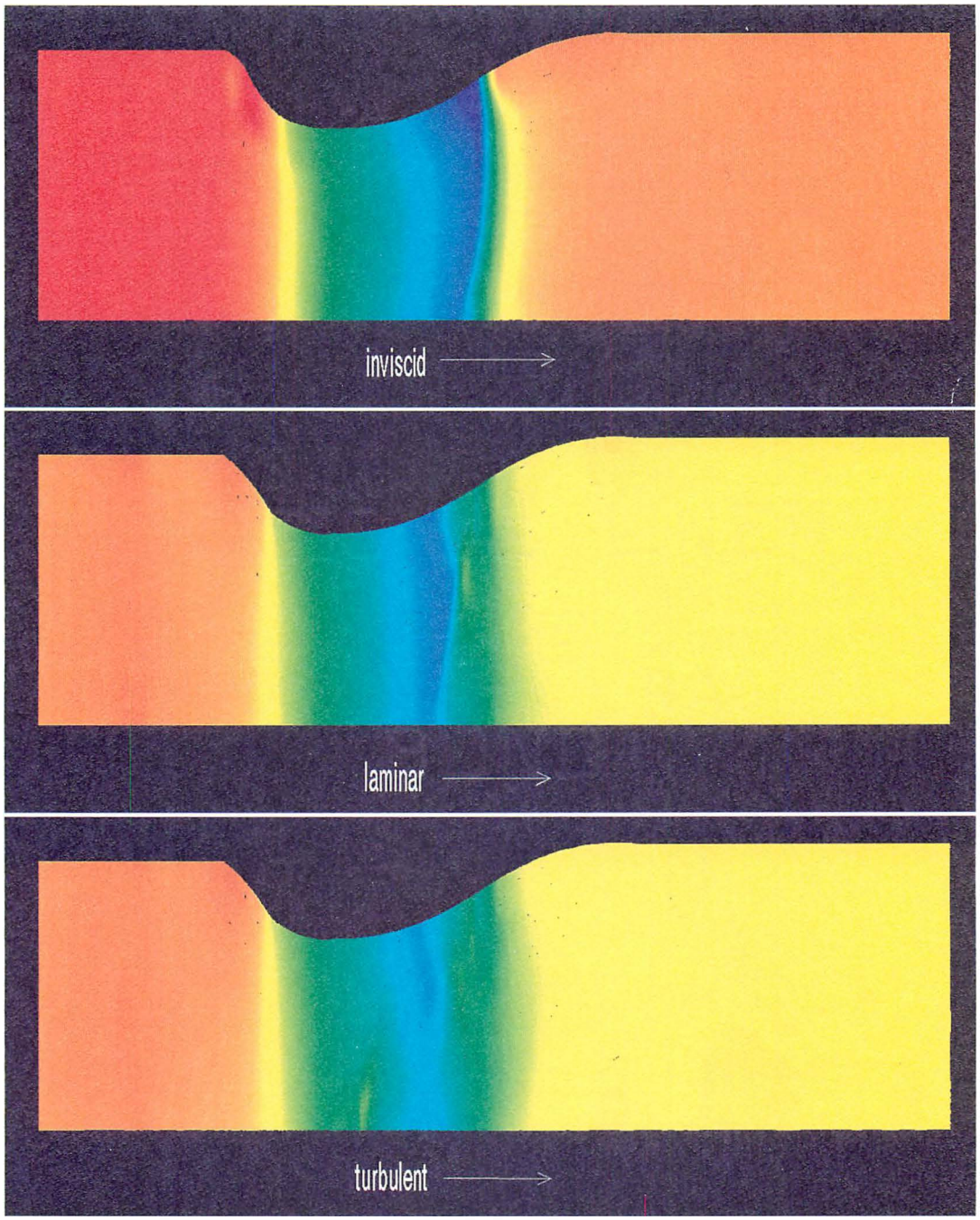


Figure 8.26: Calculated Pressure for the Inviscid, Laminar-Viscous, and Turbulent-Viscous Solution Phases of the Strong-Shock Transonic Wind Tunnel Problem (vertical scale expanded by ≈ 2 for clarity).

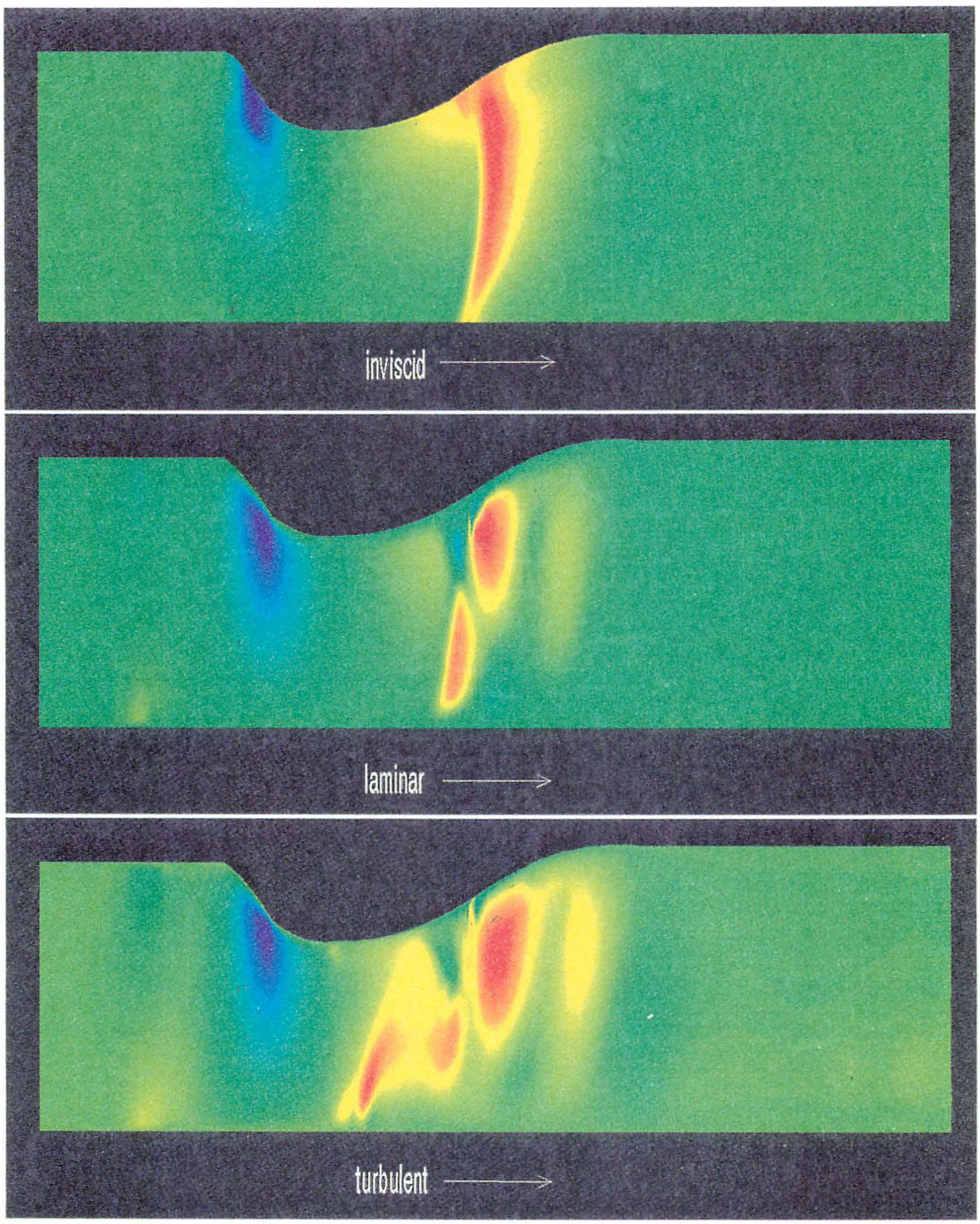


Figure 8.27: Calculated Transverse Momentum for the Inviscid, Laminar-Viscous, and Turbulent-Viscous Solution Phases of the Strong-Shock Transonic Wind Tunnel Problem (vertical scale expanded by ≈ 2 for clarity).

As the final summary, the corresponding set of Mach number contours, Figure 8.28, dramatically detail the viscous and turbulence model effects. The color base level changes from inviscid to viscous cases, since the minimum Mach number for of the inviscid case is greater than zero. The shock strength, hence shock resolution, is evidenced in the inviscid case by the much smaller decrease from maximum (red) to minimum (blue) downstream of the shock, as compared to upstream. The viscous flow region initiation is evidenced by the blue boundary layer region, along the upper and lower walls, which continues downstream to the exit plane. The boundary layer thickness increases dramatically in the shock interaction region for the turbulent case, which is unlike the experimental data. The shock is increasingly diffused going from inviscid to laminar to turbulent flow simulation, as evidenced by the mixing of color contour levels.

The boundary layer extraction flow from the wind tunnel was neither modeled nor accounted for in this 2D simulation. In the experiments, approximately 21% of the total flow was extracted along the side and bottom walls. Therefore, there is a difference between the simulated and experimental results in some variables. Since the available data is limited, the simulation problem initialization attempts to duplicate the exit Mach number and pressure distribution. Also, the calculated streamwise velocity profiles are compared to available data at the vertical centerplane of the wind tunnel.

Isentropic flow parameters associated with this transonic problem were compared to known experimental data. Differences between isentropic conditions and the experimental data give some indication of the level of inaccuracy, in the quasi-1D and 2D assumptions. Table 8.7 lists the nominal data used to initialize the quasi-1D solution. Items denoted with the asterisk are from the experimental data, while the remaining data are calculated utilizing isentropic theory and normal-shock tables. Detailed flow distributions, including at the

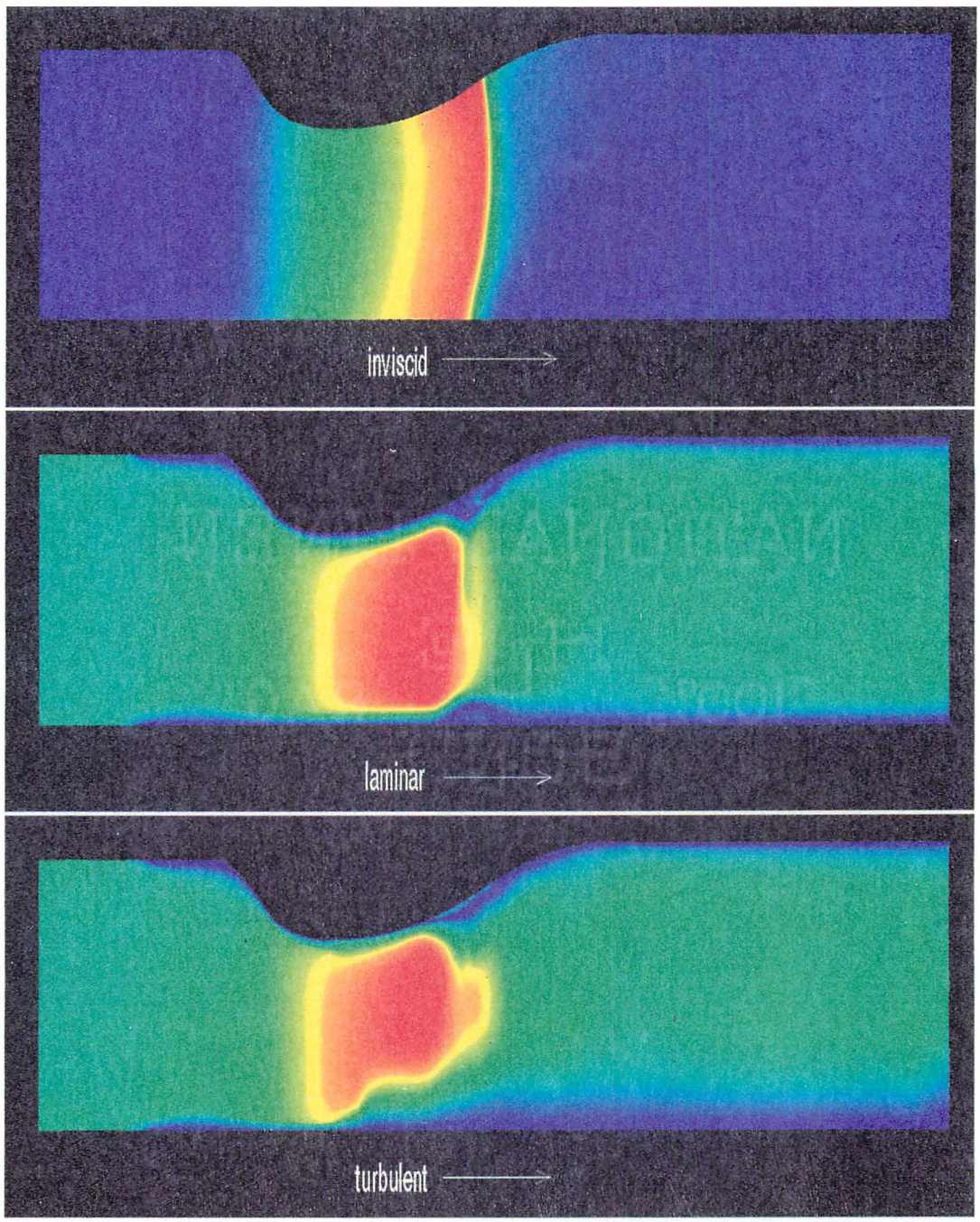


Figure 8.28: Calculated Mach Number for the Inviscid, Laminar-Viscous, and Turbulent-Viscous Solution Phases of the Strong-Shock Transonic Wind Tunnel Problem (vertical scale expanded by ≈ 2 for clarity).

Table 8.7: Transonic Wind Tunnel Problem Isentropic Initial and Boundary Conditions for the Weak-Shock Case

Quantity	Inlet	Outlet	Stagnation (ideal gas)	Stagnation (real gas)
T (^{o}R)	360.0*	357.05	375.63	375.55
A (ft^2)	.2037*	.2165*	n/a	n/a
ρ (lbm/ft^3)	.0381	.0365	.0423	.0491
u (ft/sec)	443.5	472.6	n/a	n/a
a (ft/sec)	930.5	926.67	950.5	946.1
p (lbf/ft^2)	731.0*	695.5*	848.4	983.2
M (none)	.4766*	.51*	n/a	n/a

* known or assumed quantities

inlet or outlet, were not given in the references. However, the values tabulated yielded reasonably close estimates to several quantities that were given. For example, the inlet air supply was stated at a pressure "less than 35 kPa." Therefore, the inlet pressure was set equal to 35 kPa $\approx$ 5.1 psia = 731 psfa. The inlet temperature was not given, but could be derived from the Reynolds number stated as being $6.4 \rightarrow 8.4 \times 10^5$ based on a wind tunnel throat length of .1444 ft.

The exit Mach number and pressure were given as approximately .51 and 4.83 psia respectively (actually the exit pressure was given as $p_2/p_{T_1} = .826$ where p_{T_1} is the total inlet pressure $= p_1 + u_1^2/2$). Because of the boundary layer bleed along the bottom and side walls, the flow rate implied by Table 8.7 will not conserve mass nor yield the shock Mach number given by the experiment at the normal shock. Therefore, duplication of the exit conditions is not exact (perhaps this is why the details of the complete fluid conditions were not available). The inlet fluid properties and peak Mach number at the shock were made consistent with experimental data by varying the exit pressure.

8.2.2 Discretization Specifications

The equation for the curve describing the tunnel top wall trajectory is[128]

$$h(\bar{x}) = \frac{\alpha \cosh(\zeta)}{(\alpha - 1) + \cosh(\zeta)}, \tag{8.3a}$$

where

$$\zeta = \frac{C_1 \left(\frac{\bar{x}}{l}\right) \left[1 + C_2 \left(\frac{\bar{x}}{l}\right)\right]^{C_3}}{\left[1 - \left(\frac{\bar{x}}{l}\right)\right]^{C_4}}, \tag{8.3b}$$

for $-2.598 < \bar{x} < 7.216$. The nondimensional wind tunnel length is defined as $\bar{x} = x/A_t$ where A_t is the throat area (for unit width) and x is the horizontal wind tunnel coordinate equal to zero at the throat. The nondimensional coordinate

$\bar{x}$ extends from -6.925 ($x = -1.0$ feet) to $+14.432$ ($x = +2.085$ feet). For $\bar{x} \leq -2.598$, then $h(\bar{x}) = h(\bar{x} = -2.598)$. Similarly, for $\bar{x} \geq 7.216$, then $h(\bar{x}) = h(\bar{x} = 7.216)$. Table 8.8 shows the various constants in Equations 8.3a and 8.3b defined for $\bar{x}$ occurring in the converging and diverging side of the throat.

Figure 8.29 shows three grid specifications for the strong-shock case for three distinct solution phases. The grid refinement increases as the flow detail requiring resolution, from quasi-1D to viscous-turbulent, increases. It is not apparent until closer examination that sufficient grid refinement is not present in the quasi-1D case near the region of the throat entrance, hence the 2D inviscid grid is refined for this region. Figure 8.30 gives a magnified view, showing that the smooth transition between curved and straight surfaces appears captured. However, even with 25 nonuniformly arranged finite elements bundled around this region, further refinement could be used based on the solutions obtained.

Also shown in the 2D inviscid grid of Figure 8.29 is a refinement in the region about the shock to provide required resolution. A less detailed grid will

Table 8.8: Constants Used to Define the Geometry of the Transonic Wind Tunnel Problem

	converging ($\bar{x} < 0$)	diverging ($\bar{x} > 0$)
α	1.4114	1.5
l	-2.598	7.216
C_1	0.81	2.25
C_2	1.0	0.0
C_3	0.5	n/a
C_4	0.6	0.6

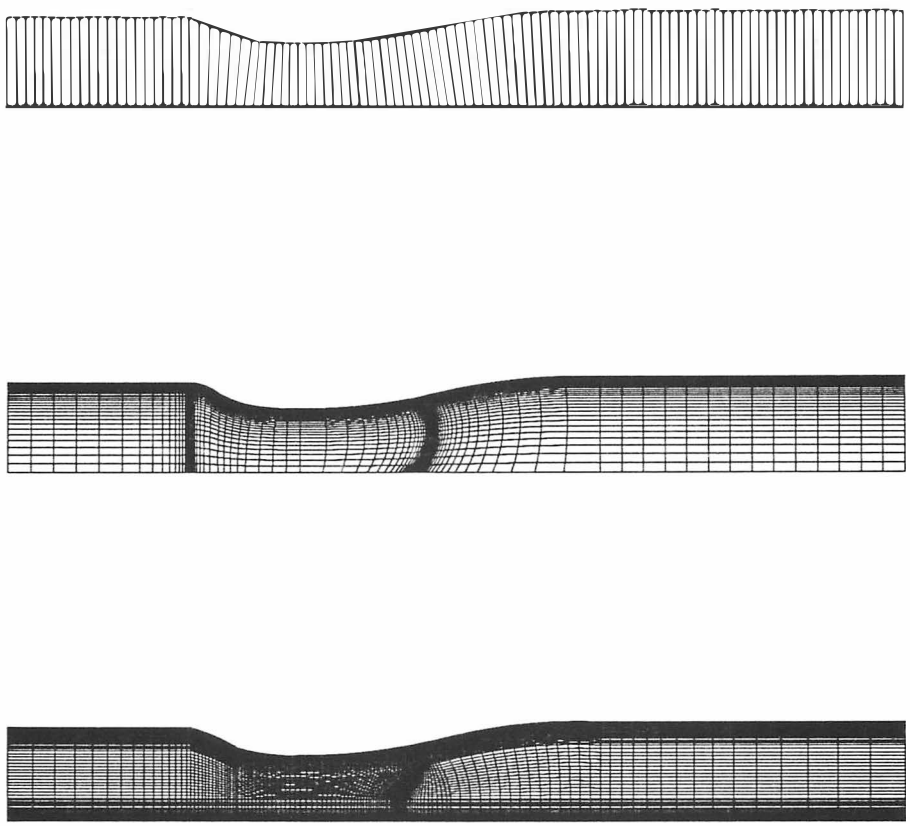


Figure 8.29: Three Mesh Adaptations for the Solution Progression of the Strong-Shock Transonic Wind Tunnel Problem

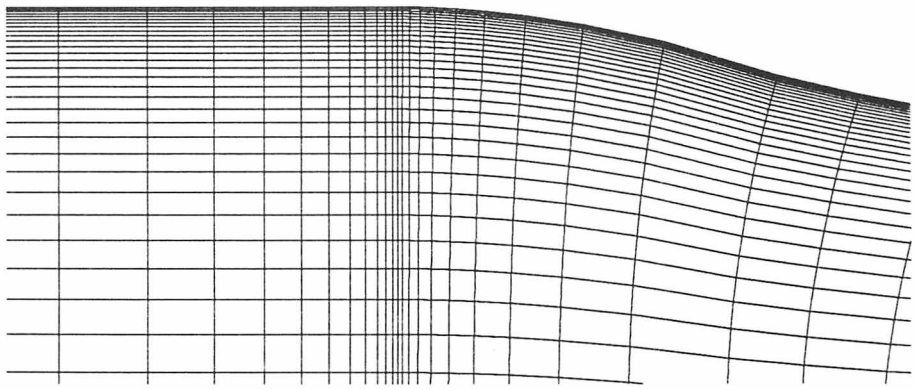


Figure 8.30: Local Magnification of Inlet Transition Region Mesh for Inviscid Simulation of the Transonic Wind Tunnel Problem

also capture the shock, but this will not approach the true distance of a shock, on the order of the molecular mean free paths, to be simulated. Typically the shock is captured on a minimum of two to three elements, regardless of the actual distance between element nodes. Therefore, the smaller the distance between these particular nodes, the nearer the solution approaches the actual limit of the shock.

Figure 8.31 shows the computational grid mapped to transform (η_1, η_2) space, for the viscous simulation of the problem. Two viscous walls exist, hence a total of six macro elements are needed to span the η_2 direction; with 9 macro elements spanning the η_1 direction, this yields 54 macro elements total. As

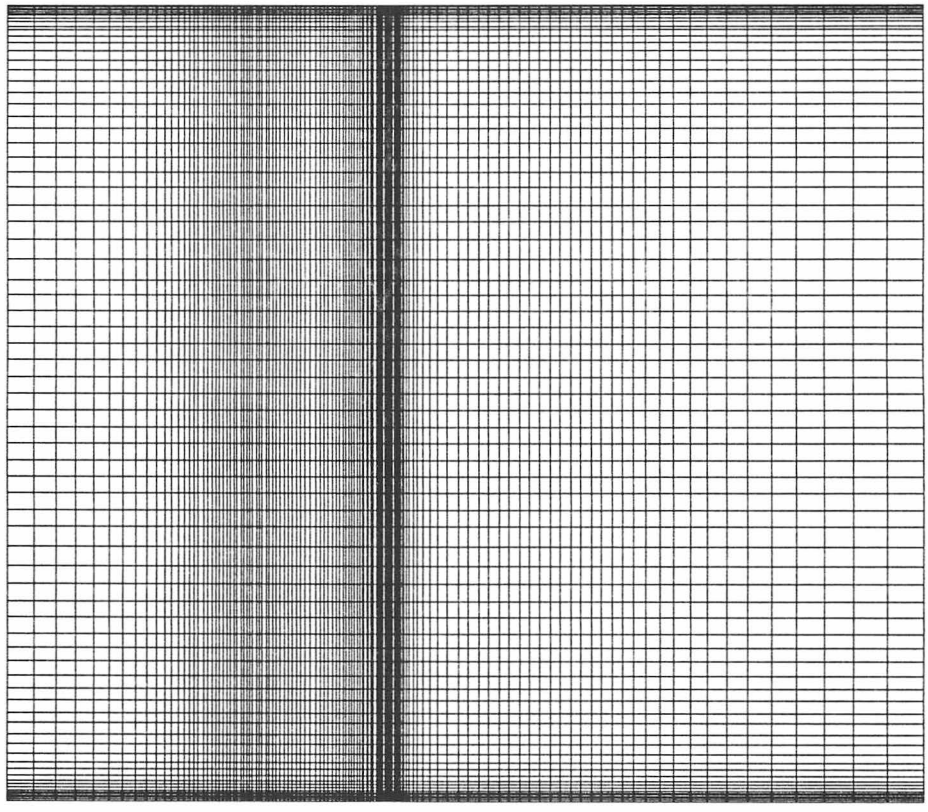


Figure 8.31: Grid Specification for the Viscous Transonic Wind Tunnel Problem Plotted in Computational Space

shown in Figure 8.23, the total distance across the entire wind tunnel height is .2083 feet. The distances, number of finite elements, and progression ratios, starting at the first macro element from the lower wall, are (.001, .049, .1) feet, (10, 15, 5) finite elements, and (1.0, 1.34, 1.0) progression ratio respectively. This design is mirrored toward the top wall. Thus the total number of finite elements spanning the η_2 direction is 60. The number of finite elements spanning the η_1 direction is 116, also of nonuniform design, resulting in 7076 total nodes. Since there are 4 degrees of freedom per node, the total of 28304 degrees of freedom are solved. An additional refinement utilized grids up to 136×76, yielding a grand total of 41344 degrees of freedom for the linear algebra process.

8.2.3 Quasi-One Dimensional Flow Simulation

The quasi-1D steady solution for the transonic wind tunnel problem is summarized in Figure 8.32, for the weak-shock case, and Figure 8.33 for the strong-shock case. The solutions are nearly identical except for a few subtle details. One difference is the exit pressure Dirichlet data, which was held at .7061 for the weak shock case and .6785 for the strong shock. Hence, in comparison, the shock strength is larger, and shock position is displaced further downstream for the strong-shock case. The solution local oscillation at $x = -.375$ feet exists because of the converging section geometry limitation, as discussed. Experimental data also revealed a local wall-pressure peak at this position, Also, the value of TWS parameter β is decreased slightly for the strong-shock case, causing the shock Mach number to be sharper but also causing the momentum to slightly overshoot, Figure 8.33, part c.

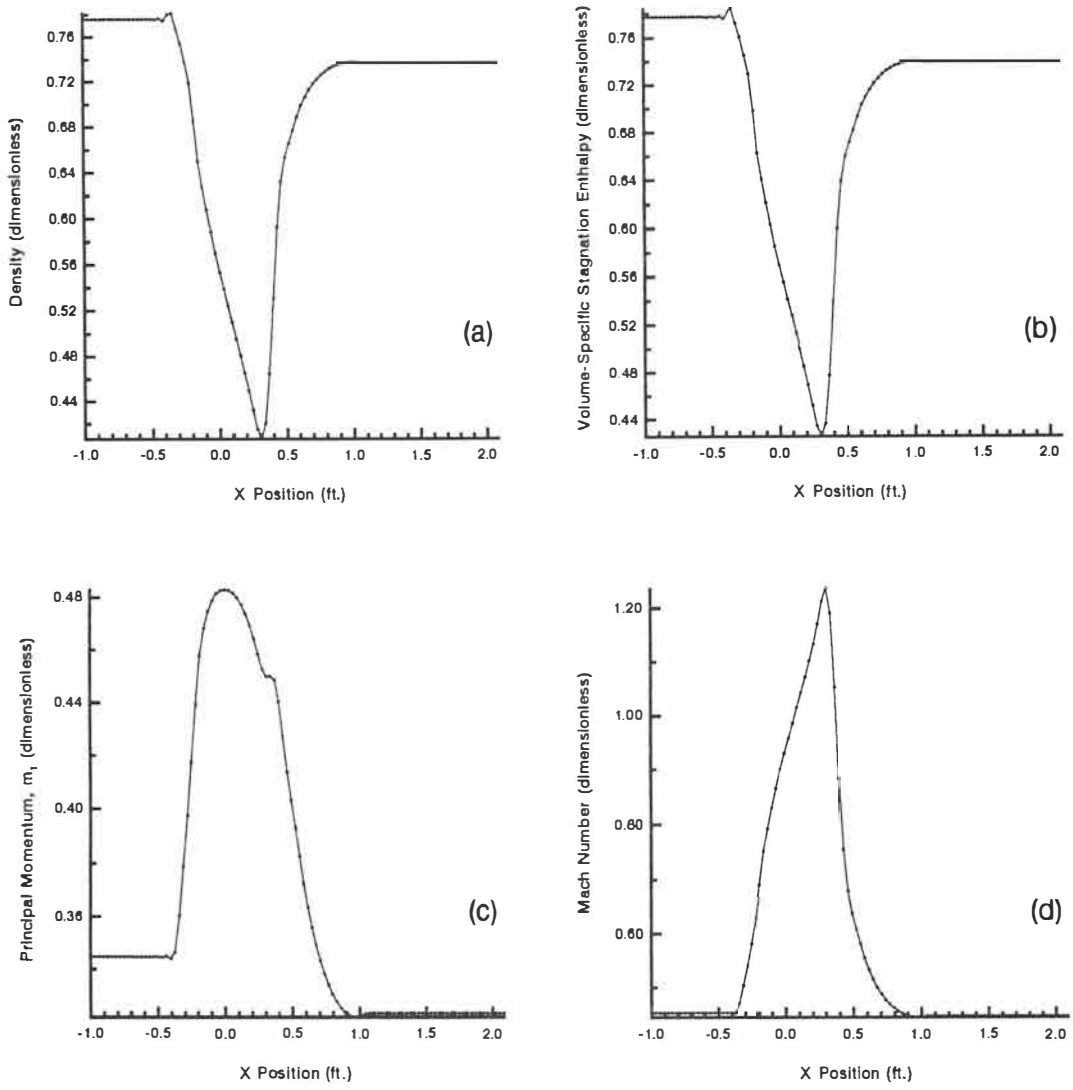


Figure 8.32: Weak-Shock Transonic Wind Tunnel Problem Quasi-1D Solution; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Velocity.

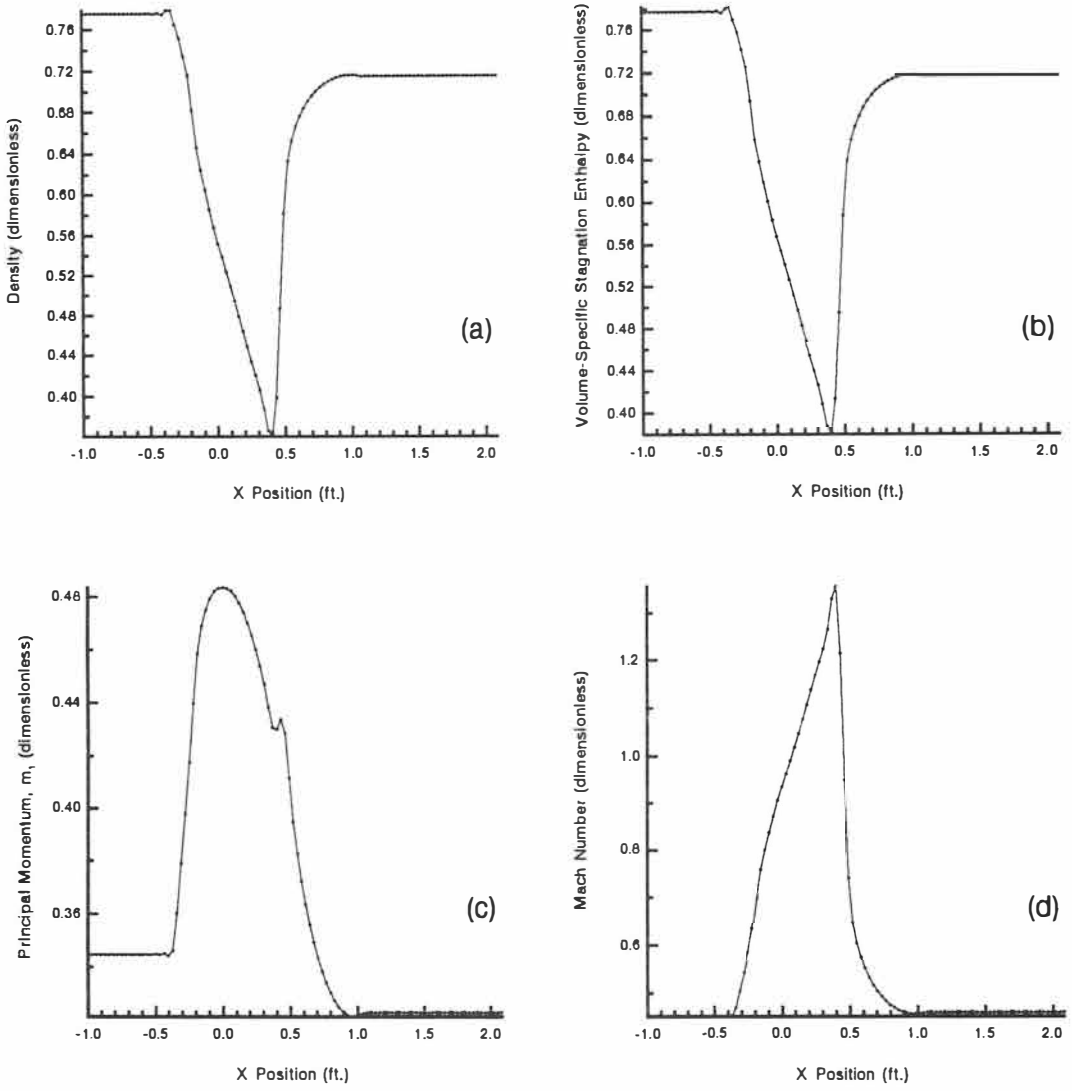


Figure 8.33: Strong-Shock Transonic Wind Tunnel Problem Quasi-1D Solution; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Velocity.

The weak-shock case was utilized for a sensitivity study of the dissipation generated by the TWS parameter β. Recall the form for β is

$$\beta = \frac{\beta_0}{2\sqrt{15}}, \tag{8.4}$$

where 2 in the denominator, appears as a result of the Taylor series expansion, and $\sqrt{15}$ is a normalizing factor.[25] In Figures 8.32 and 8.33, $\beta_0 = 3.5$ and $\beta_0 = 3.0$ respectively. Figures 8.34 and 8.35 show expanded views of the shock region principal momentum and Mach number respectively, for the weak-shock case with three sensitivity cases of $\beta_0 = 2, 4$, and 6. These three cases span the β space from under- to over-dissipation of the shock. Based on a tradeoff between the graphical sharpness of the Mach number profile in the shock region, and a goal of obtaining a constant momentum flux in the same region, $\beta_0 = 3$, hence $\beta = .3873$, was selected and held constant throughout the remainder of the simulations documented in this dissertation.

8.2.4 Inviscid Flow 2D Simulation

The steady-state quasi-1D solution initialized the 2D inviscid solution following the procedure described. The resultant steady-state inviscid approximate solution for the weak-shock case is summarized in Figure 8.36 as the state variable iso contour. The corresponding contour distributions for pressure, principal velocity, transverse velocity, and Mach number are shown in Figure 8.37.

Several interesting aspects of the solution result from the curvature of the nozzle top wall (surface 4). The first perturbation of the system, as viewed from left to right (inlet to outlet), is due to the sudden area decrease at the throat entrance. The generated transverse momentum extends nearly to the wind tunnel centerline. This causes the pressure to increase sharply as confirmed by the experimental data, Figure 8.24. Just downstream of the entrance and prior to

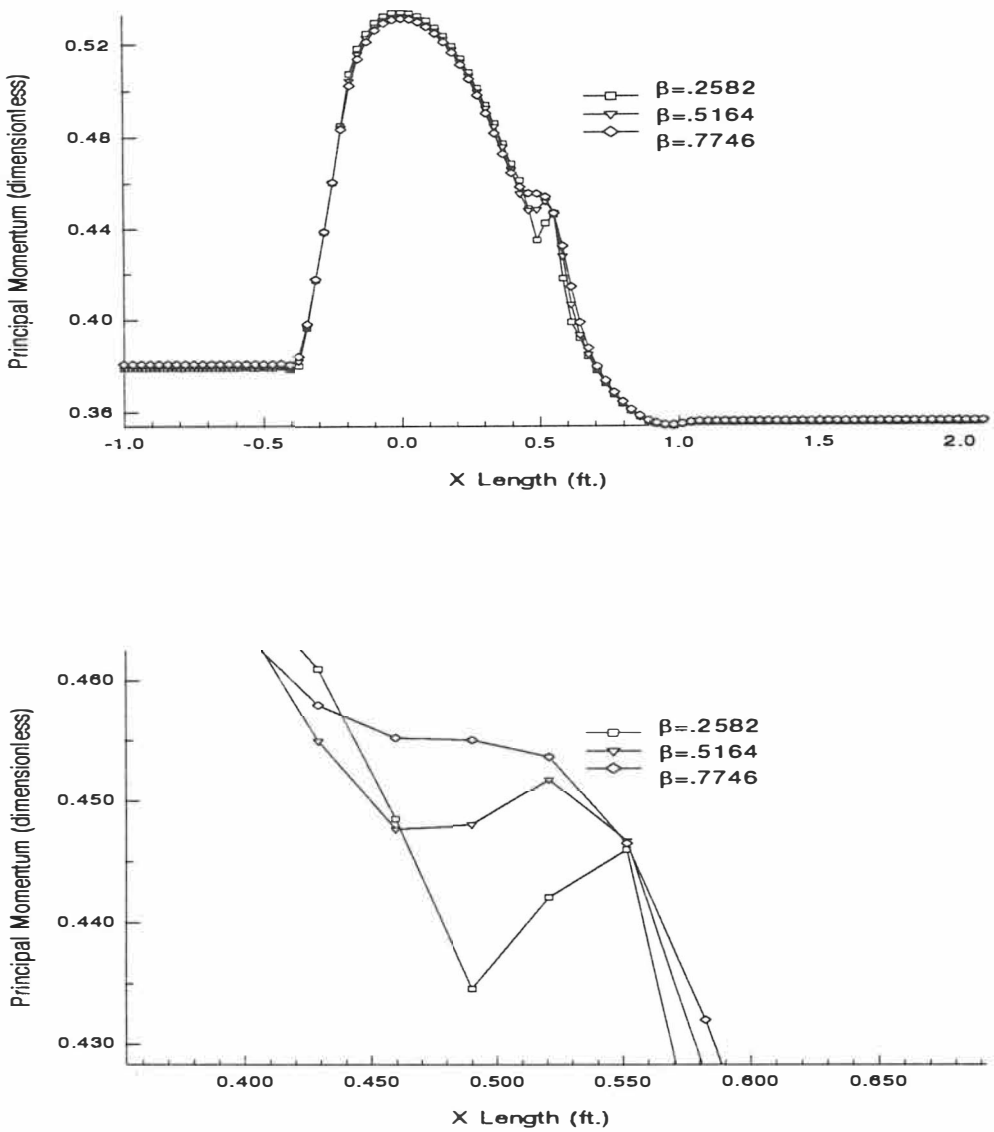


Figure 8.34: Beta Sensitivity on Principal Momentum for the Weak-Shock Transonic Wind Tunnel Problem Quasi-1D Solution.

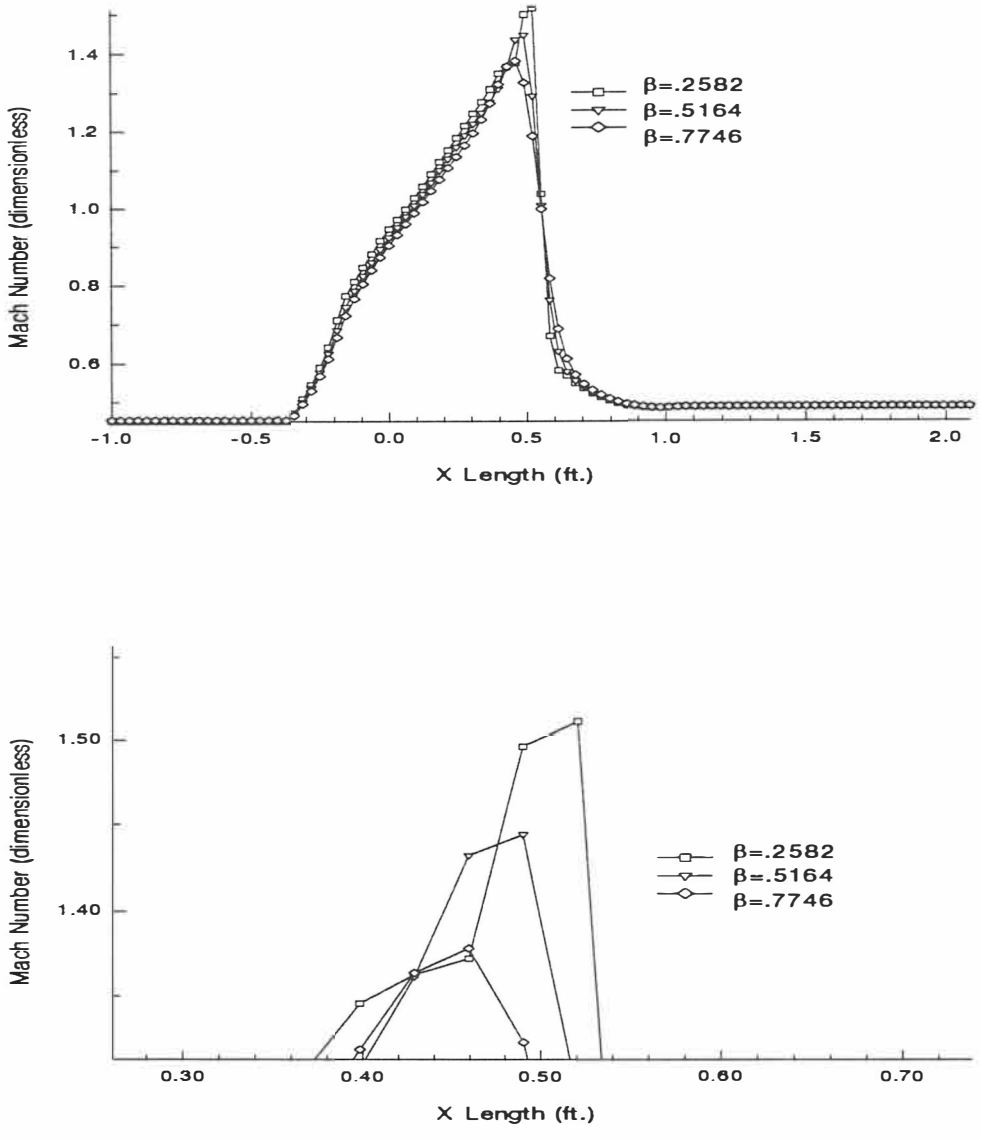


Figure 8.35: Beta Sensitivity on Mach Number for the Weak-Shock Transonic Wind Tunnel Problem Quasi-1D Solution.

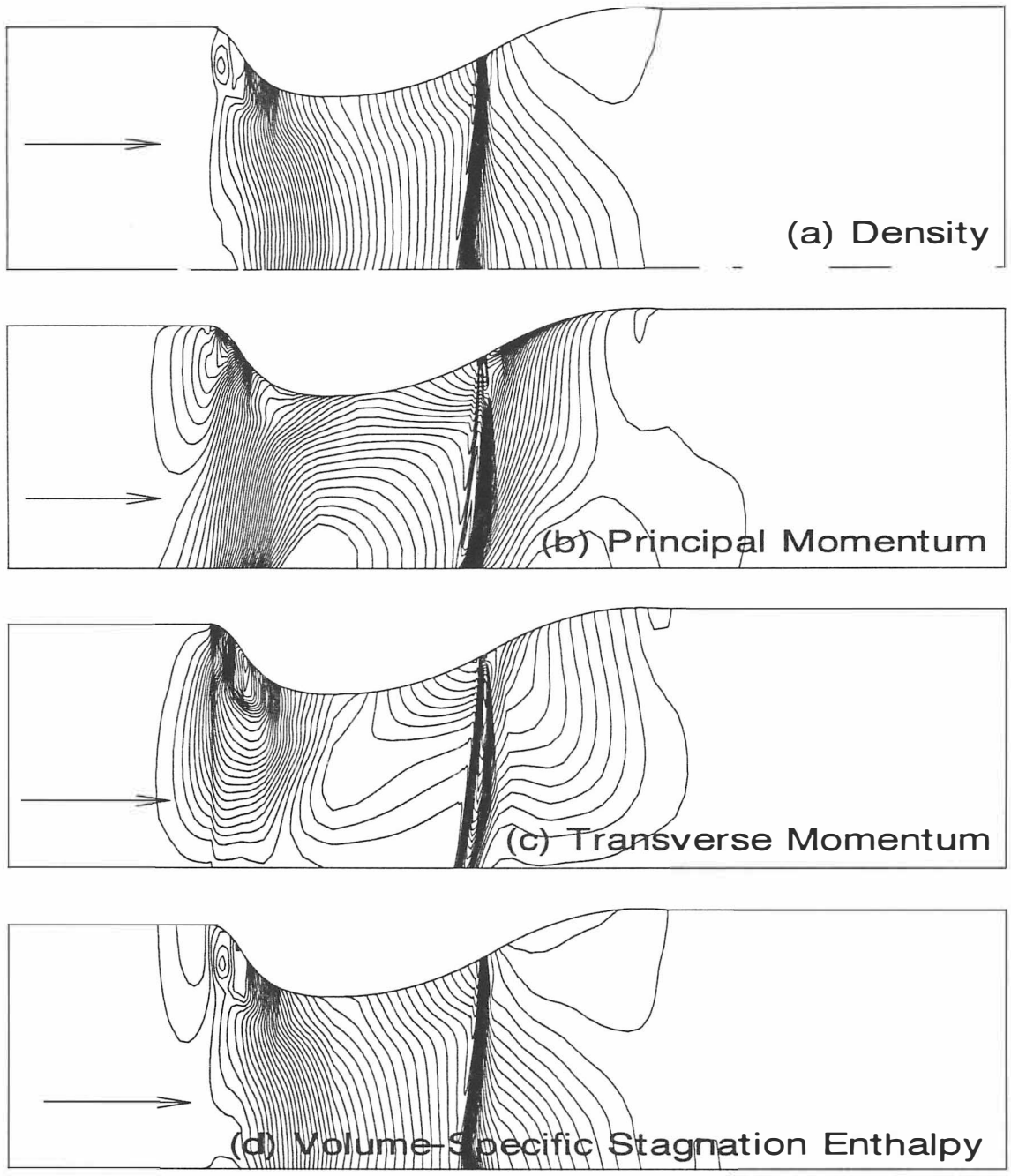


Figure 8.36: Inviscid Solution of the Weak-shock Transonic Wind Tunnel Problem Shown as Iso-Value Contours in Physical Space; (a) Density, (b) Principal Momentum, (c) Transverse Momentum, and (d) Volume-Specific Stagnation Enthalpy (vertical scale expanded for clarity).

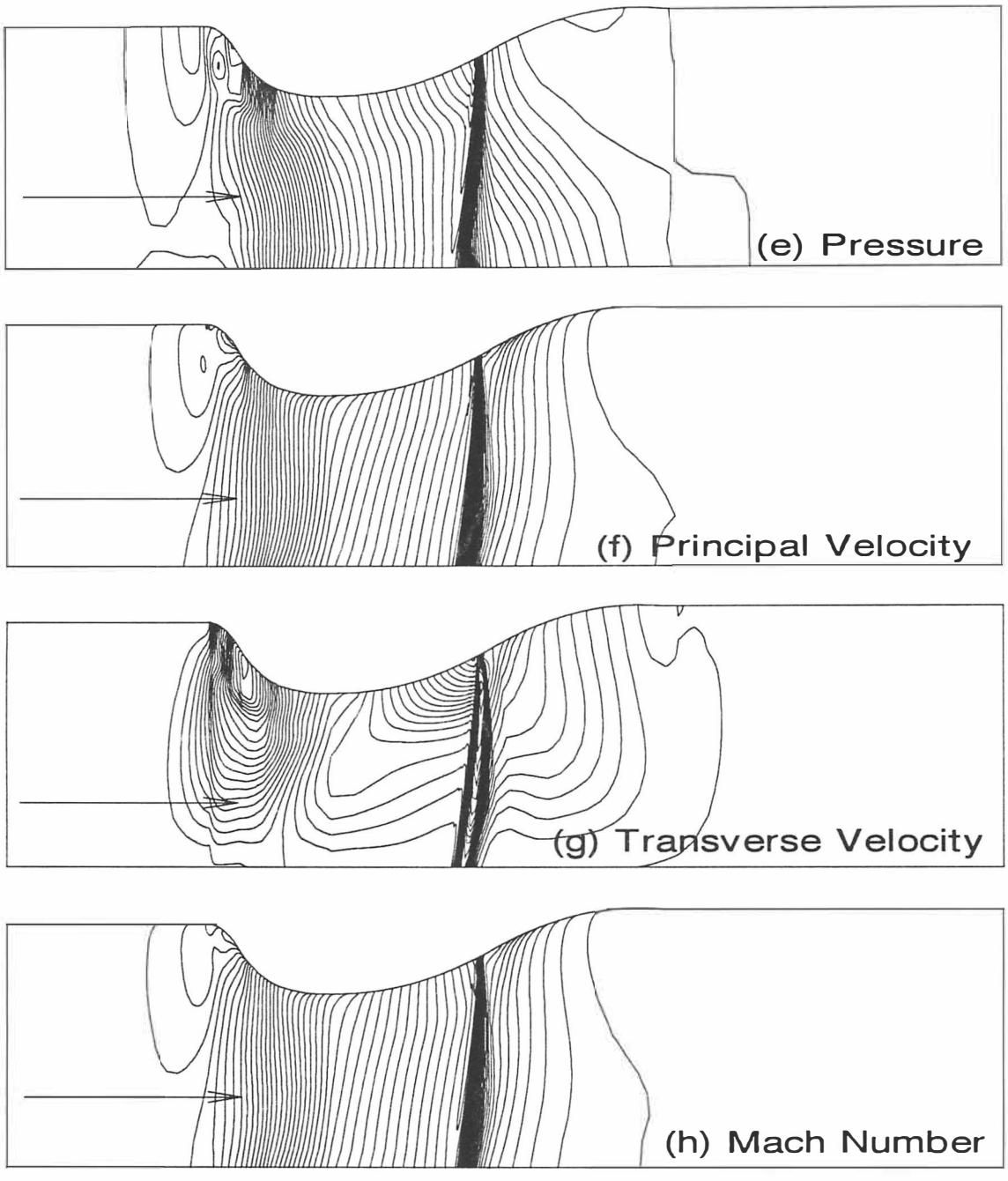


Figure 8.37: Inviscid Solution of the Weak-Shock Transonic Wind Tunnel Problem Shown as Iso-Value Contours in Physical Space; (e) Pressure, (f) Principal Velocity, (g) Transverse Velocity, and (h) Mach Number (vertical scale expanded for clarity).

the throat, a flow deflection region is developed due to the momentum exchange caused by the area decrease.

At the throat, the Mach number takes on a unit value, as graphically confirmed in the color iso contour discussed. The transverse momentum (and velocity) zero iso contour therein connects the nozzle wall to the lower tunnel wall defined by surface 3. In the shock region, the flow tangency boundary condition of surface 3 does not completely enforce the transverse velocity/momentum to absolute zero. This is later confirmed corrected by additional mesh refinement. The normal shock is captured with sufficient resolution as evidenced by the dense packing of the iso contours for all variables. All variables show very little change in the streamwise direction downstream of the upper wall curvature cessation.

Nodal carpet surface presentation of the state variable again yield a quantitative estimate of solution smoothness. Shown from the two different view points, Figures 8.38 and 8.39 highlight flow regions of interest including the dominant perturbations caused by the nozzle entrance area change, the shock, and the nonzero transverse momentum spike on the bottom wall (surface 3) caused by the shock intersection.

Perhaps the most interesting part of the solution is the near-normal shock downstream of the throat. The natural boundary condition associated with the finite element weak statement ($\boldsymbol{m} \cdot \boldsymbol{n} = 0$), not a Dirichlet constraint ($m_2 = 0$), should drive the transverse momentum toward zero at the lower wall (surface 3). As verified by examining part (d) of the nodal plot Figures 8.38 and 8.39, a nonzero transverse momentum occurs at the position of the shock along the bottom wall. This deficiency in turn causes other variables to take on a nonsymmetric feature along the same boundary. This type of inviscid flow

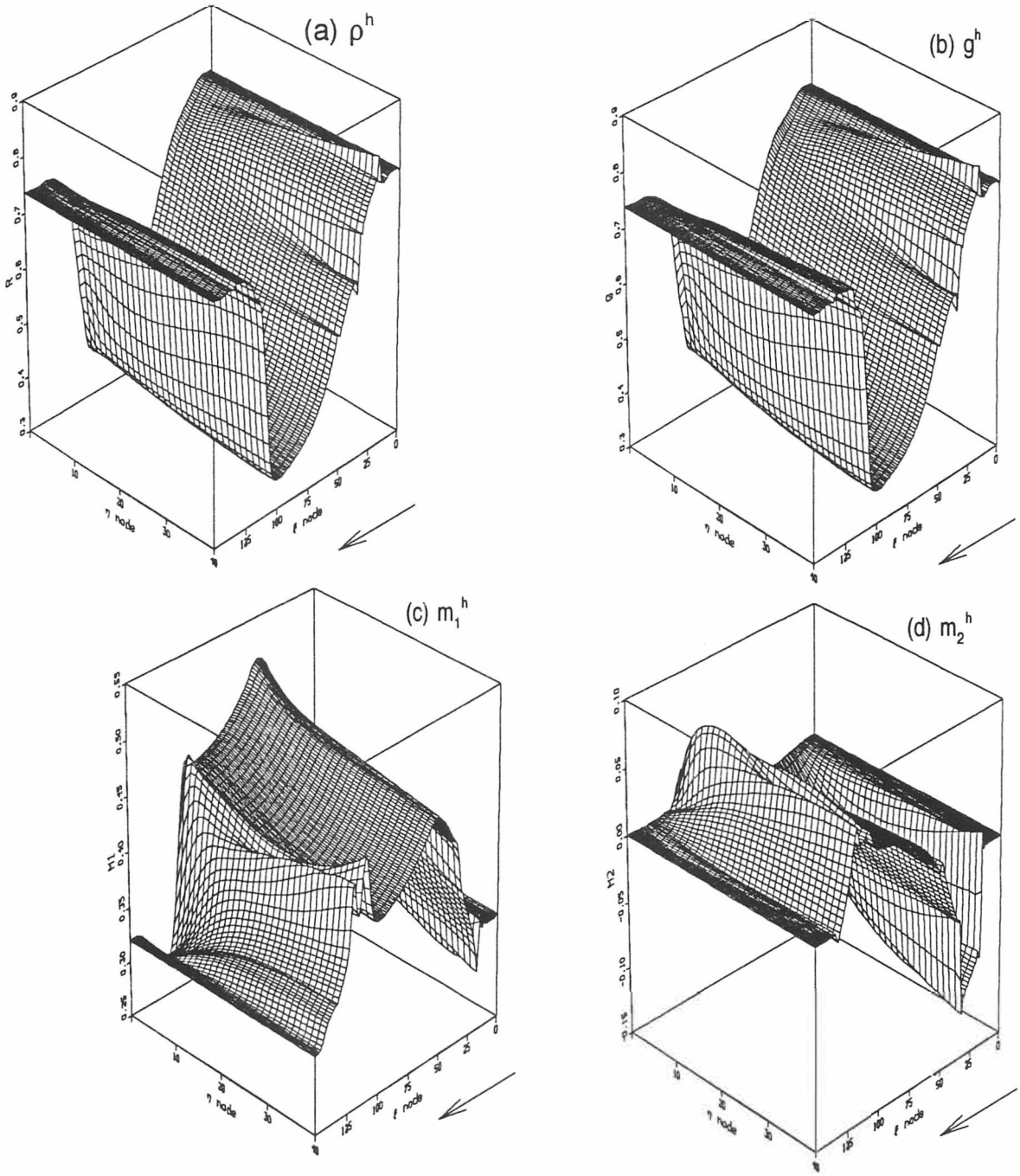


Figure 8.38: Inviscid Solution of the Weak-Shock Transonic Wind Tunnel Problem, Plotted in Nodal Space as Viewed from the Top Exit. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

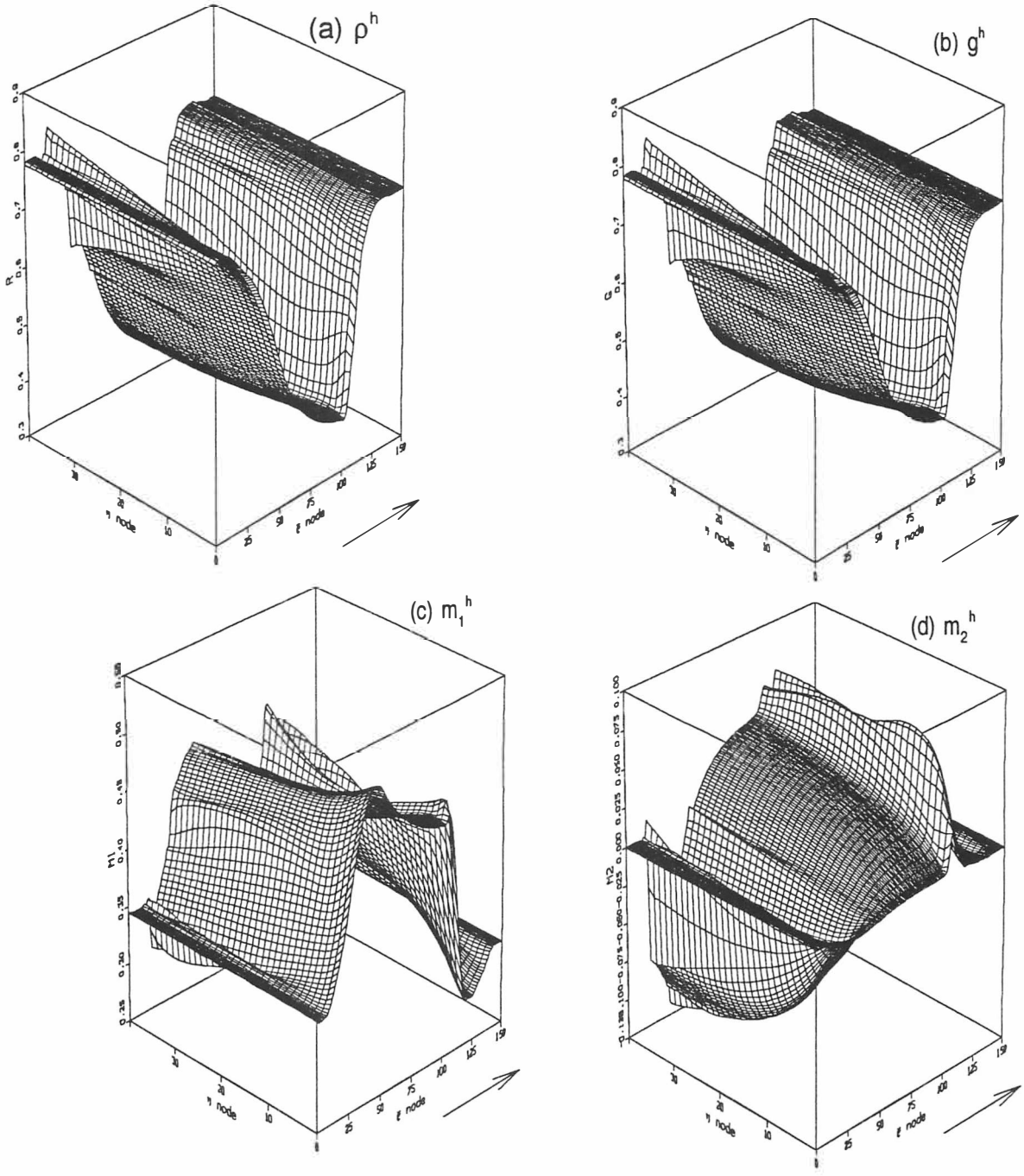


Figure 8.39: Inviscid Solution of the Weak-Shock Transonic Wind Tunnel Problem, Plotted in Nodal Space as Viewed from the Bottom Entrance. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

simulation deficiency is shown remedied in the next section (GAMM nozzle) by more-refined mesh and a unique treatment for flow-tangency boundaries. However, for this wind tunnel problem, the deficiency is remedied on surfaces 3 and 4 by additional mesh refinement along the viscous wall region.

The relationship between the physical position of the tunnel upper wall curvature and the mesh nodal positions is realized by examining the facing gridline on the density nodal plot, part (a), of Figure 8.38. The inlet, constant area region corresponds to the essentially constant density surface along the first few η_1 grid lines. Then a sudden peak in density occurs at the nozzle entrance area change of the tunnel, which verifies the need for a more refined mesh. The density continues to decrease as the throat is approached. The next oscillation in the density corresponds to the minimum in transverse momentum, part (c) of Figures 8.36, 8.37, and 8.38. The density continues to decrease with further area decrease. The next surface pattern perturbation corresponds to the minimum flow area at the throat further confirmed by the zero transverse momentum shown in part (d) of Figures 8.38 and 8.39. Thus, nearly half of the mesh is required to resolve the flow entrance region prior to the throat, which proves to be as demanding as resolution of the shock region of prime interest. The entrance region was further refined in the viscous case to smooth the observed wall curvature oscillatory features uncovered in this inviscid simulation.

The density continues to decrease, passing through the throat until it reaches a minimum on the upstream side of the shock. Along this same path, the transverse momentum increases until a sudden decrease is present at the shock, part (d) of Figure 8.38. The resolution and position of shock is evidenced by the Rankine-Hugoniot condition approximation, i.e., the constant principal momentum region clearly visible along constant η_1 lines at approximately $\eta_1 = \xi = 125$. The flow recovers from the shock very quickly as evidenced by

the sudden increase in density. Only a few additional η_1 gridlines are required to resolve the flow on out the tunnel exit.

The corresponding iso contour plots for the strong-shock inviscid simulation are shown in Figures 8.40 and 8.41. Figure 8.40 shows the state variable physical space, but the vertical scale has been expanded as before for visual clarity. Figure 8.41 contains the similar contours for pressure, principal velocity, transverse velocity, and Mach number. The strong shock position is further downstream as expected. Furthermore, the shock appears to have more curvature between the bottom and top. The increased shock strength has apparently caused the fluid to experience increased variation downstream of the shock as seen in all the contour plots. Experience has shown[36] that, for this type of inviscid transonic wind tunnel problem (converging/diverging), the shock curvature approaches a perpendicular angle to the wall curvature with increasing shock strength until eventually the shock curvature will have the appearance of intersecting oblique and normal shocks.

For this relatively uncomplicated inviscid simplification, the distinct characteristic between the weak- and strong-shock cases is realized in the generated solutions. As discussed in the problem introduction for the complete viscous problem, one experimentally observed difference between the weak- and strong-shock cases is the explanation of the flow characteristics associated with the recirculation at the upper wall surface 4 and shock region. In the weak-shock case, the pressure change dominates the recirculation and boundary layer interaction, while in the strong-shock case, the shock itself becomes the dominate influence. The distinction between the pressure change (weak) and shock (strong) influences is clearly evidenced, even for this inviscid problem, in the iso contour plots, particularly the momentum/velocity variables, by the curvature pattern of the shock as it becomes normal to the upper wall, and by the increased number

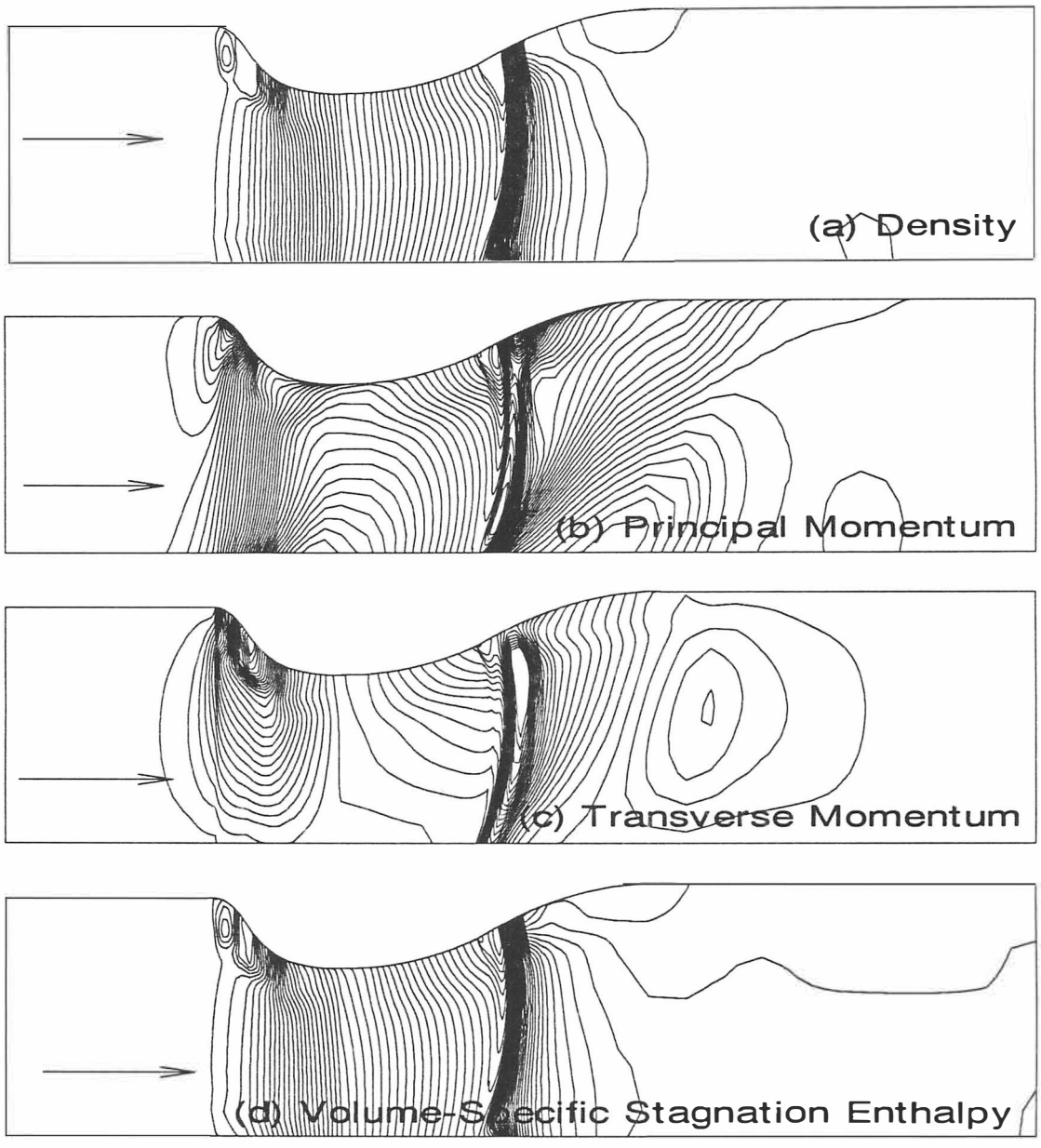


Figure 8.40: Inviscid Solution of the Strong-shock Transonic Wind Tunnel Problem Shown as Iso-Value Contours in Physical Space; (a) Density, (b) Principal Momentum, (c) Transverse Momentum, and (d) Volume-Specific Stagnation Enthalpy (vertical scale expanded for clarity).

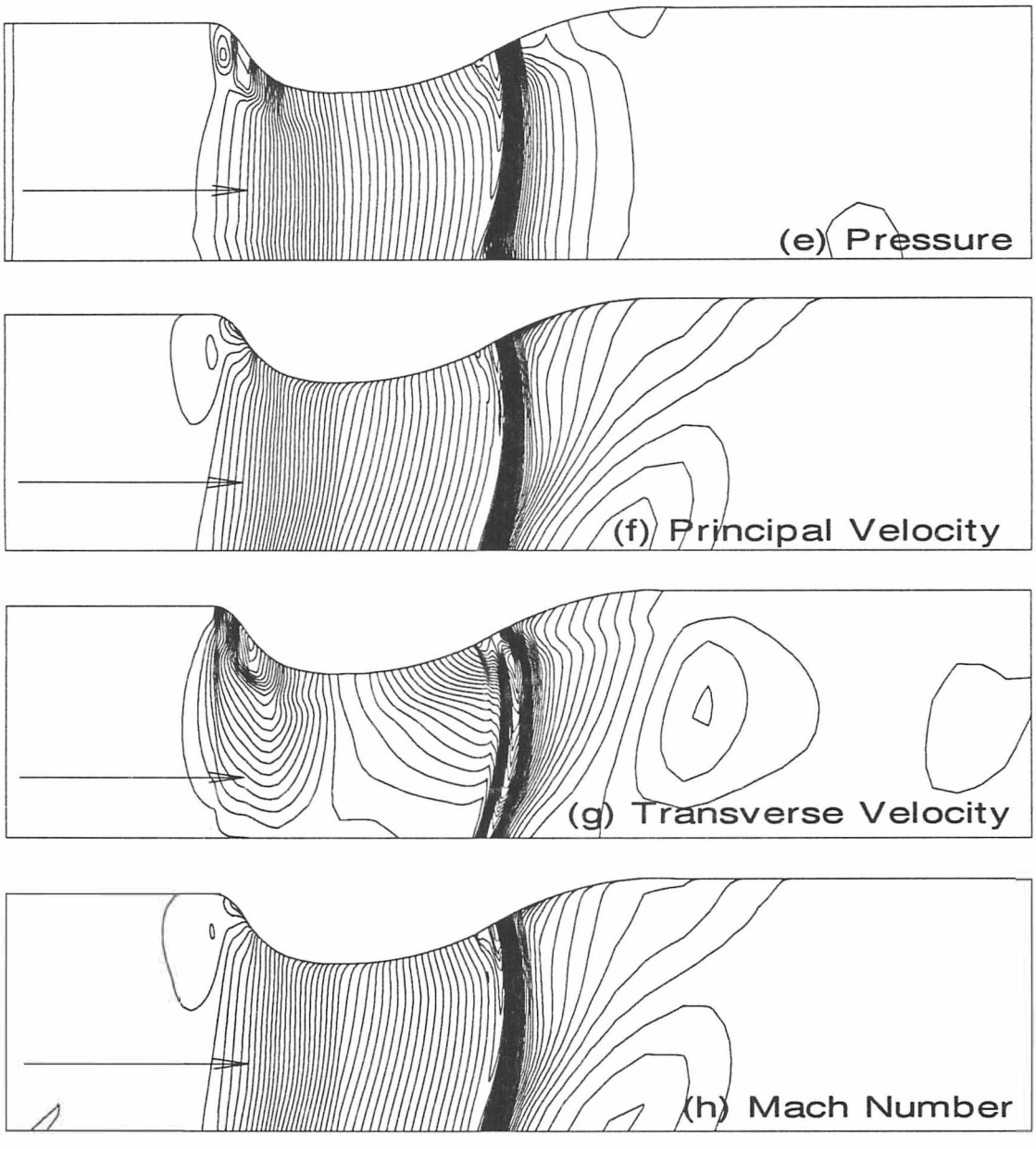


Figure 8.41: Inviscid Solution of the Strong-Shock Transonic Wind Tunnel Problem Shown as Iso-Value Contours in Physical Space; (e) Pressure, (f) Principal Velocity, (g) Transverse Velocity, and (h) Mach Number (vertical scale expanded for clarity).

of contour lines downstream of the shock about the upper wall region. Thus, even before the viscous problem is solved to simulate recirculation, the patterns that drive the recirculation become evidenced in the inviscid case.

Figures 8.42 and 8.43 show the nodal surface plots of the strong-shock case from two different view points. Smoothness of the surfaces verifies an essentially non-oscillatory solution has been obtained. The strong-shock capture quality is relatively enhanced, for this simulation, by a combination of greater shock strength and a better-designed mesh. This is evidenced by the density change across the shock in 4-5 node rows in part (a) of Figure 8.42, rather than 7-8 node rows as in the weak-shock case. The resolution of the nozzle entrance region flow is improved by mesh enhancements, as confirmed by the appearance of the transverse momentum/velocity iso contours, parts (c) and (g) of Figures 8.40 and 8.41 respectively. The mesh refinement in the shock region was improved by allocating additional mesh along the principal momentum peaks on both the upper and lower walls. This is confirmed by observing part (c) of Figure 8.42 showing only a 2-3 grid line deviation in the η_1 direction between the peaks, thus improving resolutions of the curved shape of the shock.

8.2.5 Viscous Flow Simulations

The viscous simulations were restarted from the steady inviscid solution by imposing no-slip boundary conditions at the bottom and top walls (surface 3 and 4). The final CFD experiment objective is to generate sufficient turbulent boundary layer displacement thickness to match the experimental data to the degree attainable. Before the algebraic turbulent models can be utilized, the laminar viscous solution must be obtained to generate a velocity profile as data for the algebraic turbulence model. The laminar viscous problem statement employs a Reynolds number small enough to enhance stability in the boundary layer region

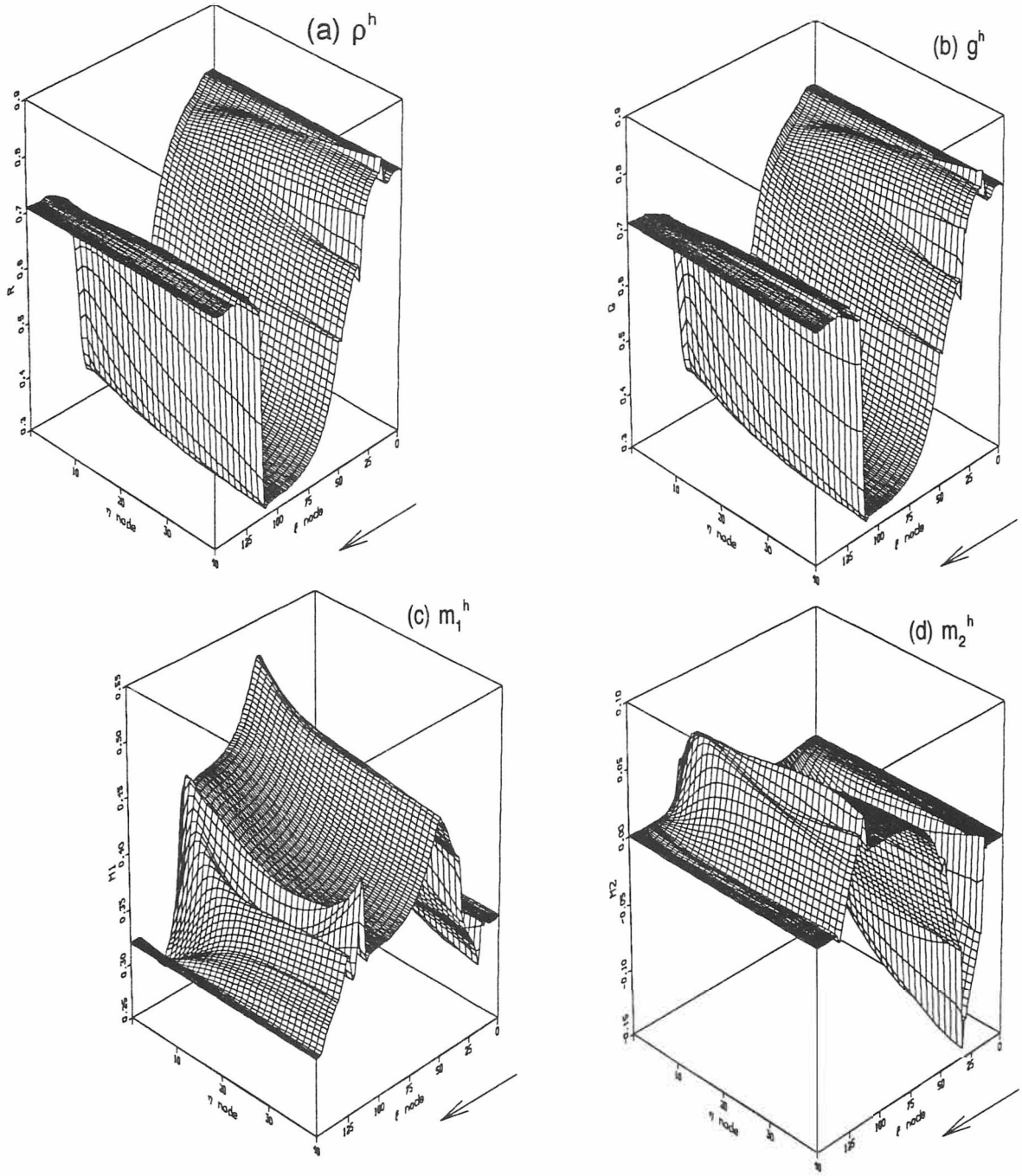


Figure 8.42: Inviscid Solution of the Strong-Shock Transonic Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Top Exit. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

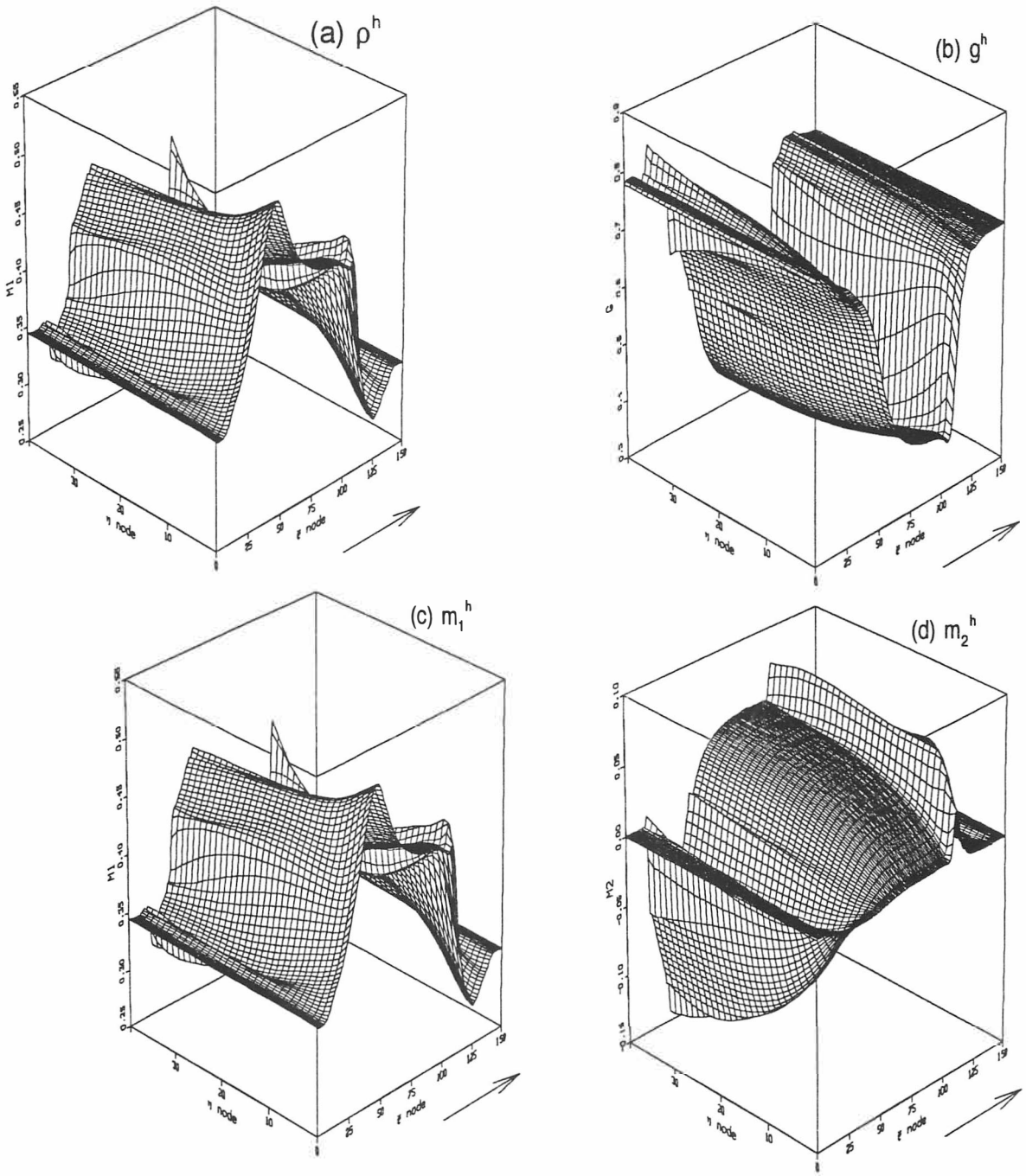


Figure 8.43: Inviscid Solution of the Strong-Shock Transonic Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Bottom Entrance. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

(where β is driven to zero), and to develop a displacement thickness of the same order of magnitude as the experiment. Specifically, for subsequent turbulence modeling, it is desired that the laminar velocity profile have a boundary layer thickness about the same as the objective turbulent case.

The Reynolds numbers to achieve the laminar initialization were 10^4 for the laminar weak-shock case and 10^3 for the strong-shock case. Upon achieving the steady laminar flow, the turbulent simulation solution involved no more than enabling the algebraic turbulence model and increasing the Reynolds number to the final value of 10^5. The Cebeci-Smith algebraic turbulence model was chosen for all turbulent flow simulations. The experimental turbulence was "triggered" at a position along the upper-wall, but the exact position is not given. Therefore, the algebraic turbulence model was enabled at the throat, and laminar flow was imposed between the wall trigger and the throat. As in all simulations in this dissertation, the no-slip wall boundary condition starts a few wall nodes downstream of the entrance, i.e., an inviscid boundary condition remains at the inlet plane. The final Reynolds number of 10^5 defined a length scale of $\approx .02$ feet, which is about the same as the boundary layer displacement thickness observed in the experiment, while still achieving a reasonable level of stability.[b]

The choice of Reynolds number is based on matching experimental data. Earlier, a discussion on the mesh adaptations, Figure 8.29, indicated the viscous mesh, shown as the bottom grid in the figure, was utilized for the final viscous turbulent simulation of the strong-shock case. It is important to realize that there is a direct relationship between the level of natural dissipation provided by the fluid viscosity and Reynolds number magnitude and the mesh refinement

[b]Stability here implies a time step that was large enough so as to obtain a solution in a reasonable amount of machine time.

requirements. Typically, the distance between adjacent grid lines normal to a viscous wall is on the order of $\frac{L}{\text{Rey}} \rightarrow \frac{L}{\sqrt{\text{Rey}}}$. During all simulations presented in this dissertation, an upper limit on Reynolds-number-bound stability was never witnessed, i.e., as long as sufficient grid resolution was provided, and the computer stored the grid distances to a sufficient number of digits, then genuine Reynolds number ($> 10^5$) solutions could be obtained. Of course, the integration time step must be correspondingly decreased which, combined with the increased mesh density, may cause the computational work to be prohibitive, but that is not a limitation of the developed algorithm.

Laminar Flow

The steady-state laminar viscous solution for the weak-shock transonic wind tunnel simulation is summarized by iso contours of pressure p, principal velocity u, transverse velocity v, and temperature T in the transverse-stretched physical space. Truly substantial differences exist between the inviscid and viscous simulations, Figure 8.37 and Figure 8.44. The principal velocity is densely contoured along surfaces 3 and 4 where the laminar boundary layer has formed. The boundary layer started slightly downstream of the inlet surface 1 by allowing the first few η_1 gridlines to remain inviscid. This removes an inconsistency in the inlet boundary conditions and allowed for a smooth transition from inviscid to viscous flow. The throat entrance area transition (as viewed from left to right) is much better resolved by the grid density shown in Figure 8.29. The shock-boundary layer interaction is clearly present, and will be discussed in more detail later. The transverse velocity is now held zero (Dirichlet constraint), correcting the flow tangency problem observed in the inviscid simulation. The increased wall region mesh density improves the pressure contour nominal-normal appearance to the bottom wall. The shock character is somewhat diffused in the wall

regions by the presence of natural viscous dissipation along the boundaries. This diffusion creates additional contours beyond the shock region, especially near the upper wall. The viscous region is completely absent of the TWS β term as described earlier. Both the transverse velocity and temperature contours show existence of a boundary layer confirmed by dense contours near the walls.

The state variable distribution for the laminar viscous weak-shock case is graphed as the carpet plot in nodal space by Figures 8.45 and 8.46. As anticipated, the only hint of oscillation is the location of minimal dissipation where the shock intersects the wall boundary layer causing flow reversal. This occurs in the momentum presentation on η_1 (or ξ) nodal planes $100 \rightarrow 125$ along both lower and upper walls in (c) and (d) of Figures 8.45 and 8.46. In this region, the principal momentum undergoes a change from zero to approximately -0.3 across a few nodes in the η_2 (or η) direction. Computational experience verifies that the solution becomes less oscillatory with decreasing Reynolds number, but if Reynolds number is too small, the objective recirculation region could become eliminated. The oscillations will respond to increasing grid density in this area, verified in the strong-shock simulation.

The viscous flow region initiation is clearly visible in part (c) of Figure 8.45 as the first few constant η_2 grid lines remain essentially equal to the inlet values ($\eta_1 = 1$) and the momentum drops to zero, corresponding to the no-slip wall, across a single node. The boundary layer develops very quickly, and continues to increase in thickness as the fluid moves downstream. As verified in the experimental data, and according to theory, the boundary layer thins in approaching the throat, but then continues to increase downstream through a local favorable pressure gradient. Variations in density and stagnation enthalpy, and hence pressure, are clearly present within the boundary layer, via the slight hump surfaces parallel to the η_2 directions toward the tunnel interior. The slight

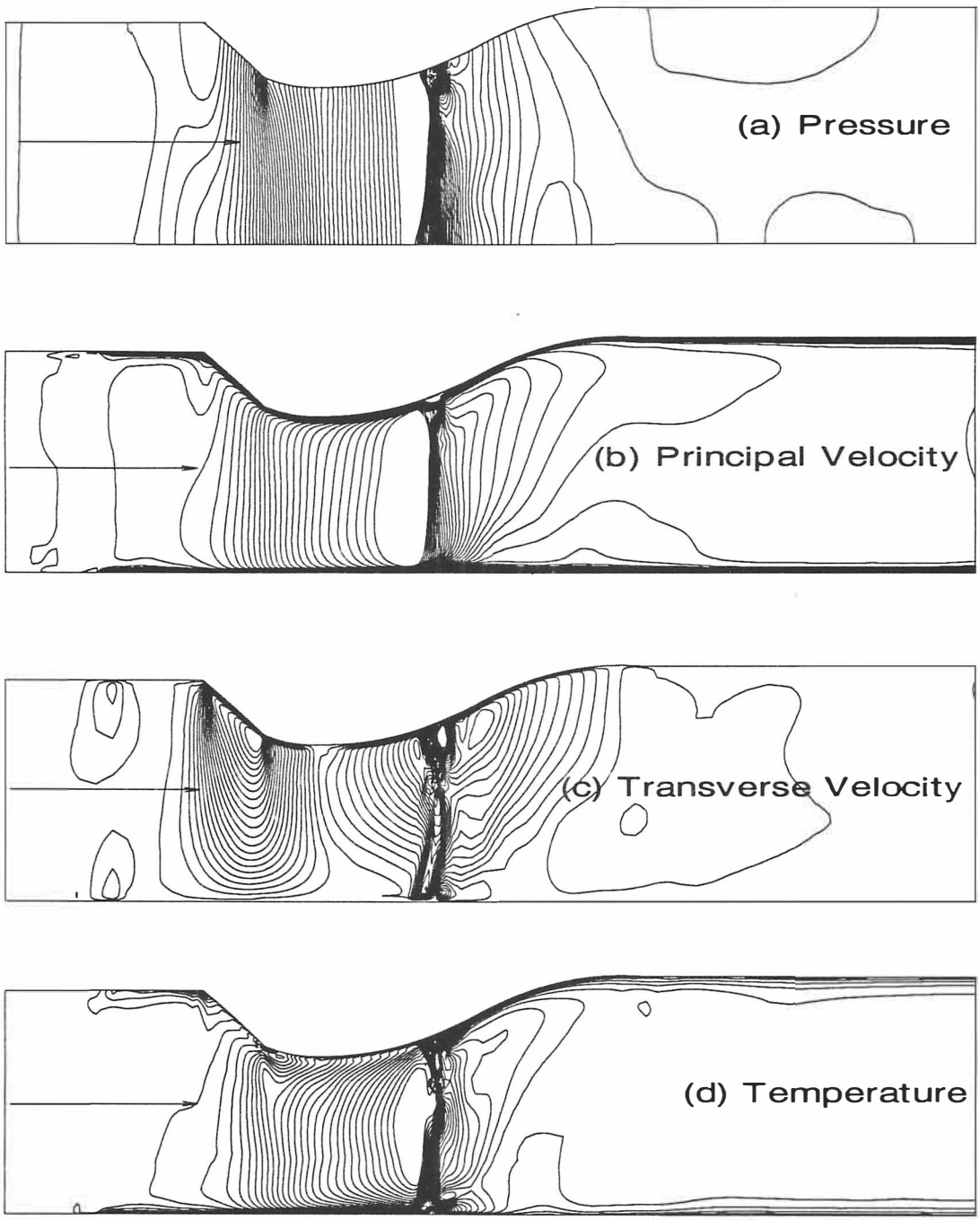


Figure 8.44: Laminar-Viscous Solution of the Weak-Shock Transonic Wind Tunnel Problem Shown as Iso-Value Contours in Physical Space; Shock Mach Number $\approx$ 1.235, Reynolds Number 10^4; (a) Pressure, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature (vertical scale expanded for clarity).

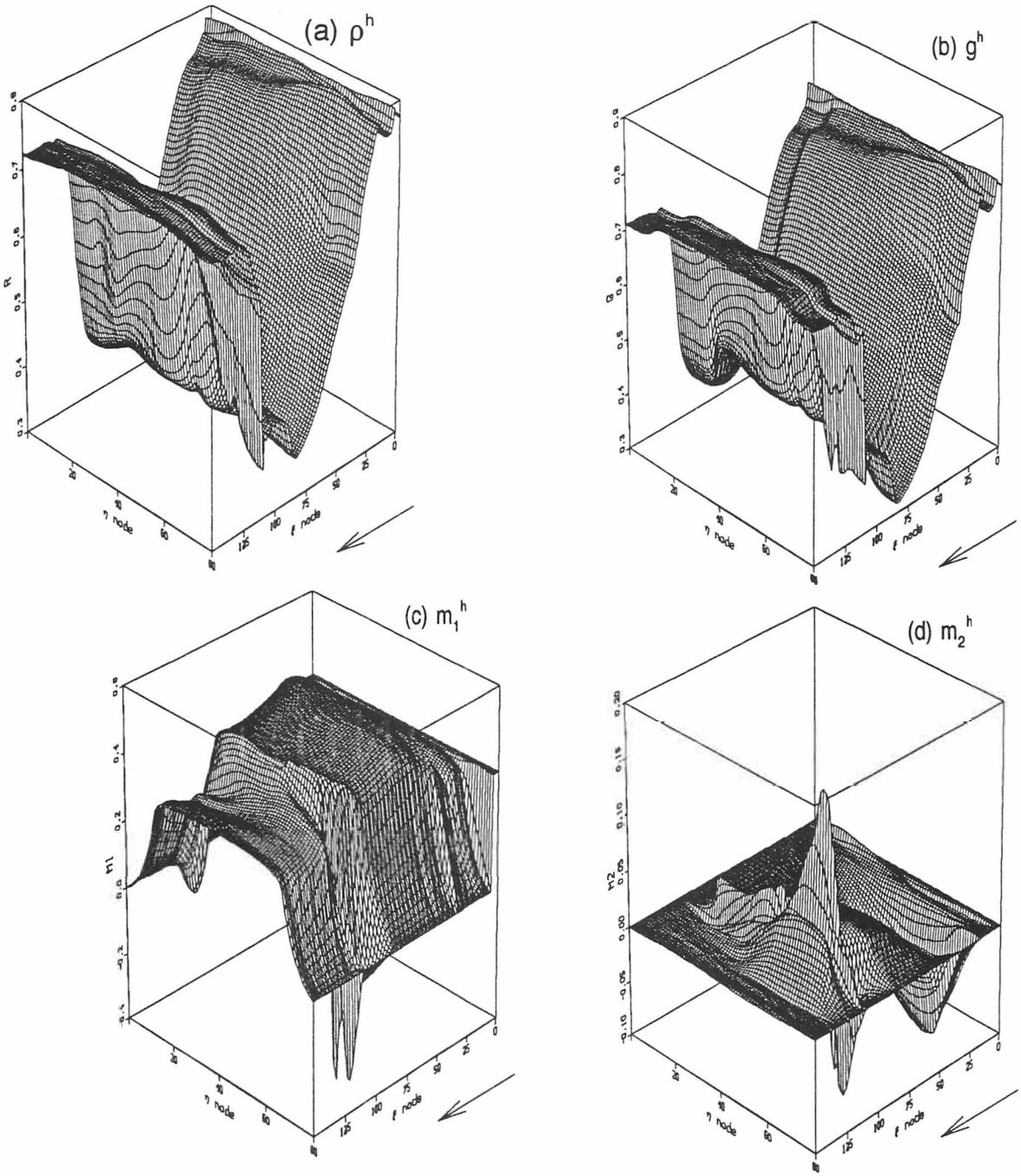


Figure 8.45: Laminar-Viscous Solution of the Weak-Shock Transonic Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Top Exit; Shock Mach Number 1.235, Reynolds Number 10^4; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

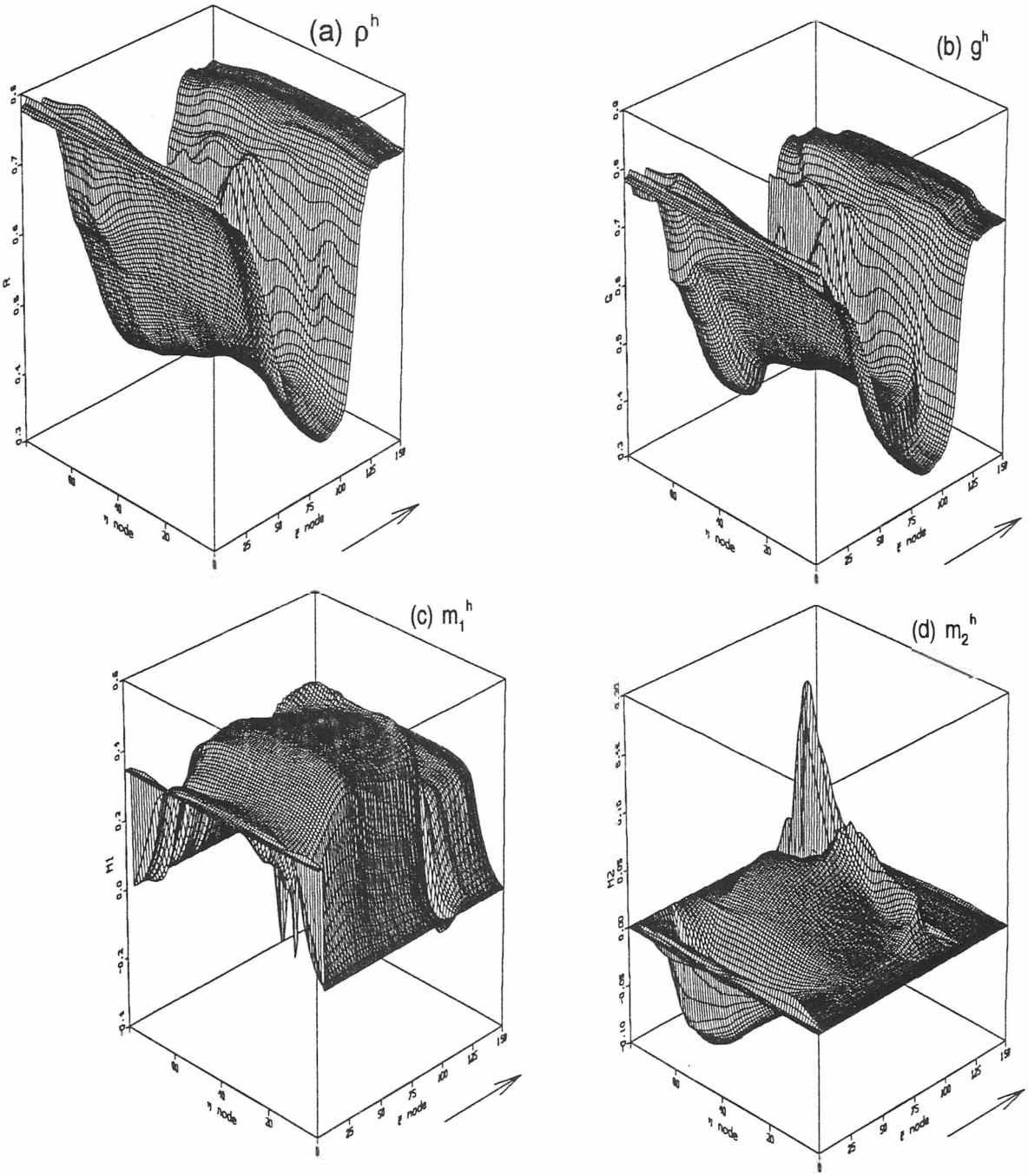


Figure 8.46: Laminar-Viscous Solution of the Weak-Shock Transonic Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Bottom Entrance; Shock Mach Number 1.235, Reynolds Number 10^4; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

oscillations in density, present in the inviscid simulation in the throat entrance region are now completely smooth, due to the increased mesh density and to natural viscous dissipation. This smoothness does not remove the pressure peak, shown via density and stagnation enthalpy surfaces, that is present from the throat entrance. The shock-boundary layer interaction is clearly visible in the density nodal surface, showing the shock captured over only 2-3 node rows at the wall and completely free of oscillation. All boundary conditions that naturally allow for vanishing derivatives are essentially non-oscillatory. In summary, the solution smoothness, hence quality, is quite satisfactory.

Figure 8.47 presents the comparison viscous solution summary for the strong-shock case. Because the Reynolds number has been decreased from 10^4 to 10^3, the boundary layer thickness has increased and the shock character is correspondingly somewhat further diffused. The lower Reynolds number was verified as appropriate for establishing the initial conditions for the turbulent simulation, to achieve the displacement thickness tabulated in the experimental data. The laminar weak-shock solution, Figure 8.44 and the strong-shock solution in Figure 8.47, are otherwise essentially similar except that the shock position is slightly downstream in the strong-shock case.

The state variable nodal-carpet plots for the steady strong-shock case are shown in Figures 8.48 and 8.49. These solutions appear very similar to the weak-shock set, except that the shock boundary layer interaction is clearly smoother. In particular, the recirculation region on both lower and upper walls is completely free of oscillation, while the shock remains clearly captured (the density surfaces) across 2-3 node columns. This can be attributed to the increased stability brought about by decreasing the Reynolds number to 10^3 and by a better grid refinement distribution about the shock region. The boundary

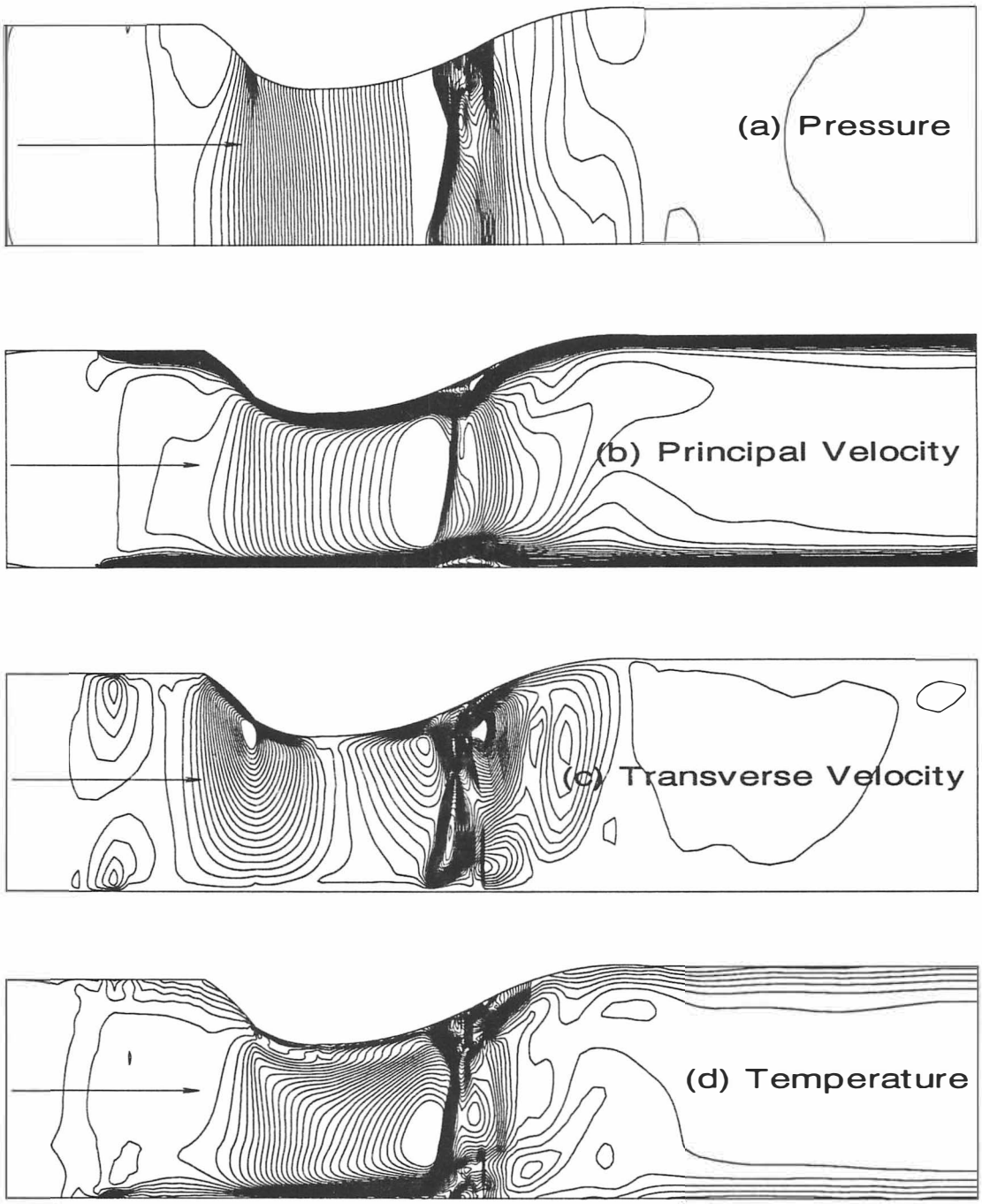


Figure 8.47: Laminar-Viscous Solution of the Strong-Shock Transonic Wind Tunnel Problem Shown as Iso-Value Contours in Physical Space; Shock Mach Number $\approx$ 1.353, Reynolds Number 10^3; (a) Pressure, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature (vertical scale expanded for clarity).

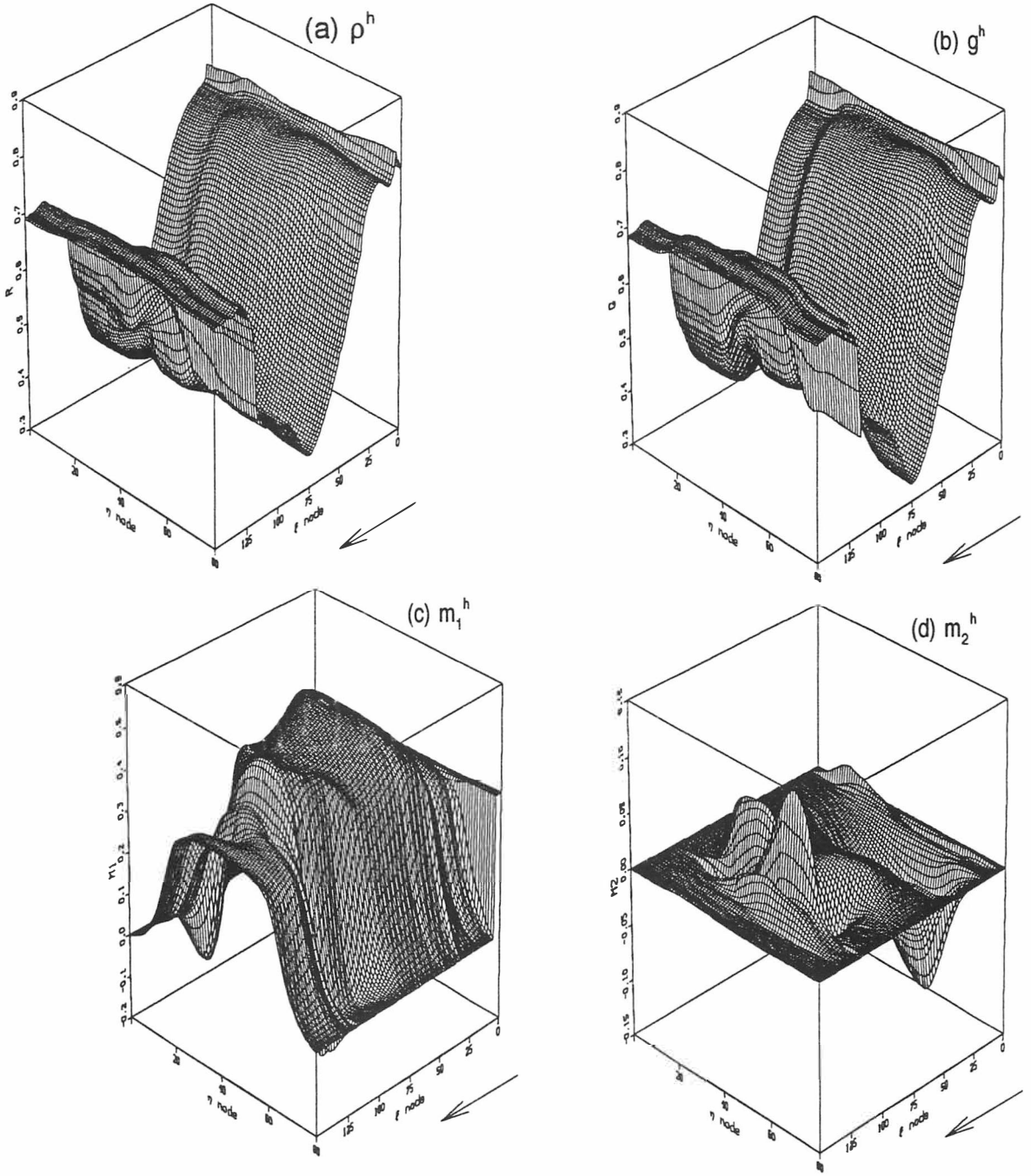


Figure 8.48: Laminar-Viscous Solution of the Strong-Shock Transonic Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Top Exit; Shock Mach Number 1.353, Reynolds Number 10^3; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

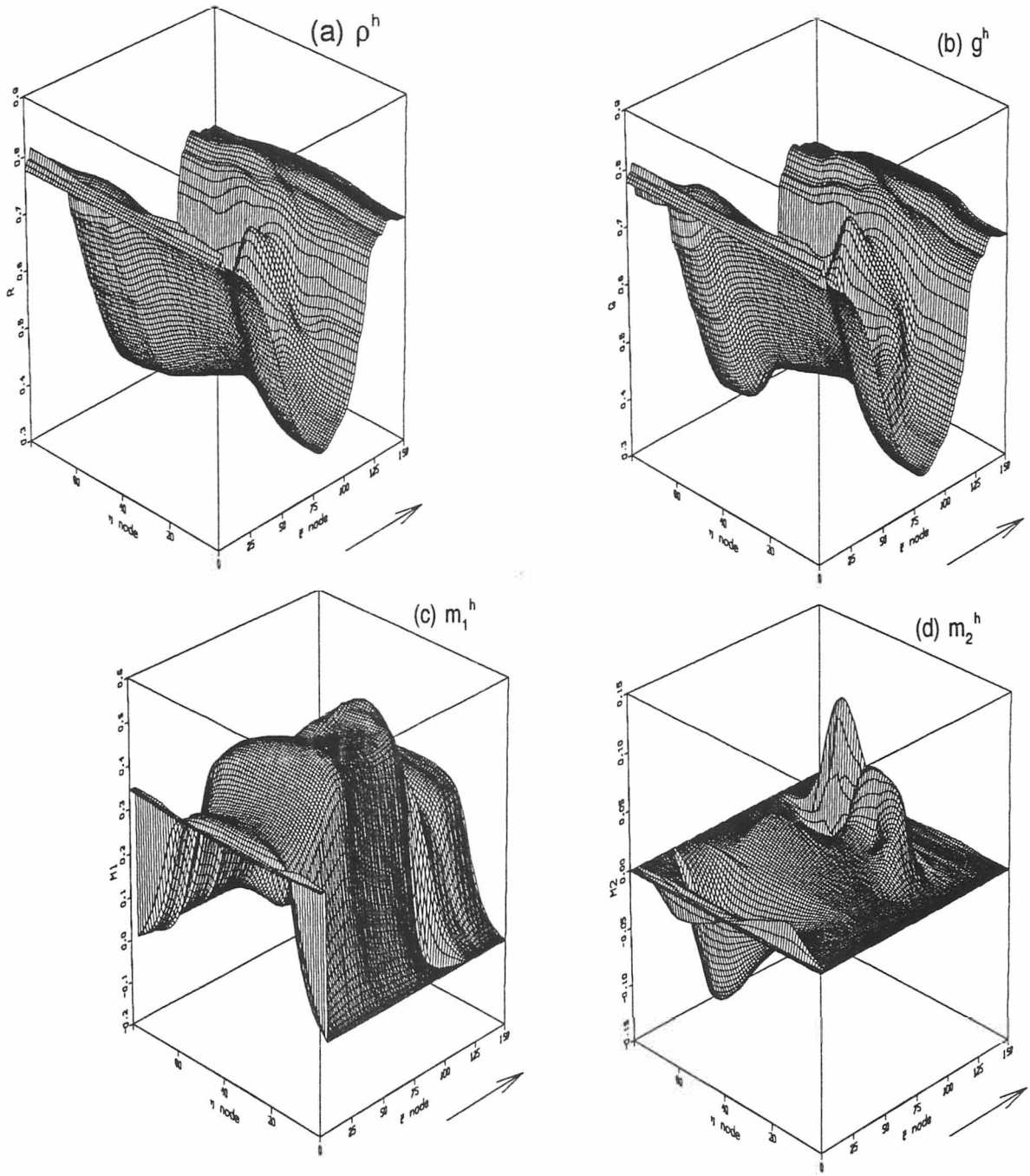


Figure 8.49: Laminar-Viscous Solution of the Strong-Shock Transonic Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Bottom Entrance; Shock Mach Number 1.353, Reynolds Number 10^3; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

layer thickness distribution is clearly larger, when compared to the weak-shock case, as is verified by the experimental data.

Vector plots showing magnitude and direction of selected velocity vector distributions for the laminar weak-shock and strong-shock simulations are shown in Figures 8.50 and 8.51. Each of the plot sequence figure shows progressively magnified views of the region of the shock-boundary layer interaction and recirculation. The weak-shock case, with a higher Reynolds number (10^4) and poorer mesh resolution, appears more oscillatory, as discussed. Both flow fields reattach as a stable boundary layer downstream of the shock, with axial laminar boundary layer profiles. These data certainly confirm that accurately placed mesh in the region of the shock-boundary layer interaction enhances solution accuracy and improved resolution of the shock recirculation. In both cases, it appears that the grid density normal to the wall is sufficient, hence refinements in the axial direction are required to be improved.

Turbulent Flow

The turbulent viscous solution of the weak-shock case is summarized as iso contour distributions in physical space by Figure 8.52. The solution appearance is very similar to Figure 8.44, the laminar solution, except for the boundary layer thickness distribution. Also transverse velocity and temperature levels have slightly increased for the turbulent case. The turbulence-enhance dissipation has diffused the shock over a slightly larger region, and overall increased viscous-turbulent dissipation causes an increased gradient to exist on all variables along the tunnel walls well downstream of the shock. This is confirmed by the presence of contour lines in the turbulent solution where none existed in the laminar results. Nonzero outflow Neumann boundary conditions were not incorporated into this solution, instead vanishing derivatives were imposed which are inaccurate

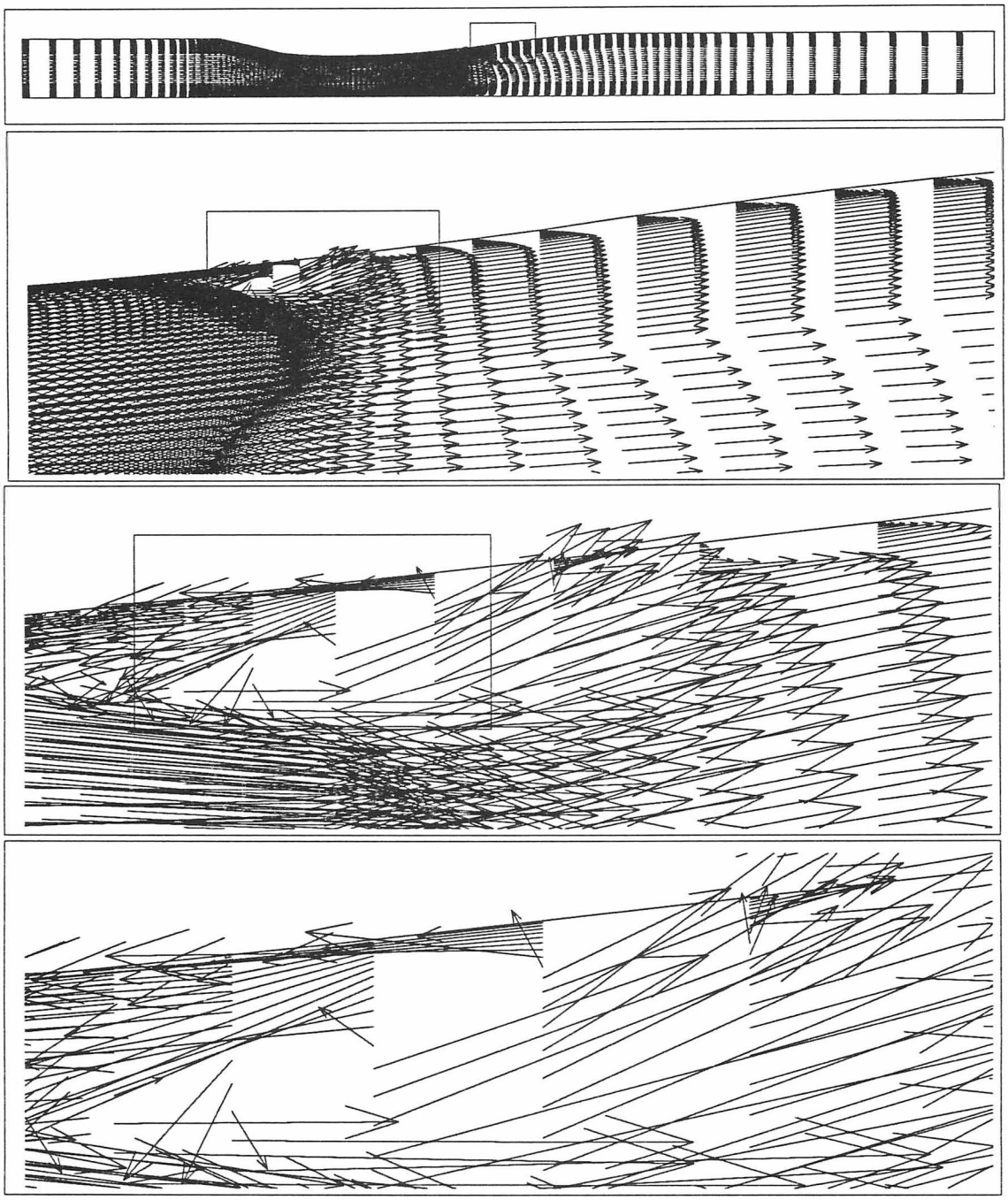


Figure 8.50: Laminar-Viscous Solution of the Weak-Shock Transonic Wind Tunnel Problem Shown as Velocity Vectors in Physical Space; Shock Mach Number $\approx$ 1.235, Reynolds Number 10^3; Magnifications increase from top to bottom as defined by box.

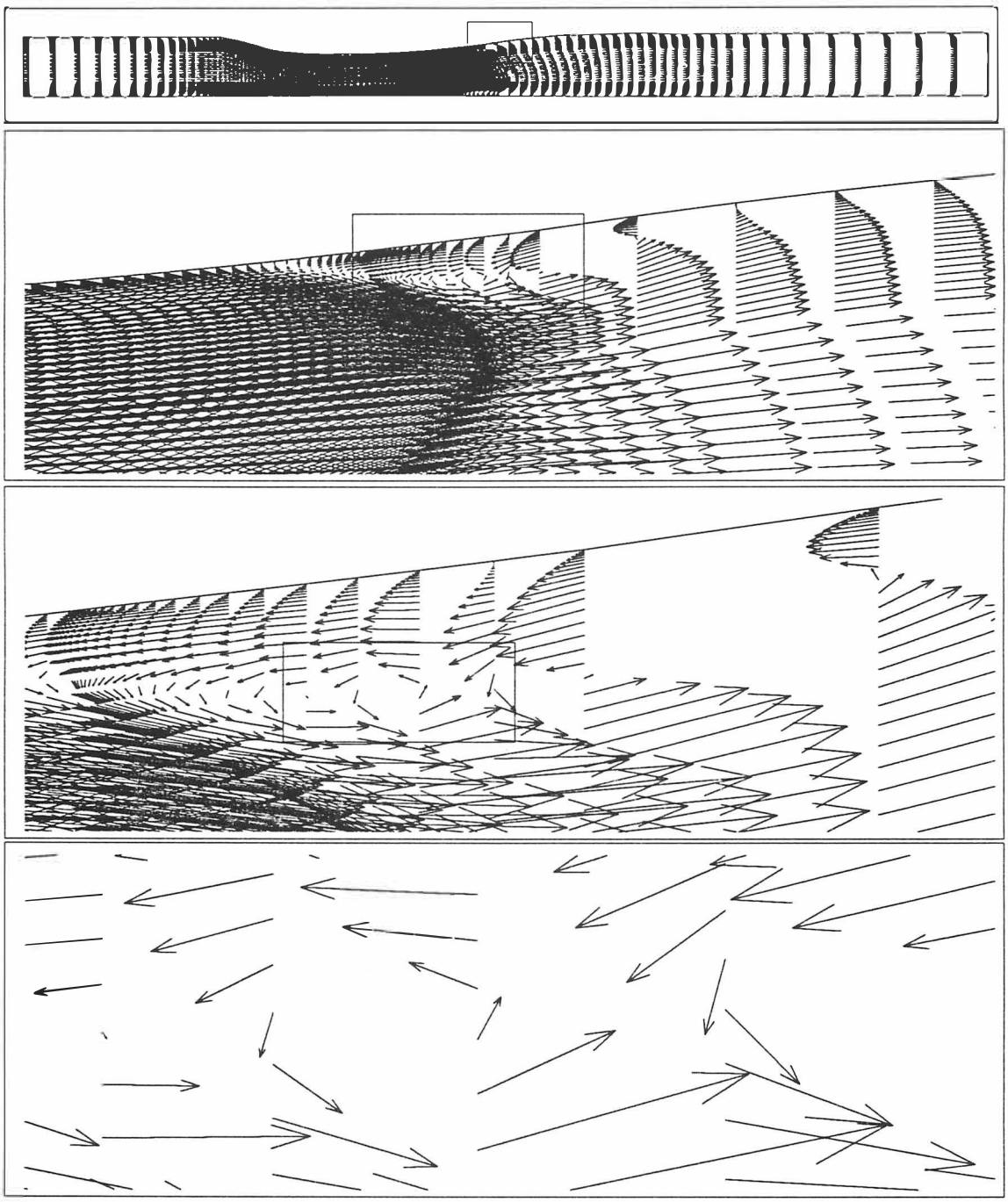


Figure 8.51: Laminar-Viscous Solution of the Strong-Shock Transonic Wind Tunnel Problem Shown as Velocity Vectors in Physical Space; Shock Mach Number $\approx$ 1.353, Reynolds Number 10^3; Magnifications increase from top to bottom as defined by box.

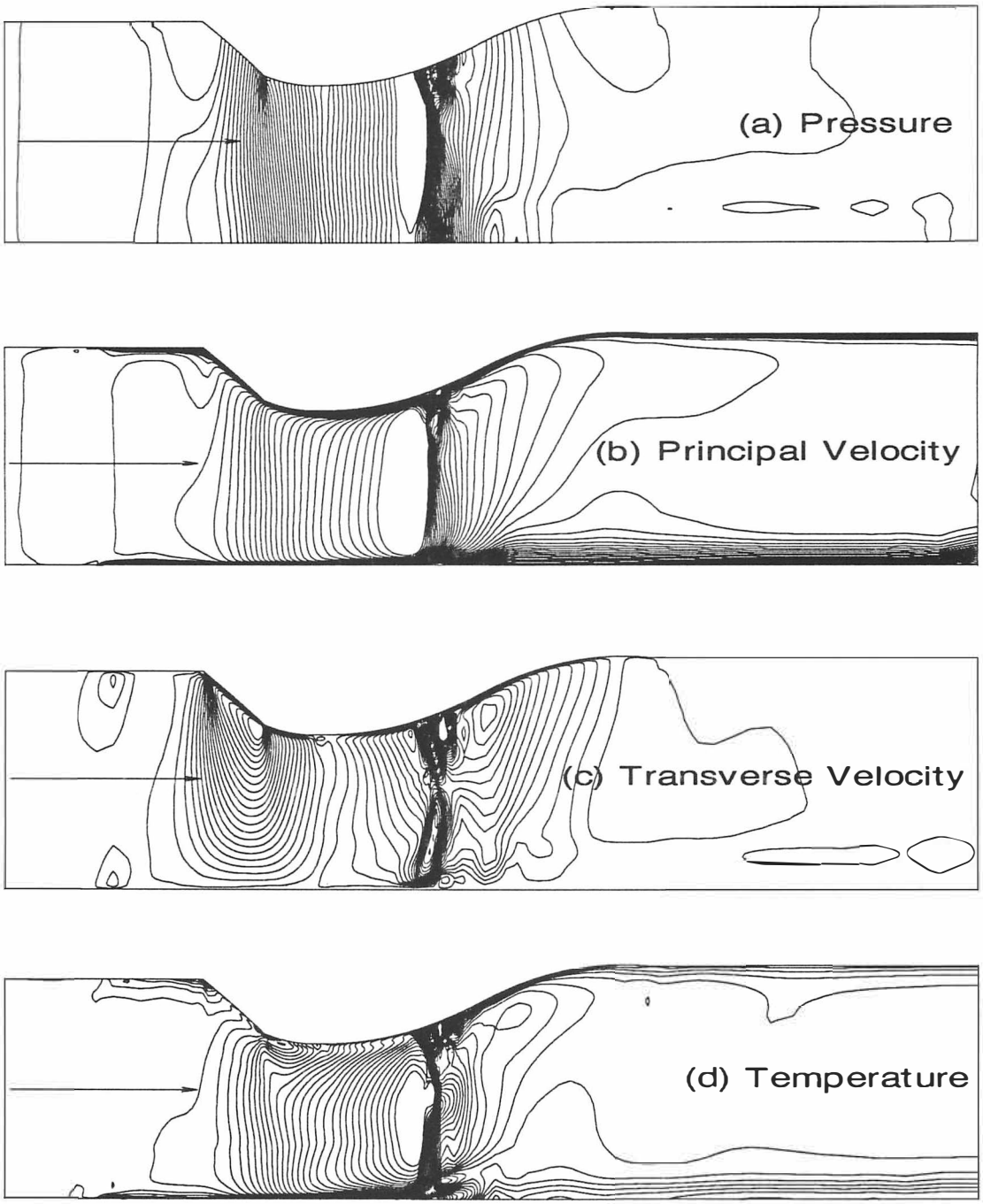


Figure 8.52: Turbulent-Viscous Solution of the Weak-Shock Wind Tunnel Problem Shown as Iso-Value Contours in Physical Space; Shock Mach Number $\approx$ 1.235, Reynolds Number 10^5; (a) Pressure, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature (vertical scale expanded for clarity).

for this length duct. Thus, an oscillation occurs across the last two η_1 grid planes of the mesh, as confirmed in the exit region of the principal velocity, part (b) of the figure.

The nodal-space surface plots of the state variable shown in Figures 8.53 and 8.54 also appear very similar to the laminar counterparts, Figures 8.45 and 8.46. Closer examination of the turbulent nodal plots reveals increased oscillation in the shock boundary layer region on the upper wall surface, most notable in the density and stagnation enthalpy, parts (a) and (b) of Figure 8.53. Also along the both walls, but particularly the upper wall, the turbulent velocity profiles are visible, although not as pronounced as expected. The most significant visible velocity profile change due to the turbulence effects is in the transverse momentum surface plots part (d). In both views of the transverse momentum approximate solution the shape of the peak has changed from the laminar case. However, in general, the turbulence influence upon the weak-shock case is minimal, according to the calculated results, which is not confirmed by the experiment. This is attributed to the relatively poor initial conditions provided by the laminar weak-shock solution, particularly in the shock-boundary layer region as seen by the vector plots of Figure 8.50, which has further been attributed to the lack of grid resolution in that region. Thus, it is concluded that simply increasing dissipation alone, via turbulence, will not always provide a smoother solution, i.e., an adequate grid resolution must also be present.

The effect of turbulence is significantly greater in the strong-shock case. The turbulent solution is shown as iso contour plots in physical space by Figure 8.55. All variables show increased diffusion across nearly the entire wind tunnel cross section. As discussed in the experimental data, the turbulence is "triggered" downstream of the wind tunnel throat by an external device on the top wall only. To simulate the trigger, the algebraic turbulence model eddy viscosity

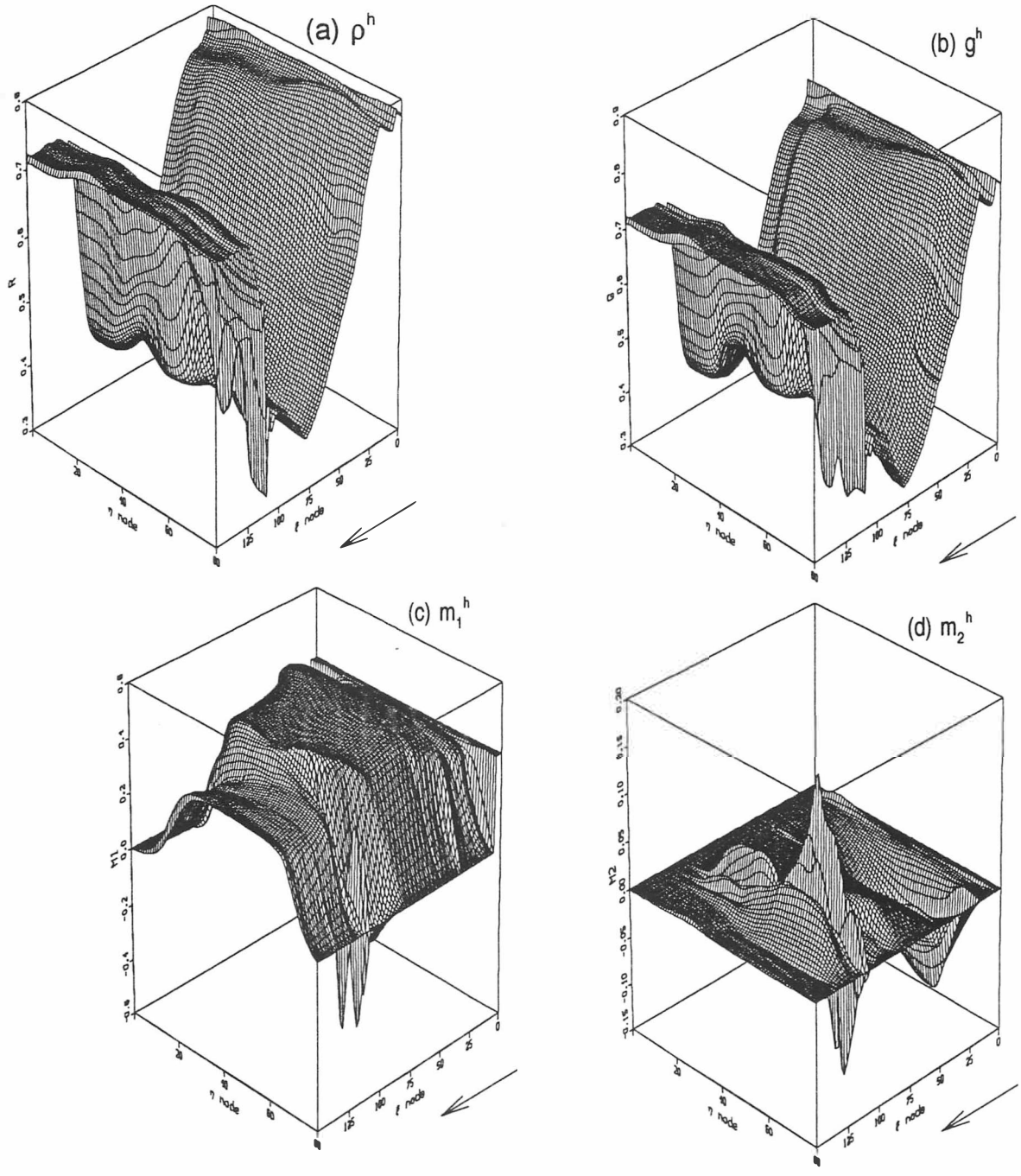


Figure 8.53: Turbulent-Viscous Solution of the Weak-Shock Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Top Exit; Shock Mach Number 1.235, Reynolds Number 10^5; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

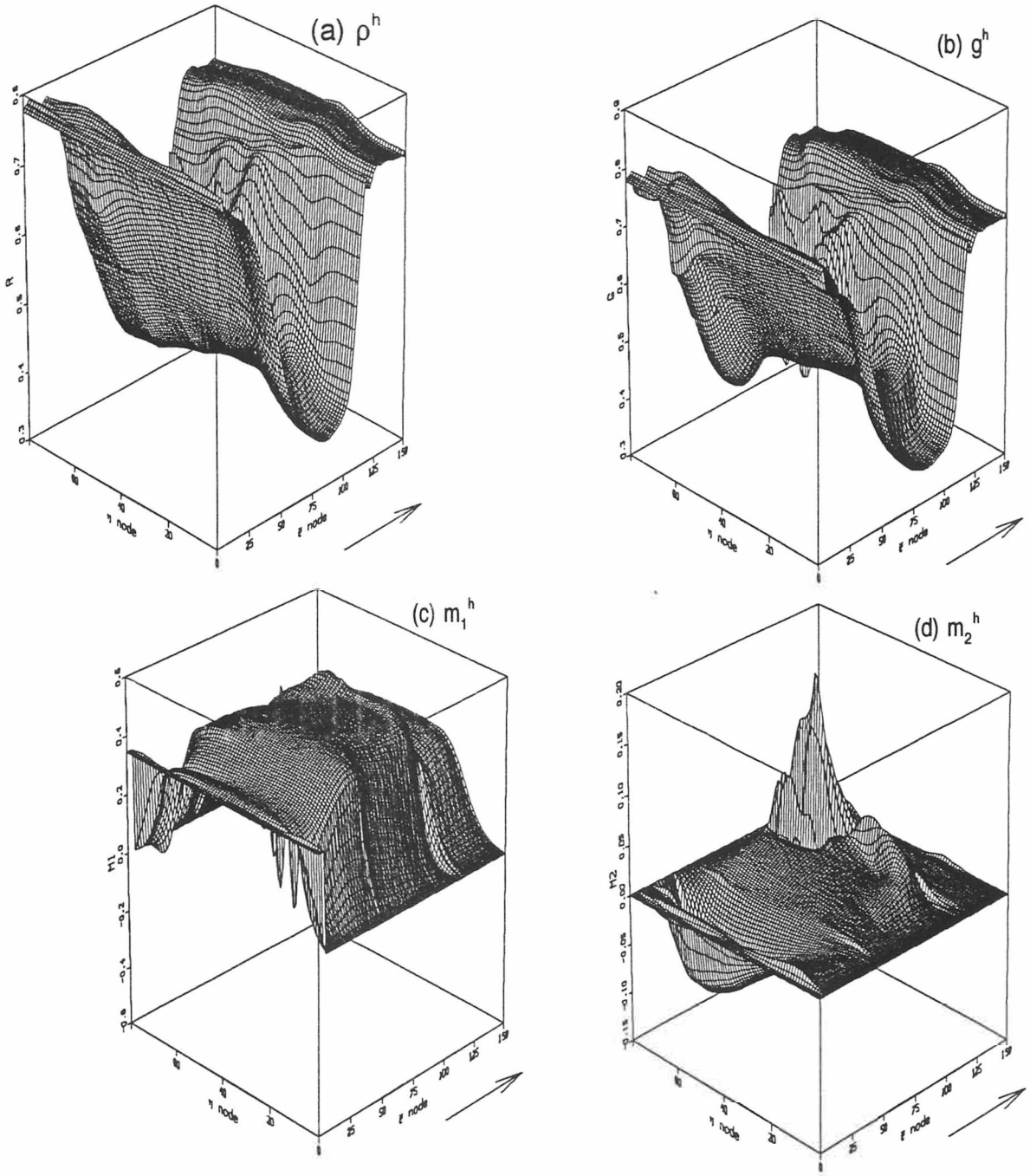


Figure 8.54: Turbulent-Viscous Solution of the Weak-Shock Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Bottom Entrance; Shock Mach Number 1.235, Reynolds Number 10^5; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

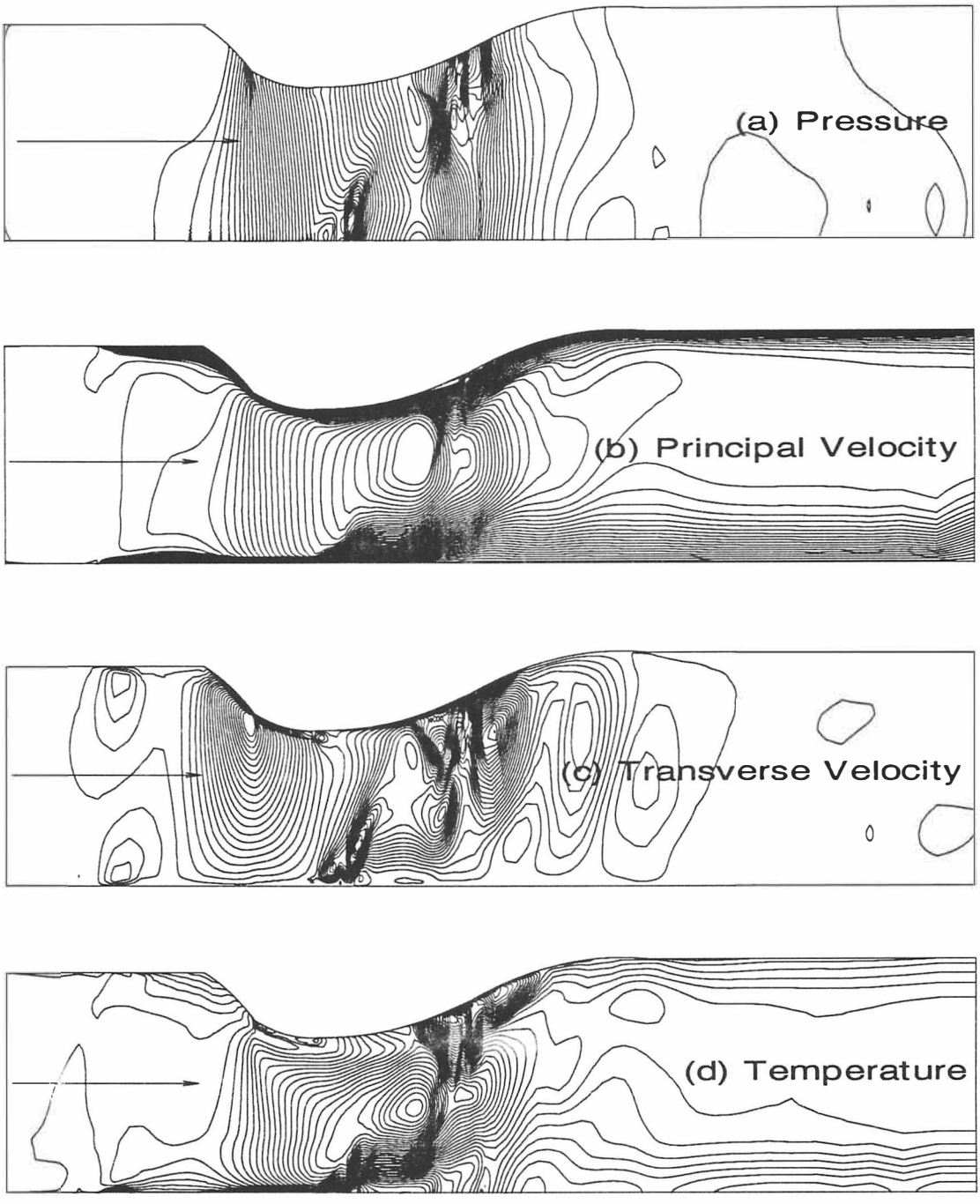


Figure 8.55: Turbulent-Viscous Solution of the Strong-Shock Wind Tunnel Problem Shown as Iso-Value Contours in Physical Space; Shock Mach Number $\approx$ 1.353, Reynolds Number 10^5; (a) Pressure, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature (vertical scale expanded for clarity).

is not computed upstream of the wind tunnel throat as stated. The transition from laminar to turbulent flow is therefore clearly visible in increased contour densities, hence increased boundary layer thickness, on both upper and lower walls downstream of the throat. The increased viscous dissipation caused by the turbulence also causes an increased pressure change across the wind tunnel downstream of the throat. Similar to the converging nozzle problem discussed in the previous section, a pressure gradient is present to the exit plane due to the viscous shear losses. Therefore, a conflict exists between the assumed vanishing derivative boundary condition, and the equation of state, hence a non-vanishing derivative is present at the outflow boundary. Since application of nonzero gradient outflow boundary condition has clearly been demonstrated in the previous converging duct problem and the GAMM nozzle problem in the next section, this transonic wind tunnel solution was not re-calculated to correct this deficiency. In fact, it is more valuable for future reference to see how a deficient outflow boundary conditions can be detected.

The level of genuine dissipation (laminar plus turbulent) in this problem is so significant that the boundary layers of both surfaces 3 and 4 extend to nearly half the cross section at the wind tunnel exit plane. The shock is still visible, but the physical dissipation has significantly diffused it across a wider region in the diverging section. The level of dissipation produced by the turbulence model in this computation is consistent with the experimental data, since the computed displacement thickness is close to the given tabulated values. However, the experiment does not indicate that the shock is diffused as much as in this simulation. Therefore, the algebraic turbulence model (Baldwin-Lomax or Cebeci-Smith) does not appear sufficiently accurate to predict these details of turbulent boundary layer/shock interaction. This is not surprising, since neither model has been proposed to accomplish this. However, the objective of this

research project has been met, the goal being to demonstrate the ability of the algorithm to incorporate a turbulence model, with sufficient mesh refinement, for flowfield resolution.

The nodal-space surface carpet plots of the state variable are shown in Figures 8.56 and 8.57. The boundary layer thickness and shape caused by the turbulence is clearly visible in all plots. As before, the problematic area is near the upper wall at the shock intersection, node line $\eta_1 = \xi \approx 125$ on Figure 8.56. The start of the turbulence region is visible at $\eta_1 = \xi \approx 46$, where the throat begins, hence the principal momentum (part (c) of Figure 8.56) boundary layer profile changes from laminar to turbulent. The thinner boundary layer thickness in the throat area is also more apparent in this turbulent case. Mesh resolution encompassing the recirculation region is much improved, with approximately thirty η_1 grid lines and nearly a fourth of the η_2 gridlines present therein. The turbulence has so diffused the shock that the peak principal momentum, in the predominantly inviscid region, has been reduced from that of the laminar and inviscid simulations. The outflow boundary continues to pose a conflict with the imposed inaccurate vanishing derivative, but this has a local effect only, and does not affect the flow pattern in the shock region. The shock is again clearly captured, as shown in the density and enthalpy surface plots (parts (a) and (b)) of Figure 8.56, by 4-6 node rows across the shock, even within the turbulent-diffused recirculation region. It is difficult to explain all the surface orientations of transverse momentum, part (d), except to note that all regions of majority negative or positive sign, prior to the shock, are also regions of tunnel area decrease and increase respectively. Along the bottom wall, Figure 8.57, the turbulent model is also enabled but is not as influential on the flowfield. All surfaces coming away from the bottom wall appear very smooth, with the only complications arising from the upper-wall influence.

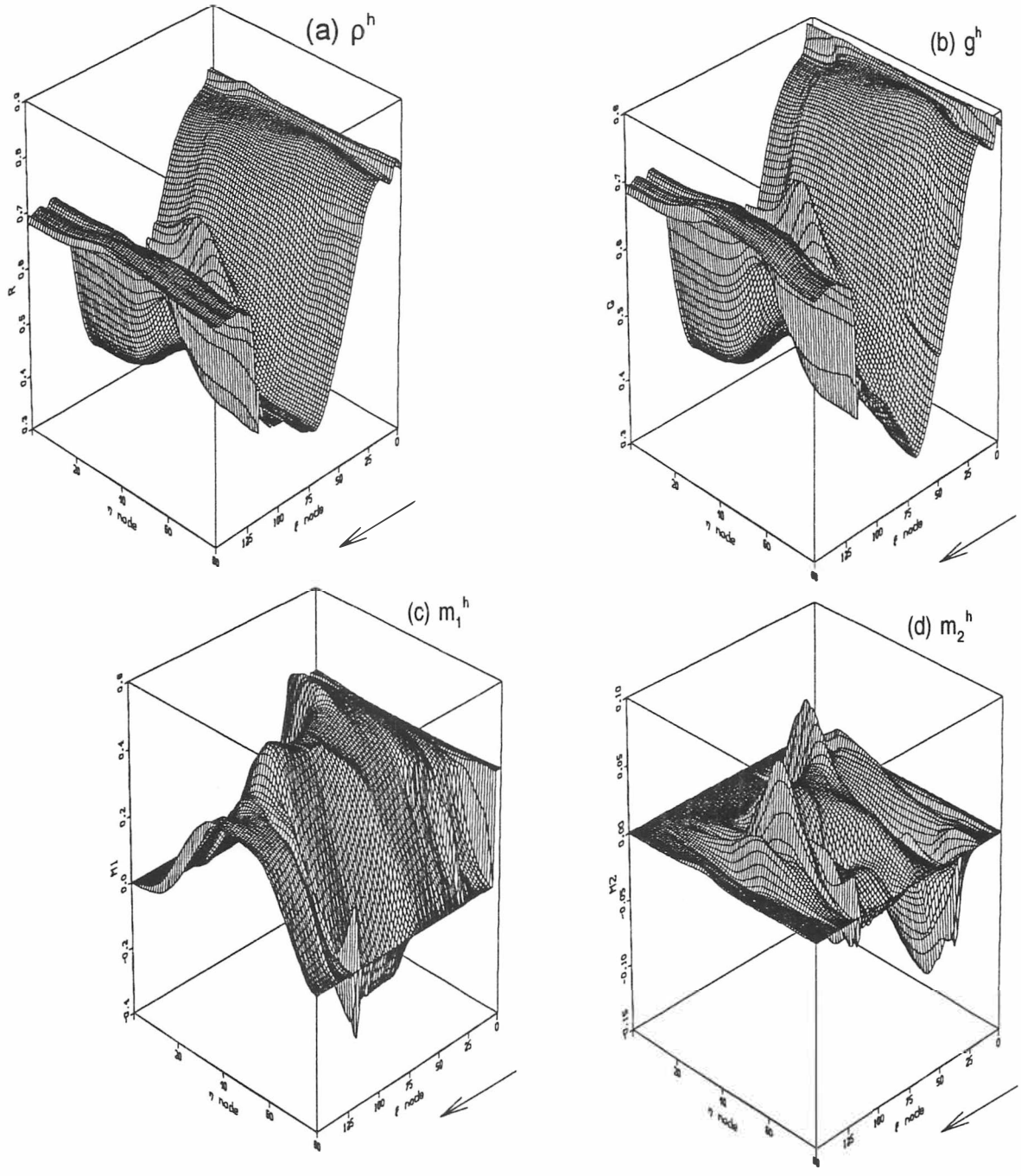


Figure 8.56: Turbulent-Viscous Solution of the Strong-Shock Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Top Exit; Shock Mach Number 1.353, Reynolds Number 10^5; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

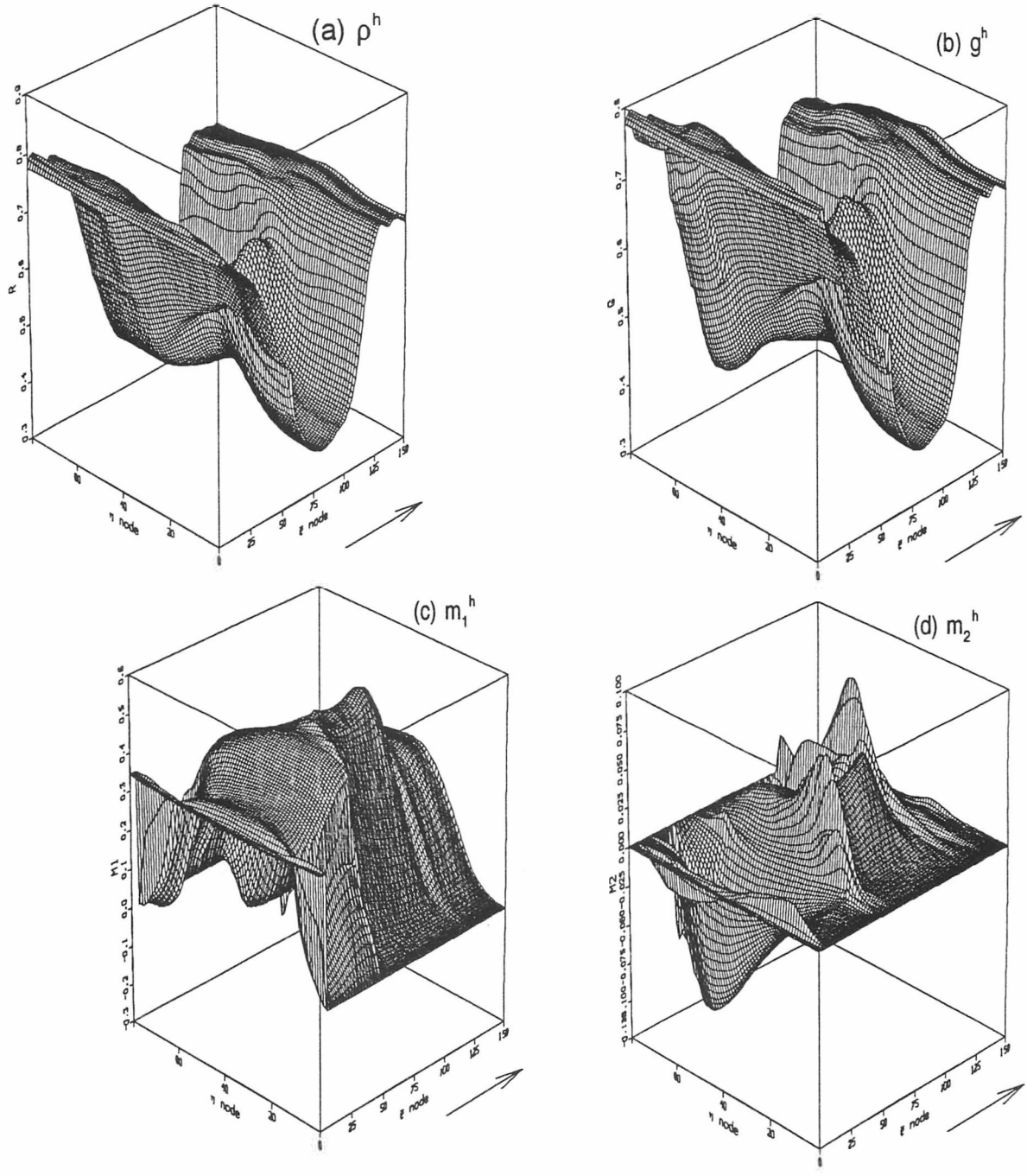


Figure 8.57: Turbulent-Viscous Solution of the Strong-Shock Wind Tunnel Problem Plotted in Nodal Space as Viewed from the Bottom Entrance; Shock Mach Number 1.353, Reynolds Number 10^5; (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

Vector plots of the weak-shock and strong-shock turbulent cases, similar to those shown previously for the laminar cases, are shown in Figures 8.58 and 8.59. Both the weak-shock and strong-shock simulations were completed at Reynolds number = 10^5. The weak-shock solution field appears more oscillatory, which may be attributed to the relatively poor initial conditions generated by the laminar solution for the turbulence model. Indeed, the weak-shock case turbulence does not appear to fully develop. Both simulations recover to a stable boundary layer profile downstream of the shock, particularly the strong-shock case. The turbulent, double-slope boundary layer velocity profiles are clearly visible along each streamwise grid line in the strong-shock case, Figure 8.59.

Based on this study, it is clearly important to produce stable laminar solutions before attempting to generate the turbulent counterpart. Also, an accurately placed mesh refinement is important to solution stability. An improvement in the relatively crude mesh adaptation procedure employed herein would certainly be welcome.

Comparison with Experimental Data

The raw data summarized in the referenced technical papers were not available. Therefore, truly detailed comparisons, could not be performed. However, there exists sufficient tabulated data to allow for some global comparisons. Table 8.9 compares experimental and calculated results for key global measures for both the weak- and strong-shock simulations. The Cebeci-Smith algebraic turbulence model computes displacement thickness, hence was expanded to compute the momentum thickness for the comparison. All other comparison quantities are obtained directly from the computation.

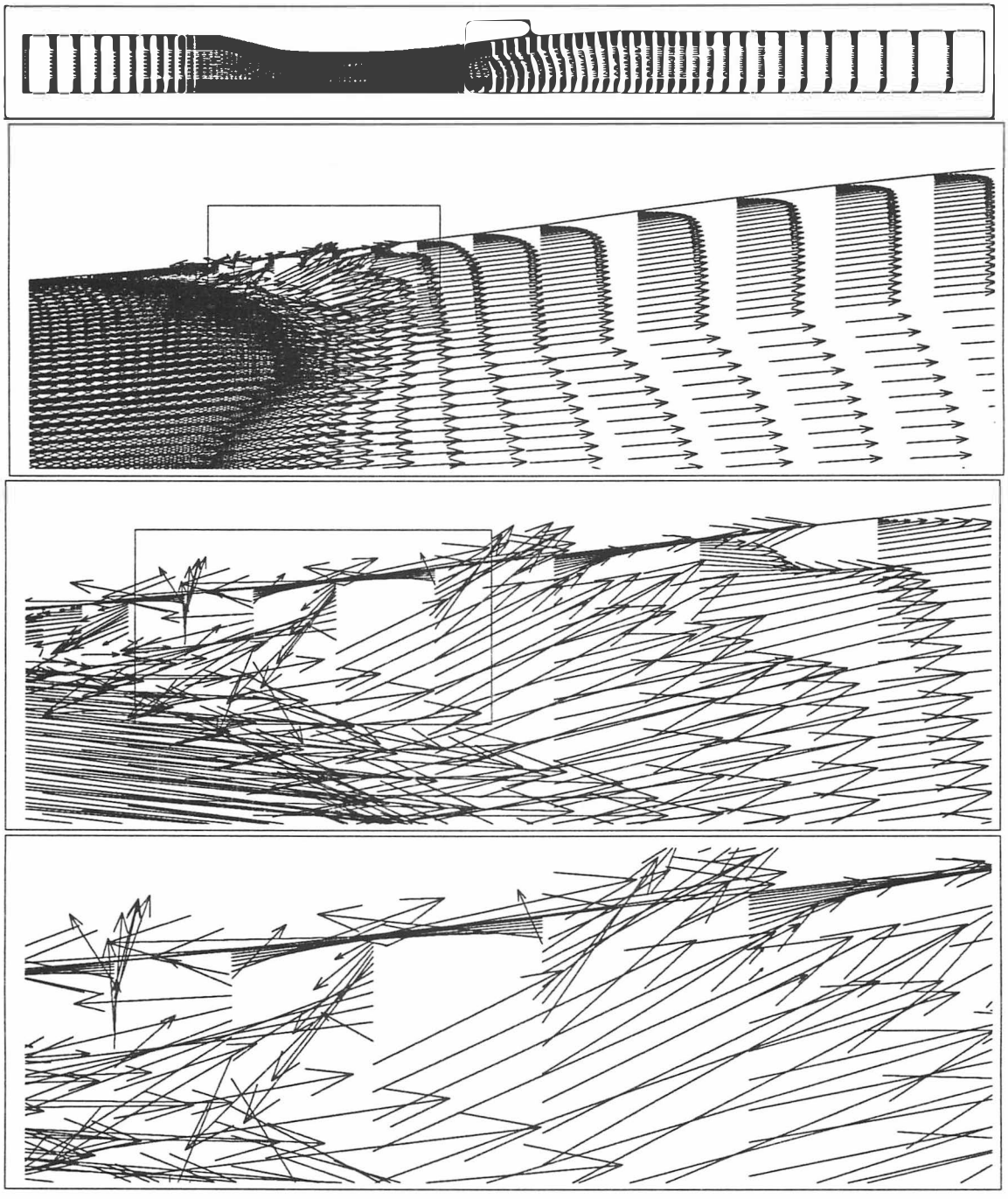


Figure 8.58: Turbulent-Viscous Solution of the Weak-Shock Wind Tunnel Problem Shown as Velocity Vectors in Physical Space; Shock Mach Number ≈ 1.235, Reynolds Number 10^5; Magnifications increase from top to bottom as defined by box.

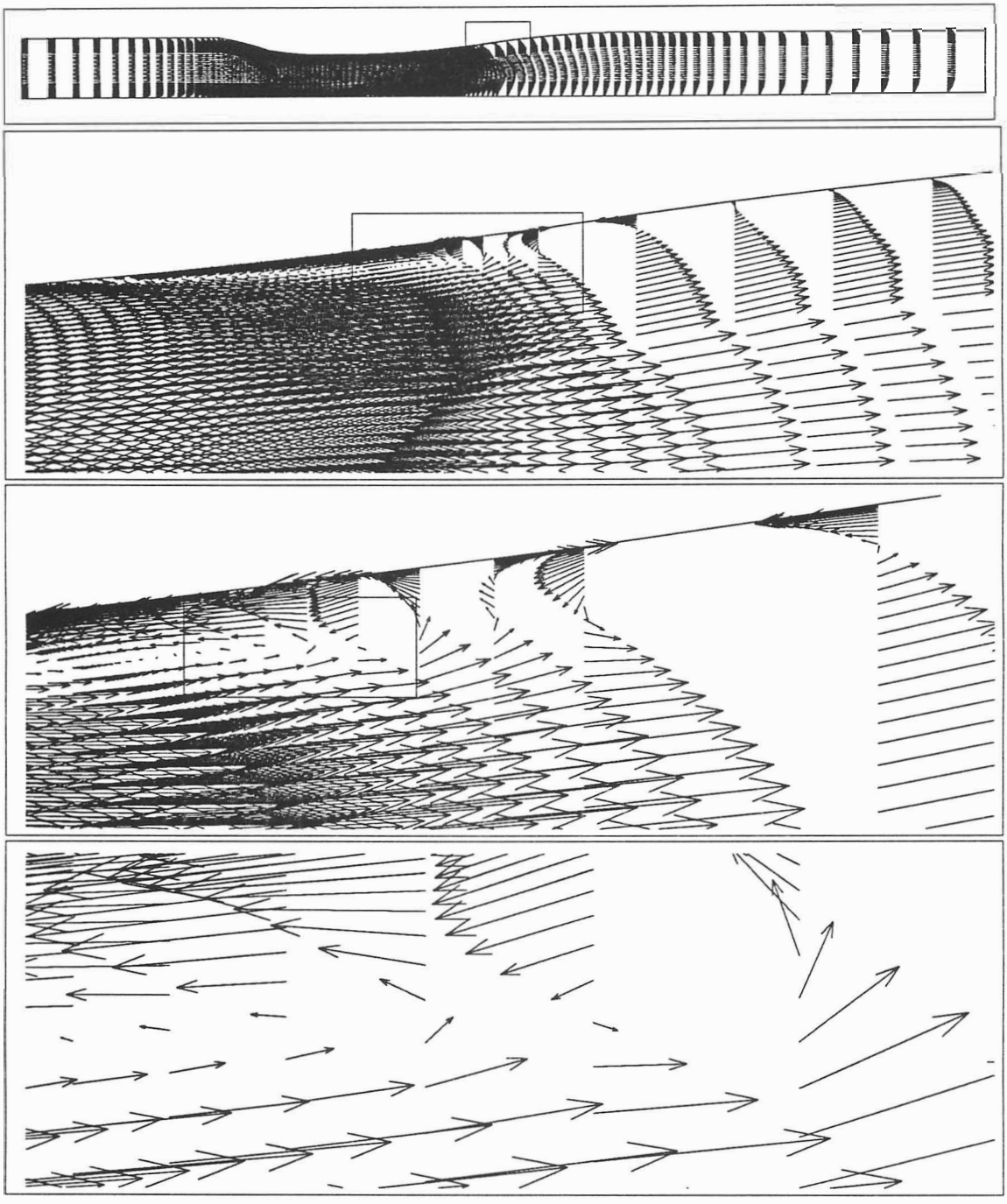


Figure 8.59: Turbulent-Viscous Solution of the Strong-Shock Wind Tunnel Problem Shown as Velocity Vectors in Physical Space; Shock Mach Number ≈ 1.353, Reynolds Number 10^5; Magnifications increase from top to bottom as defined by box.

Table 8.9: Representative Comparison of Calculated Results with Experimental Data for the Transonic Wind Tunnel Problem Top Wall (Surface 4)

quantity	weak shock		strong shock	
(dimensionless)	experiment	calculated	experiment	calculated
Shock Position	.22	.30	.34	.40
Exit Mach Number	.51	.45	.62	.47
Displacement Thickness				
Inlet	.017	.015	.016	.013
Throat	.0065	.025	.0065	.054
Exit	.11	.107	.32	.298
Momentum Thickness				
Inlet	.0105	.010	.011	.0075
Throat	.0035	.0113	.0035	.028
Exit	.07	.0704	.15	.139
Pressure				
Inlet	.74	.74	.74	.74
Throat	.45	.48	.45	.48
Minimum	.34	.32	.27	.28
Exit	.71	.71	.62	.68

The shock position and exit Mach number show significant differences for both the weak- and strong-shock cases. The CFD simulation is initialized by specifying known input flow conditions, and an exit pressure. In the quasi-1D simulation, the TWS β parameter was also modulated for dissipation control determination, and combined with the exit pressure modulation, matched the shock Mach number. The boundary layer bleed ports could not be included in the 2D simulation, therefore, the tunnel mass flow rate differed. The calculation flow rate is less than the experiment by (apparently but not conclusively) approximately 20%. Therefore, it is not surprising that the calculated shock position and exit Mach numbers are different by the indicated amounts.

Both the displacement and momentum thickness comparisons are good everywhere except at the throat. The throat is the region of highest mass flux, and the calculation does show a thinner boundary layer at the throat than upstream, Figures 8.56 and 8.57. The turbulence model is enabled at the throat in the calculation, so perhaps even though the cited turbulence trigger location is not known, the actual start of turbulence could be prior to the throat. Thus, it is clearly plausible that the turbulence model contributes significantly to the discrepancy in the throat boundary layer thickness. The calculation was controlled by changing the Reynolds number to try and match the experimental exit displacement thickness. Therefore, that the inlet and exit displacement and momentum thicknesses should have good comparisons was the design point in the CFD experiment.

The pressure distribution shows a very good comparison at all locations tabulated. The experimental data pressure was referenced to the total inlet pressure. Since the calculation utilized a different reference pressure, the experimental pressure was calibrated to the calculated reference pressure. This allowed the inlet pressures to be exactly the same for both experiment and

calculation. This is valid since the fluid thermodynamic state is input Dirichlet data to the calculation. Figure 8.60 shows the calculated pressure distribution along the upper wall for the turbulent viscous solution. Comparing to Figure

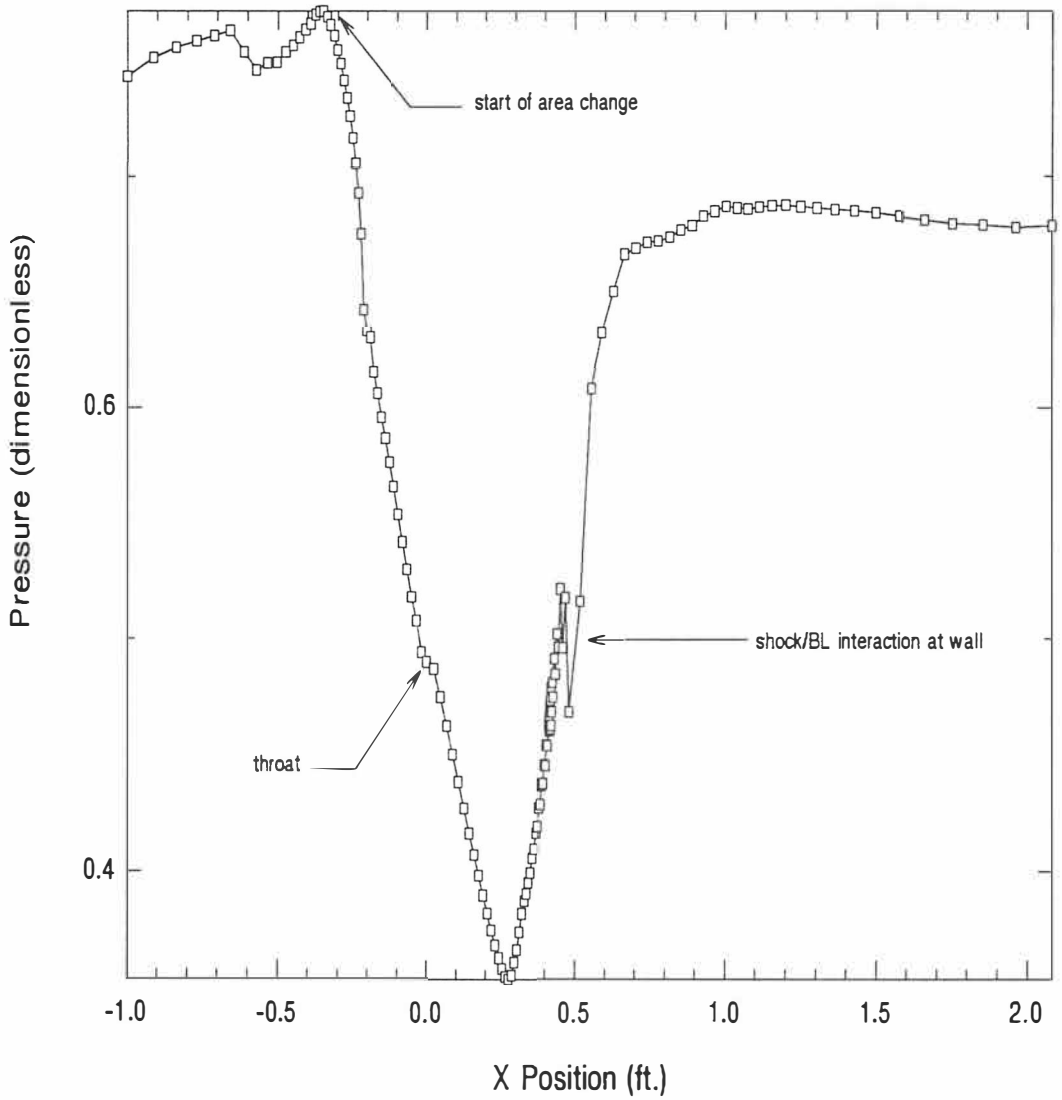


Figure 8.60: Calculated Pressure Distribution Along the Top Wall (Surface 4) for the Turbulent Viscous Solution of the Strong-Shock Transonic Wind Tunnel Problem

8.24, and recalling the difference in reference values, these data show good agreement. The experiment shows a more definitive plateau at the throat and does not indicate any oscillation at the shock boundary layer interaction point. The experiment also indicates a more diffuse pressure distribution everywhere downstream of the shock. It is especially interesting to see the pressure peak at the nozzle entrance region, in both the calculation and the experiment.

Calculated and experimental axial velocity profiles, starting approximately at the shock and proceeding downstream are shown in Figures 8.61 and 8.25 respectively. The abscissa is referenced by the throat cross-section, with the sign reversed in Figure 8.61 for graphics comparison to Figure 8.25. Each figure contains six streamwise stations downstream from the throat, with the left ordinate the top wall and the right the bottom wall. All calculated results show the boundary layer thickness along the bottom wall to be significantly larger and of different shape than the experiment. Along the top wall, the boundary layer profile is also significantly different than the experiment, particularly beyond the recirculation region. Since the calculation has been designed to conserve the displacement and momentum thicknesses (Table 8.9), the shape difference is primarily attributed to the algebraic turbulence model. The comparison for the laminar solution, Figure 8.62, shows the effect of the algebraic turbulence model of the velocity profiles. The velocity magnitude in the essentially inviscid region matches the experiment fairly well on all graphs. The recirculation region in the first three streamwise stations along the upper wall shows good agreement with experiment, indicating the shock location is relatively consistent. The difference in magnitude and shape of the velocity profiles in the recirculation region again can be attributed largely to the algebraic turbulence model. Therefore, aspects of the velocity distributions that are independent of the turbulence model show good agreement.

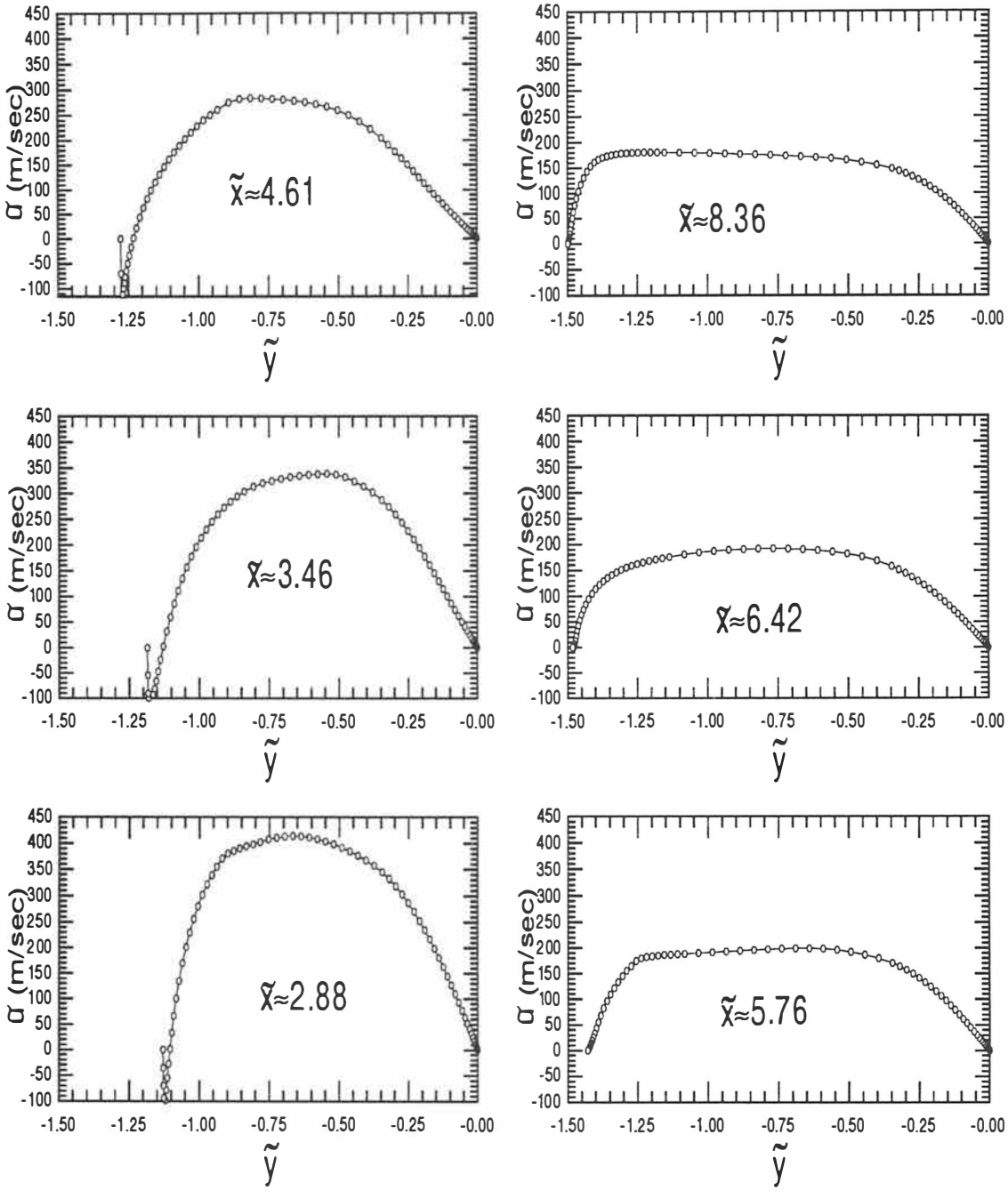


Figure 8.61: Calculated Principal Velocity Profiles Downstream of the Shock for the Turbulent Viscous Solution of the Strong-Shock Transonic Wind Tunnel Problem

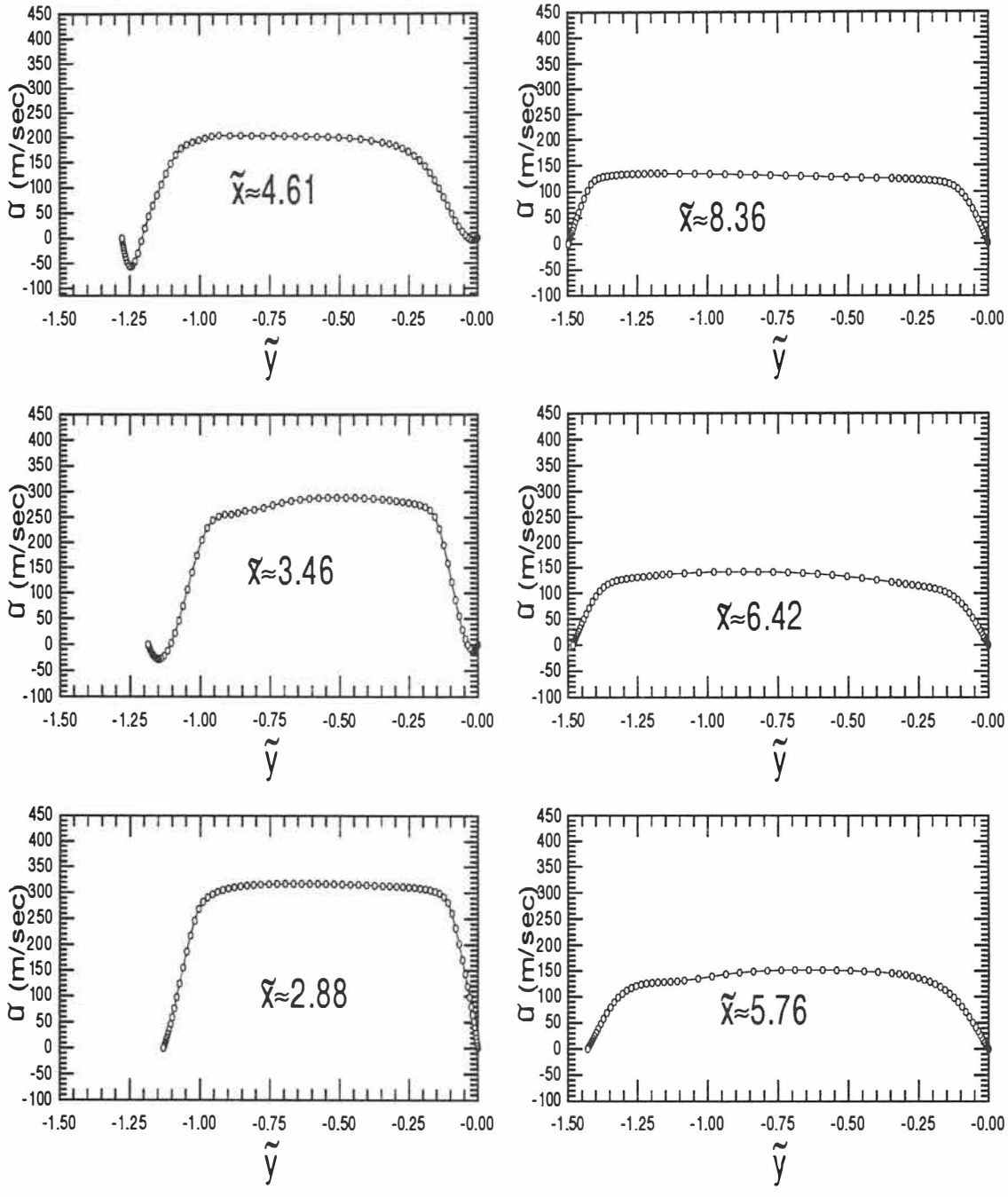


Figure 8.62: Calculated Principal Velocity Profiles Downstream of the Shock for the Laminar Viscous Solution of the Strong-Shock Transonic Wind Tunnel Problem

Comments on Stability

The nonlinear iterative solution procedure, and the associated linear-algebra computations, remained stable for all calculations presented herein. The full jacobian formulation, with all off-diagonal submatrices included, was used for all viscous simulations. Essentially, as long as the time step remains small enough such that the Lyapunov functional, defined as the Euclidean norm of the most current right-hand side residual array, continually decreases with time, then the solution is assured to be stable. The more complicated physics associated with the shock boundary layer interaction and the turbulence problem caused an increased stability constraint, i.e., the maximum allowable time step had to be relatively smaller. However, for the time step carefully increased, while maintaining a decreasing Lyapunov functional, a stable, steady-state solution process is assured.

Real-Gas Considerations

A real- and ideal-gas comparison of the quasi-1D solution Mach number distribution of the weak-shock case is shown in Figure 8.63. The differences are very minor, therefore, the simplifying effect of ideal-gas thermodynamics would not significantly affect there results. Nevertheless, since all calculations in this section have been compared to experiments, real-gas thermodynamics have been utilized throughout.

Summary

The derived finite element algorithm has been conclusively demonstrated to accurately capture transonic flow shocks in 2D via the TWS β term. The magnitude of β was calibrated in 1D and extended to 2D without change

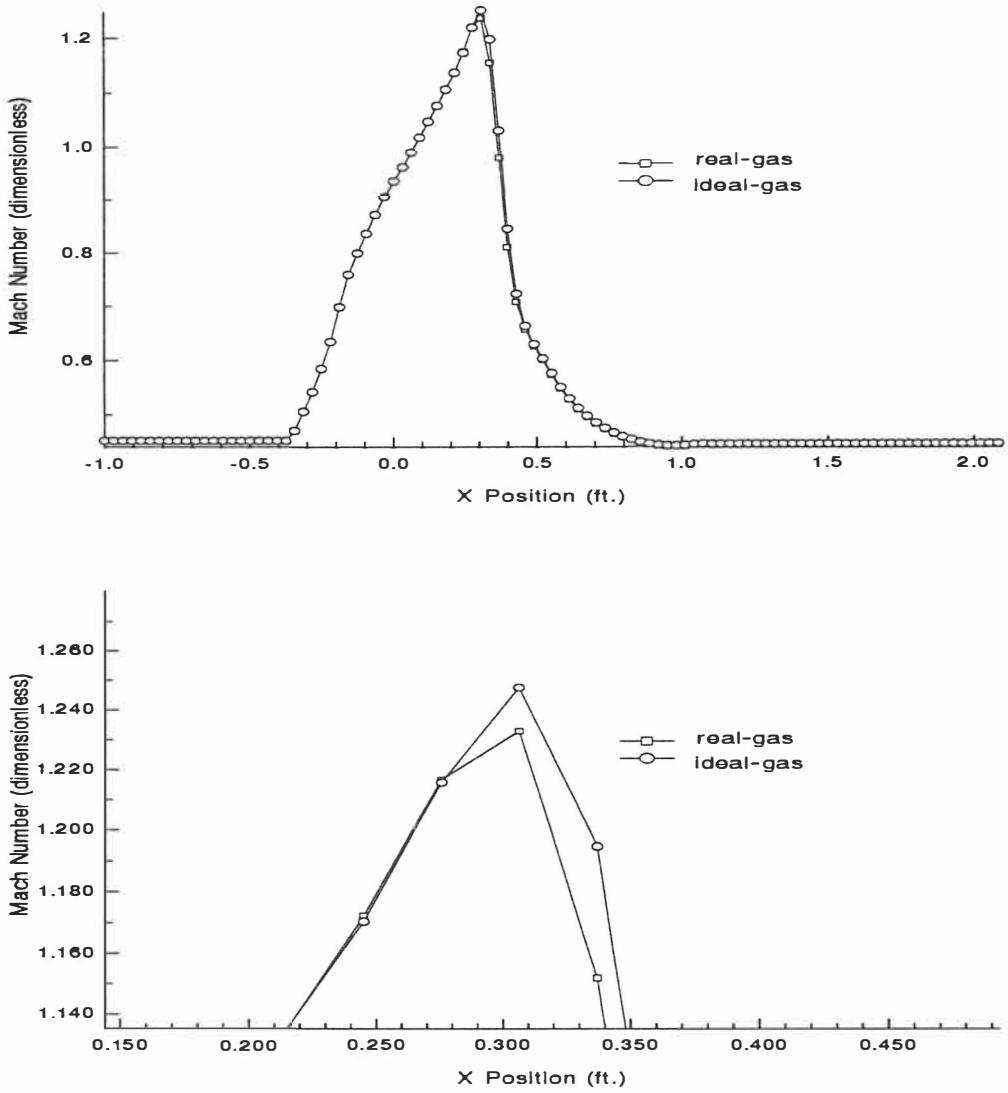


Figure 8.63: Comparison Between Real- and Ideal-Gas Equations of State for the Quasi-1D Solution of the Weak-Shock Transonic Wind Tunnel Problem

for application in inviscid flow regions. Furthermore, β is smoothly removed from flowfield viscous regions, hence natural viscous dissipation was present for accurate shock boundary layer interaction simulation. The algorithm results have yielded solutions consistent with data for shock position and magnitude, pressure distribution, and the issues of shock-boundary layer recirculation. The selected algebraic turbulence models are properly incorporated into the algorithm, but are not sufficient to accurately simulate all turbulence-induced affects for this transonic internal-flow problem. The adjustment of turbulent-model length scale is not sufficient to accurately simulate the turbulent boundary layers in a geometry different from the originally-derived model application. The algorithm and solution accuracy are significantly affected by the refinement and proper placement of the mesh.

8.3 GAMM Double-Throat Nozzle

The final problem assessing algorithm performance is the double-throat nozzle defined as a GAMM workshop CFD benchmark.[131] The problem was specifically designed to critically assess shock capturing and outflow boundary conditions. Hence, the CFD design calculations determine performance characteristics of a prototype nozzle. Thus, much like a production environment, the verified computational tool is now utilized to perform a detailed CFD experiment, via the computer, to determine design limitations. The workshop proceedings[132] give several computational results for comparison to those developed herein.

In addition to expanding the assessment to a design problem, several other goals and/or new findings were made on this problem. These include

- confirming, and for the first time in this research, examining significantly different results produced using the real-gas and ideal-gas closure models,
- further assess grid resolution requirements and procedures,
- critically validate non-vanishing outflow Neumann boundary condition specifications, and
- bring the computer code into the production environment of the Cornell National Supercomputer Facility (CNSF).

8.3.1 Problem Statement

Figure 8.64 shows the geometry for the GAMM double-throat nozzle (hereafter referred to as GAMM nozzle). The nozzle wall (surface 4) is defined as a function of the axial coordinate x, which scales from -12 to $+14$, with $x = 0$ defined as the location of the first and most narrow throat. The length is

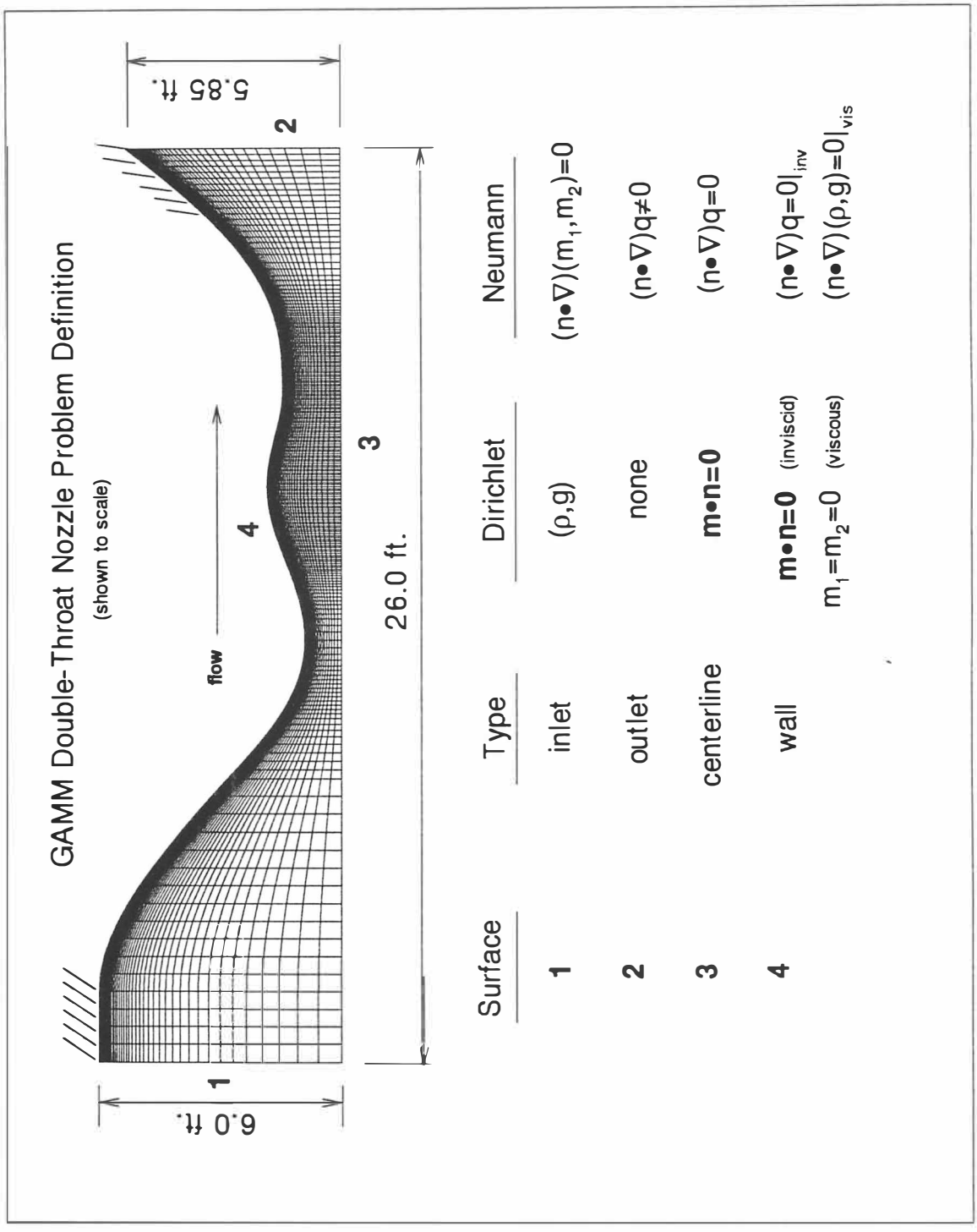


Figure 8.64: GAMM Double-Throat Nozzle Problem Definition

rescaled in the code from zero to +26. Table 8.10 tabulates the functional relationship defining the nozzle wall. The constants in Table 8.10 are truncated to 4 significant digits for clarity.

The length scale enters into the equation system via the Reynolds number and impacts the computation through the viscous boundary layer thickness. Therefore, it is important to know the true geometry of the problem to correctly characterize viscous length scales. This attention is precisely what leads to the consistency of the results with an experiment, as discussed in the previous section. The GAMM nozzle length was not specified in the announcement, requiring a choice of length for the simulation. It is therefore assumed that the throat length is one foot, which yields an overall nozzle length of 26 feet, and allows for direct input of the geometry into the pilot code, built to accept U.S. common engineering units. This is important when comparing to other computations since the Reynolds number, specified 1600 by the GAMM-workshop announcement, also specifies the length scale.

The assumption (by GAMM) that this nozzle exhausts to free space indicates the outflow at surface 2 will be supersonic. Therefore, the flow is

Table 8.10: Functional Relationship of the Wall Surface for the GAMM Nozzle

Arc	x Range	Function
I	$-12 \rightarrow -10$	6.6
II	$-10 \rightarrow -4$	$3.112 - .872(x + 4) + .008074(x + 4)^3$
III	$-4 \rightarrow 2.3$	$1 + .138x^2 - .0085x^3 - .0025x^4$
IV	$2.3 \rightarrow 7$	$1.557 + .3782(x - 2.3) - .0497(x - 2.3)^3 + .0070(x - 2.3)^4$
V	$7 \rightarrow 14$	$1.6 - .001978(x - 7)^4$

choked, and no Dirichlet data may be specified at that boundary. Furthermore, since the inflow at surface 1 is subsonic, fluid thermodynamic properties become selected, via (ρ, g), specified as Dirichlet data. The "reservoir" thermodynamic state is not specified, hence is assumed to be at 100°F and 100 psia. The steady flow rate is ultimately established by flow being choked, yielding a throat velocity of exactly Mach= 1.0 for the one-dimensional isentropic reduction. For the given area ratios, and isentropic air tables,[118] the inlet mass flow rate corresponds to Mach= 0.09030669.

The remaining boundary conditions shown in Figure 8.64 are

- flow tangency along the line of symmetry (surface 3) for both the inviscid and viscous cases,
- flow tangency along the wall (surface 4) for the inviscid case,
- nonslip (zero) velocity along the wall (surface 4) for the viscous cases,
- vanishing normal derivative along the line of symmetry, and the wall formed by surfaces 3 and 4, for all remaining variables, and
- non-zero normal derivative (Neumann) boundary conditions at the outflow (surface 2) for all state variable members.

8.3.2 Geometry Specification

The grid specification for the GAMM nozzle problem follows the discussed approach. From Table 8.10 it is obvious that the wall curvature is highly nonlinear, containing polynomials up to fourth degree. Therefore, a sufficient number of second degree macro elements must be distributed along the wall to capture the wall curvature in sufficient detail. To generate the GAMM nozzle

discretization, a specific FORTRAN program, which embeds the functional form of the GAMM nozzle wall, was written to create the macro-element structure subsequently read by the code. The resulting design for all the grids utilized for solutions contained 26 macro elements spanning the η_1 direction.

Several variations on the number of macro elements in the η_2 direction were investigated as a function of Reynolds numbers. However, for the low Reynolds number (1600) specified in the GAMM workshop announcement, a grid of sufficient density was formed by a single macro element in the η_2 direction for all the results generated herein. Figure 8.64 illustrates the most refined mesh (221×71) for the turbulent viscous simulation of the GAMM nozzle. Finite elements are densely distributed in the region of most interest between the two throats. For the inviscid simulation, several variations of the number of finite elements within each macro element were investigated and are discussed in a later section.

8.3.3 Solution Procedure

The solution procedure for the GAMM nozzle problem follows that developed from the previous problems. The sequence is:

- obtain the steady quasi-1D solution utilizing essential boundary conditions,
- obtain the steady inviscid 2D solution, for specified boundary conditions, and initial conditions from the quasi-1D solution via mapping the state variable onto the mesh,
- obtain the laminar viscous 2D solution utilizing the appropriate boundary conditions, with initial conditions generated from mapping the inviscid 2D solution onto the viscous grid, and specifying velocities of zero, and

- generate the turbulent viscous solution by restarting the laminar viscous solution and initiating the algebraic turbulence model.

8.3.4 Quasi-One Dimensional Flow

Prior to discovering the proper way to implement outflow nonvanishing Neumann boundary conditions, the GAMM nozzle problem was solved assuming an outflow vanishing normal derivative. Figure 8.65 summarizes the steady quasi-1D solution for pressure, temperature, Mach number, and principal velocity. The TWS β term is retained, since it was not known beforehand if a shock might form for the quasi-1D simplification of this problem. As a result, without specific action, a vanishing normal derivative becomes applied at the outflow boundary. In this case, an unrealistic discontinuity was therein predicted. Figure 8.66 shows the comparison solution with outflow boundary conditions for nonvanishing normal derivatives for the state variable. From Chapter 6, recall that nonvanishing derivatives are enforced by setting the appropriate entries in the algorithm assembled matrices to zero. The smooth results in Figure 8.66 confirm that correct Neumann boundary conditions have been enforced at the nozzle exit.

One aspect regarding the quasi-1D simulation is that the Mach number achieves unity at the first throat, and remains supersonic thereafter. The Mach number decreases in the converging region prior to the second throat, as expected, but no normal shock is predicted. In the time history of the quasi-1D solution evolution, a normal shock did initiate downstream of the first throat, and then traveled down the length of the nozzle and out the exit. These results confirmed that, if a normal shock is present in the quasi-1D solution, that is in fact an unconverged 2D or unsteady solution.

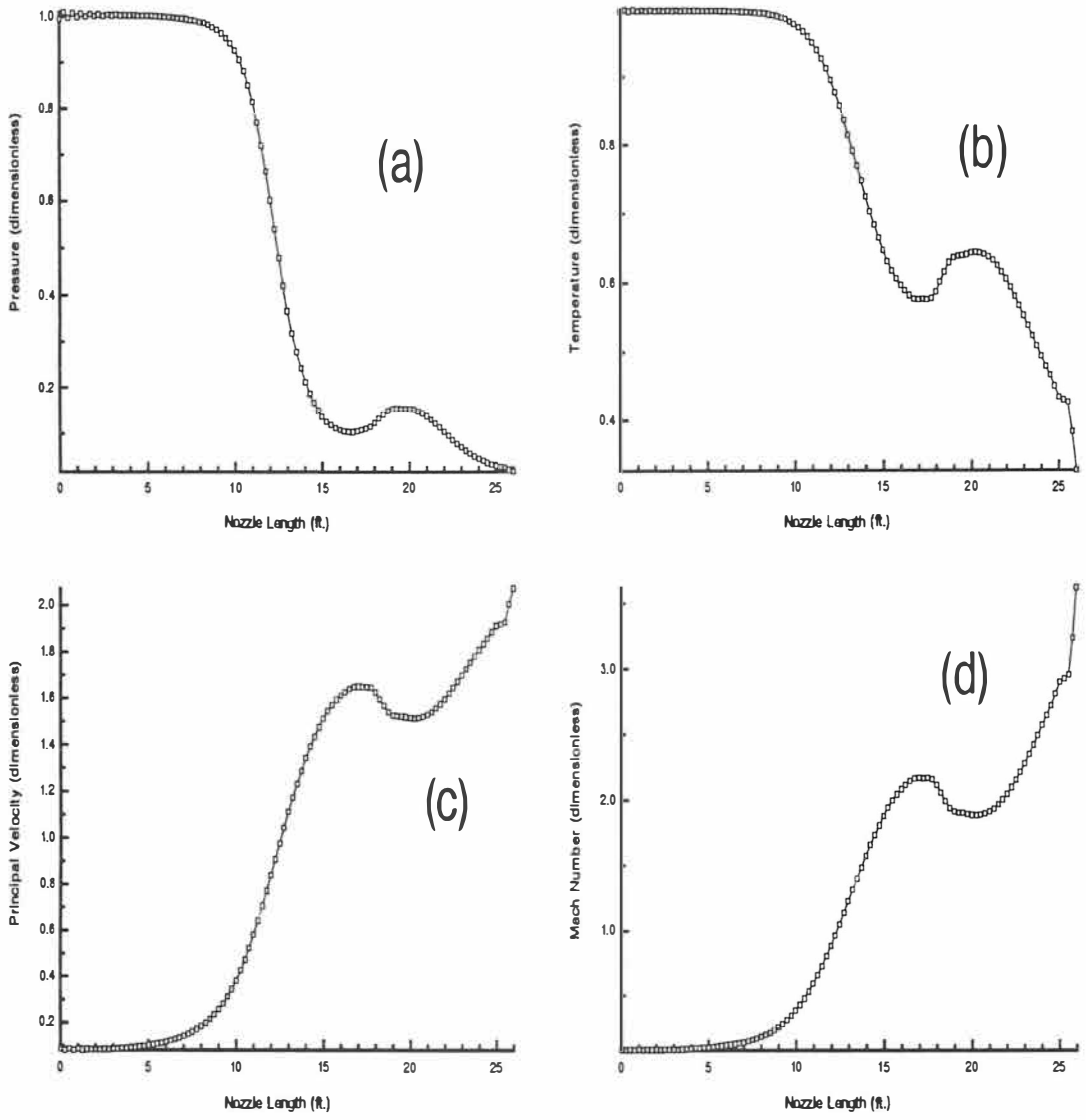


Figure 8.65: GAMM Double-Throat Nozzle Problem Ideal-Gas Quasi-1D Solution; Vanishing Derivative Exit Boundary Conditions; Variables Plotted in Physical Space; (a) Pressure, (b) Temperature, (c) Principal Velocity, (d) Mach Number.

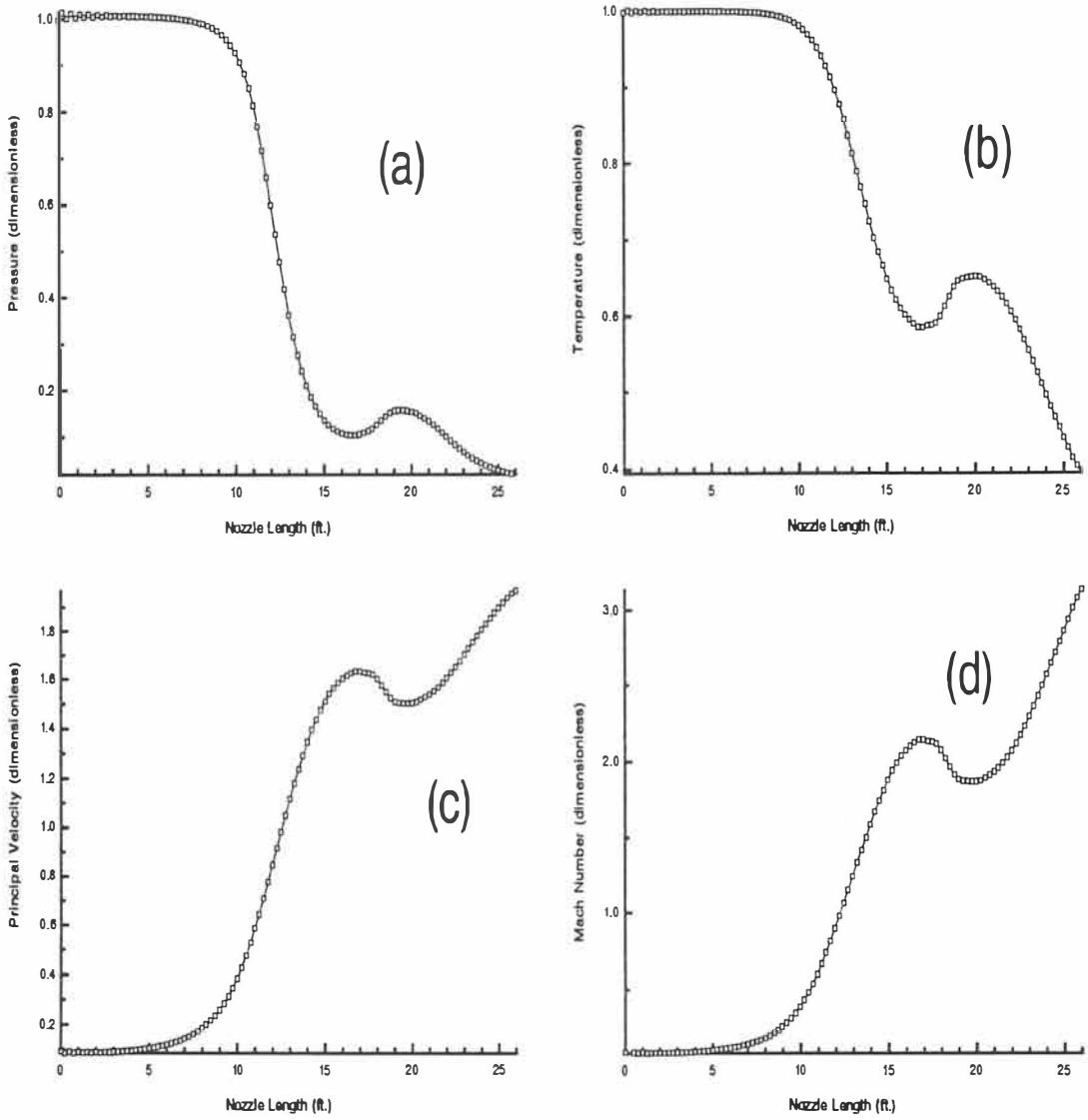


Figure 8.66: GAMM Double-Throat Nozzle Problem Ideal-Gas Quasi-1D Solution; Non-vanishing Derivative Exit Boundary Conditions; Variables Plotted in Physical Space; (a) Pressure, (b) Temperature, (c) Principal Velocity, (d) Mach Number.

8.3.5 Inviscid 2D Flow Simulation

The state variable iso contours in physical space for a first, relatively coarse-grid 2D ideal-gas inviscid solution for the GAMM nozzle are shown in Figure 8.67. All contours are smooth indicating the solution is ENO. However, not apparent from the state variable contours is a computational problem associated with the region of extreme fluid expansion near the outflow intersection of surfaces 2 and 4 (Figure 8.64). Herein, the fluid expands so rapidly that the temperature, hence pressure, drop to zero for the real-gas simulation.

This problem was analyzed further by comparing real-gas and ideal-gas solutions at various grid resolutions. The coarse-grid inviscid temperature solution of the GAMM nozzle shown in nodal space as viewed upstream from the exit plane is shown in Figure 8.68. The top plot is the ideal-gas steady solution, while the bottom contains the real-gas solution at termination. The real-gas solution would not progress further because of the discontinuous equation of state in the exit region of interest.

Since a real-gas inviscid simulation would not yield a steady-state solution, the ideal-gas closure model was utilized for all GAMM nozzle solutions shown herein. A viscous ideal-gas solution restarted from the unsteady real-gas inviscid solution (prior to the equation-of-state corruption in the exit corner) was attempted in hopes that the adiabatic boundary conditions would allow the top exit region to heat up sufficiently, via kinetic energy deposition, to allow for a nonzero fluid temperature, hence realistic thermodynamic state, to exist. The results of this experiment are discussed later. However, the fluid temperature rise was only local, i.e., near the wall, and was not sufficient to keep the problem from occurring, which indicates that the nozzle design is not practical because of this flaw.

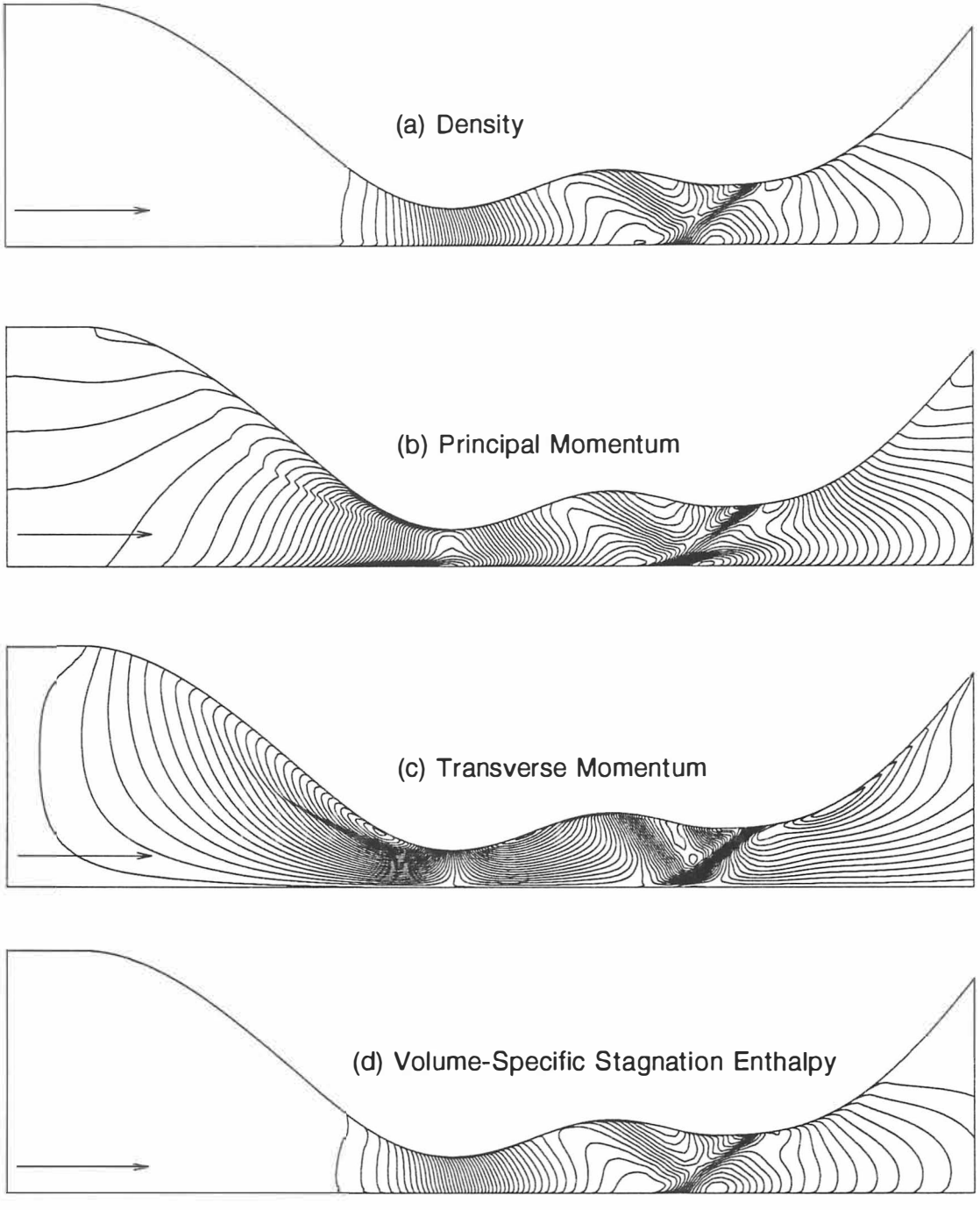


Figure 8.67: Coarse-Grid Ideal-Gas Inviscid Flow Solution of the GAMM Double-Throat Nozzle Problem Shown as Iso Contours in Physical Space; (a) Density, (b) Principal Momentum, (c) Transverse Momentum, and (d) Volume-Specific Stagnation Enthalpy ($\approx$ to scale).

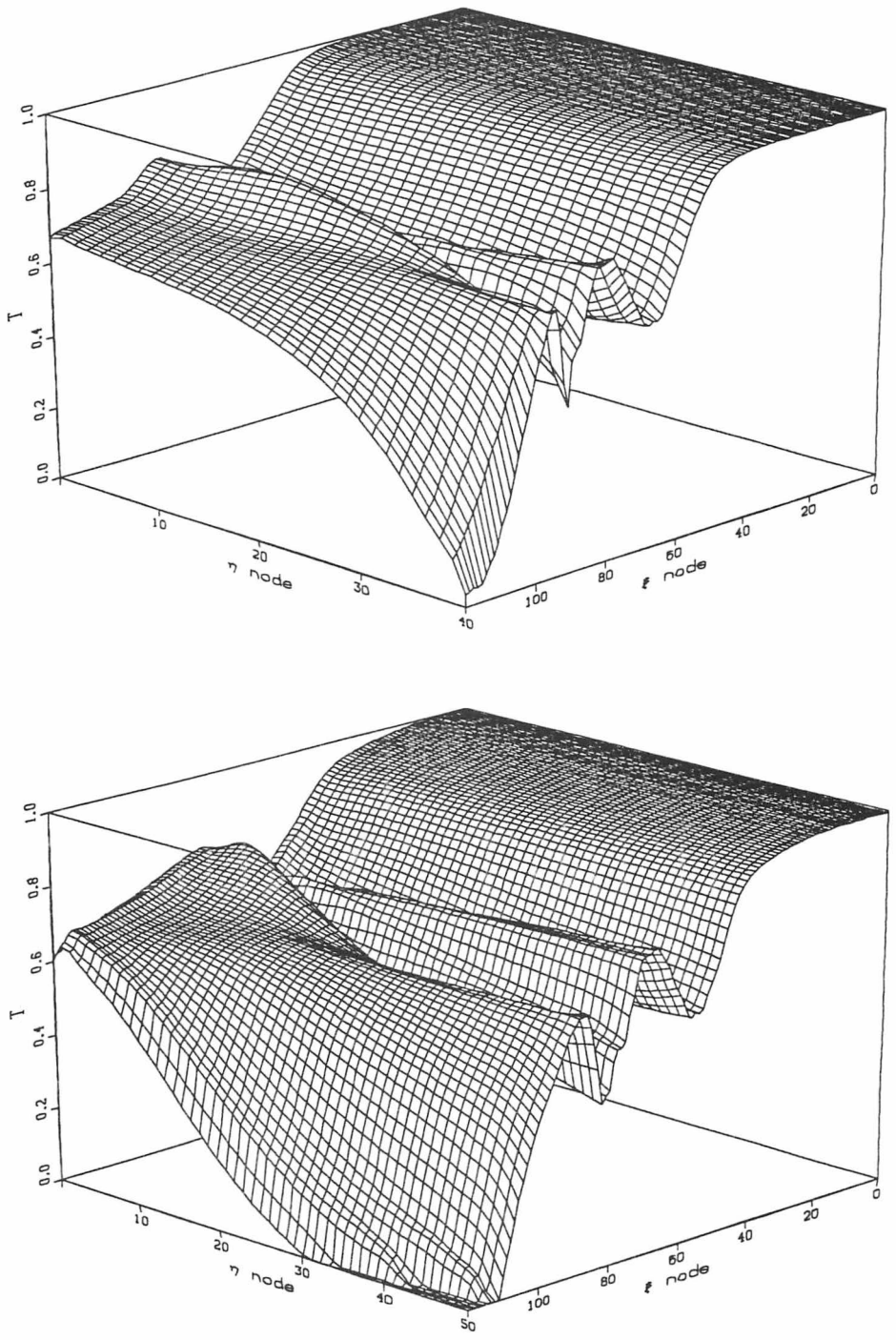


Figure 8.68: Coarse-Grid Inviscid Temperature Solution of the GAMM Double-Throat Nozzle Problem Plotted in Nodal Space as Viewed from the Top Exit. Ideal-Gas (top), Real-Gas (bottom).

Several aspects of the coarse-grid solution of Figure 8.67 point out the need for more mesh refinement in the region between the two throats in order to accurately capture flow details. Furthermore, a refined mesh in the transverse direction reduces the asymmetrical appearance of the solution contours at the centerline. Figure 8.69 shows three mesh refinements for improved solution

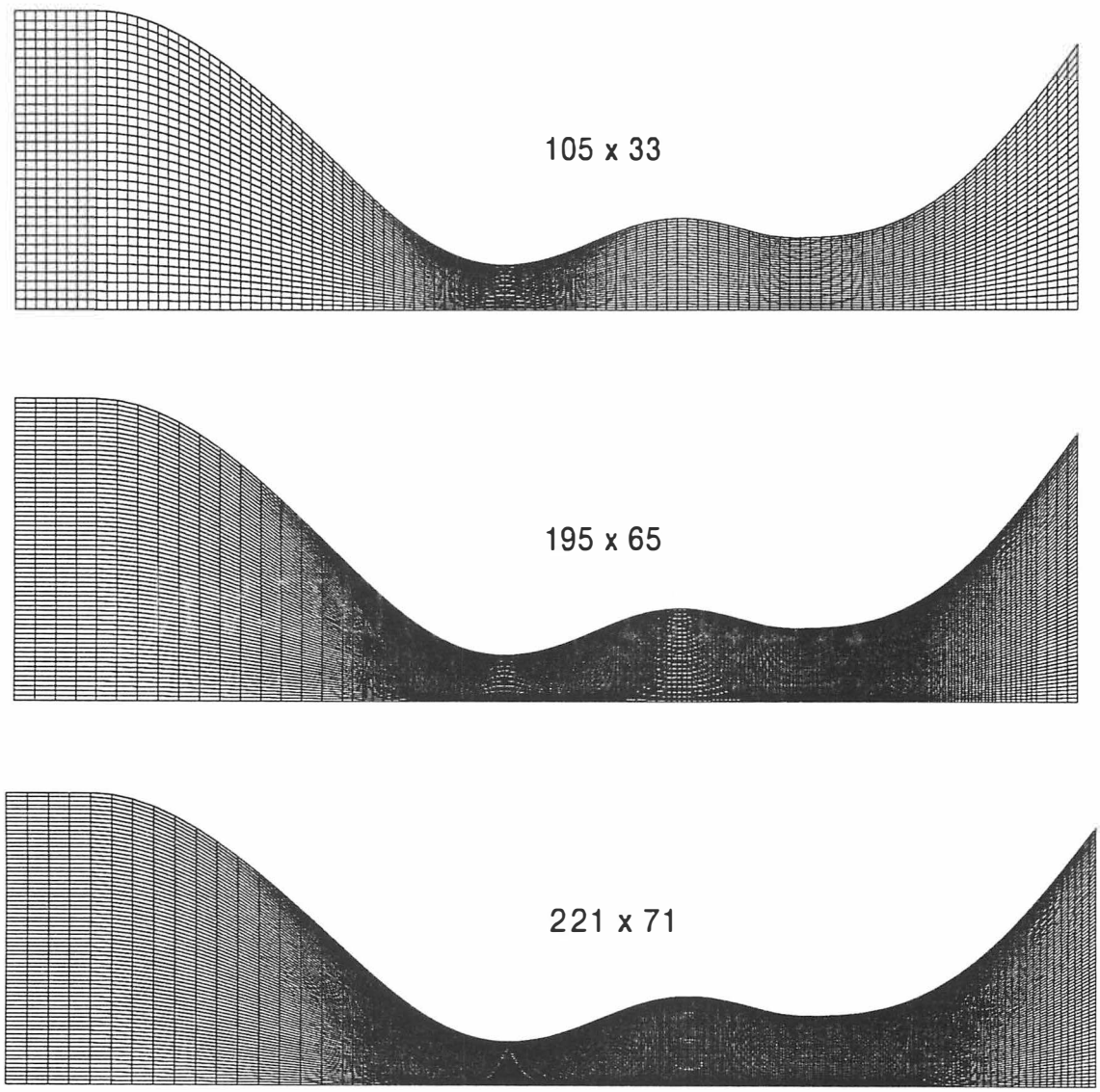


Figure 8.69: Mesh Density Refinement for the GAMM Double-Throat Nozzle Inviscid Problem.

quality. All three grids have a unit progression ratio in the transverse direction across a single macro element.

The resulting inviscid flow pressure isobar distributions corresponding to the mesh designs in Figure 8.69 are shown in Figure 8.70. The main improvement

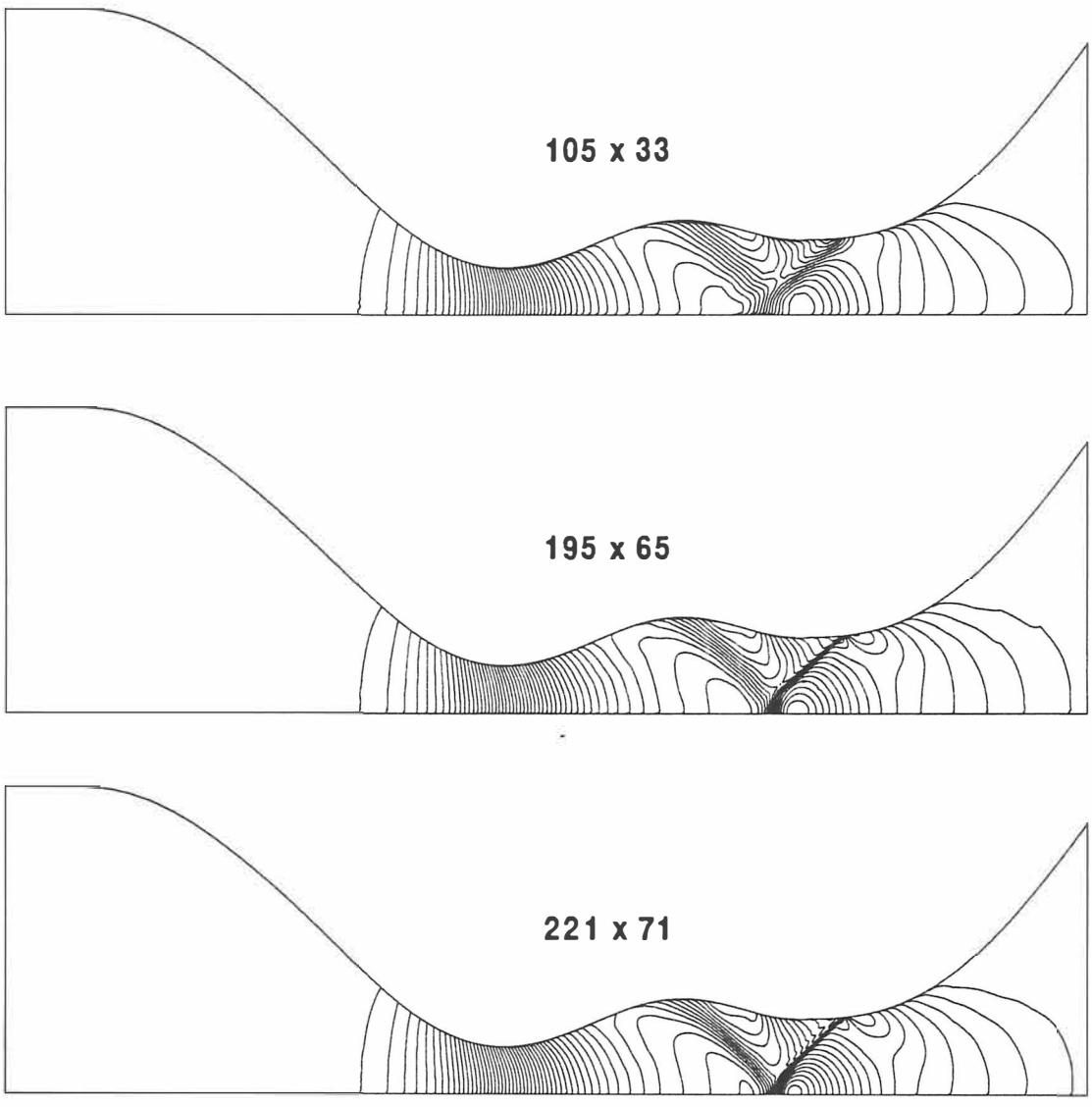


Figure 8.70: Ideal-Gas Pressure Solutions of Respective Mesh Density Refinements for the GAMM Double-Throat Nozzle Inviscid Problem.

is in supersonic wave reflection and oblique shock resolution reflecting off the centerline, just prior to the second throat. In addition, the symmetry of the pressure contours about the centerline is improved with each mesh density increase.

The final fine-grid inviscid solution of the GAMM nozzle is shown as iso contours in physical space in Figure 8.71. For all variables shown, the oblique shock and supersonic compression wave are excellently captured with the more refined mesh. Except for the transverse momentum, part (c), relatively little change appears in the contour plots from the flow entrance up to the first throat for the refined mesh solution. In this region, the flow experiences an area deflection causing the transverse velocity to become negative, undergo a large negative peak, and become zero across the entire cross section of the first throat area. The flow is choked at the first throat area, and remains supersonic thereafter (Figure 8.82). The fluid expands, via area change, until a supersonic wave compression is formed that is directed toward the nozzle centerline. At the point of intersection of this wave with the centerline, an oblique shock forms from that point up to the second throat. At the wave, shock, and centerline intersection, the Mach number experiences a minimum but remains supersonic (Figure 8.82), hence avoiding development of a strong oblique shock. Beyond the weak oblique shock, the fluid continues to expand until the nozzle exit. The area expansion at the exit is dramatic since both a velocity maximum and fluid pressure/temperature minimum are present at the upper wall exit (intersection of surfaces 2 and 4). The appearance of oscillation in the principal velocity iso contour (part b of Figure 8.71) is an apparent anomaly in the graphics since the nodal surface plots (shown next) are free of this oscillation.

The smoothness of the solution about the oblique shock and at all entrances and exits is confirmed by the nodal-surface plots of Figures 8.72 and

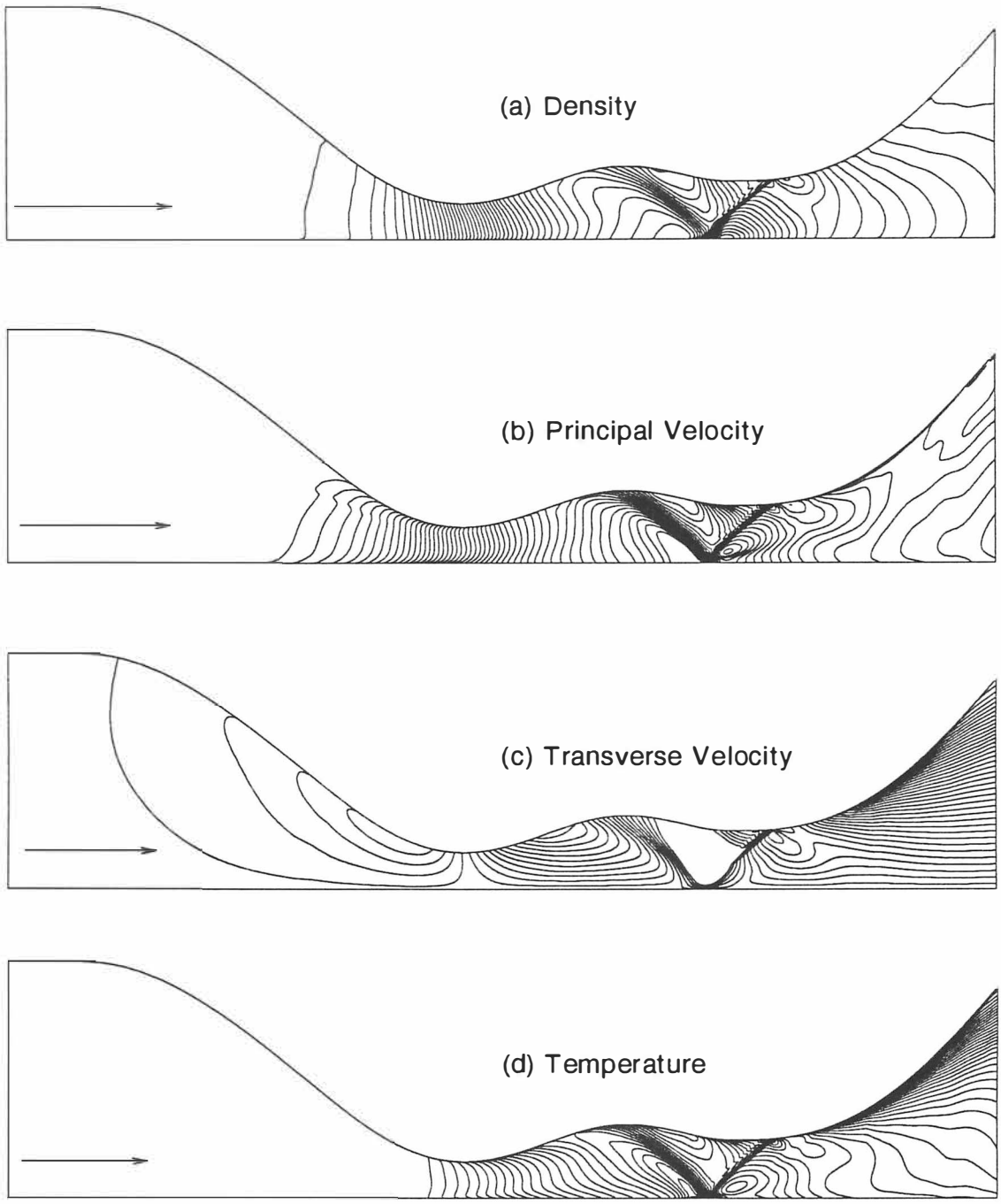


Figure 8.71: Fine-Grid Ideal-Gas Inviscid Solution of the GAMM Double-Throat Nozzle Problem Shown as Iso Contours in Physical Space; (a) Density, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature ($\approx$ to scale).

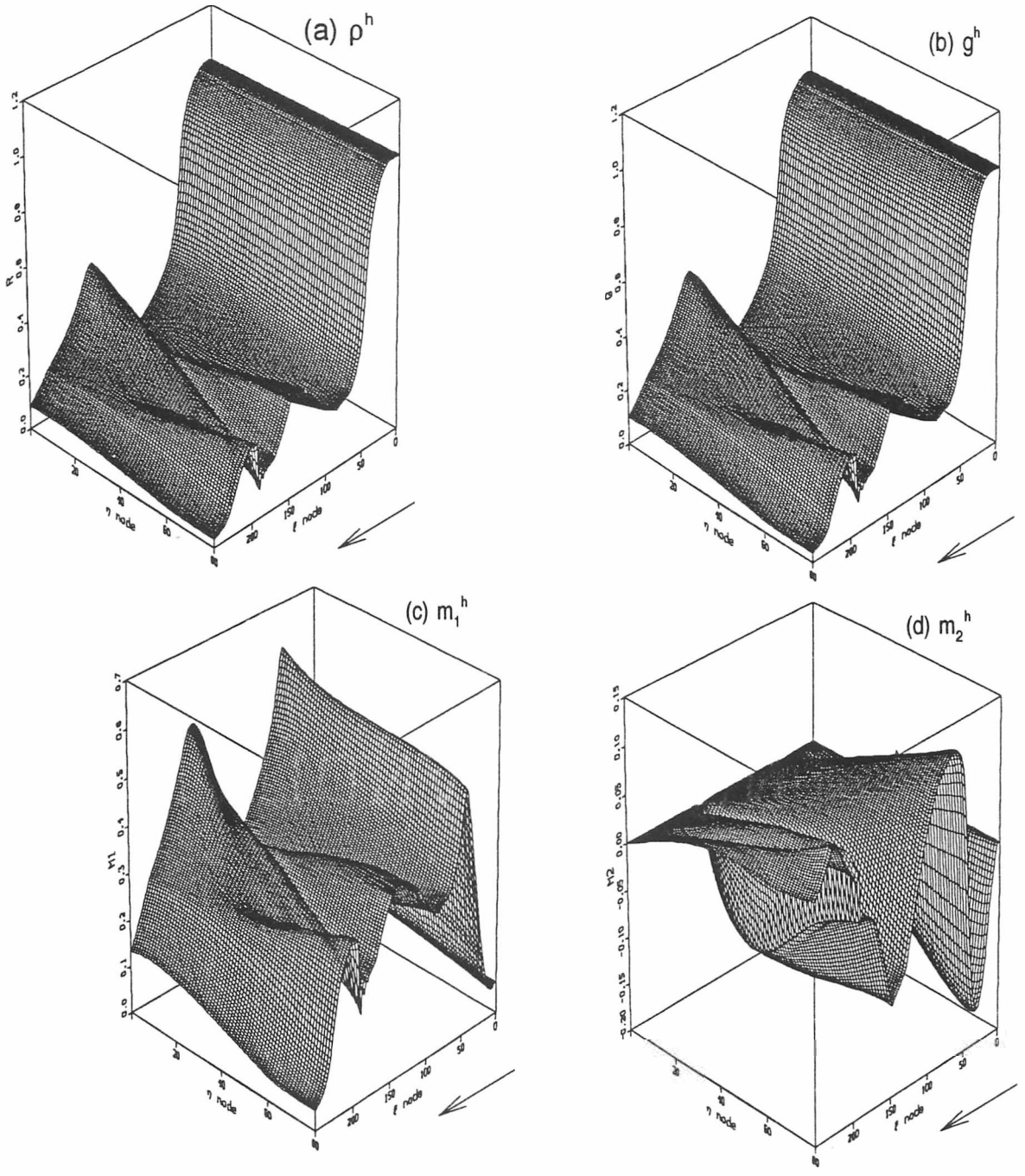


Figure 8.72: Fine-Grid Ideal-Gas Inviscid Flow Solution of the GAMM Double-Throat Nozzle Problem Plotted in Nodal Space as Viewed from the Top Exit. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

8.73. In particular, the non vanishing outflow boundary condition, present on all variables for this problem at surface 2, is shown free of oscillation on all surface plots in Figure 8.72, thus demonstrating that the inaccurate outflow boundary conditions shown in the wind tunnel problem can be corrected. The two peaks in principal momentum along surface 3, zero η_2 (or η), identify the two throat locations, and hence zero transverse momentum. The location of the oblique shock is clearly visible as a local minimum in ρ^h, g^h, and m_1^h nodal plots in Figure 8.72, along surface 4 at approximately η_1 (or ξ) = 180. The nodal plot of transverse momentum in this same region (part (d) of Figure 8.72) dramatically details the ability of the shock captured without requiring the grid aligned with the shock (Figure 8.69). Other notable details revealed by the nodal plots are the relatively few η_1 grid lines in the flow entrance region, and the extremely smooth wall and centerline (surfaces 4 and 3 respectively) surface carpet plots due to the TWS β enforced vanishing derivatives boundary conditions.

8.3.6 Viscous Flow Simulation

As before, the viscous extensions of the simulation are obtained by restarting from the 2D inviscid solution, and then solving the laminar problem. With a laminar viscous solution established, the algebraic turbulence model is then initiated to solve the turbulent flow extension. Stable laminar flow solutions with Reynolds numbers of 10^4, 10^5, 10^6, and 10^7 were obtained using appropriate wall region mesh refinements. However, the GAMM workshop announcement specified that a Reynolds number of 1600 should be used. Therefore, in order to compare with published results of the workshop, the smaller Reynolds number results are shown here. Furthermore, the algebraic turbulence model did not function properly until a larger displacement thickness was computed at the smaller Reynolds numbers.

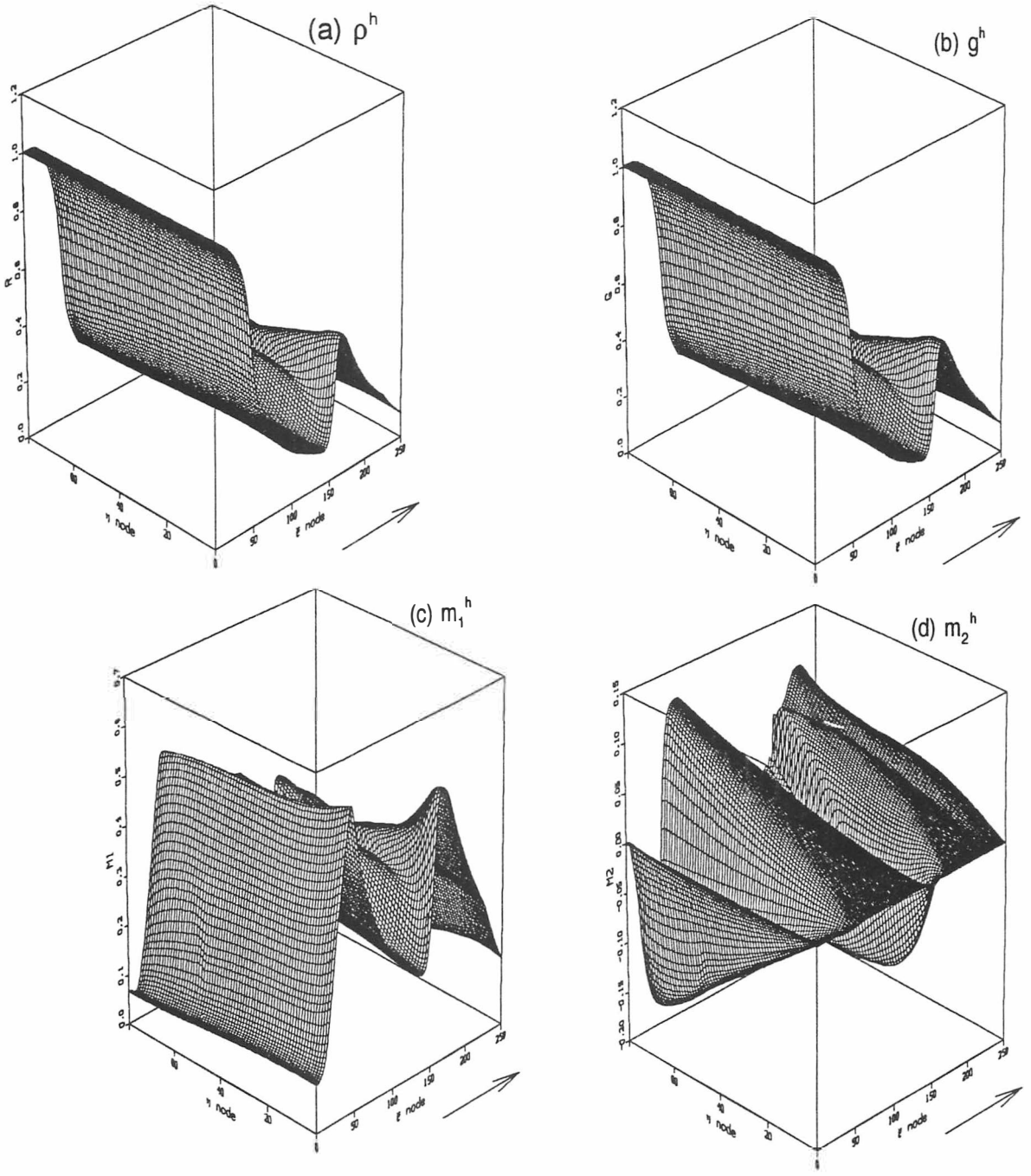


Figure 8.73: Fine-Grid Ideal-Gas Inviscid Flow Solution of the GAMM Double-Throat Nozzle Problem Plotted in Nodal Space as Viewed from the Bottom Entrance. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

Laminar Flow

The laminar viscous solution computed on a 221×71 mesh is summarized as iso contours in physical space in Figure 8.74. The appearance is essentially unchanged from the inviscid counterpart, Figure 8.71. Close examination of the wall regions reveal velocity and temperature boundary layers that have developed from the no-slip and adiabatic boundary conditions at Reynolds number 1600. Thus, even with these relatively small Reynolds number, on such a large-scale body of several feet in diameter ($\approx$ 12 ft. at the exit), the viscous boundary layer is visually very thin. This is different from the published results of unknown length references and scales.[132] The grid design was changed from the inviscid case to resolve all wall details while compromising the centerline detail and associated symmetry resolution. A comparison of the pressure (part (a) of Figure 8.74) with the inviscid solution (221×71 mesh of Figure 8.70) reveals the last four outflow isobars at the centerline are drawn across a single node, hence, the loss of symmetry detail.

The corresponding nodal-space surfaces of the state variable viewed from two different angles are shown in Figures 8.75 and 8.76. The laminar boundary layer is captured by essentially half the η_2 grid and smooth everywhere. The first few η_1 grid lines at the flow inlet (surface 1) remain inviscid, i.e., the no-slip velocity was not imposed at the wall (surface 4) until slightly beyond the flow inlet. Shown starting at $\eta_1 = 0$ on part (c) of both nodal plots, the inviscid start-up region is required for all viscous cases throughout this dissertation because the density and volume-specific stagnation enthalpy imposed Dirichlet data are not known in the boundary layer. Therefore, to avoid an inconsistency in the equation of thermodynamic state, and establish the correct mass flow rate, the present approach is used. At the flow exit (surface 2), consistent Neumann

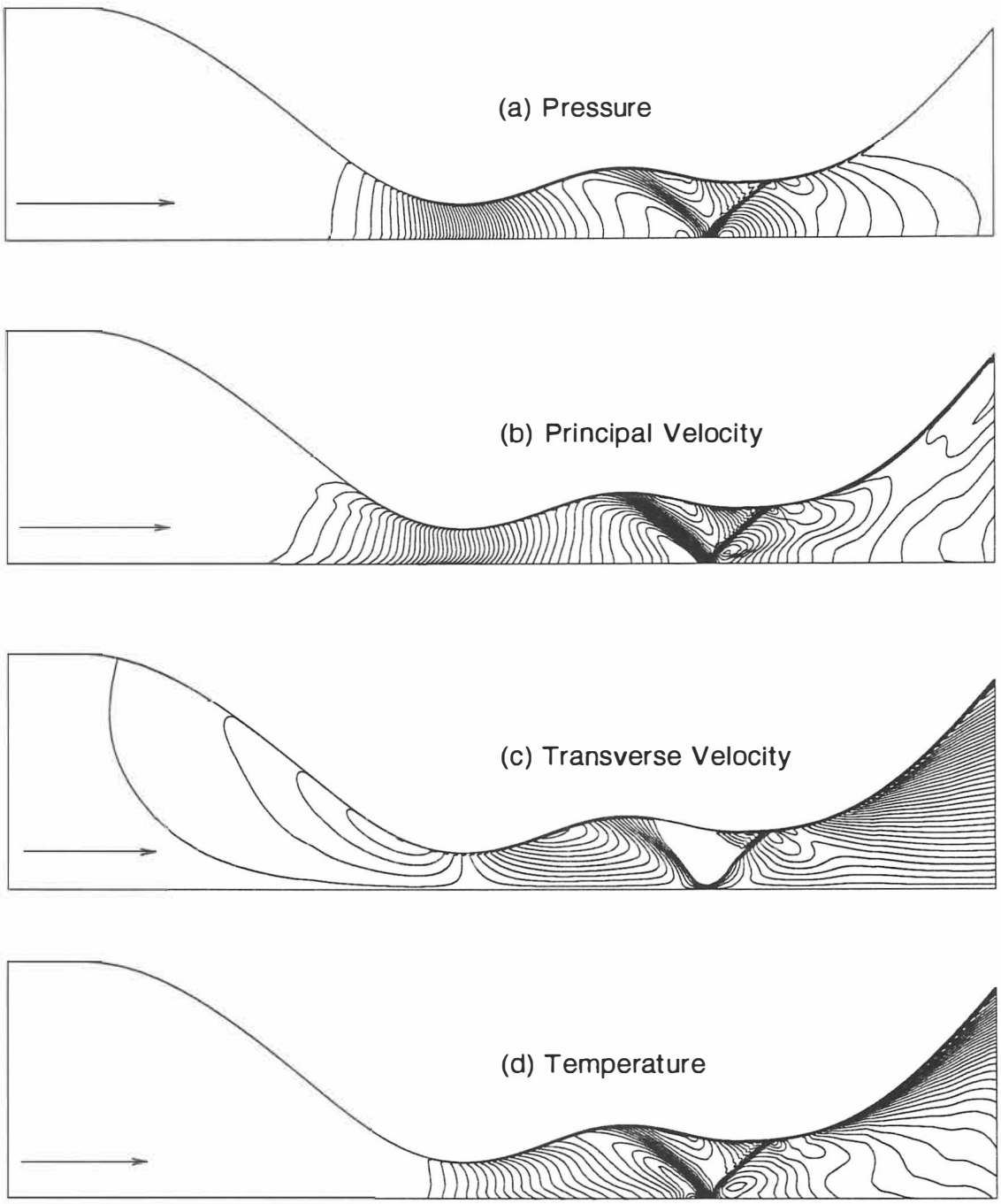


Figure 8.74: Laminar Ideal-Gas Viscous Solution of the GAMM Double-Throat Nozzle Problem Shown as Iso Contours in Physical Space; (a) Pressure, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature (approximately to scale).

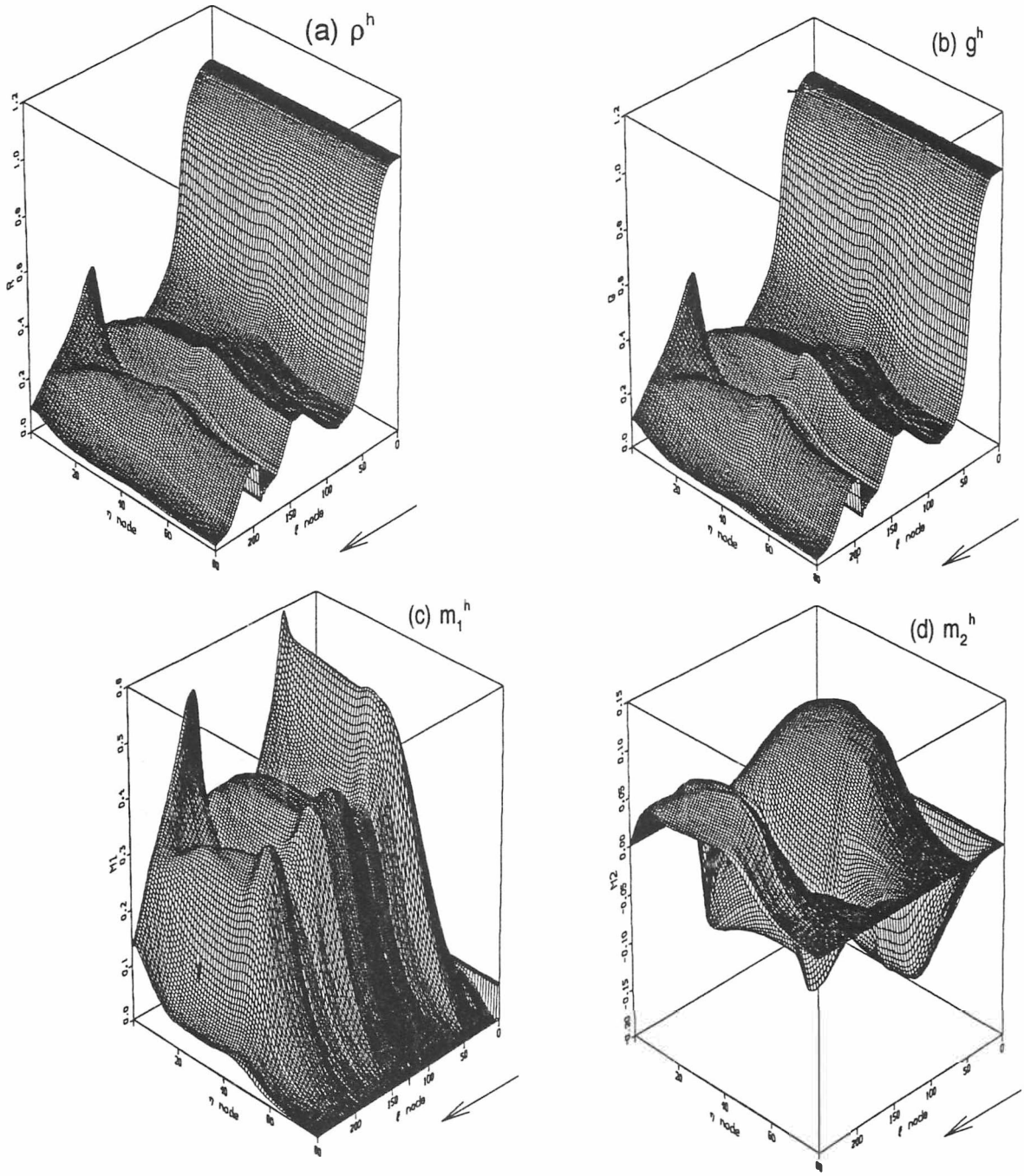


Figure 8.75: Laminar Ideal-Gas Viscous Solution of the GAMM Double-Throat Nozzle Problem Plotted in Nodal Space as Viewed from the Top Exit. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

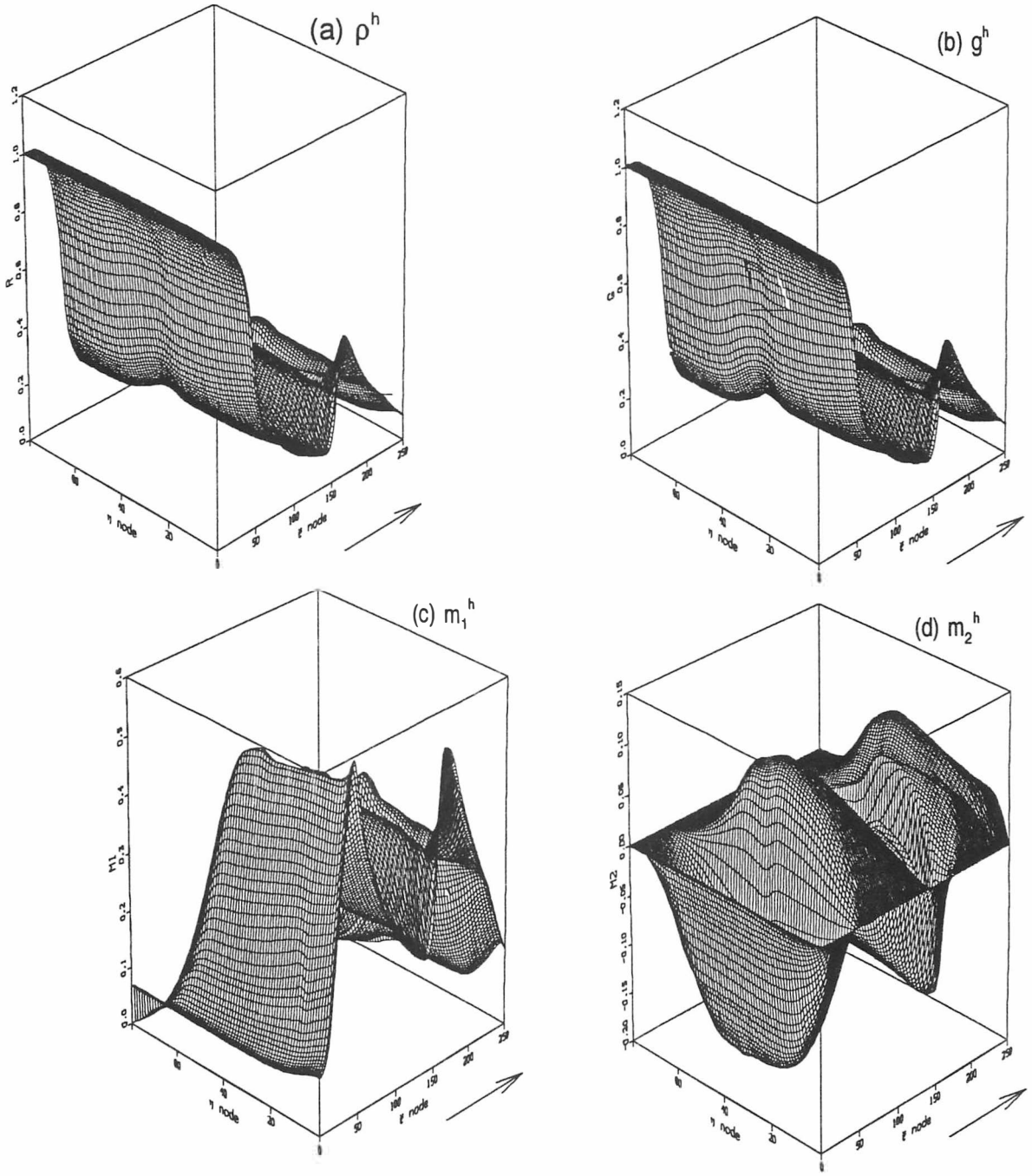


Figure 8.76: Laminar Ideal-Gas Viscous Solution of the GAMM Double-Throat Nozzle Problem Plotted in Nodal Space as Viewed from the Bottom Entrance. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

boundary conditions are present throughout. A slight oscillation can be detected in the boundary layer region of the outflow on both momentum surfaces. This is attributed to the very small magnitude of the viscous terms applied on a relatively large grid spacing in the η_1 direction, i.e., a more detailed mesh at the flow exit would correct this slight deficiency. The shock is captured all the way to the wall as confirmed by density and enthalpy surfaces part (a) and (b) of Figure 8.75. Also notable is the zero transverse momentum at all surfaces except the outflow boundary.

Turbulent Flow

The turbulent viscous solution computed on a 221×71 mesh is shown as iso contours in physical space by Figure 8.77. The appearance is essentially unchanged from the laminar counterpart shown in Figure 8.74. Closer examination of the wall regions reveals that the boundary layer has become thicker than the laminar case as expected for the same Reynolds number. Therefore, viewing the solution in physical space only, with a Reynolds number of 1600, is not sufficient to detect the turbulence effects.

The corresponding nodal-space surfaces of the state variable shown from two different views is given by Figures 8.78 and 8.79. The density and enthalpy nodal plots, parts (a) and (b), remain essentially the same as the laminar case. However, the momentum nodal plots, parts (c) and (d), show a dramatic difference in the boundary layer region of the flow. On both momentum surfaces, the two-slope turbulent velocity profile is apparent causing the laminar region to appear extremely steep near the wall. Furthermore, because of the increased dissipation provided by the eddy viscosity, the outflow boundary surface appears smoother. The turbulent boundary layer possesses the typical non smooth character between the wall-normal grid lines due to the discrete nature of the

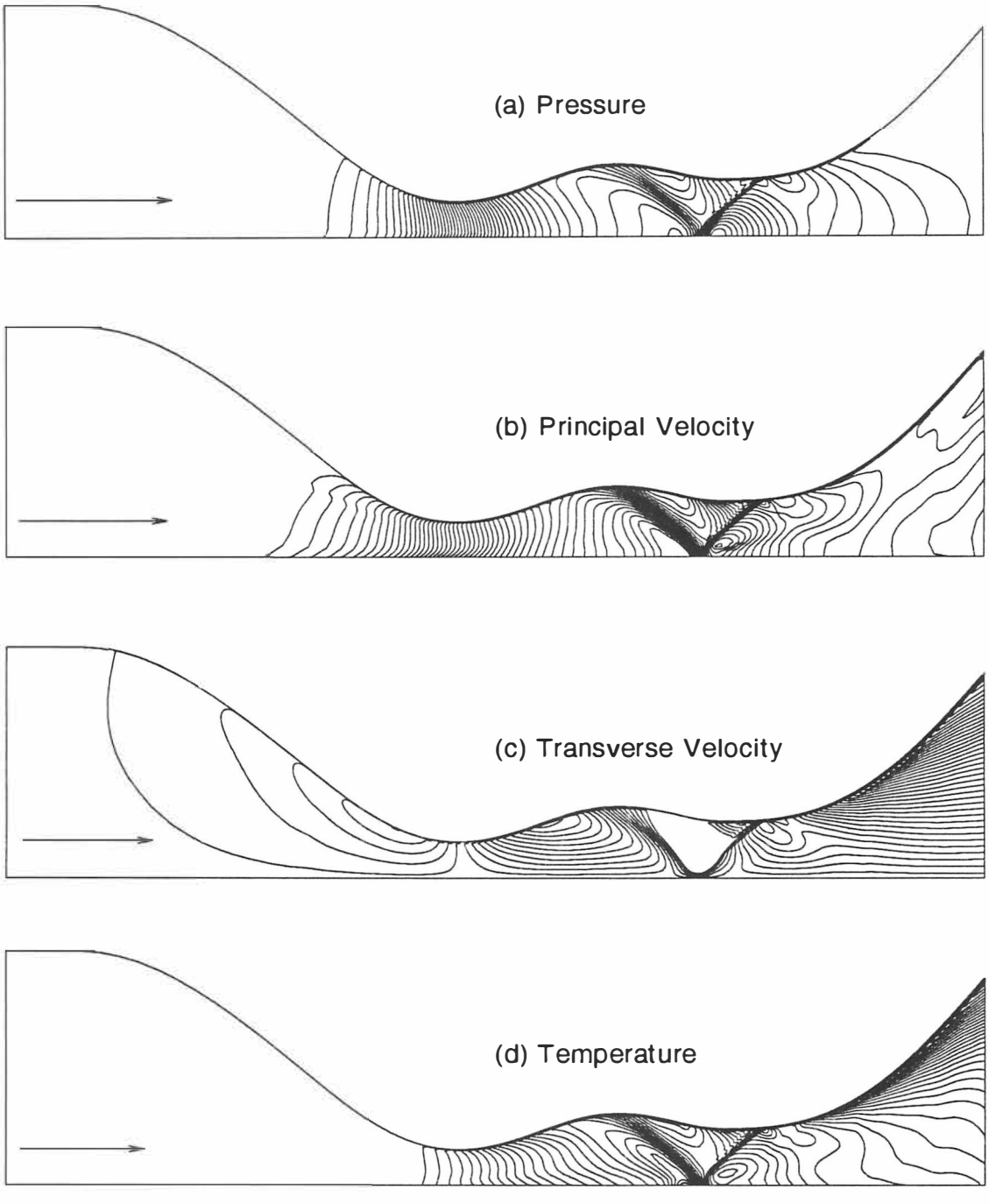


Figure 8.77: Turbulent Ideal-Gas Viscous Solution of the GAMM Double-Throat Nozzle Problem Shown as Iso Contours in Physical Space; (a) Pressure, (b) Principal Velocity, (c) Transverse Velocity, and (d) Temperature (approximately to scale).

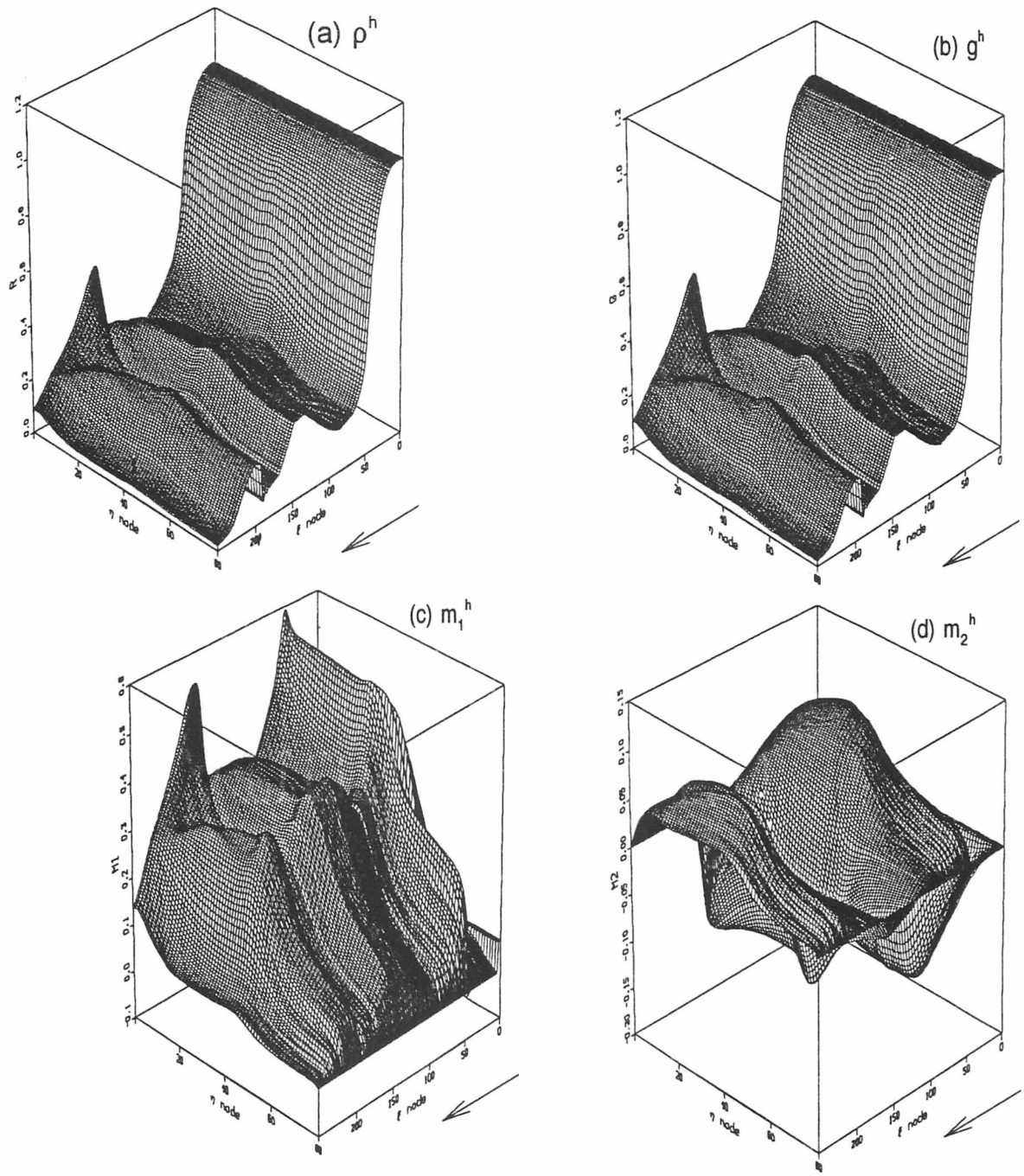


Figure 8.78: Turbulent Ideal-Gas Viscous Solution of the GAMM Double-Throat Nozzle Problem Plotted in Nodal Space as Viewed from the Top Exit. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

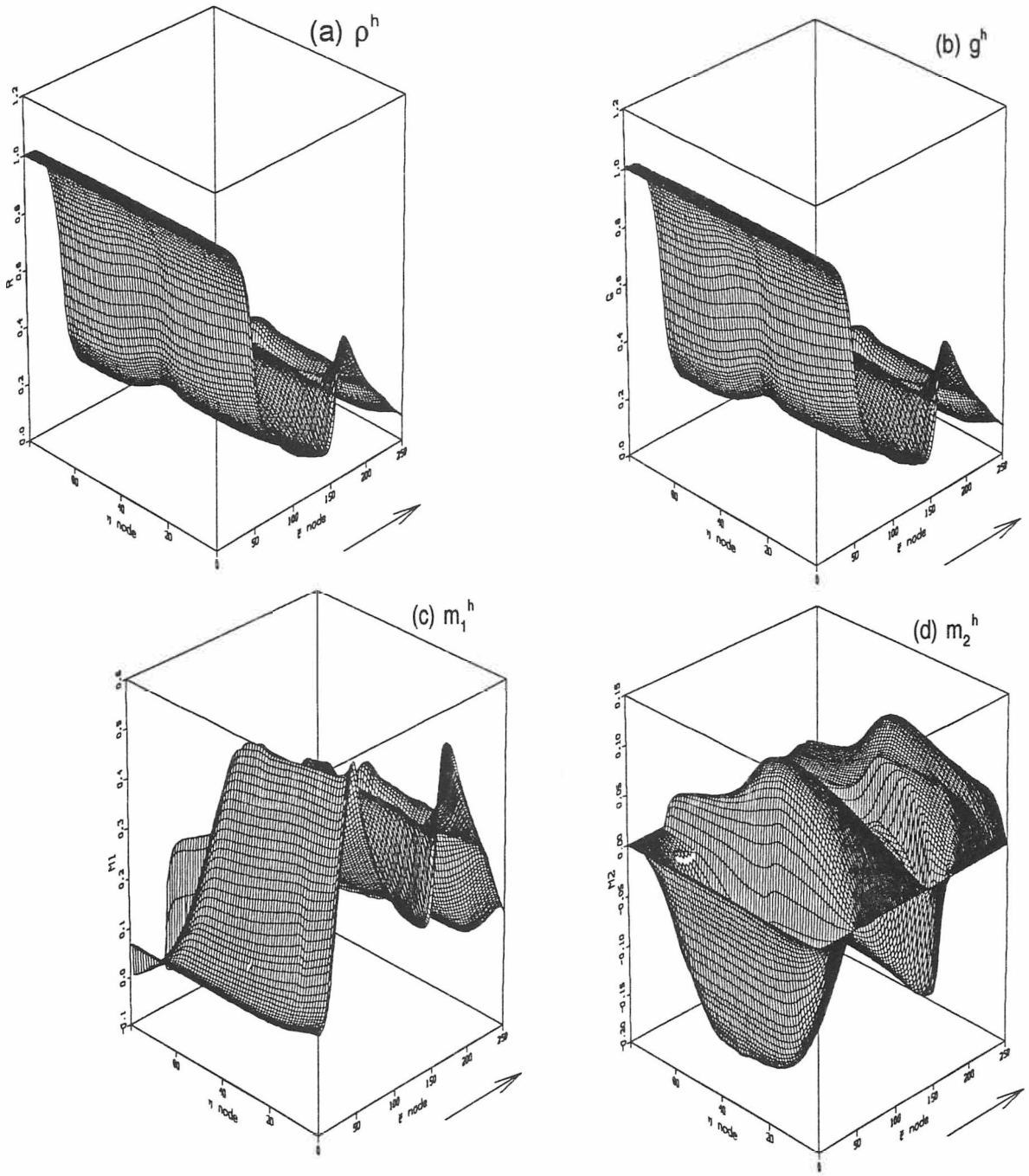


Figure 8.79: Turbulent Ideal-Gas Viscous Solution of the GAMM Double-Throat Nozzle Problem Plotted in Nodal Space as Viewed from the Bottom Entrance. (a) Density, (b) Volume-Specific Stagnation Enthalpy, (c) Principal Momentum, (d) Transverse Momentum.

algebraic turbulence model. The Baldwin-Lomax algebraic turbulence model is enabled starting at the first throat (i.e., the turbulent eddy viscosity is zero upstream of the first throat) which can also be detected on both momentum nodal plots.

Comparisons

The fine-grid steady temperature solutions for the inviscid, laminar, and turbulent cases are shown in Figure 8.80. Even when the wall temperature is increased to nearly the inlet reservoir temperature, and using the adiabatic wall boundary, the kinetic energy dissipation is not sufficient to increase the extremely low temperature predicted in the exit wall region. If the real-gas colusure model is used, this region becomes a Dirichlet constraint causing unreasonable design nozzle flow characteristics. This nozzle design deficiency has been realized by careful attention to real-gas fluid properties and proper enforcement of nonvanishing outflow boundary conditions.

The results published in the GAMM workshop proceedings[132] reveals all published methods are finite-difference schemes, and outflow boundary conditions are extrapolated. This could explain why none of the published results indicate problems experienced with the outflow region. No mention is made in the workshop proceedings of real-gas fluid properties, turbulence, or realistic geometry values.

Figure 8.81 shows a typical pressure and Mach number contour set from the GAMM workshop proceedings for a Reynolds number of 1600. In comparison to the present results, the most significant difference is the much thicker boundary layer, which combined with the fluid expansion, hence, low fluid thermodynamic state (pressure and temperature) in the top exit corner, cause the

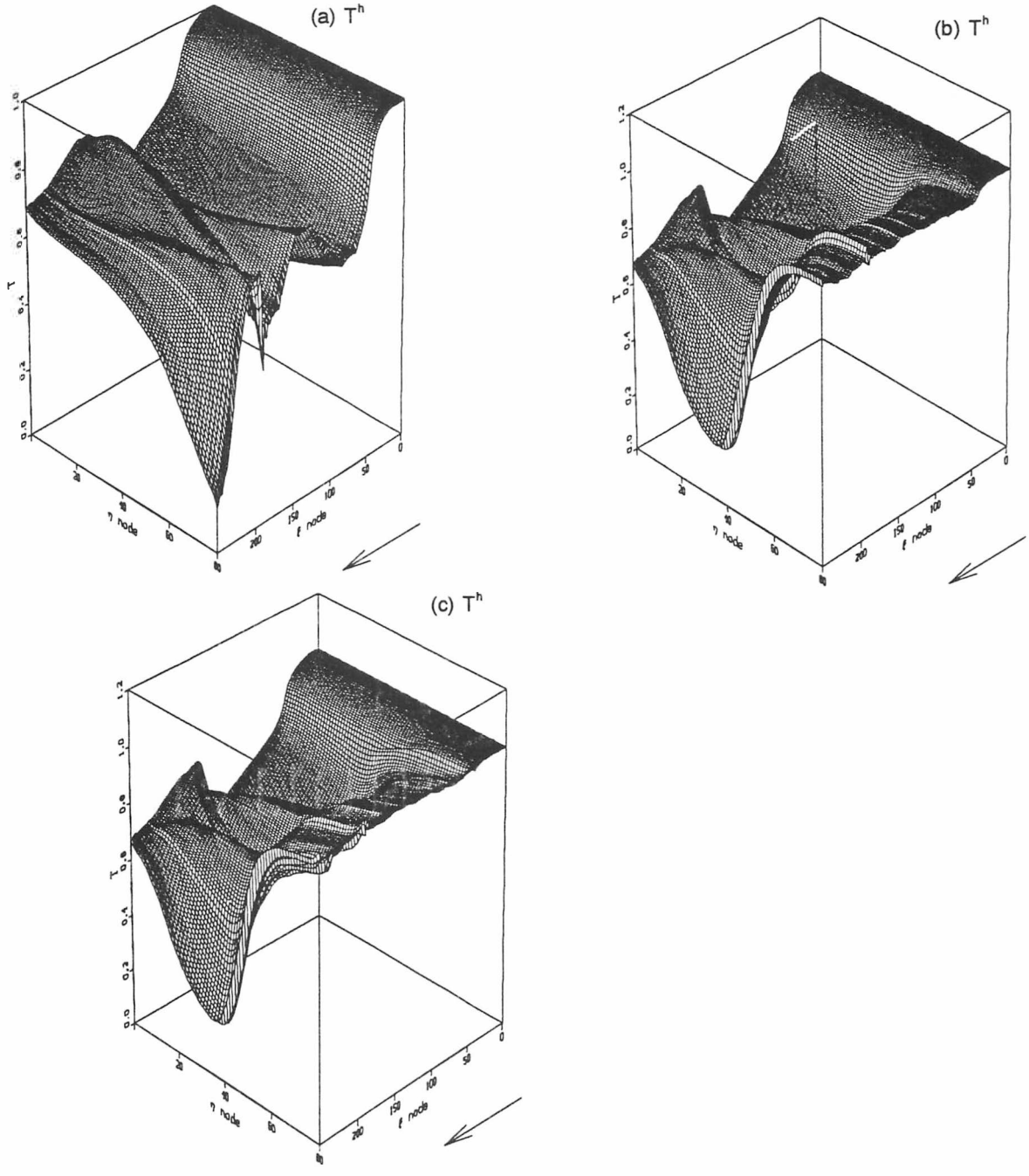


Figure 8.80: Fine-Grid Temperature Solution of the GAMM Double-Throat Nozzle Problem Plotted in Nodal Space as Viewed from the Top Exit. (a) Inviscid, (b) Laminar Viscous, (c) Turbulent Viscous.

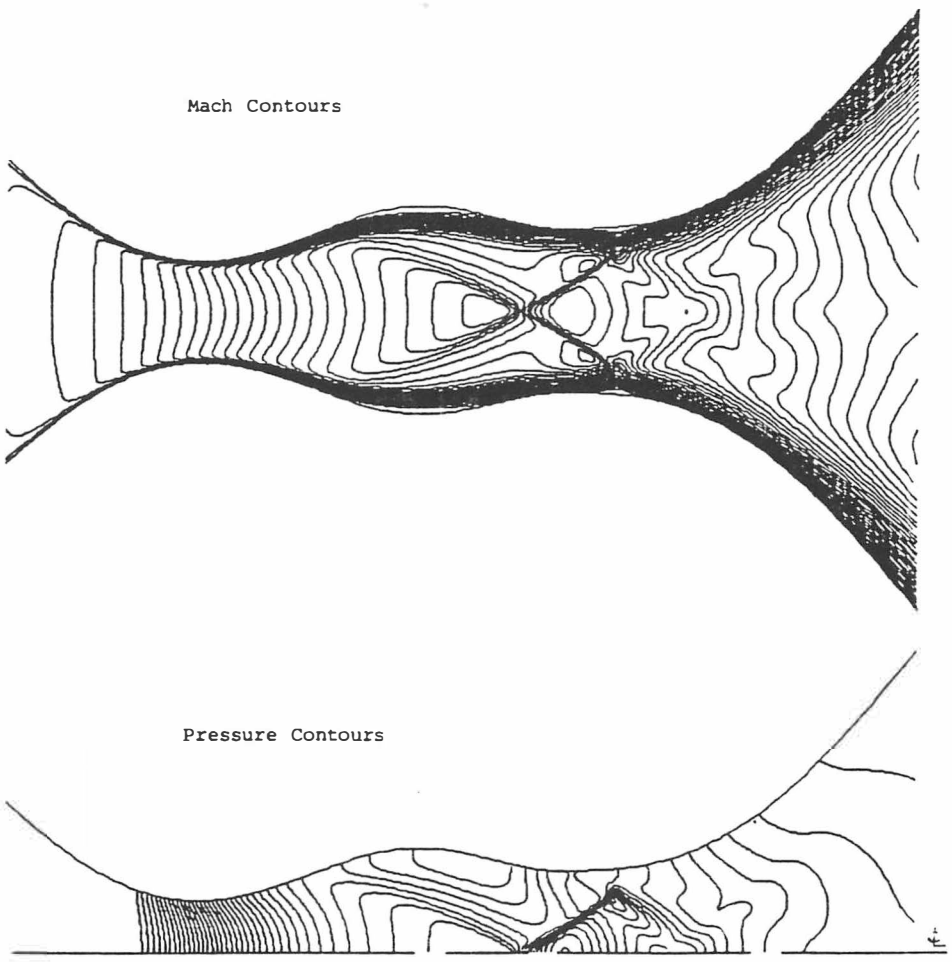


Figure 8.81: Pressure and Mach Number Iso Contours for Reynolds Number of 1600, Thomas and Walters, from GAMM Workshop Proceedings, Braunschweig, Wiesbaden, Bristeau, et al.[133]

pressure wave reflection/oblique shock to be significantly offset from the wall. Other published results from the proceedings indicate, as expected, that the boundary layer thickness and oblique shock wall offset is controlled by variations in Reynolds number (several variations are shown with the largest being 1600). This suggests that the present simulation is different for two reasons:

- the length scale associated with Reynolds number 1600 does not cause a visually significant boundary layer thickness for the physical geometry chosen, and
- the shock is present all the way to the wall for variables not directly operated on by viscosity, i.e., density, volume-specific stagnation enthalpy, and pressure (ρ, g, p).

The Mach number for the present GAMM nozzle results becomes large in the exit region, adjacent to the wall, due to low temperature. Figure 8.82 shows a magnified view of the double throat region Mach number solution as a color contour for the ideal-gas inviscid case. The inviscid case is chosen because velocity/temperature gradients are so steep near the wall for the viscous case, hence, the shock reflection details prior to the second throat region are only visible in the inviscid case. Except for the lack of a thick boundary layer, Figure 8.82 compares qualitatively with 8.81. Furthermore, the color shows the lack of a subsonic region beyond the first throat and confirms the nozzle is void of normal shocks. Also notable, the pressure solution given in Figures 8.83, and others shown previously in Figures 8.70, 8.74 part(a), and 8.77 part(a) all indicate good agreement with the published results, typical by Figure 8.81, except for the boundary layer offset previously discussed. However, many of the GAMM-published results do not show dense isobar contours, implying grid refinement and resulting accuracy is increased in the present work and dramatically confirmed

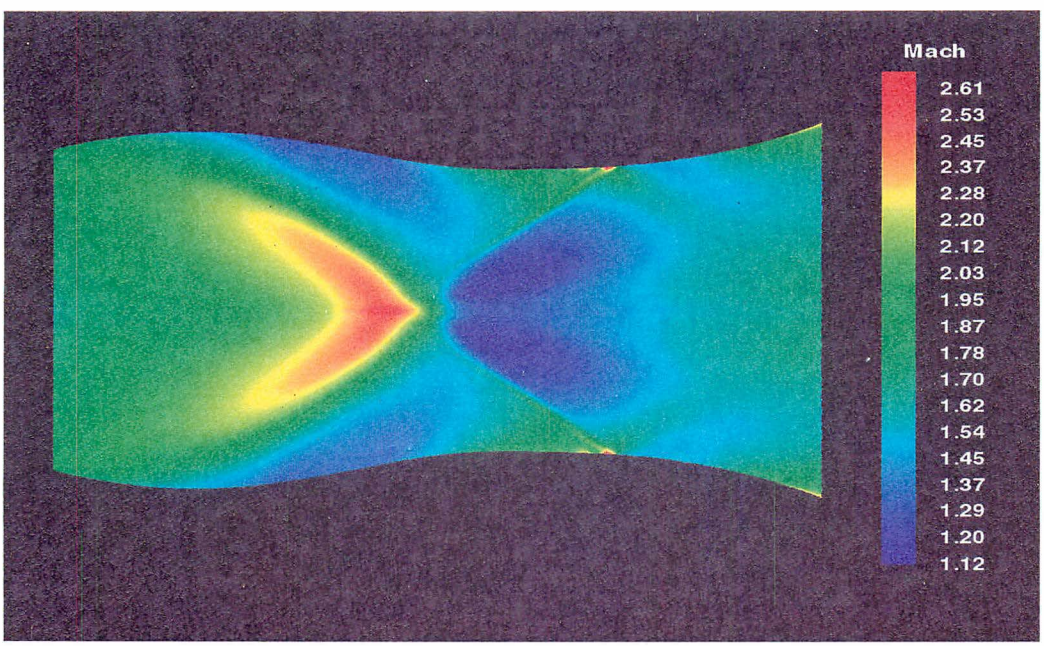


Figure 8.82: Calculated Mach Number for the Ideal-Gas Invsicid Solution of the GAMM Double-Throat Nozzle Problem, Expanded Shock-Reflection Region Shown. ($\approx$ to scale).

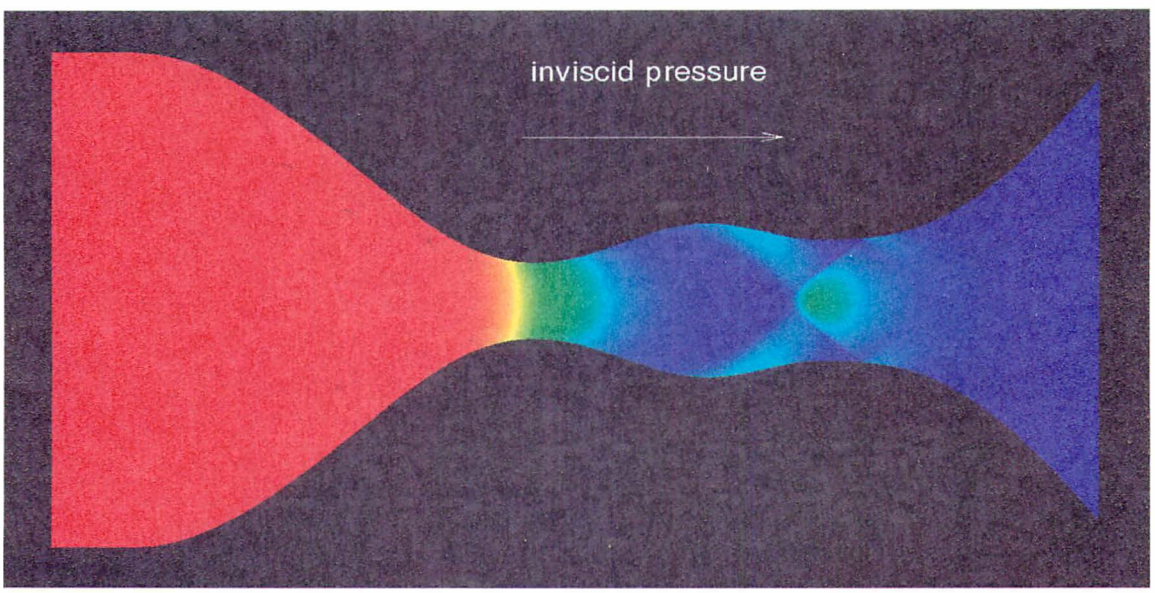


Figure 8.83: Calculated Pressure for the Ideal-Gas Inviscid Solution of the GAMM Double-Throat Nozzle Problem ($\approx$ to scale).

by the color spectrum gradients of Figure 8.83. Finally, a color iso contour of the laminar viscous solution of transverse momentum, Figure 8.84, reveals a number of flow details including:

- 2D flow symmetry about the nozzle centerline,
- zero transverse momentum at the inlet, centerline, walls, and both throats indicated by the green color,
- negative values in regions of area decrease and positive values in regions of area increase as indicated by blue and red respectively,

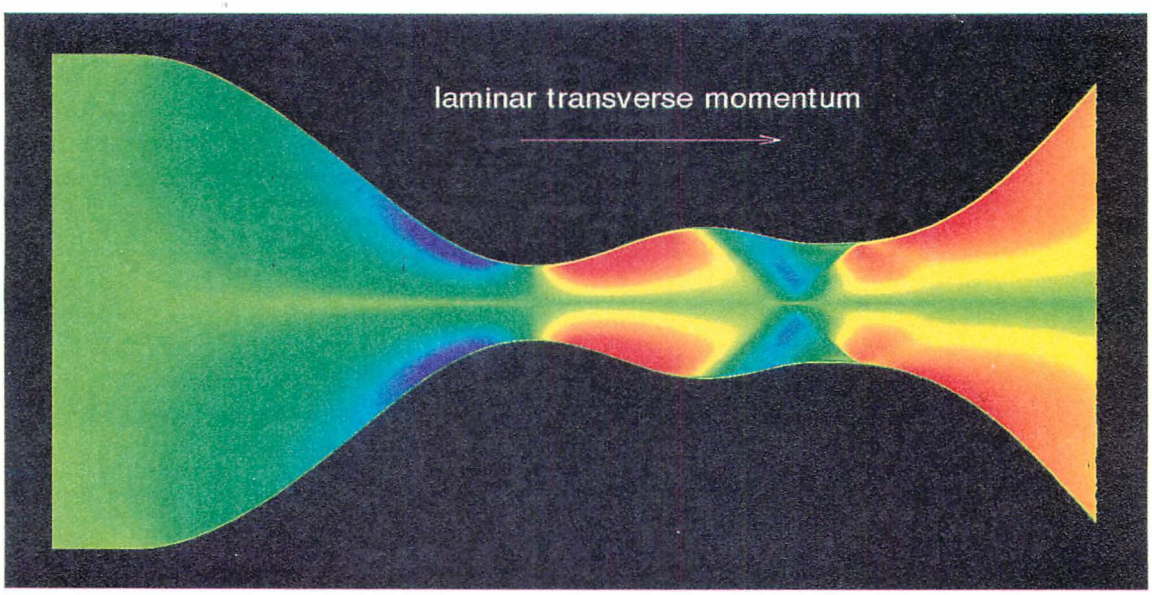


Figure 8.84: Calculated Transverse Momentum for the Ideal-Gas Laminar Viscous Solution of the GAMM Double-Throat Nozzle Problem (≈ to scale).

- supersonic wave compression and oblique shock captured with crispness indicated by the high density of color gradients,

- smooth non-vanishing normal outflow boundary conditions while simultaneously resolving dramatic changes in transverse velocity profile, from zero to maximum peak value back to zero, and extreme demand on the thermodynamic equation of state (outflow wall region).

Chapter 9

Supercomputer Utilization

The numerical algorithm and verification code, described herein, is well suited for supercomputer utilization. This section briefly describes how the code, hence naturally the algorithm, are formulated to exploit supercomputer features. Techniques employed in vectorization and parallelization are discussed, with implementation specific to the IBM-3090 with vector facility (VF). The analysis techniques are equally applicable to other supercomputers. The impact of real-gas fluid property determination on code performance is also discussed.

9.1 General Organization

The organization of the code naturally follows the algorithm construction, Figure 9.1. The major portions are initialization, finite element assembly, solution operations, and output. The initialization portion (subroutine `INIT` and lower levels) only occurs one time per execution. Therefore, no major effort was expended to exploit the supercomputer in this portion of the code. Similarly, the output portion (subroutine `DUMPIT` and lower levels) is only executed periodically. Because the `DUMPIT` routine requirements are intrinsically inefficient, from a vectorization

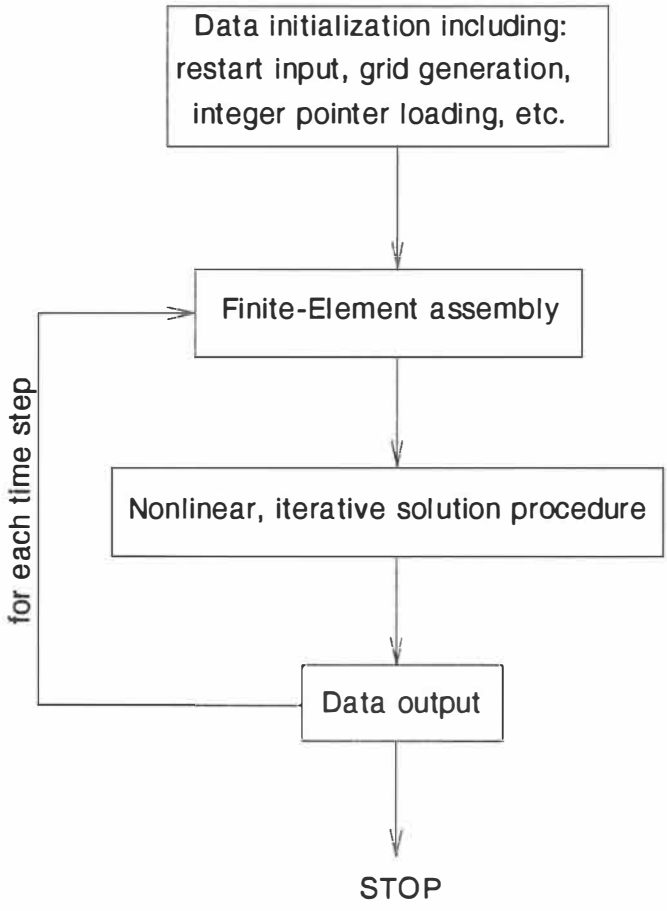


Figure 9.1: Overall Organization of the Pilot Code

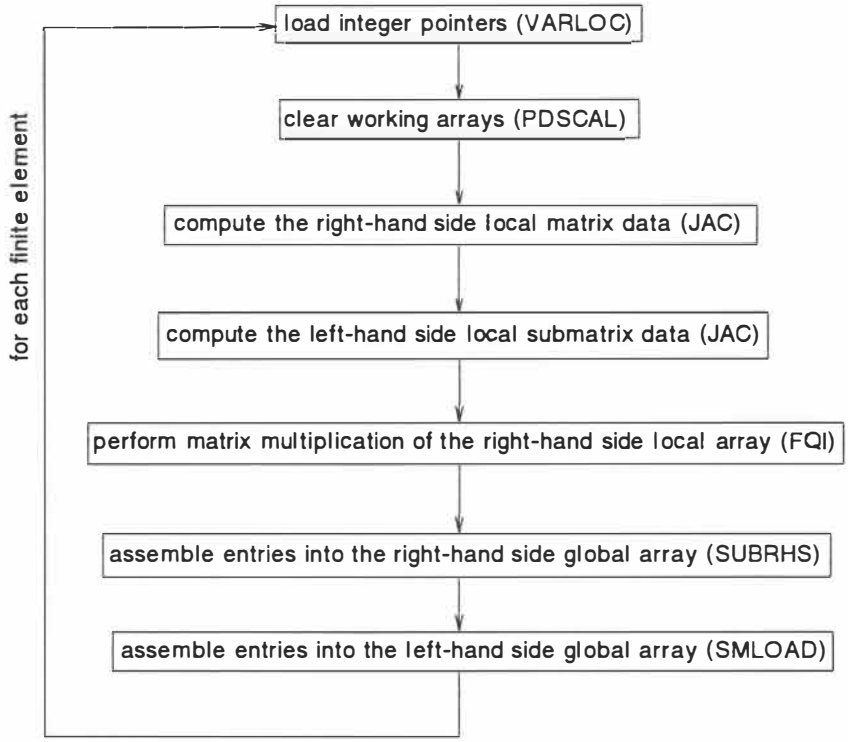


Figure 9.2: Organization of the Finite Element Assembly Loop of the Pilot Code

or parallelization viewpoint, a marked reduction in the computational efficiency in code computational efficiency is expected when output is requested too often.

Figure 9.2 depicts the structure of the finite element assembly loop, which serves two purposes. The extremum vector (right-hand side or RHS) is assembled, with the level of complexity depending on the degree of the finite-element basis function, the state variable order (number of governing equations), and the number of terms in each algorithm statement. For the selected two-dimensional, linear tensor-product basis, this results in the generation and subsequent multiplication of a minimum of a 4×4 matrix and an array of length 4 for each term in each equation. The energy equation alone has nine terms (in two dimensions) exclusive of the time-derivative term. Hence, a large

number of floating-point operations generate the RHS and the code has thereby been structured to exploit vectorization a supercomputer.

The finite element assembly loop also generates and loads the quasi-Newton jacobian matrix. Because there are four entries in the state variable, there are sixteen (4×4) submatrices within the full-jacobian matrix definition. Furthermore, since the two-dimensional jacobian is approximated by two sweeps in each dimension, 32 submatrices are required. Each of these 2×2 submatrices brings the total to 128 matrix elements to be computed and assembled. Each computation is detailed, depending on the complexity and size of the (simplified) jacobian matrix (bandwidth$\geq$ 3) described earlier.

Where possible, these computations have been vectorized and also utilize available parallel techniques. However, this portion of the code has also been the most difficult to exploit the supercomputer. A code rewrite, possibly in assembler language, following redesigns of the computational sequences in the basic algorithm, are necessary to take full advantage of the supercomputer features.

The matrix solution procedure and code iteration loop are shown in Figure 9.3. Numerous places in this part of the code exploit supercomputer resources due to a number of `DO` loops over every degree of freedom. At various points in the iteration loop, logic tests are executed to determine stopping criteria. These criteria include achieving steady-state, CPU time limit, a computed residual being too large, and a maximum number of iterations. If the steady-state criteria is too stringent, the program will normally end when the CPU time limit is approached. All halting of code execution is achieved with no loss of computed data via the output of periodic restart records, including the end point.

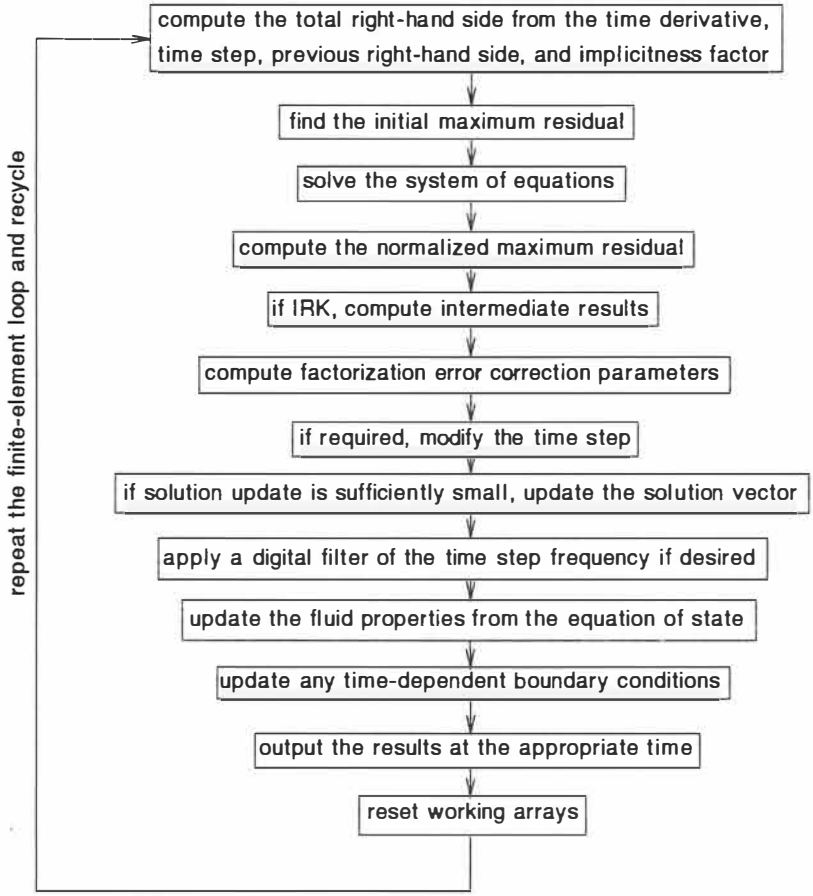


Figure 9.3: Organization of the Nonlinear Iterative Solution Procedure of the Pilot Code

The solution time step is very tightly controlled via a solution residual "window" about a solution iterate (see Chapter 5) based on the quasi-Newton iteration procedure, a maximum Courant number, and a minimum and maximum time step. Furthermore, the Θ implicitness factor (for a non-SIRK solution), and the Reynolds number may optionally vary based on the Courant number, for a controlled approach to steady-state.

9.2 Vectorization Techniques

The primary objective for utilizing supercomputer vector computation hardware is to translate `DO` loops from scalar to vector, causing huge savings in CPU and wall-clock time for the same amount of computing work. From the experience of this research, there are three principal limitations toward achieving this objective: (1) the length must be sufficient to offset the startup cost (overhead) of vectorization; (2) no recursive properties can be present, i.e., the calculation of any given element of the vector is not affected by a previous element calculation; and (3) the syntax must be simple enough for the `FORTRAN` compiler to translate the scalar `DO` loop into a vector equivalent. The compiler will not always determine these limiting criteria correctly, particularly when the loop end points are variables. Therefore, it is recommended that all vector translations be carefully scrutinized before executing in a production mode since it is possible to compute incorrect results and/or cause more CPU and wall-clock time to be required for the same amount of work (just the opposite of the desired effect!).

The dominating criteria for vectorization, which varies between difference supercomputers is the threshold length by which a vector loop becomes more economical (utilizes less computer resources) than a scalar loop. The IBM-3090VF requires at least 128 contiguous bytes or 16 `REAL*8` floating-point words

in order to vectorize.[133] There is also an upper limit threshold (the IBM-3090VF upper threshold is about 256 REAL*8 floating point words) whereby longer vectors will not decrease the computational effort. For example, two REAL*8 floating-point loops of length 256 each will process at about the same speed as a single loop with a length of 512 REAL*8 words on the IBM-3090VF.

If a simple DO loop meets the length criteria, then the FORTRAN compiler will automatically vectorize the loop. Figure 9.4 shows the result of the vectorization of simple loops within subroutine SWITCH of the code. The top part of Figure 9.4 is the original listing of the loops and the bottom part shows that the loops have been vectorized.

Often the DO loop structure is not so simple. The algorithm presented herein results in loops that occupy a large block of memory that are much greater than the lower threshold and meet the non-recursive computational requirement. However, the natural computational sequence of the loop does not result in a simple contiguous relationship. This problem has been remedied, without major rewriting of the code, through the utilization of "indirect addressing." For example, subroutine FQI performs the multiplication sequence to create the local RHS residual array. This sequence is made up of a set of 4×4 matrices multiplying a single array of length 4 (for each node of the linear tensor-product basis finite element). The total sequence is quite long and results in a large number of floating-point operations. Figure 9.5 shows the original FQI subroutine prior to rewriting and vectorization. The occurrence of each term of the governing equations appears quite naturally. For example, the multiplication term [C1OR] {RU} represents the convection of the primary momentum in the primary direction within the continuity equation. Prior to vectorization, subroutine FQI required approximately 40% of the total CPU time of a typical CFD2D execution.

```
          89          IF(TYPE)THEN

                   C zeta direction to eta direction

                   CPDIR PREFER VECTOR PARALLEL
                   CVDIR PREFER VECTOR
  1        90            DO 10000 I=1,NNODE
  1  1     91    10000     RERR(ZTET(I))=RHSV(ZTEF(I))

  1        92          ELSE IF(.NOT.TYPE)THEN

                   C eta direction to zeta direction
                   CPDIR PREFER VECTOR PARALLEL
                   CVDIR PREFER VECTOR
  1        93            DO 10001 I=1,NNODE
  1  1     94    10001     RERR(ETZT(I))=RHSV(ETZF(I))

  1        95          ENDIF

                   CPDIR PREFER VECTOR PARALLEL
                   CVDIR PREFER VECTOR
           96            DO 10002 I=1,NNODE
     1     97    10002     RHSV(I)=RERR(I)

           98          TYPE=.NOT.TYPE
```

Original Code Listing Prior to Vectorization

```
0089                     IF(TYPE)THEN

                    C zeta direction to eta direction

                    CPDIR PREFER VECTOR PARALLEL
                    CVDIR PREFER VECTOR
0090 VECT +-------          DO 10000 I=1,NNODE
0091      |_______10000       RERR(ZTET(I))=RHSV(ZTEF(I))

0092                     ELSE IF(.NOT.TYPE)THEN

                    C eta direction to zeta direction

                    CPDIR PREFER VECTOR PARALLEL
                    CVDIR PREFER VECTOR
0093 VECT +-------          DO 10001 I=1,NNODE
0094      |_______10001       RERR(ETZT(I))=RHSV(ETZF(I))

0095                     ENDIF

                    CPDIR PREFER VECTOR PARALLEL
                    CVDIR PREFER VECTOR
0096 VECT +-------          DO 10002 I=1,NNODE
0097      |_______10002       RHSV(I)=RERR(I)

0098                     TYPE=.NOT.TYPE
```

Code Listing After Vectorization

Figure 9.4: Vectorization of a Simple Loop Within Subroutine `SWITCH` of the Pilot Code

```
      DO II=1,NPE
      DO JJ=1,NPE
* continuity equation
        M1 (II) = M1 (II)    + T1L (II,JJ) * DR      ! dr/dt
        R1 (II) = R1 (II)    + C10R(II,JJ) * RU      ! dm1/dx
     >                       + C20R(II,JJ) * RV      ! dm2/dy
     >                       + T1R (II,JJ) * R       ! tws term
     >                       + S10 (II,JJ)           ! source term
* principal momentum equation
        M2 (II) = M2 (II)    + T2L (II,JJ) * DRU     ! dm1/dt
        R2 (II) = R2 (II)    + C10M1(II,JJ)* U * RU  ! d(um1)/dx
     >                       + C01M1(II,JJ)* R       ! a**2 dr/dx
     >                       + C20M1(II,JJ)* U * RV  ! dum2/dy
     >                       + D2U (II,JJ) * U       ! (d**2/dx**2+d**2/dy**2)u
     >                       + D2V (II,JJ) * V       ! (d**2/dxdy)v
     >                       + T2R (II,JJ) * RU      ! tws term
     >                       + S20 (II,JJ)           ! source term
* transverse momentum equation
        M3 (II) = M3 (II)    + T3L (II,JJ) * DRV     ! dm2/dt
        R3 (II) = R3 (II)    + C10M2(II,JJ)* V *RU   ! dvm1/dx
     >                       + C20M2(II,JJ)* V * RV  ! d(vm2)/dy
     >                       + C02M2(II,JJ)* R       ! a**2 dr/dy
     >                       + D3V (II,JJ) * V       !(d**2/dx**2+d**2/dy**2)v
     >                       + D3U (II,JJ) * U       !(d**2/dxdy)u
     >                       + T3R (II,JJ) * RV      ! tws term
     >                       + S30 (II,JJ)           ! source term
* energy equation
        M4 (II) = M4 (II)    + T4L (II,JJ) * DG      ! dg/dt
        R4 (II) = R4 (II)    + C10GA(II,JJ) * U *G   ! dug/dx
     >                       + C20GA(II,JJ) * V *G   ! dvg/dy
     >                       + C10GB(II,JJ) * RU     ! a**2(GAM0-1) dru/dx
     >                       + C20GB(II,JJ) * RV     ! a**2(GAM0-1) drv/dy
     >                       + D4T (II,JJ) * T       ! d**2 T
     >                       + D4U (II,JJ) * U       ! d**2 u
     >                       + D4V (II,JJ) * V       ! d**2 v
     >                       + T4R (II,JJ) * G       ! tws term
     >                       + S40 (II,JJ)           ! source term
      ENDDO
      ENDDO
```

Figure 9.5: Selected Portion of the Original `FQI` Subroutine Prior to Vectorization

Subroutine FQI appears after rewriting and vectorization by indirect addressing, is shown in Figure 9.6. The upper part is the original listing and the lower part is the vector report. The subroutine is now composed of two outer loop structures which are the time derivative and non-time derivative portions of the RHS respectively. The inner loops are sufficiently long to vectorize. The array QMQ is composed of the entire set of nodal-value arrays of length 4 (such as the M1TMP array). The array AMQ is composed of the entire set of 4 $\times$ 4 matrices (such as the [C10R] matrix). The vectorization method is called "indirect addressing" because the index loop (variable II in this case) is utilized to reference an additional array of indices (array LAMQ in this case) which in turn references the actual data being multiplied. Incidentally, the inner loop is also a vector reduction operation since the variable MQ(JJ) appears as a constant to the compiler.

The creation of this loop set is possible through the use of array equivalencing and the fixed structure property of FORTRAN COMMON blocks as shown in Figure 9.7. The first location in COMMON block FBUILD is equivalenced to the first location in array AMQ. Similarly the first location in COMMON block VARTMP is equivalenced to array DRTMP. The net result is that the CPU time required for subroutine FQI has been reduced from the previous 40% to approximately 20% of the total execution time.

Some DO loop structures are so complex that the compiler cannot vectorize the loop. This can be very frustrating when the algorithm developer knows that the structure fits all the criteria for vectorization. A typical example of such a DO loop occurrence is shown in Figure 9.8 within subroutine SMLOAD of the code. The loop is sufficiently long (256 REAL*8 words) and does not possess recursive properties since the same location of the array SMAT appears on both sides of the equation. However, because of the complexity of the integer pointer

```
          1          SUBROUTINE FQI
          2          IMPLICIT REAL*8(A-H,O-Z)
                C    INCLUDE 'COMMON.ONE'
                C    INCLUDE 'COMMON.THR'
                * routine to load the rhs vector F = [J] * {&Q}

         89          CALL PDSCAL(2*MNPE,ZERO,MQ)

         90          DO 100 II=1,4
      1  91            DO 100 JJ=1,MNPE
      2  92              MQ(JJ) = MQ(JJ) + QMQ(     (JJ-1)*LOFL(JJ)+II)
      2                >                     * AMQ(LAMQ((JJ-1)*LOFL(JJ)+II))
      2  93       100    CONTINUE

         94          DO 200 JJ=MNPE+1,2*MNPE
      1  95            DO 200 II=4*MNPE+1,4*MNPE+LOFL(JJ)
      2  96              MQ(JJ) = MQ(JJ) + QMQ(     (JJ-MNPE-1)*LOFL(JJ)+II)
      2                >                     * AMQ(LAMQ((JJ-MNPE-1)*LOFL(JJ)+II))
      2  97       200    CONTINUE

         98          RETURN
         99          END
```

Original `FQI` Listing

```
0001                     SUBROUTINE FQI
0002                     IMPLICIT REAL*8(A-H,O-Z)
                   C     INCLUDE 'COMMON.ONE'
                   C     INCLUDE 'COMMON.THR'
                   * routine to load the rhs vector F = [J] * {&Q}

0089                     CALL PDSCAL(2*MNPE,ZERO,MQ)

0090 ELIG +-------       DO 100 II=1,4
0091 VECT |+------         DO 100 JJ=1,MNPE
0092      ||                 MQ(JJ) = MQ(JJ) + QMQ(     (JJ-1)*LOFL(JJ)+II)
0092      ||_____         >                     * AMQ(LAMQ((JJ-1)*LOFL(JJ)+II))
          |_______

0094 RECR +-------       DO 200 JJ=MNPE+1,2*MNPE
0095 VECT |+------         DO 200 II=4*MNPE+1,4*MNPE+LOFL(JJ)
0096      ||                 MQ(JJ) = MQ(JJ)+QMQ(     (JJ-MNPE-1)*LOFL(JJ)+II)
0096      ||_____         >                    *AMQ(LAMQ((JJ-MNPE-1)*LOFL(JJ)+II))
          |_______

0098                     RETURN
0099                     END
```

`FQI` Vector Report

Figure 9.6: Selected Portion of the `FQI` Subroutine After Rewriting and Vectorization

```
      COMMON/FBUILD/                                                    $$$$$$
     >  T1L,     T2L,     T3L,     T4L,                                $$$$$$
     >  C10R,    C20R,    T1R,     S10,     F01,  F02,  F03,  F04,  F05,  $$$$$$
     >  C10M1,   C01M1,   C20M1,   D2U,     D2V,  T2R,  S20,  F06,  F07,  $$$$$$
     >  C10M2,   C20M2,   C02M2,   D3V,     D3U,  T3R,  S30,  F08,  F09,  $$$$$$
     >  C10GA,   C20GA,   C10GB,   C20GB,   D4T,  D4U,  D4V,  T4R,  S40,  $$$$$$
     >  QMQ,              LMQ,              LAMQ,       LQMQ               $$$$$$
C                                                                       $$$$$$
      REAL*8                                                            $$$$$$
     >  T1L(4,4),     T2L(4,4),     T3L(4,4),     T4L(4,4),             $$$$$$
     >  C10R(4,4),    C20R(4,4),    T1R(4,4),     S10(4,4),             $$$$$$
     >  C10M1(4,4),   C01M1(4,4),   C20M1(4,4),   D2U(4,4),    D2V(4,4), $$$$$$
.
.
.
      REAL*8                                                            $$$$$$
     >  T1LV(16),     T2LV(16),     T3LV(16),     T4LV(16),             $$$$$$
     >  C10RV(16),    C20RV(16),    T1RV(16),     S10V(16),             $$$$$$
     >  C10M1V(16),   C01M1V(16),   C20M1V(16),   D2UV(16),    D2VV(16), $$$$$$
.
.
.
      INTEGER         LRQ(MRQ),     LMQ(MMQ+MRQ),                       $$$$$$
     >                LARQ(MRQ),    LAMQ(MMQ+MRQ),                      $$$$$$
     >                LQRQ(MRQ),    LQMQ(MMQ+MRQ)                       $$$$$$
C                                                                       $$$$$$
      EQUIVALENCE                                                       $$$$$$
     >   ( AMQ(1) , T1LV  (1) , T1L  (1,1) ),                           $$$$$$
     >              ( T2LV  (1) , T2L  (1,1) ),                         $$$$$$
     >              ( T3LV  (1) , T3L  (1,1) ),                         $$$$$$
     >              ( T4LV  (1) , T4L  (1,1) ),                         $$$$$$
     >   ( ARQ(1) , C10RV (1) , C10R (1,1) ),                           $$$$$$
     >              ( C20RV (1) , C20R (1,1) ),                         $$$$$$
     >              ( T1RV  (1) , T1R  (1,1) ),                         $$$$$$
     >              ( S10V  (1) , S10  (1,1) ),                         $$$$$$
.
.
.
     >              ( LAMQ(MMQ+1), LARQ(1) ),                           $$$$$$
     >              (  LMQ(MMQ+1),  LRQ(1) ),                           $$$$$$
     >              ( LQMQ(MMQ+1), LQRQ(1) ),                           $$$$$$
     >              (  QMQ(MMQ+1),  QRQ(1) )                            $$$$$$
.
.
.
      COMMON/VARTMP/                                                    $$$$$$
     >   DRTMP,     DM1TMP,     DM2TMP,      DGTMP,                     $$$$$$
     >   M1TMP,      M2TMP,       RTMP,       OTMP,       ZTMP,         $$$$$$
     >  UM1TMP,     UM2TMP,      UTMP,       VTMP,     VM1TMP,         $$$$$$
     >  VM2TMP,      UGTMP,     VGTMP,       TTMP,       GTMP,         $$$$$$
     >    PTMP                                                          $$$$$$
C                                                                       $$$$$$
      REAL*8                                                            $$$$$$
     >   DRTMP(4),  DM1TMP(4),  DM2TMP(4),   DGTMP(4),    PTMP(4),     $$$$$$
     >   M1TMP(4),   M2TMP(4),    RTMP(4),    OTMP(4),    ZTMP(4),     $$$$$$
     >  UM1TMP(4),  UM2TMP(4),    UTMP(4),    VTMP(4),  VM1TMP(4),     $$$$$$
     >  VM2TMP(4),   UGTMP(4),   VGTMP(4),    TTMP(4),    GTMP(4),     $$$$$$
     >  ALLTMP(80)                                                     $$$$$$
C                                                                       $$$$$$
      EQUIVALENCE(ALLTMP(1),DRTMP(1))                                   $$$$$$
```

Figure 9.7: Array Equivalencing Used by the Vectorized `FQI` Subroutine

```
      SUBROUTINE SMLOAD(IELEM,JELEM,SQRTER)
      IMPLICIT REAL*8(A-H,O-Z)
      INCLUDE 'COMMON.ONE/NOLIST'
      INCLUDE 'COMMON.THR/NOLIST'

*  routine to "assemble" the jacobian matrix given
*  the element number in the first dimension = IELEM and
*  the element number in the second dimension = JELEM
*
*  the routine is good for general N, NVAR, K
* (up to two dimensions, any number of variables, both
*  linear and quadratic basis) and now factored jacobians
*
.
.
.
      DO 10001 J=1,NVAR**2,NNSKIP

        ID=((J-1)/NVAR)*NPE
        JF=1
        JCT=JCT+1
        JC= N1SKIP(SOLTYP+1)*(NPE*((J-1)/NVAR)+1)
     >    + N2SKIP(SOLTYP+1)*(NPE*(JCT-1)     +1)

      DO 10000 I=1,NPEGO**2

        IF=IF+1
        ID=ID+1
        IPT =IHOLD +KIL(ID         ,1)*JHOLD + KIL(JC         ,1)
        IPT2=IHOLD2+KIL(ID+FACP(IF),2)*JHOLD + KIL(JC+FACP(JF),2)

        ICT=ICT+KADD(I,K,KADNPT)

        IF(FACT)THEN

          SMAT(IPT ,1) = SMAT(IPT ,1) + ZZ1(ICT)*SQRTER
          SMAT(IPT2,2) = SMAT(IPT2,2) + ZZ2(ICT)*SQRTER

        ELSE IF(.NOT.FACT)THEN

          SMAT(IPT ,1) = SMAT(IPT ,1) + ZZ1(ICT)*SQRTER**2

        ENDIF

        IF(MOD(ID,NPEGO).EQ.0)THEN
          ID=((J-1)/NVAR)*NPE
          JC=JC+1
          IF=0
          JF=JF+1
        END IF

10000 CONTINUE

      IF(MOD(J,NVAR).EQ.0)JCT=0
      ICT=ICT+KADD(17,K,KADNPT) + N1SKIP(SOLTYP+1)*NVAR*NPE**2

10001 CONTINUE

      RETURN
      END
```

Figure 9.8: Subroutine SMLOAD as an Example of Code Which Cannot be Vectorized

calculation of the variable IPT, the loop will not vectorize. A rewrite of this loop structure was left for later development. Perhaps the compiler may become more intelligent in the near future!

After utilizing these vectorization techniques throughout the code where possible, quantizing the impact of the effort (vectorization) has on the pilot code performance was accomplished at the CNSF using several CPU timers. The numbers shown in columns 2 through 5 of Table 9.1 are the percentage of the total CPU spent in the corresponding subroutine shown in the first column. The data listed in columns 2 and 3 correspond to the code compiled in scalar mode while the data listed in columns 4 and 5 correspond to the code compiled in vector mode where appropriate. Data in columns 2 and 4 are for small bandwidth (3) quasi-Newton jacobian matrix while data in columns 3 and 5 are for the larger bandwidth (15) quasi-Newton jacobian matrix. The order of the subroutines is in descending order based on column 2. The following list of key words should be utilized to determine the meaning of column 1 of the table.

- JAC computes the local submatrix and array data for both the RHS and LHS data. The JAC subroutine is further subdivided and described as follows:

  -- JACa A scalar loop over every node of a single finite element that computes finite-element averages and temporary arrays in the RHS calculation.

  -- JACb A vector loop which loads multiple temporary arrays into a larger single temporary array for subsequent RHS calculation.

  -- JACc Additional scalar and serial finite-element averaging calculations.

Table 9.1: Comparative Percent CPU Utilization of the IBM-3090 Vector Facility on a Model Problem Utilizing the Pilot Code

Subroutine Sampled	Scalar Bandwidth		Vector Bandwidth	
	3	15	3	15
JAC	42.49	31.18	43.93	28.62
JACa	(4.37)	(3.15)	(6.55)	(4.15)
*JACb	(10.07)	(7.02)	(6.28)	(3.84)
JACc	(.93)	(.62)	(1.19)	(.74)
JACd	(20.57)	(14.12)	(20.91)	(12.74)
JACe	(3.57)	(3.55)	(4.74)	(4.04)
JACf	(2.85)	(2.65)	(4.09)	(2.95)
JACg	(.13)	(.09)	(.22)	(.14)
*FQI	21.10	14.40	20.81	13.56
*PDSCAL	15.41	11.39	7.41	4.88
SMLOAD	6.25	15.24	8.50	19.18
*SOLVR	4.28	20.19	5.85	25.01
VARLOC	3.24	2.38	4.56	2.98
SQRT	1.61	1.12	2.06	1.30
SUBRHS	1.17	.71	2.09	1.03
JACADD	.65	.56	.94	.82
*PAR1	.64	.44	.21	.14
*SOLUTI	.35	.21	.37	.25
*all others	2.81	2.18	3.27	3.26
Effective Work Done				
F_{eff}	.346	.234	.478	.311
Vector Facility Utilization				
F_{vec}	.0	.0	.367	.243

* these particular routine sections have been converted to execute in parallel

  - -- JACd A vector loop of length 16 which computes all the local matrices for loading into the RHS (the largest CPU intensive loop in the entire pilot code).
  - -- JACe Computes the local submatrix data for the ξ sweep.
  - -- JACf Computes the local submatrix data for the η sweep.
  - -- JACg Computes all surface terms.

- FQI Performs matrix multiplication of the RHS local array.
- PDSCAL Sets a vector equal to zero.
- SMLOAD Loads the local quasi-Newton jacobian matrix into the corresponding global array.
- SOLVR Solves the system of equations written as $[A]\{x\} = \{b\}$ where $[A]$ is stored in banded form with a minimum of storage.
- VARLOC Retrieves the integer location of local variables from the global array storage given the local finite element indices.
- SQRT FORTRAN intrinsic function that computes the square root.
- SUBRHS Loads the local RHS array into the global RHS array.
- JACADD Computes the surface terms for both the RHS and LHS local arrays.
- PAR1 A single vector/parallel loop for computing the RHS residual from the time derivative, time step, old (previous time step) data and an implicitness factor.
- SOLUTI Performs the solution procedure including calling the SOLVR subroutine.

- F_{eff} A measure of the efficiency of the computational sequence and is equal to the product of the degrees of freedom and the number of iterations divided by the CPU time in 10^4 seconds. The larger this number, the more efficient the algorithm performance.

- F_{vec} A measure of the VF utilization equal to the CPU spent on the VF divided by the total CPU time. The larger this number, the more time spent on the VF.

The dominate CPU consuming subroutine is JAC, further broken down as JACa through JACg in column 1. The calculation of the RHS, when combined among the appropriate subsets (summing the CPU percentages associated with JACa/2 + JACb + JACc/2 + JACd + FQI + SUBRHS + JACADD/2 from column 4 yields 54.4%), is the entity that dominates CPU consumption. This is followed by the calculation of the left-hand side (LHS, summing CPU percentages from JACa/2 + JACe + JACf + SMLOAD + JACADD/2 the linear-algebra matrix solution in subroutine SOLVR (5.85%). Within the RHS calculation, a loop within subroutine JAC, which calculates the individual elements of the contributing 4 $\times$ 4 matrices, is the dominate CPU consumer (JACd of column 1). From the data shown in Table 9.1, it is obvious that the small bandwidth formulation is more efficient in CPU utilization then the large bandwidth. The overall impact of vectorization is that the code spends approximately 28% and 16% less CPU time in vector mode for the small bandwidth formulation and large bandwidth formulations respectively. This is done by spending approximately 37% and 15% of the total CPU time on the VF for the small and large bandwidth formulation respectively.

The largest impact of CPU consumption in going from the small to large bandwidth formulations is in subroutine SOLVR. Subroutine SMLOAD also consumes

a significant percentage of CPU time in the large bandwidth formulation because of the larger number of submatrix elements that are "stuffed" into the global matrix array. This is because the bandwidth of the quasi-Newton jacobian matrix increases from 3 (tridiagonal) to 15. Furthermore, because the SOLVR subroutine cannot be vectorized, utilization of the VF cannot be realized. This lead to a mini-investigation into the availability of a matrix solver, which utilizes direct methods (as opposed to iterative methods), and which can utilize the VF. The mini-investigation is documented as Appendix E of this dissertation. The final result is that subroutine SOLVR is the best choice until a more efficient, vectorized banded-matrix direct solver can be found or developed.

Several other aspects of the vector timing study reveal some interesting results. Subroutine PDSCAL, which simply sets an entire vector equal to zero, requires significant resources. Subroutine VARLOC, which simply returns integer locations corresponding to a finite element, requires 2% to 5% of the CPU time. The FORTRAN intrinsic function SQRT also consumes 1% to 2% of the CPU time. Because this model problem utilizes ideal gas properties, the fluid property determination (calculated by subroutine CPROP) is minimal. As will be seen in a later section, real-gas fluid properties significantly impact the CPU utilization.

9.3 Parallelization Techniques

Parallel processing is relatively less mature. Presented techniques demonstrate how the algorithm is executed in a given parallel configuration without a major rewrite of the code. Ideas are also discussed on how to improve on the concepts presented.

Parallel processing is the concept of utilizing multiple processors in parallel to perform tasks of a single processor for a net decrease in wall-clock time. A secondary goal of parallel processing is to perform the net work while utilizing the same net resources (CPU, IO's, etc.), and is the most difficult to achieve, i.e., "you can't get something for nothing."

At present, there are two main types of parallel computer configurations: (1) shared-memory; and (2) non-shared memory. The parallel computer configurations utilized in this research, IBM-3090VF/200E, IBM-3090VF/600E, and DEC-VAX-6340, are shared memory machines and are the preferred configuration for minimizing programming effort.

Even with a shared-memory machine, the fundamental rule when transforming a serial code to parallel (or developing a parallel code from scratch) is that memory updating (or writing) among processors upon the shared portion of memory must be minimized and ideally eliminated altogether. In other words, read-only memory access poses no problem for parallel processing, whereas, writing or updating common portions of memory by more than one processor must be controlled. Indeed, the science of parallel processing is now practiced by performing the art of developing code that eliminates common-memory writing among multiple processors.

Two compilers, the IBM Parallel FORTRAN Prototype (PFP) and the DEC VAX FORTRAN Version 5, have been investigated and applied to the pilot code for parallel processing. Both compilers translate standard FORTRAN code and special machine-dependent parallel FORTRAN code (IBM-3090/200E/600E and DEC-VAX-6340 or similar series) to the machine code required for parallel execution on multiple processors. This allows the scientist and engineer, who cannot be bothered by details of the machine code, to focus on the algorithm

development at a high-level coding language. Because the IBM compiler and computer are currently superior to that of the VAX, application focused on the IBM machine. Where possible, machine-dependent coding was minimized.

As with vectorization, the simplest FORTRAN statement sequence to parallelize is the DO loop (currently, the only VAX parallel function in FORTRAN is the DO loop). Figure 9.9 shows how the PFP compiler translates simple DO loops into parallel code. The original source is identical to the serial counterpart as shown in Figure 9.4. The parallel code is generated by accessing the PFP compiler and declaring a parallel directive. The parallel compilation report indicates PAVE for parallel and vector loop whereas the vector compilation report indicates VECT for a serial and vector loop.

```
0001                    SUBROUTINE SWITCH

                *   ROUTINE TO TRANSFER RESIDUAL WORKING AREAS FOR
                *   FACTORED JACOBIAN ALGORITHMS

0088                    LOGICAL*1 TYPE/.TRUE./

0089                    IF(TYPE)THEN

                C ZETA DIRECTION TO ETA DIRECTION

                CVDIR PREFER VECTOR
                CPDIR PREFER VECTOR PARALLEL
0090 PAVE +-------        DO 10000 I=1,NNODE
0091      |_______          RERR(ZTET(I))=RHSV(ZTEF(I))

0092                    ELSE IF(.NOT.TYPE)THEN

                C ETA DIRECTION TO ZETA DIRECTION

                CVDIR PREFER VECTOR
                CPDIR PREFER VECTOR PARALLEL
0093 PAVE +-------        DO 10001 I=1,NNODE
0094      |_______          RERR(ETZT(I))=RHSV(ETZF(I))

0095                    ENDIF

                CVDIR PREFER VECTOR
                CPDIR PREFER VECTOR PARALLEL
0096 PAVE +-------        DO 10002 I=1,NNODE
0097      |_______          RHSV(I)=RERR(I)

0098                    TYPE=.NOT.TYPE

0099                    RETURN
0100                    END
```

Figure 9.9: Parallelization of Subroutine SWITCH of the Pilot Code

Another DO loop parallelization technique is the PARALLEL LOOP statement (PARALLEL DO in VAX FORTRAN). Figure 9.10 shows how the FQI subroutine is translated to execute in parallel utilizing the PARALLEL LOOP statement. Comparing Figure 9.10 with the serial counterpart shown in Figure 9.6, the outer loops are PARALLEL LOOP statements as opposed to serial DO statements. The parallel report indicates PARA rather than ELIG or RECR in the serial-vector report of Figure 9.6.

A DO loop does not have to be a vectored loop to execute in parallel. However, in a similar manner to the minimum threshold for vectorization, IBM recommends that approximately 128 contiguous bytes be available per processor[134] in order for parallel processing to be economical. As will be seen, considerable overhead is generated by the PFP compiler in order to execute code in parallel. Therefore, not all DO loops were converted to parallel in the code. Indeed, it was not possible for all DO loops to be converted because of the memory conflict problems discussed earlier.

One additional parallelization technique, the concept of "originated tasks" was also incorporated into the code. Subroutine SOLUTI, as shown in Figure 9.11, was converted to execute in parallel utilizing subroutine SOLVR as an originated task. The original equivalent serial counterpart SOLUTI subroutine is reproduced by eliminating the non-standard FORTRAN and replacing the CALLING statement with a standard FORTRAN CALL statement. The net effect is to allocate the linear-algebra solution among multiple processors instead of a single processor. The only real trick is to determine the end node within a solution block such that it lies at the end of a solution line. This is because the last node on a line is not coupled to the first node on the next line due to the structure of the approximate quasi-Newton jacobian matrix.

```
          1          SUBROUTINE FQI
         88          INTEGER LOFL(2*MNPE)/MNPE*4,MNPE*36/

                   * routine to load the rhs vector F = [J] * [&Q]

         89          CALL PDSCAL(2*MNPE,ZERO,MQ)

         90          IF(.NOT.IRK)THEN
  1      91            PARALLEL LOOP 100 II=1,4
  1  1   92              DO 100 JJ=1,MNPE
  1  2   93                MQ(JJ) = MQ(JJ) + QMQ(     (JJ-1)*LOFL(JJ)+II)
  1  2                  >                     * AMQ(LAMQ((JJ-1)*LOFL(JJ)+II))
  1  2   94       100      CONTINUE
  1      95          ENDIF

         96          PARALLEL LOOP 200 JJ=MNPE+1,2*MNPE
     1   97            DO 200 II=4*MNPE+1,4*MNPE+LOFL(JJ)
     2   98              MQ(JJ) = MQ(JJ) + QMQ(     (JJ-MNPE-1)*LOFL(JJ)+II)
     2                 >                     * AMQ(LAMQ((JJ-MNPE-1)*LOFL(JJ)+II))
     2   99       200    CONTINUE

        100          RETURN
        101          END
```

FQI Listing Prior to Parallelization

```
0001                     SUBROUTINE FQI
0088                     INTEGER LOFL(2*MNPE)/MNPE*4,MNPE*36/

                   * ROUTINE TO LOAD THE RHS VECTOR F = [J] * {&Q}

0089                     CALL PDSCAL(2*MNPE,ZERO,MQ)

0090                     IF(.NOT.IRK)THEN
0091 PARA                  PARALLEL LOOP 100 II=1,4
0092 VECT +--------          DO 100 JJ=1,MNPE
0093      |                    MQ(JJ) = MQ(JJ) + QMQ(     (JJ-1)*LOFL(JJ)+II)
0093      |_______           >                     * AMQ(LAMQ((JJ-1)*LOFL(JJ)+II))
0095                       ENDIF

0096 PARA                  PARALLEL LOOP 200 JJ=MNPE+1,2*MNPE
0097 VECT +--------          DO 200 II=4*MNPE+1,4*MNPE+LOFL(JJ)
0098      |                    MQ(JJ) = MQ(JJ) + QMQ(     (JJ-MNPE-1)*LOFL(JJ)+II)
0098      |_______           >                     * AMQ(LAMQ((JJ-MNPE-1)*LOFL(JJ)+II))

0100                       RETURN
0101                       END
```

FQI Listing Showing Parallel Report

Figure 9.10: Parallelization of Subroutine FQI of the Pilot Code

```
0001                        SUBROUTINE SOLUTI
0088                        INTEGER*4 ITASK(MAXPRC)

0089                        IF(MAXPRC.LT.NUMPRC)THEN
0090                          WRITE(4,12345)NUMPRC,MAXPRC
0091                  12345   FORMAT(' Error in SOLUTI. NUMPRC =',I5,' MAXPRC =',I5)
0092                          STOP
0093                        ENDIF

                    * ROUTINE TO SOLVE A BANDED MATRIX EQUATION SET IN TWO SWEEPS

0094                        CALL FIXED
0095 UNAN                   DO 10000 KLOOP=1,NUMPRC
0096                  10000   ORIGINATE ANY TASK ITASK(KLOOP)
0097 UNAN                   DO 10001 KLOOP=1,NUMPRC
0098                          NLOOP  = NNODE/NI/NUMPRC+1
0099                          NLOOP  = NLOOP*NI
0100                          KSTART = (KLOOP-1)*NLOOP+1
0101                          KEND   = MIN(NNODE,KLOOP*NLOOP)
0102                          KSMAT  = (KSTART-1)*NBAND+1
0103                          KLENG  = KEND-KSTART+1
0104                          SCHEDULE TASK ITASK(KLOOP),
0104                       >  CALLING SOLVR(KLENG,NBAND,SMAT(KSMAT,1),RHSV(KSTART))
                    C           CALL SOLVR(KLENG,NBAND,SMAT(KSMAT,1),RHSV(KSTART))
0105                10001 CONTINUE
0106                        WAIT FOR ALL TASKS
0107 UNAN                   DO 10002 KLOOP=1,NUMPRC
0108                  10002   TERMINATE TASK ITASK(KLOOP)

0109                        IF(FACT)THEN
0110                          CALL SWITCH
0111                          CALL FIXED
0112 UNAN                     DO 10003 KLOOP=1,NUMPRC
0113                  10003     ORIGINATE ANY TASK ITASK(KLOOP)
0114 UNAN                     DO 10004 KLOOP=1,NUMPRC
0115                            NLOOP  = NNODE/NJ/NUMPRC+1
0116                            NLOOP  = NLOOP*NJ
0117                            KSTART = (KLOOP-1)*NLOOP+1
0118                            KEND   = MIN(NNODE,KLOOP*NLOOP)
0119                            KSMAT  = (KSTART-1)*NBAND+1
0120                            KLENG  = KEND-KSTART+1
0121                            SCHEDULE TASK ITASK(KLOOP),
0121                         >  CALLING SOLVR(KLENG,NBAND,SMAT(KSMAT,2),RHSV(KSTART))
                    C           CALL SOLVR(KLENG,NBAND,SMAT(KSMAT,2),RHSV(KSTART))
0122                10004     CONTINUE
0123                          WAIT FOR ALL TASKS
0124 UNAN                     DO 10005 KLOOP=1,NUMPRC
0125                  10005     TERMINATE TASK ITASK(KLOOP)
0126                          CALL SWITCH
0127                        ENDIF

0128                        IF(LDETJ)THEN

                    CVDIR PREFER VECTOR
                    CPDIR PREFER VECTOR PARALLEL
0129 PAVE +-------          DO 10006 I=1,NNODE
0130      |_______            RERR(I)=RHSV(I)/DETJST(I)

                    CVDIR PREFER VECTOR
                    CPDIR PREFER VECTOR PARALLEL
0131 PAVE +-------          DO 10007 I=1,NNODE
0132      |_______            RHSV(I)=RERR(I)

0133                        ENDIF

0134                        RETURN
0135                        END
```

Figure 9.11: Parallelization of Subroutine SOLUTI of the Pilot Code

Where possible, without a major redesign of the algorithm or rewrite of the code, the entire code was converted to execute in parallel. The asterisks shown adjacent to the subroutine names in column 1 of Table 9.1 indicate that this portion of the serial code has been converted to parallel. There are also many other portions of the code, not shown in Table 9.1, which have been converted to parallel equivalent. Approximately 44% of the code's CPU time in the most efficient mode of execution (column 3 of Table 9.1) has been converted to parallel capability. The code was carefully tested to ascertain that the correct results were obtained. It was often easy to restructure and compile a subroutine in parallel that would appear (based on looking at the code) to execute correctly in parallel. However, upon actually executing the code in parallel, the results would be very similar but not the same (i.e., wrong!). Usually, careful cross examination revealed a subtle, but deadly, shared-memory update dependence. Many times, the cause of the parallel problem was not obvious. This forced the code to be reverted back to serial mode. Consistent with IBM recommendations, many portions of the code that could not be vectorized could not be parallelized.

Another compilation problem appeared due to incorrect documentation of the default parameter options of the compile step itself. The PFP compiler is supposed to compile (at least the documentation states) with `NOVECTOR` as the default parameter. In fact, the PFP compiler has `VECTOR` as the default parameter at CNSF. Until discovered, this caused significant degradation in performance due to improper vectorization of small loops.

Another problem is that PFP syntax and code optimization is equivalent to IBM FORTVS version 2.1 whereas the serial IBM FORTVS compiler is at version 2.3. There are significant syntax differences between FORTVS version 2.1 and 2.3. The code was not bench marked using FORTVS version 2.1 in

serial mode to determine the effect of the later compiler version on the code optimization and performance.

The code was carefully benchmarked utilizing a model problem on the IBM-3090VF/600E at CNSF under the VM/XA-CMS operating system and on the IBM-3090VF/200E at UTCC under the MVS/XA operating system. The benchmarks were executed on both machines in parallel mode utilizing the FORTVS Version 2.1 PFP compiler and in serial mode utilizing the FORTVS Version 2.3 compiler. At CNSF, the code executed correctly on up to five processors simultaneously with a single processor apparently dedicated to the CMS operating system (there are six processors on each machine). At UTCC, the code executed correctly on up to two processors. The batch MVS/XA operating system at UTCC allowed for both of its processors to execute simultaneously. CNSF had extensive CPU timer utilities available for measuring each processor's individual performance while UTCC presently does not have any CPU timers that will work in parallel mode. Furthermore, CNSF has a second IBM-3090VF/600E machine which eventually will simultaneously execute with the first IBM-3090VF/600E in a parallel mode. However, this option did not function at this writing.

The measured CPU time (at CNSF) was essentially the same across each processor. This was a surprise because the expected response is for one processor (root task) to have significantly more CPU time accumulation while the other processors (originated tasks both deliberate in subroutine `SOLUTI` and as a consequence of parallel `DO` loops) would have less CPU time accumulation. Apparently because the IBM-3090VF operates in a time-sharing mode with other jobs in the system, the execution is continuously and randomly swapped among the processors. Apparently this is true even when executing in serial mode. Therefore, the CPU time of individual processors need not be quoted.

Because the IBM-3090VF is a virtual, time-sharing machine, the actual savings in wall-clock time as a result of parallel processing cannot be directly measured. In order to arrive at an estimate of the desired savings in wall-clock time, it is assumed that CPU time is equal to wall-clock time. This would be essentially the same as if the machine were dedicated to a single process.

The results of the parallel-processing benchmark of the CFD2D code on the IBM-3090VF are shown in Table 9.2. In order to properly interpret the results shown in the table, the following definitions are stated:

- F_{eff} A measure of the efficiency of the computational sequence and is equal to the product of the degrees of freedom and the number of iterations divided by the CPU time in 10^4 seconds. The larger this number, the more efficient the algorithm performance.
- F_{vec} A measure of the VF utilization equal to the CPU spent on the VF divided by the total CPU time. The larger this number, the more time spent on the VF.
- F_{eff_o} The value of F_{eff} for a single processor in serial mode.
- WORK A measure of the work accomplished by the execution computed by the quantity F_{eff_o}/F_{eff}.
- WALL An estimated wall-clock time computed by the quantity WORK / NPROCS, where NPROCS is the number of processors.
- SUF Speed-up factor computed as the inverse of WALL.
- C/B Cost benefit ratio computed as WORK / SUF.

Theoretically, if the parallel processing were ideal, the expected the speed-up factor (SUF) should equal to the number of processors (NPROCS). Furthermore,

Table 9.2: Parallel Performance of the IBM-3090 Vector Facility on a Model Problem Utilizing the Pilot Code

NPROCS	F_{vec}	F_{eff}	WORK	WALL	SUF	C/B
CNSF IBM-3090VF/600E small bandwidth formulation						
0	.367	.478	1.000	1.000	1.000	1.000
1	.343	.385	1.244	1.244	0.804	1.547
2	.329	.338	1.416	0.708	1.412	1.003
3	.239	.228	2.100	0.700	1.429	1.470
4	.244	.228	2.100	0.525	1.905	1.102
5	.210	.193	2.478	0.496	2.016	1.229
CNSF IBM-3090VF/600E large bandwidth formulation						
0	.243	.311	1.000	1.000	1.000	1.000
1	.240	.261	1.189	1.189	0.841	1.414
2	.238	.230	1.351	0.676	1.480	0.913
3	.211	.187	1.658	0.553	1.808	0.917
4	.180	.155	2.003	0.501	1.996	1.004
5	.158	.134	2.323	0.465	2.151	1.080
UTCC IBM-3090VF/200E small bandwidth formulation						
0	.365	.503	1.000	1.000	1.000	1.000
1	N/A	.398	1.261	1.261	0.793	1.590
2	N/A	.309	1.628	0.814	1.229	1.325
UTCC IBM-3090VF/200E large bandwidth formulation						
0	.240	.305	1.000	1.000	1.000	1.000
1	N/A	.262	1.165	1.165	0.858	1.358
2	N/A	.202	1.509	0.755	1.325	1.139

for any value of the cost/benefit ratio that is less than unity, parallel processing is beneficial.

The results indicate several departures from ideal parallelism. When the number of processors (NPROCS) goes from zero (relating to a single processor in serial mode) to one (relating to a single processor in parallel mode), a significant degradation in performance occurs. Apparently, this can be exclusively attributed to the PFP overhead. Furthermore, as each processor is included in the execution, about half of the expected work is actually accomplished while the amount of time spent on the vector facility correspondingly decreases (the time spent on the vector facility is expected to remain approximately constant since sufficient vector length exists). The speed-up factor (SUF) is less that half of the expected result. In all cases, however, the wall-clock time is significantly reduced. Assuming that computing costs are of concern, (predominantly CPU time for this type of executing), and that the wall-clock time equals the CPU time (for the sake of comparison), the cost-benefit ratio (C/B) is seldom less than unity (only for 2 and 3 processors for the large bandwidth problems at CNSF). It is also interesting that the CNSF and UTCC results are slightly different over the applicable range of processors. Apparently, this can be attributed to the different operating systems and perhaps different compiler versions.

In summary, this research aspect found that parallel processing of the code without a major algorithm modification or code rewrite is only marginally beneficial. The cost of CPU time in the research environment is not critical. However, because the computer is configured as a time-sharing machine, the actual wall-clock time could be even greater in parallel-processing mode if it is realized that parallel execution is contributing significantly to the overall load on the machine. If the cost-benefit ratio is defined as in Table 9.2, the following

equation can be written as

$$\frac{\text{Cost}}{\text{Benefit}} = \frac{C}{B} = \frac{\text{WORK}^2}{\text{NPROCS}}. \tag{9.1}$$

Since the work accomplished tends to flatten as the number of processors increases, it is expected that the cost-benefit ratio to become smaller as the number of processors increases. This of course assumes that in the future the efficiency of the parallel compiler will be the same or better.

The algorithm described herein possesses many modes of parallelism. Application of these alternative parallel schemes was not investigated but discussing them here is appropriate. To begin with there are many other IBM parallel FORTRAN extensions that could be investigated to get around the shared-memory conflict problems inherent in the present CFD2D code. In particular, the LOCK statement might be used to reserve particular memory locations during the write process and protect multiple processors from simultaneously updating the same location. This might increase the parallelism significantly in the present code but would require a major effort.

Another idea that would involve some additional rewriting of the code is the concept of "block parallelism." Almost the entire code outside the finite element loop (see Figure 9.3) will presently execute in parallel. The difficult portion of the code to transform into parallel and/or vector mode is the finite-element assembly loop. The concept is to allocate blocks of finite elements to the number of processors available. The finite-element assembly loop would then be processed on each block as if they were independent. After the finite-element loop is completed, the separate blocks of left- and right-hand sides would be combined to form the total system as before.

A further extension of the "block parallelism" idea is a family of algorithms which have come to be called Element-by-Element (EBE) methods.[135] The EBE

method could potentially utilize many processors. From the EBE algorithm standpoint, the number of processors would be limited to the number of finite elements in the solution domain. Furthermore, there are many places in the code where a vector processor can be utilized at an element level. Incorporating an EBE method into the present algorithm could utilize the capabilities of modern hypercube parallel computers. Another feature of EBE methods is that it can be demonstrated on a serial computer.

9.4 Impact of Real-Gas Closure

Fluid properties are calculated by specification of the NUPD variable (see Appendix D). If NUPD is zero, an ideal gas is assumed and viscosity and thermal conductivity are held constant at a non-dimensional value of unity, i.e., fixed at the dimensional value of the reference stagnation point. All benchmarks quoted previously were calculated with the ideal gas option (NUPD=0).

If NUPD is set to a positive non-zero integer, then real-gas relationships are utilized to determine all fluid properties. If NUPD equals 1 (recommended), then fluid properties are updated at each time step iteration. If NUPD equals 2, then fluid properties are updated every two time steps, and so forth. An equilibrium real-gas closure package[95] has been interfaced for this purpose. The purpose of this section is to document the impact of including this real-gas closure procedure on supercomputer performance.

A model problem was chosen (subsonic, adiabatic, inviscid, essentially isothermal flow) for a sensitivity study so that only minor differences between the ideal-gas and real-gas fluid properties would result and is analogous to the small bandwidth, vector, serial problem quoted previously. Two benchmarks were executed: (1) NUPD=0, specifying ideal gas, and (2) identical to the first

case except NUPD=1, specifying real-gas fluid property updates at each time-step iteration. The performance measure is the F_{eff} factor. These calculations determined that $F_{eff} = 0.495$ for ideal gases, and $F_{eff} = 0.457$ for real gases. This result indicates that a nominal 8% performance degradation results when including real-gas effects in the calculation.

The algebraic turbulence model has a more significant effect on the computational efficiency. Typically an additional 10% $\rightarrow$ 20% reduction in F_{eff} occurs when the algebraic turbulence model is implemented (independent of the method of determining the equation of state). The computational impact range is because two different turbulence models were investigated, Baldwin-Lomax and Cebeci-Smith, with the later requiring significantly more computation from the need to compute the boundary layer displacement thickness.

Chapter 10

Summary and Conclusions

The following commitment summary from the dissertation proposal[6] clarifies the originality of this Ph.D. research and summarizes the conclusions found (original commitment shown in boldface type):

- A **compressible 2D Navier-Stokes equation system with volume-specific stagnation enthalpy as the energy variable** has been derived and transformed to dimensionless form. A generalized coordinate transformation has been derived and implemented into the algorithm.

- Both an **ideal- and real-gas thermodynamic equation of state for air** have been integrated into the algorithm, verified operational and compared with known data.

- **Laminar and two forms of algebraic turbulent modeling of physical dissipation** have been integrated into the algorithm, verified operational and compared with known data.

- Two reduced forms of the governing equations have been derived and researched. A quasi-1D form has been implemented to establish initial conditions for the 2D problem. A detailed analysis of a PNS form of

the equations has determined the stability envelope for space-marching applicability. The essential theoretical development of a finite-element based PNS algorithm and code was presented. A reduced PNS form of the governing equations was used to validate finite element convergence rates for the compressible viscous case.

- A **Taylor Weak Statement for numerical dissipation effects** has been derived and implemented into the algorithm. Several formulations of the inviscid flux vector jacobian matrix, primarily dependant on the treatment of the momentum pressure gradients, have been examined with conclusions reached. A filtering function for eliminating artificial dissipation where physical dissipation is important has been derived and implemented.

- A **finite element algorithm** semi-discretization has been derived and implemented taking advantage of nomenclature and other accomplishments of past research. A functional 2D grid generation procedure has been integrated into supported use of the finite element algorithm.

- A **nonlinear, iterative solution algorithm utilizing a weighted approximation variably-implicit time integration** has been implemented to form the fully-discrete approximation. Both a quasi-Newton iteration and a semi-implicit Runge-Kutta (non-iterative) procedure have been implemented and validated.

- A **comparison of various tensor-product matrix factorization schemes and arrival of a recommended approach** has been completed. Full details of the quasi-Newton jacobian matrix and factored approximations have been disclosed.

- Additional reduction in jacobian complexity, resulting in significant computer resource requirements, has been investigated. Recommendations for the proper implementation of the reduced jacobian have been made.

- Well-posed Dirichlet and Neumann boundary conditions have been theorized and confirmed. Special cases for flow tangency, outflow boundaries, blowing, momentum flux, and heat flux boundary conditions have also been investigated.

- A unique approach to initial condition generation, based on a vigorous implementation of restart capability, has been implemented with a minimal amount of manual input and manipulation of data.

- **Confirmation of stability, convergence (time and space), and consistency** has been performed. This has resulted in a unique time-step selection criteria, an on-line stability monitor, and implementation of a digital filter of the time-step frequency.

- A **demonstration of the algorithm on three independent problems** has been completed. Problems of increasing complexity allowed for independent confirmation of algorithm essentials. Comparisons with analytical, experimental, and independent calculations were made giving strong credibility to both the algorithm theory and the results.

- Finally, an **efficient utilization of supercomputer (vector/parallel) resources** with complete details was demonstrated in a production environment.

Chapter 11

Recommendations for Future Research

Although the objectives listed in the previous chapter have certainly been meet, during the course of the research, certain other questions and interesting related topics were also uncovered. The following itemized discussion attempts to recount many of these topics for which future scholars and/or engineering scientists may want to research:

- **Turbulence models-** The algebraic turbulence models utilized in this research were certainly adequate to meet the dissertation objective. It has been demonstrated that the developed algorithm was capable of handling mesh and computational complexities for turbulent simulations. However, the algebraic models were definitely deficient in several ways including: (1) the independence of the model along wall-normal grid lines resulting in the appearance of oscillations in the streamwise direction, (2) the only mechanism of control of the placement of turbulent dissipation was by varying the turbulent length scale, indirectly via the Reynolds number, resulting in a global rather than a local change in dissipation, and (3) the computational burden imposed by the algebraic equations is not trivial.

Thus, the incorporation of a differential turbulence model, such as the two-equation $k - \epsilon$ model, is certainly worth investigating.

- **Grid generation-** The grid generation procedure incorporated was functional but primitive. Each time a new grid was needed, a significant amount of manual effort was required. The integration of more automated regridding is certainly possible and is highly recommended.

- **Parallel processing-** A few methods of incorporation of parallel processing into the algorithm were investigated during this research. The fundamental criteria for incorporation of these methods was to minimize the impact on the current computer code structure. Several other methods and/or ideas should be investigated that could prove more promising. It is proposed herein that the most rewarding of these ideas might be based on "physical" methods, i.e., break the problem up based on the physics of the problem and not purely based on ideas from mathematics and/or computer science. For example, the physical domain (grid) could be divided into blocks each independently solved in parallel. Another example is to define the blocks based on time-step size and hence resource requirements. These methods have been called "domain decomposition."

- **TWS expansion-** This dissertation project has considered only the β term in the TWS theoretical formulation. Related research has demonstrated some success with including the other TWS terms (α, μ, γ).[28,25] This success could be verified and extended within the present pilot code, as most of these terms are already included.

- **Thermal-hydraulics-** The nuclear industry dominates the utilization of thermal-hydraulic (T-H) computer simulations. The large majority of these computer analyses/codes are limited to 1D, unsteady calculations based on

explicit and/or non-fully implicit time integration schemes. The significant complication in all the approaches is the determination of the two-phase (liquid-vapor) thermodynamics and interactions; a non-trivial task! In general, the development of T-H technologies has focused on the proper physics and remained far behind the state of the art in numerical algorithm implementation. As a result, even the 1D calculations take huge amounts of computer cpu time, and the multi-dimensional problem is usually not even considered possible, i.e., the more accurate multidimensional solution is compromised. Therefore, there is a monumental amount of research and development that could be carried out in the T-H field, and the algorithm fundamentals discussed in this dissertation are certainly applicable.

- **Nonlinear stability-** The issue of nonlinear stability was addressed in this dissertation and was a principal focal point in related research.[35] The knowledge base is essentially untapped in this field. The algorithm/code developed during this research could serve as a test bed for more detailed analyses on the stability issue.

- **Time accuracy-** Not a goal of this research project, the important problem of time-accurate unsteady solutions is possible to examine using the iterative procedures described and contained in the code.

- **3D extension-** Because of the straight forward manner in which the finite element algorithm discussed in this dissertation has been developed,[87] the extension to 3D is a straight forward, but difficult task. However, the pilot code has been written in a modular manner such that a 3D extension could be done without a complete rewrite (see the next item).

- **PNS coupling-** An ongoing project is the research and development of a finite element based space marching or PNS algorithm.[86] The theoretical

quasi-1D, steady equation development and implementation is quite similar to the volume specific stagnation enthalpy formulation developed herein. Furthermore, the developed PNS code is a direct extension of the dissertation code, i.e., specified on input as an option. Therefore, a natural extension would be to continue the PNS code development to provide a direct initial condition generation for the PNS space marching within the same code. This development could be further enhanced by development of 3D initial condition generator for PNS.

BIBLIOGRAPHY

Bibliography

[1] Stephen Whitaker. *Introduction to Fluid Mechanics*. Prentice-Hall, Englewood Cliffs, NJ, 1968.

[2] C. M. Allison et al. *SCDAP/RELAP5/MOD2 Code Manual, Volumes 1-3*. Technical Report EGG-2555 and NUREG/CR-5273, Idaho National Engineering Laboratory, June 1989.

[3] EI International Inc. *RETRAN-02-A, Program for Transient Thermal-Hydraulic Analysis of Complex Fluid Flow Systems*. Technical Report EPRI-NP-1850-CCM-A, Electric Power Research Institute, 1988.

[4] D. R. Liles et al. *TRAC-PF1/MOD1 Correlations and Models*. Technical Report LA-11208-MS, Los Alamos National Laboratory, 1988.

[5] University of California, Davis. *4th International Symposium on Computational Fluid Dynamics*, Davis, CA, USA, September 1991. A Collection of Technical Papers, Vol. I, pages 1-692, Vol. II, pages 693-1384, Vol. III (late papers) pages 1-112.

[6] James D. Freels. A taylor weak-statement finite element algorithm for real-gas computational fluid dynamics. A Proposal for a Dissertation Topic Submitted to Department of Engineering Science and Mechanics, The University of Tennessee, Knoxville, December 5, 1988.

[7] A. J. Baker and P. D. Manhardt. *An Arbitrary Grid CFD Algorithm for Configuration Aerodynamics Analysis*. Technical Report COMCO-TR-87-2.1, NASA SBIR Phase I Contract NAS2-12347, 1987.

[8] A. J. Baker. On current aspects of finite element computational fluid mechanics for turbulent flows. *Computer Methods in Applied Mechanics and Engineering*, 32(1-3):261--282, Sept. 1982.

[9] A. J. Baker and Soliman M. O. Current topics on finite element analysis for flows with large reynolds number. In Tadahiko Kawai, editor, *Finite Element Flow Analysis, Proceedings of the Fourth International Conference on Finite Element Methods in Flow Problems*, pages 129--136, Chuo University, Tokyo, Japan, July 1982. University of Tokyo Press.

[10] A. J. Baker. A finite element algorithm for the navier-stokes equations. In W.G. Habashi, editor, *Computational Methods in Viscous Flow*, chapter 9, pages 309--340, Pineridge Press Ltd., Swansea, U.K., 1984. vol.3, Section 2 (ISBN 0-906674-27-1).

[11] A. J. Baker. Finite element algorithm for high-speed flow. In *Finite Elements in Fluids, Volume 5*, pages 197--214, John Wiley & Sons, Chichester, England and New York, NY, USA, 1984. Presented at the Fifth International Symposium of Finite Elements for Flow Problems, Tokyo, Japan, 1982.

[12] A. J. Baker. *An Improved CFD Design Analysis for Hypersonic Air-Breathing Engine Inlets*. Technical Report, Department of the Navy, Naval Sea Systems Command, 1985.

[13] A. J. Baker, J. D. Freels, J. W. Kim, and J. A. Orzechowski. On a finite element cfd algorithm for compressible viscous and turbulent aerodynamic flows. *International Journal for Numerical Methods in Fluids*, 7:1235--1259, 1987. also in Finite Elements in Fluids - Volume 7, chapter 13, the 6th International Conference on Finite Elements in Flow Problems, John Wiley & Sons, 1988, pages 247-272.

[14] A. J. Baker, G. S. Iannelli, and W. P. Noronha. Accuracy, convergence and stability of finite element cfd algorithms. In T.J. Chung and G.R. Karr, editors, *Proceedings of the Seventh International Conference on Finite Element Methods in Flow Problems*, pages 782--789,1609, University of Alabama in Huntsville Press, Huntsville, AL, April 1989.

[15] A. J. Baker, J. D. Freels, and G. S. Iannelli. *Finite Element Computational Aerodynamics-A Response to a Grand Challenge.* Technical Report CFDL/90-1.1, College of Engineering, University of Tennessee, Knoxville, Tennessee, January 1990. prepared for the 1989 IBM 3090 Supercomputing Competition, Sponsored by the IBM Corporation, USA.

[16] A. J. Baker. Finite element weak statement cfd algorithms for fluid-thermal system analysis. In M.R. Heller, editor, *Proceedings of the 2nd nuclear simulation symposium and mathematical modeling workshop*, pages 142--156,248, Springer, Berlin, Germany, October 1990. sent to DOE from Federal Republic of Germany.

[17] A. J. Baker and J. D. Freels et al. *Progress on an Accurate and Efficient Finite Element CFD Algorithm for Hypersonic External Aerothermal Flowfield Prediction.* Technical Report CFDL/85-5, College of Engineering, University of Tennessee, December 1985. for NASA under Grant NAG1-319/Phase II by UT Computational Fluid Dynamics Laboratory.

[18] A. J. Baker, J. D. Freels, et al. An arbitrary grid aerodynamics cfd algorithm. In *AIAA 8th Conference on Computational Fluid Dynamics*, June 1987.

[19] A. J. Baker, J. D. Freels, G. S. Iannelli, J. W. Kim, and W. P. Noronha. *Advances in Finite Element Analysis for CFD Applications*. Technical Report CFDL/88-1, College of Engineering, University of Tennessee, February 1988. for International Conference on Computational Methods in Flow Analysis, Okayama, Japan, September, 1988, by UT Computational Fluid Dynamics Laboratory.

[20] A. J. Baker, G. S. Iannelli, and J. D. Freels. Recent advances on a finite element algorithm for computational aerodynamics: transonics -- hypersonics. In *4th International Symposium on Computational Fluid Dynamics*, pages 48--53, University of California, Davis, Davis, California, September 1991. a collection of technical papers.

[21] Baker A. J. and M.O. Soliman. On the utility of finite element theory for computational fluid dynamics. In *Collection of Tech. Papers, AIAA 5th Computational Fluid Dynamics Conference*, pages 333--346, Palo Alto, CA., June 1981. AIAA Paper No.81-1031.

[22] A. J. Baker and M. O. Soliman. On the accuracy and efficiency of a finite element tensor product algorithm for fluid dynamics applications. *Computer Methods in Applied Mechanics and Engineering*, 27(2):215--237, July 1981.

[23] A. J. Baker and M. O. Soliman. A finite element algorithm for computational fluid dynamics. *AIAA Journal*, 21(6):816--827, June 1983.

[24] Mohamed Osama Soliman. *Accuracy and Convergence of a Finite Element Algorithm for Computational Fluid Dynamics*. PhD thesis, The University of Tennessee, Knoxville, Tennessee, June 1978.

[25] George Elliot Moore. *An Analysis of Factors Affecting the Convergence and Efficiency of an Implicit Finite Element Algorithm for Solution of the Quasi-One Dimensional Euler Equations*. Master's thesis, The University of Tennessee, Knoxville, TN, June 1984.

[26] A. J. Baker and J. W. Kim. *Analysis of a Taylor Weak-Statement Algorithm for CFD Applications*. Technical Report CFDL/85-2, College of Engineering, University of Tennessee, Knoxville, Tennessee, August 1985.

[27] A.J. Baker and J.W. Kim. A taylor weak statement for hyperbolic conservation laws. *Int. J. Num. Mtd. Fluids*, 7:489--520, 1987.

[28] Jin Whan Kim. *On a Taylor Weak Statement for Finite Element Computations in Gas Dynamics*. PhD thesis, The University of Tennessee, December 1988.

[29] Wilbert Paul Noronha. *Accuracy, Convergence, and Stability of Finite Element Algorithms for Incompressible Flow*. PhD thesis, The University of Tennessee, December 1989.

[30] A. J. Baker, W. P. Noronha, and E. L. Wachspress. Consistent sparse factorization of elliptic difference equations. *Applied Mathematics Letters*, 2:215--220, 1989.

[31] G.S. Iannelli and A.J. Baker. A stiffly-stable implicit runge-kutta algorithm for cfd applications. AIAA 26th Aerospace Sciences Meeting, AIAA-88-0416, January 1988.

[32] G.S. Iannelli and A.J. Baker. An implicit and stiffly stable finite element cfd algorithm for unsteady aerodynamics. AIAA 27th Aerospace Sciences Meeting, AIAA-89-0656, January 1989.

[33] G.S. Iannelli and A.J. Baker. An efficient solution-adaptive implicit finite element cfd navier-stokes algorithm. AIAA 28th Aerospace Sciences Meeting, AIAA-90-0400, January 1990.

[34] G.S. Iannelli and A.J. Baker. An intrinsically n-dimensional generalized flux vector splitting implicit finite element euler algorithm. AIAA 29th Aerospace Sciences Meeting, AIAA-91-0123, January 1991.

[35] Giuseppe S. Iannelli. *A Globally Well Posed Accurate and Efficient Finite Element CFD Algorithm for Compressible Aerodynamics*. PhD thesis, The University of Tennessee, Knoxville, Tennessee, 1991.

[36] J.D. Freels and A.J. Baker. Progress on a taylor weak statement finite element algorithm for high speed aerodynamic flows. AIAA 27th Aerospace Sciences Meeting, AIAA-89-0654, January 1989.

[37] T. J. R. Hughes, T. E. Tezduyar, and A. N. Brooks. Streamline upwind formulations for advection-diffusion, navier-stokes, and first-order hyperbolic equations. In Tadahiko Kawai, editor, *Finite Element Flow Analysis, Proceedings of the Fourth International Conference on Finite Element Methods in Flow Problems*, pages 97--104, Chuo University, Tokyo, Japan, July 1982. University of Tokyo Press.

[38] T. J. R. Hughes and M. Mallet. A high-precision finite element method for shock-tube calculations. In *Finite Elements in Fluids. Volume 6*, pages 339--353, Wiley-Interscience, Chichester, England and New York, 1985.

[39] T.J.R. Hughes, L.P. Franca, and M. Mallet. A new finite element formulation for computational fluid dynamics. i. symmetric forms of the compressible euler and navier-stokes equations and the second law of thermodynamics. *Computer Methods in Applied Mechanics and Engineering*, 54(2):223--234, February 1986.

[40] Thomas J. R. Hughes, Michel Mallet, and Akira Mizukami. A new finite element formulation for computational fluid dynamics. ii. beyond supg. *Computer Methods in Applied Mechanics and Engineering*, 54(3):341--355, 1986.

[41] Thomas J. R. Hughes. Recent progress in the development and understanding of supg methods with special reference to the compressible euler and navier-stokes equations. *International Journal for Numerical Methods in Fluids*, 7:1261--1275, 1987.

[42] Thomas J. R. Hughes. Galerkin/least-squares procedures in computational fluid dynamics. In T.J. Chung and G.R. Karr, editors, *Proceedings of the Seventh International Conference on Finite Element Methods in Flow Problems*, pages 311--321, University of Alabama in Huntsville Press, Huntsville, AL, April 1989.

[43] J. T. Oden. *Adaptive Computational Methods for SSME Internal Flow Analysis*. Final Report NASA-CR-178864 and NAS 1.26:178864 and TR86-05, Computational Mechanics Consultants, Austin, Texas, 1986. 58 pages, AIAA Technical Library.

[44] G.F. Carey and J. T. Tinsley Oden. *Finite elements: Fluid mechanics. Volume VI.* Prentice-Hall, Englewood Cliffs, U.S.A., 1986. 333 pages, ISBN 0-13-317132-9.

[45] Philippe Devloo, J. Tinsley Oden, and Paresh Pattani. An h-p adaptive finite element method for the numerical stimulation of compressible flow. *Computer Methods in Applied Mechanics and Engineering*, 70(2):203--235, September 1988.

[46] J. T. Oden, T. Strouboulis, and Ph. Devloo. Adaptive finite element methods for compressible flow problems. In *Applied Mechanics Division Symposia Series, volume 78*, pages 115--126, American Society of Mechanical Engineers, Applied Mechanics Div, New York, NY, USA, December 1986. Numerical Methods for Compressible Flows - Finite Difference, Element and Volume Techniques, Presented at the Winter Annual Meeting of the American Society of Mechanical Engineers, Anaheim, CA, USA.

[47] J.T. Oden and T. Strouboulis amd Ph. Devloo. Adaptive finite element methods for high-speed compressible flows. *International Journal for Numerical Methods in Fluids*, 7:1211--1228, 1987.

[48] W.G. Habashi, G.S. Baruzzi, and M.F. Peeters. A newton-galerkin algorithm for computational fluid dynamics. In *Collection of Technical Papers, International Symposium on CFD*, pages 347--352, Nagoya Trade and Ind. Center, Nagoya, Japan, 1989.

[49] W.G. Habashi, M. Robichaud, V.N. Nguyen, W.S. Ghaly, and M. Fortin. Large-scale computational fluid dynamics by the finite element method. In *29th Aerospace Sciences Meeting*, page 17, AIAA, Reno, Nevada, USA, January 1991. AIAA-Paper-91-0120.

[50] W. G. Habashi, G. Baruzzi, and M. Hafez. A second order finite element formulation for viscous transonic flows. In *4th International Symposium on Computational Fluid Dynamics*, University of California, Davis, Davis, CA, USA, September 1991.

[51] Dominique Pelletier, Andre Garon, and Ricardo Camarero. A new finite element method for computing turbulent flow near a propeller. In *26th Aerospace Sciences Meeting*, page 9, AIAA, Reno, Nevada, USA, January 1988. AIAA PAPER 88-0048.

[52] D. Pelletier and J.-F. Hetu. Adaptive finite element techniques for the navier-stokes equations. In *4th International Symposium on Computational Fluid Dynamics*, University of California, Davis, Davis, CA, USA, September 1991.

[53] R. Loehner, K. Morgan, and O. C. Zienkiewicz. An adaptive finite element method for high speed compressible flow. In *Presented at the 9th International Conference on Numerical Methods in Fluid Dynamics*, page 7, Institute for Numerical Methods in Engineering, University Coll. of Swansea, Wales, June 1984. AIAA Technical Library.

[54] R. Loehner, K. Morgan, J. Peraire, O.C. Zienkiewicz, and L. Kong. Finite element methods for compressible flow. In K.W. Morton; M.J. Baines, editor, *Numerical Methods for Fluid Dynamics II*, pages 27--53, Oxford Univ. Press, Swansea Univ. Coll.,, 1985. conference proceedings, ISBN 0-19-853610-0.

[55] R. Loehner, K. Morgan, J. Peraire, and O.C. Zienkiewicz. Finite element methods for high speed flows. In *7th Computational Fluid Dynamics Con-*

ference, pages 403--410, AIAA, Cincinnati, OH, USA, July 1985. AIAA Paper 85-1533.

[56] Rainald Loehner, Ken Morgan, Jaime Peraire, and Mehdi Vahdati. Finite element flux-corrected transport (fem-fct) for the euler and navier-stokes equations. In *Finite Elements in Fluids, Volume 7*, pages 105--121, John Wiley & Sons, 1987.

[57] Rainald Lohner and Dorothee Martin. Snakes, linelets, and implicit unstructured flow solvers. In *4th International Symposium on Computational Fluid Dynamics*, pages 717--722, University of California, Davis, Davis, CA, USA, September 1991.

[58] M. O. Bristeau, R. Glowinski, B. Mantel, J. Periaux, and P. Perrier. Numerical methods for the time dependent compressible navier-stokes equations. In *Computing methods in applied sciences and engineering, VI; Proceedings of the Sixth International Symposium*, pages 565--583, Versailles, France, December 1983. AIAA Technical Library also in Proceedings Paper in Mathematical Reviews of the American Mathematical Society, 1984.

[59] Marie Odile Bristeau, Roland Glowinski, Jacques Periaux, and Henri Viviand. Numerical simulation of compressible navier-stokes flows. In *Notes on Numerical Fluid Mechanics*, page 349, Friedr. Vieweg und Sohn, Brunswick, Federal Republic of Germany, 1987. volume 18.

[60] M.O. Bristeau and J. Periaux. Finite element methods for the calculation of compressible viscous flows using self-adaptive mesh refinements. In *Lectures on Computational Fluid Dynamics*, chapter vol. 2, paper 3, page 108, VonKarman Inst. Fluid Dyn., Rhode Saint Genese, Belgium, 1986. Lecture Ser. 1986-04.

[61] M. O. Bristeau, R. Glowinski, and J. Periaux. Finite element methods for the numerical simulation of compressible viscous flows using self-adaptive mesh refinement. In *Numerical Methods for Compressible Flows - Finite Difference, Element and Volume Techniques*, pages vol. 78, p 1--3, American Society of Mechanical Engineers, Applied Mechanics Division, Applied Mechanics Div, New York, NY, USA, December 1986. Presented at the Winter Annual Meeting of the American Society of Mechanical Engineers.

[62] M. O. Bristeau, R. Glowinski, and J. Periaux. Numerical methods for the navier-stokes equations. applications to the simulation of compressible and incompressible viscous flows. In *Finite elements in physics*, pages 73--187,

American Mathematical Society, North-Holland, Amsterdam-New York, 1986. Proceedings Paper in Mathematical Reviews.

[63] M. O. Bristeau, R. Glowinski, L. Dutto, J Periaux, and G. Roge. Compressible viscous flow calculations using compatible finite element approximations. In T.J. Chung and G.R. Karr, editors, *Proceedings of the Seventh International Conference on Finite Element Methods in Flow Problems*, pages 322--329,1609, University of Alabama in Huntsville Press, Huntsville, AL, April 1989.

[64] Bo-Nan Au-Jiang and Louis A. Povinelli. Least-squares finite element method for fluid dynamics. In *Finite element analysis in fluids; Proceedings of the Seventh International Conference on Finite Element Methods in Flow Problems*, pages 105--110, University of Alabama in Huntsville Press, Huntsville, AL, USA, April 1989. also in NASA Lewis Research Center report NASA-TM-102352 or E-5061 or ICOMP-89-23 or NAS 1.15:102352, 37 pages.

[65] R. Glowinski. Viscous flow simulation by finite element methods and related numerical techniques. In *Progress and supercomputing in computational fluid dynamics;*, pages 173--210, AIAA Technical Library, Jerusalem, Israel, December 1984. Proceedings of U.S.-Israel Workshop.

[66] R. Glowinski and J. Periaux. Finite element, least squares and domains decomposition methods for the numerical solution of nonlinear problems in fluid dynamics. In *Numerical Methods in Fluid Dynamics*, pages 1--114, Springer-Verlag in Berlin and New York, 1985. AIAA Technical Library.

[67] P. Amestoy and R. Tilch. Solving the compressible navier-stokes equations with finite elements using a multifrontal method. *Impact of Computing in Science and Engineering*, 1(1):93--107, March 1989.

[68] C.H. Cooke and D.K. Blanchard. A higher order finite element algorithm for the unsteady navier-stokes equations. *Mathematics and Computers in Simulation*, 22(2):127--132, July 1980.

[69] C. A. J. Fletcher. An alternating direction implicit finite element method for compressible, viscous flow. In W. C. Reynolds and R. W. MacCormack, editors, *Seventh International Conference on Numerical Methods in Fluid Dynamics*, pages 182--187, Stanford, CA, USA and NASA/Ames, CA, USA, Springer-Verlag, Berlin, West Germany, 1981.

[70] K. C. Reddy and J. N. Reddy. A locally implicit finite element analysis algorithm for 3d compressible flows. In T.J. Chung and G.R. Karr, editors,

Proceedings of the Seventh International Conference on Finite Element Methods in Flow Problems, pages 339--344, University of Alabama in Huntsville Press, Huntsville, AL, April 1989.

[71] L. W. Spradley, J. F. Stalnaker, M. A. Robinson, and K. E. Xiques. Finite element algorithms for compressible flow computations on a supercomputer. In *Finite Elements in Fluids. Volume 6*, pages 135--156, Wiley-Interscience, Chichester, England and New York, 1985.

[72] S. G. Rubin. *Numerical and Physical Aspects of Aero Flows*. Springer-Verlag, 1982.

[73] S. G. Rubin and A. Lin. Pns. *AIAA Journal*, 20:1500--1507, 1982.

[74] S. G. Rubin and D. R. Reddy. AIAA-84-1627.

[75] Y.C. Vigneron, J.V. Rakich, and J.C. Tannehill. Calculation of supersonic viscous flow over delta wings with sharp subsonic leading edges. Technical Paper AIAA-78-1137.

[76] J. C. Tannehill, E. Venkatapathy, and J. V. Rakich. Pns. *AIAA Journal*, 20:203--210, 1981.

[77] J. C. Tannehill, E. Venkatapathy, and J. V. Rakich. AIAA-82-0028.

[78] L. B. Schiff and J. L. Steger. Pns. *AIAA Journal*, 18:1421--1430, 1979.

[79] D. S. Chausee, J. L. Patterson, P. Kutler, T. H. Pulliam, and J. L. Steger. AIAA-81-0050.

[80] S. P. Shanks, G. R. Srinivasan, and W. E. Nicolet. AFWAL-TR-82-3034, 1982.

[81] W. E. Nicolet. AFWAL-TM-83-164-FIMG, 1983.

[82] W. K. Anderson, J. L. Thomas, and B. vanLeer. AIAA-85-0122, 1985.

[83] R. C. Buggeln, Y. N. Kim, and H. McDonald. NASA CR-4021, 1986.

[84] R. W. Walters. *Ph.D. Dissertation*. PhD thesis, North Carolina State University, Raleigh, NC, 1984.

[85] A. J. Baker and J. D. Freels. *A Comprehensive PNS Upgrade for Three-Dimensional Reentry Flow Prediction*. Technical Report COMCO TR 88-4.0, Computational Mechanics Corporation, June 1988. Final Report, AF/BMO SBIR Phase I Contract F04704-87-C-0100.

[86] A. J. Baker and P. D. Manhardt. *A Finite Element PNS Aerodynamics Code*. Technical Report F-04704-89-C-0024, Computational Mechanics Corporation, 1990-91. for Norton Air Force Base, HQ Ballistic Missles Org. SBIR Phase II quarterly progress reports 6/89 through 6/91.

[87] A. J. Baker. *Finite Element Computational Fluid Mechanics*. Hemisphere Publishers, New York, NY, 1983.

[88] A. J. Baker. A finite element penalty algorithm for the parabolic navier-stokes equations for turbulent three-dimensional flow. *Computer Methods in Applied Mechanics and Engineering*, 46(3):277--293, Nov. 1984.

[89] J.D. Freels and A.J. Baker. Symbolic stability analysis for space-marching of reduced reynolds-averaged navier-stokes equation systems. AIAA 28th Aerospace Sciences Meeting, AIAA-90-0584, January 1990.

[90] A. J. Baker, J. D. Freels, and G. S. Iannelli. *The Finite Element PRANS for Real Gas Missile Aerodynamics Simulation*. Technical Report F-04704-89-C-0024, Computational Mechanics Corporation, 1991-92. for Norton Air Force Base, HQ Ballistic Missles Org. SBIR Phase II Draft Final Report, November 1991, COMCO Report 91TR-6.1.

[91] C.A.J. Fletcher. Computational exploitation of the reduced navier-stokes equations. Proceedings ISCFD- Sydney, Australia, August 1987.

[92] S.W. Armfield and C.A.J. Fletcher. Symbolic analysis of partial differential equations. Department of Mechanical Engineering, University of Sydney, Sydney, Australia, AMS(MOS) subject classification 65M30.

[93] C.A.J. Fletcher and S.W. Armfield. Symbolic analysis of the reduced incompressible navier-stokes equations. to be published.

[94] S.T. Zalesak. Generalized finite volume (gfv) schemes on unstructured meshes: cell averages, point values, and finite elements. In *Collection of Technical Papers, 4th International Symposium on CFD*, University of California, Davis, September 1991.

[95] S. Srinvasan, J. C. Tannehill, and K. J. Weilmuehster. *Simplified Curve Fits for the Transport Properties of Equilibrium Air*. Technical Report NASA CR 178411, Engineering Research Institute, Iowa State University, Ames, Iowa, September 1987. for NASA under Grant NAG-1-313.

[96] Fredrick S. Sherman. *Viscous Flow*. McGraw-Hill, New York, NY, 1990.

[97] B. S. Baldwin and H. Lomax. Thin layer approximation and algebraic model for separated turbulent flows. AIAA 16th Aerospace Sciences Meeting, AIAA-78-0257.

[98] Tuncer Cebeci. Calculation of compressible turbulent boundary layers with heat and mass transfer. AIAA 3rd Fluid and Plasma Dynamics Conference AIAA-70-741.

[99] Bram Van Leer and William A. Mulder. Flux-vector splitting for the euler equations. In *Eighth International Conference on Numerical Methods in Fluid Dynamics, Volume 170 of Lecture Notes in Physics*, pages 507--521, 1982. E. Krause, ed.,Springer-Verlag.

[100] R. M. Beam and R. F. Warming. An implicit finite-difference algorithm for hyperbolic systems in conservation law form. *Journal of Computational Physics*, 87--110, 1976.

[101] P. D. Lax and B. Wendroff. Systems of conservation laws. *Comm. Pure Appl. Math.*, 13:217--237, 1960.

[102] Raymond. W. H. and A. Garder. Selective damping in a galerkin method for solving wave problems with variable grids. *Monthly Weather Review*, 104:1583--1590, 1976.

[103] J. Donea. A taylor-galerkin method for convective transport problems. *International Journal of Numerical Methods in Engineering*, 20:101--119, 1984.

[104] James B. Jones and George A. Hawkins. *Engineering Thermodynamics, An Introductory Textbook*. John Wiley & Sons, 1960.

[105] Dale A. Anderson, John C. Tannehill, and Richard H. Pletcher. *Computational Fluid Mechanics and Heat Transfer*. McGraw-Hill Book Comparny, 1984.

[106] Lester Haar, John S. Gallagher, and George S. Kell. *NBS/NRC Steam Tables:Thermodynamic and Transport Properties and Computer Programs for Vapor and Liquid States of Water in SI Units*. McGraw-Hill Book Company, 1984.

[107] C. Millikan. On the steady motion of viscous incompressible fluids with particular reference to a variational principle. *Phil. Magazine*, 7:641--662, 1929.

[108] J. N. Reddy. *Applied Functional Analysis and Variational Methods in Engineering*. McGraw-Hill, Inc., 1986.

[109] L. Lapidus and G. F. Pinder. *Numerical Solution of Partial Differential Equations in Science and Engineering*. John Wiley and Sons, Inc., New York, NY, 1982.

[110] G. D. Smith. *Numerical Solution of Partial Differential Equations: Finite Difference Methods*. Oxford University Press, 1978.

[111] Garth A. Baker, Vassilios A. Dougalis, and Ohannes A. Karakashian. Convergence of galerkin approximations for the korteweg-de vries equation. *Mathematics of Computation*, 40(162):419--433, April 1983.

[112] Jerry L. Bona, Vassilios A. Dougalis, and Ohannes A. Karakashian. Fully discrete galerkin methods for the korteweg-de vries equation. *Comp. & Maths. with Applis.*, 12A(7):859--884, 1986.

[113] J. D. Lambert. *Computational Methods in Ordinary Differential Equations*. John Wiley & Sons, N.Y., 1973.

[114] W. H. Hundidorfer and M. N. Spigker. A note on *B*-stability of runge-kutta methods. *Num. Math.*, 36:319--331, 1981.

[115] P. R. Halmos. *Finite Dimensional Vector Spaces*. Van Nostrand, Princeton, New Jersey, 1958.

[116] J.C. Strickwerda. Initial boundary value problems for incompletely parabolic systems. *Communications of Pure Applied Mathematics*, 30:797--822, 1977.

[117] S.R. Chakravarthy. Euler equations-implicit schemes and implicit boundary conditions. AIAA 20th Aerospace Sciences Meeting, AIAA-82-0228.

[118] Michel A. Saad. *Compressible Fluid Flow*. Prentice-Hall Inc., Englewood Cliffs, New Jersey 07632, 1985. ISBN 0-13-163486-0.

[119] K. H. Huebner. *The Finite Element Method for Engineers*. John Wiley & Sons, 1975.

[120] T. W. Kerlin. Class notes on nuclear reactor dynamics and control. 1977. Chapter 6, Stability Analysis.

[121] R. A. Hedrick and J. R. Penland. Derivation of a digital filter. personal communication, 1979-1990.

[122] Hermann Schlichting. *Boundary-Layer Theory*. McGraw-Hill, New York, NY, 1979.

[123] D. E. Coles. The law of the wake in turbulent boundary layers. *Journal of Fluid Mechanics*, 1:191--226, 1956.

[124] A. J. Baker, J. D. Freels, J. W. Kim, and J. A. Orzechowski. On a finite element cfd algorithm for compressible viscous and turbulent aerodynamic flows. In *Finite Elements in Fluids - Volume 7*, chapter 13, pages 247--272, John Wiley & Sons, 1988.

[125] H. W. Stock and W. Haase. Determination of length scales in algebraic turbulence models for navier-stokes methods. *AIAA Journal*, 27(1):5--14, January 1989.

[126] D. A. Anderson, J. C. Tannehill, and R. H. Pletcher. *Computational Fluid Mechanics and Heat Transfer*. Hemisphere Publishers, New York, NY, 1984.

[127] J. T. Salmon, T. J. Bogar, and M. Sajben. Laser doppler velocimeter measurements in unsteady, separated, transonic diffuser flows. *AIAA Journal*, 21(12):1690--1697, December 1983.

[128] T. J. Bogar, M. Sajben, and J. C. Kroutil. Characteristic frequencies of transonic diffuser flow oscillations. *AIAA Journal*, 21:1232--1240, September 1983.

[129] T. Hsieh, A. B. Wardlaw Jr., P. Collins, and T. Coakley. Numerical investigation of unsteady inlet flow fields. In *AIAA 22nd Aerospace Sciences Meeting*, January 1984. AIAA-84-0031.

[130] Steven R. Allmaras and Michael B. Giles. A coupled euler/navier-stokes algorithm for 2-d unsteady transonic flows. In *AIAA 27th Aerospace Sciences Meeting*, January 1989. AIAA-89-0556.

[131] Announcement of the gamm-workshop on numerical simulation of compressible navier-stokes flows. GAMM-Committee for Numerical Methods in Fluid Mechanics, December 1985.

[132] GAMM Workshop on the Numerical Simulation of Compressible Navier-Stokes Flows. 1985 : nice, france. In *Notes on numerical fluid mechanics ; v. 18*, December 1985. Braunschweig : Wiesbaden : F. Vieweg & Sohn, 1987, editor: M.O. Bristeau et.al.

[133] David B. Soll. *Vectorization and Vector Migration Techniques*. Technical Report SR20-4966-00, IBM Data Systems Division, Kingston, NY, June 1986. First Edition, Technical Bulletin.

[134] *Parallel FORTRAN Language and Library Reference.* IBM, March 1988. SC23-0431-00.

[135] J. D. Freels. *An Improved Element-By-Element Algorithm for Parallel Computational Fluid Dynamics*. Technology for Energy Corporation SBIR Proposal P-87-388 for the DARPA and SBIR Proposal P-88-077 for the NSF.

APPENDICES

Appendix A

Derivation of the Governing Equations

A.1 Introduction

The governing equations for the problem class are derived here. It is assumed that the conservation laws of mass, momentum, and energy, established using Reynolds transport theorem, are valid. Reynolds transport theorem is stated by the following equation

$$\frac{D}{Dt}\int_V \alpha dV = \int_V \frac{\partial \alpha}{\partial t}\, dV + \int_A (\alpha \boldsymbol{u}) \cdot \boldsymbol{n}\, dA = E(\alpha) \qquad \text{(A.1)}$$

where $\frac{D}{Dt}$ denotes the total derivative, A is the surface area surrounding an arbitrary volume, V, $\boldsymbol{u}$ is the fluid velocity, $\boldsymbol{n}$ is an outward unit vector normal to A facing in a direction away from V, α is an arbitrary intensive property of the fluid, and E is an extensive property of the fluid which is a function of α. For α equal to the fluid density, Equation (A.1) describes conservation of mass (continuity) and E is identically zero. For α equal to the fluid momentum, conservation of momentum (Newton's second law) is expressed and E is equal to the sum of forces acting on the fluid. For α equal to the fluid total energy, conservation of energy holds (first law of thermodynamics) and E is equal to the sum of all energy transferred to the fluid. Assuming α sufficiently smooth

(differentiable), the divergence theorem may be applied to the surface integral in Equation (A.1), yielding Reynolds transport theorem written as

$$\frac{D}{Dt}\int_V \alpha dV = \int_V \left(\frac{\partial \alpha}{\partial t} + \nabla \cdot (\alpha \boldsymbol{u})\right) dV = E(\alpha). \tag{A.2}$$

A.2 Conservation of Mass

Applying Reynolds transport theorem with fluid density as the intensive property results in the conservation of mass or continuity equation

$$\mathcal{L}(\rho) = \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{u}) = 0. \tag{A.3}$$

The time dependent (unsteady) form is preserved, and the problem definition for this research is limited to two dimensions. Assuming a Cartesian coordinate system, the continuity equation in scalar divergence form is

$$\mathcal{L}(\rho) = \frac{\partial \rho}{\partial t} + \frac{\partial (\rho u)}{\partial x} + \frac{\partial (\rho v)}{\partial y} = 0. \tag{A.4}$$

Introducing index notation, and letting $m_1 = \rho u$, and $m_2 = \rho v$, Equation (A.4) yields the equivalent form

$$\mathcal{L}(\rho) = \frac{\partial \rho}{\partial t} + \frac{\partial m_1}{\partial x_1} + \frac{\partial m_2}{\partial x_2} = 0. \tag{A.5}$$

A.3 Conservation of Momentum

Applying Reynolds transport theorem with the fluid momentum as the intensive property and utilizing Newton's second law yields a general momentum equation written on a fluid element

$$\mathcal{L}(\boldsymbol{m}) = \frac{\partial \boldsymbol{m}}{\partial t} + \nabla \cdot (\boldsymbol{m}\boldsymbol{u}) - \rho \boldsymbol{b} - \nabla \cdot \overline{\boldsymbol{T}} = \boldsymbol{0}. \tag{A.6}$$

The quantity $\nabla \cdot (\boldsymbol{mu})$ represents the divergence of the convective momentum flux tensor. Tensors are more precisely interpreted in index notation, therefore, this practice will be carried out henceforth. It is assumed that the body force (for example, gravity) is small allowing the body force term ($\rho \boldsymbol{b}$) to be neglected. A Newtonian fluid is also assumed allowing a constitutive equation to be written for the stress tensor of the form $\overline{\boldsymbol{T}} = -p\overline{\boldsymbol{I}} + \overline{\boldsymbol{\tau}}$. This allows the momentum equation to be written as

$$\mathcal{L}(m_i) = \frac{\partial m_i}{\partial t} + \frac{\partial}{\partial x_j}(m_i u_j + \delta_{ij} p - \tau_{ij}) = 0. \tag{A.7}$$

As in the continuity equation, an unsteady, two dimensional Cartesian coordinate system is utilized yielding two scalar equations from the single vector Equation (A.7). Replacing m_i by ρu_i, the x_1 momentum equation is

$$\mathcal{L}(m_1) = \frac{\partial(\rho u_1)}{\partial t} + \frac{\partial(u_1 \rho u_1)}{\partial x_1} + \frac{\partial(u_2 \rho u_1)}{\partial x_2} + \frac{\partial p}{\partial x_1} - \frac{\partial \tau_{11}}{\partial x_1} - \frac{\partial \tau_{21}}{\partial x_2} = 0. \tag{A.8}$$

The x_2 momentum equation is

$$\mathcal{L}(m_2) = \frac{\partial(\rho u_2)}{\partial t} + \frac{\partial(u_1 \rho u_2)}{\partial x_1} + \frac{\partial(u_2 \rho u_2)}{\partial x_2} + \frac{\partial p}{\partial x_2} - \frac{\partial \tau_{12}}{\partial x_1} - \frac{\partial \tau_{22}}{\partial x_2} = 0. \tag{A.9}$$

A general relationship for the viscous stress tensor in terms of the rate of strain tensor for a Newtonian fluid in index notation for a general flow regime is

$$\tau_{ij} = 2\mu \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) + \left(\left(\kappa - \frac{2}{3}\mu\right)\frac{\partial u_k}{\partial x_k}\right)\delta_{ij}. \tag{A.10}$$

It is assumed that the bulk viscosity, κ, is zero (often called Stokes' relation). This significantly reduces the complexity of the viscous stress tensor allowing the following equations to be written

$$\tau_{11} = 2\mu \frac{\partial u_1}{\partial x_1} - \frac{2}{3}\mu\left(\frac{\partial u_1}{\partial x_1} + \frac{\partial u_2}{\partial x_2}\right), \tag{A.11a}$$

$$\tau_{22} = 2\mu \frac{\partial u_2}{\partial x_2} - \frac{2}{3}\mu\left(\frac{\partial u_1}{\partial x_1} + \frac{\partial u_2}{\partial x_2}\right), \tag{A.11b}$$

$$\tau_{12}=\mu\left(\frac{\partial u_1}{\partial x_2}+\frac{\partial u_2}{\partial x_1}\right), \text{and} \tag{A.11c}$$

$$\tau_{21}=\mu\left(\frac{\partial u_2}{\partial x_1}+\frac{\partial u_1}{\partial x_2}\right). \tag{A.11d}$$

Equations (A.11a) through (A.11d) are substituted into Equations (A.8) and (A.9), and the viscous terms are combined yielding

$$\begin{aligned}\mathcal{L}(m_1)=0 = &\frac{\partial(m_1)}{\partial t}+\frac{\partial(u_1 m_1)}{\partial x_1}+\frac{\partial(u_2 m_1)}{\partial x_2}+\frac{\partial p}{\partial x_1}\\ &-\frac{4}{3}\frac{\partial}{\partial x_1}\left(\mu\frac{\partial u_1}{\partial x_1}\right)+\frac{2}{3}\frac{\partial}{\partial x_1}\left(\mu\frac{\partial u_2}{\partial x_2}\right)-\frac{\partial}{\partial x_2}\left(\mu\frac{\partial u_1}{\partial x_2}\right)-\frac{\partial}{\partial x_2}\left(\mu\frac{\partial u_2}{\partial x_1}\right)\end{aligned} \tag{A.12}$$

for the x_1 momentum equation, and

$$\begin{aligned}\mathcal{L}(m_2)=0 = &\frac{\partial(m_2)}{\partial t}+\frac{\partial(u_1 m_2)}{\partial x_1}+\frac{\partial(u_2 m_2)}{\partial x_2}+\frac{\partial p}{\partial x_2}\\ &-\frac{4}{3}\frac{\partial}{\partial x_2}\left(\mu\frac{\partial u_2}{\partial x_2}\right)+\frac{2}{3}\frac{\partial}{\partial x_2}\left(\mu\frac{\partial u_1}{\partial x_1}\right)-\frac{\partial}{\partial x_1}\left(\mu\frac{\partial u_1}{\partial x_2}\right)-\frac{\partial}{\partial x_1}\left(\mu\frac{\partial u_2}{\partial x_1}\right)\end{aligned} \tag{A.13}$$

for the x_2 momentum equation.

A useful relationship for relating the pressure term with the state variable array is

$$\partial p=\frac{\partial p}{\partial \rho}\partial\rho=\frac{a^2}{\gamma}\partial\rho. \tag{A.14}$$

A.4 Conservation of Energy

Applying Reynolds transport theorem with the fluid energy as the arbitrary intensive property and utilizing the first law of thermodynamics applied to a fluid element in motion results in the following equation

$$\mathcal{L}(g)=\frac{D}{Dt}\left(\rho(e+\frac{\omega^2}{2}+\phi)\right)+\nabla\cdot\boldsymbol{q}-\nabla\cdot(\overline{\boldsymbol{T}}\cdot\boldsymbol{u})=0, \tag{A.15}$$

where

e=the internal energy per unit mass,

ω=the magnitude of the fluid velocity

$=\sqrt{u^2 + v^2 + w^2}$, and

ϕ=the work required to overcome the body forces represented by a potential energy, where $\rho \boldsymbol{b} \cdot \boldsymbol{u} \equiv -\frac{D\rho\phi}{Dt}$.

The total internal energy is defined by $\Omega \equiv e + \frac{\omega^2}{2} + \phi$ which when substituted in along with the definition of the total derivative yields

$$\mathcal{L}(g) = \frac{\partial}{\partial t}(\rho\Omega) + \nabla \cdot (\rho \boldsymbol{u}\Omega) + \nabla \cdot \boldsymbol{q} - \nabla \cdot (\overline{\boldsymbol{T}} \cdot \boldsymbol{u}) = 0. \qquad \text{(A.17)}$$

For this research, it is assumed that no internal heat is generated (although this is not a restriction of the algorithm) allowing the energy contribution due to heat to be expressed by the Fourier conduction law $\boldsymbol{q} = -k\nabla T$. Substituting the stress tensor given by Equation (A.10) yields

$$\mathcal{L}(g) = \frac{\partial}{\partial t}(\rho\Omega) + \nabla \cdot (\rho \boldsymbol{u}\Omega) - \nabla \cdot k\nabla T + \nabla p \cdot \boldsymbol{u} - \nabla \cdot (\overline{\tau} \cdot \boldsymbol{u}) = 0. \qquad \text{(A.18)}$$

It is desired to formulate the energy equation with the time derivative written on the volume specific stagnation enthalpy defined as $g = \rho\Omega + p = \rho e + \frac{\rho\omega^2}{2} + \rho\phi + p$. The differential relationship between pressure and density given by Equation A.14 results in the following equation

$$\mathcal{L}(g) = \frac{\partial g}{\partial t} + \nabla \cdot \boldsymbol{u}g + \frac{\partial p}{\partial \rho}(\nabla \cdot \rho \boldsymbol{u}) - \nabla \cdot k\nabla T - \nabla \cdot (\overline{\tau} \cdot \boldsymbol{u}) = 0. \qquad \text{(A.19)}$$

The last term in Equation (A.19) is commonly called the dissipation function (Φ); expansion in three dimensional scalar form yields

$$\nabla \cdot (\overline{\tau} \cdot \boldsymbol{u}) = \frac{\partial}{\partial x}\left(\tau_{xx}u + \tau_{xy}v + \tau_{xz}w\right)$$

$$+\frac{\partial}{\partial y}\left(\tau_{yx}u + \tau_{yy}v + \tau_{yz}w\right)$$
$$+\frac{\partial}{\partial z}\left(\tau_{zx}u + \tau_{zy}v + \tau_{zz}w\right). \tag{A.20}$$

Substituting Equations (A.11a), (A.11b), and (A.11c), which represent the elements of the viscous stress tensor, into Equation (A.20), and again restricting the development to two dimensions leads to the following equation

$$\Phi = \frac{\partial}{\partial x}\left[u\left(2\mu\frac{\partial u}{\partial x} - \frac{2}{3}\mu\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)\right) + v\left(\mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\right)\right]$$
$$+\frac{\partial}{\partial y}\left[u\left(\mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\right) + v\left(2\mu\frac{\partial v}{\partial y} - \frac{2}{3}\mu\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)\right)\right]. \tag{A.21}$$

Introducing index notation results in the final two dimensional form of the energy equation

$$\mathcal{L}(g) = \frac{\partial g}{\partial t} + \frac{\partial u_1 g}{\partial x_1} + \frac{\partial u_2 g}{\partial x_2} + \frac{\partial p}{\partial \rho}\frac{\partial m_1}{\partial x_1} + \frac{\partial p}{\partial \rho}\frac{\partial m_2}{\partial x_2}$$
$$-\frac{\partial}{\partial x_1}\left[u_1\left(2\mu\frac{\partial u_1}{\partial x_1} - \frac{2}{3}\mu\left(\frac{\partial u_1}{\partial x_1} + \frac{\partial u_2}{\partial x_2}\right)\right) + u_2\left(\mu\left(\frac{\partial u_1}{\partial x_2} + \frac{\partial u_2}{\partial x_1}\right)\right)\right]$$
$$-\frac{\partial}{\partial x_2}\left[u_1\left(\mu\left(\frac{\partial u_1}{\partial x_2} + \frac{\partial u_2}{\partial x_1}\right)\right) + u_2\left(2\mu\frac{\partial u_2}{\partial x_2} - \frac{2}{3}\mu\left(\frac{\partial u_1}{\partial x_1} + \frac{\partial u_2}{\partial x_2}\right)\right)\right]$$
$$-\frac{\partial}{\partial x_1}\left(k\frac{\partial T}{\partial x_1}\right) - \frac{\partial}{\partial x_2}\left(k\frac{\partial T}{\partial x_2}\right) = 0. \tag{A.22}$$

A.5 Extension to Three Dimensions

The same formulation applied up to this point may be applied to a three dimensional solution space (as opposed to the two dimensional solution space). The following compact index notation equations result for the conservation of mass, momentum, and energy respectively

$$\mathcal{L}(\rho) = \frac{\partial \rho}{\partial t} + \frac{\partial m_i}{\partial x_i} = 0. \tag{A.23a}$$
$$\mathcal{L}(m_i) = \frac{\partial m_i}{\partial t} + \frac{\partial (u_j m_i)}{\partial x_j} + \frac{\partial p}{\partial x_i}$$

$$-\frac{\partial}{\partial x_j}\left(\mu\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right)-\left(\frac{2}{3}\mu\frac{\partial u_k}{\partial x_k}\right)\delta_{ij}\right)=0. \tag{A.23b}$$

$$\mathcal{L}(g)=\frac{\partial g}{\partial t}+\frac{\partial(u_i g)}{\partial x_i}+\frac{\partial p}{\partial \rho}\frac{\partial m_i}{\partial x_i}-\frac{\partial}{\partial x_i}\left(k\frac{\partial T}{\partial x_i}\right)$$
$$+\frac{\partial}{\partial x_i}\left(u_j\left[\mu\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right)-\left(\frac{2}{3}\mu\frac{\partial u_k}{\partial x_k}\right)\delta_{ij}\right]\right)=0. \tag{A.23c}$$

A.6 Extension to Turbulent Form

In order to extend the governing conservation equations given by Equations (A.23a), (A.23b), and (A.23c) to allow for simulation of more general turbulent flows, it is necessary to perform a time averaging procedure. Let $q(\boldsymbol{x},t)$ be any dependent variable of the governing equations that is a function of space $(\boldsymbol{x})$ and time (t). The time-average procedure replaces $q(\boldsymbol{x},t)$ by

$$q(\boldsymbol{x},t)=\overline{q(\boldsymbol{x})}+q'(\boldsymbol{x},t), \tag{A.24}$$

where $\overline{q(\boldsymbol{x})}$ is a time average or "mean" of $q(\boldsymbol{x})$ defined as

$$\overline{q(\boldsymbol{x})}\equiv\lim_{\Delta t\to\infty}\left(\frac{1}{\Delta t}\right)\int_{t_o}^{t_o+\Delta t}q(\boldsymbol{x},t)dt, \tag{A.25}$$

and $q'(\boldsymbol{x},t)$ represents time fluctuations about the mean. The time average of the fluctuating component is by definition equal to zero. Replacing each dependent variable of Equations (A.23a), (A.23b), and (A.23c) by their respective time averaged and fluctuation components yields

$$\mathcal{L}(\rho)=\frac{\partial(\overline{\rho}+\rho')}{\partial t}+\frac{\partial(\overline{m_i}+m_i')}{\partial x_i}=0, \tag{A.26a}$$

$$\mathcal{L}(m_i)=\frac{\partial(\overline{m_i}+m_i')}{\partial t}$$
$$+\frac{\partial}{\partial x_j}\left\{[(\overline{u_j}+u_j')(\overline{m_i}+m_i')]+\delta_{ij}(\overline{p}+p')+(\overline{\tau_{ij}}+\tau_{ij}')\right\}$$
$$=0,\text{and} \tag{A.26b}$$

$$\mathcal{L}(g)=\frac{\partial}{\partial t}[(\overline{g}+g')-(\overline{p}+p')]$$

$$+\frac{\partial}{\partial x_i}\left\{[(\overline{u_i}+u_i')(\overline{g}+g')]+(\overline{q_i}+q_i')+[(\overline{\tau_{ij}}+\tau_{ij}')(\overline{u_j}+u_j')]\right\}$$
$$=0. \qquad \text{(A.26c)}$$

Performing a time-averaging procedure on Equations (A.26a), (A.26b), and (A.26c) yields the following equations

$$\mathcal{L}(\rho)=\frac{\partial\overline{\rho}}{\partial t}+\frac{\partial\overline{m_i}}{\partial x_i}=0, \qquad \text{(A.27a)}$$

$$\mathcal{L}(m_i)=\frac{\partial\overline{m_i}}{\partial t}+\frac{\partial}{\partial x_j}\left\{\overline{u_j m_i}+\delta_{ij}\overline{p}+\overline{\tau_{ij}}+\overline{u_j' m_i'}\right\}=0, \text{and} \qquad \text{(A.27b)}$$

$$\mathcal{L}(g)=\frac{\partial(\overline{g-p})}{\partial t}+\frac{\partial}{\partial x_i}\left\{\overline{u_i g}+\overline{q_i}+\overline{\tau_{ij}u_j}+\overline{u_i' g'}+\overline{\tau_{ij}' u_j'}\right\}=0. \qquad \text{(A.27c)}$$

The time average over bar notation may now be replaced by the original variables to yield a final turbulent form

$$\mathcal{L}(\rho)=\frac{\partial\rho}{\partial t}+\frac{\partial m_i}{\partial x_i}=0, \qquad \text{(A.28a)}$$

$$\mathcal{L}(m_i)=\frac{\partial m_i}{\partial t}+\frac{\partial}{\partial x_j}\left\{u_j m_i+\delta_{ij}p+\tau_{ij}+\overline{u_j' m_i'}\right\}=0, \text{and} \qquad \text{(A.28b)}$$

$$\mathcal{L}(g)=\frac{\partial(g-p)}{\partial t}+\frac{\partial}{\partial x_i}\left\{u_i g+q_i+\tau_{ij}u_j+\overline{u_i' g'}+\overline{\tau_{ij}' u_j'}\right\}=0. \qquad \text{(A.28c)}$$

The turbulent conservation of mass Equation (A.28a) is identical to the laminar form, Equation (A.23a). The turbulent conservation of momentum Equation (A.28b) includes one additional term ($\frac{\partial(\overline{u_j' m_i'})}{\partial x_j}$) that is not present in the laminar form, Equation (A.23b). The turbulent conservation of energy Equation (A.28c) includes two additional terms ($\frac{\partial(\overline{u_i' g'}+\overline{\tau_{ij}' u_j'})}{\partial x_i}$) that are not present in the laminar form, Equation (A.23c). These additional terms are later replaced by empirical relationships as established via a turbulence closure modeling hypothesis.

If the governing equations had been written in non-conservative form, an additional step called "mass-weighted averaging"[98] is usually performed. Because the governing equations are written in conservative form, the density only

appears as a state variable in the time derivative term of the continuity equation. Therefore, additional terms which would have been generated by the explicit appearance of density (removal of which is the impetus for the mass-weighted-averaging procedure) do not appear when the time averaging is performed on the conservative form of the governing equations. The final form of Equations (A.28a),(A.28b), and (A.28c) are identical to the final form of the mass-weighted, time-averaged, non-conservative form of the governing equations.

A.7 Constitutive Equations

A.7.1 Equations of State

Until this point, no specification has been made for the fluid type. Although not a restriction on the algorithm, for this research, it is assumed that the fluid is described by a real gas law with γ (the specific heat ratio) a function of the thermodynamic state of the fluid. This allows the following relationships to be used

$$p = z\rho RT \tag{A.29a}$$

$= $pressure,

$$z = \text{compressibility factor} \tag{A.29b}$$

$$C_v = \frac{R}{(\gamma - 1)} \tag{A.29c}$$

$= $specific heat at constant volume,

$$C_p = \frac{\gamma R}{(\gamma - 1)} \tag{A.29d}$$

$= $specific heat at constant pressure,

$$\gamma = \frac{C_p}{C_v} \tag{A.29e}$$

$= $specific heat ratio,

$$g = \rho C_p T + \frac{\rho\omega^2}{2} \tag{A.29f}$$

=volume specific enthalpy (ignoring body forces), and

$$\mathcal{L}\left(\frac{\partial p}{\partial \rho}\right) = \frac{\partial p}{\partial \rho} - zRT = \frac{\partial p}{\partial \rho} - \frac{a^2}{\gamma} \tag{A.29g}$$

=the closure equation for the partial derivative of pressure with respect to density.

A.7.2 Turbulence Closure

The classical Boussinesq approximation for the Reynolds stress tensor is

$$\overline{u_j' m_i'} = -\mu_T\left(+\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) + \frac{2}{3}\delta_{ij}\left(\mu_T \frac{\partial u_k}{\partial x_k} + \rho\bar{k}\right), \tag{A.30}$$

where μ_T is the turbulent eddy viscosity, and $\bar{k}$ is the kinetic energy of turbulent fluctuation correlation $\frac{u_i' u_i'}{2}$. Equation (A.30) allows the momentum turbulent terms to be accounted for by replacing μ in the laminar flow momentum equations by $\mu + \mu_T$.

All the turbulent contributions in the energy equation are lumped into an averaged Reynolds heat flux term in an assumed Boussinesq form

$$u_j' g' + u_i' \tau_{ij}' = -k_T \frac{\partial T}{\partial x_j}, \tag{A.31}$$

where k_T is a turbulent fluid thermal conductivity accounting for the additional diffusion of heat by turbulent motion.

The governing equation set is closed by determining adequate approximations for the turbulent eddy viscosity and conductivity, μ_T and k_T, respectively. The Boussinesq approximation has been shown to be adequate for real-gas flows up to about Mach 5. More complicated expressions, such as stress-equation models or Reynolds stress models, are often utilized for more accurate descriptions of the turbulent contributions. For this research, the Boussinesq approximation

will be utilized, not as a limitation of the algorithm, but to demonstrate that the model physics of correlated turbulent flow may be included in the algorithm construction.

The turbulent eddy viscosity coefficient for two-dimensional boundary-layer flow has been previously demonstrated by Baldwin and Lomax[97] to be adequately approximated by

$$\mu_T = \left\{ \begin{array}{ll} \mu_{T_{inner}} & \text{for } r \leq r_c \\ \mu_{T_{outer}} & \text{for } r > r_c \end{array} \right\}, \tag{A.32}$$

where r is interpreted as the distance away from the wall on which the turbulent boundary layer is formed and r_c is the "crossover" minimum distance where the inner and outer values of μ_T are equal.

The inner turbulent eddy viscosity is expressed by the Prandtl-Van Driest formula

$$\mu_{T_{inner}} = \rho l^2 |\omega|, \tag{A.33}$$

where $|\omega|$ is the magnitude of the vorticity vector written in three dimensional Cartesian coordinates as

$$|\omega| = \sqrt{\left(\frac{\partial w}{\partial y} - \frac{\partial v}{\partial z}\right)^2 + \left(\frac{\partial w}{\partial x} - \frac{\partial u}{\partial z}\right)^2 + \left(\frac{\partial u}{\partial y} - \frac{\partial v}{\partial x}\right)^2}. \tag{A.34}$$

In two dimensions, Equation (A.34) reduces to

$$|\omega| = \left|\frac{\partial u}{\partial y} - \frac{\partial v}{\partial x}\right|. \tag{A.35}$$

The Prandtl mixing length is given by the following empirical formula[97]

$$l = kr[1 - \exp(-r^+/A^+)], \tag{A.36}$$

where the parameters in Equation (A.36) are given by

$$k = \text{constant} = 0.4,$$

$$r^+ = \frac{\rho_w u_\tau r}{\mu_w} = \frac{\sqrt{\rho_w \tau_w} r}{\mu_w},$$

A^+=constant = 26.0,

$\rho_w = \rho(r = 0)$,

$\mu_w = \mu(r = 0)$, and

$$\tau_w = \tau(r = 0) = \mu_w \frac{\partial \sqrt{u^2 + v^2 + w^2}}{\partial r}.$$

The outer turbulent eddy viscosity is given by the following empirical formula[97]

$$\mu_{T_{outer}} = \rho K C_{CP} F_{WAKE} F_{KLEB}(r), \tag{A.38}$$

where the parameters in Equation (A.38) are given by

K=Clauser constant = 0.0168,

C_{CP}=constant = 1.6,

$$F_{WAKE} = \text{minimum of } \left\{ \begin{array}{l} r_{max} F_{max}, \\ \frac{C_{WK} r_{max} u_{DIF}^2}{F_{max}} \end{array} \right\},$$

r_{max}=value of r where F_{max}occurs,

F_{max}=maximum of $F(r)$,

$$F(r) = \frac{l(r)|\omega|}{k},$$

C_{WK}=constant = 0.25,

$u_{DIF}^2 = (u^2 + v^2 + w^2)_{max} - (u^2 + v^2 + w^2)_{min}$,

$$F_{KLEB}(r) = \left[1 + 5.5\left(\frac{C_{KLEB} r}{r_{max}}\right)^6\right]^{-1}, \text{and}$$

C_{KLEB}=constant = 0.3.

Baldwin and Lomax performed a further restriction on μ_T by resetting μ_T to zero when the maximum calculated value of μ_T in a profile is less than $C_{MUTM}\mu_\infty$. C_{MUTM} is a constant equal to 14.0 and μ_∞ is the "free stream" value of μ.

A representation for the turbulent thermal conductivity coefficient has been previously demonstrated as[97]

$$k_T = \frac{\mu_T C_p}{Pr_T}, \tag{A.40}$$

where Pr_T is a turbulent Prandtl number that varies over the boundary layer but is often averaged with a value of 0.9 or 0.8.

Appendix B

Transformation to Dimensionless Form

B.1 Introduction

Before transforming the governing equations to a dimensionless form, a choice must be made for the reference fluid properties. For this research, the choice is:

M_∞=free stream Mach number,

L=length scale,

τ=time scale,

a_o=fluid speed of sound at freestream stagnation,

μ_o=fluid viscosity at freestream stagnation,

p_o=fluid pressure at freestream stagnation,

k_o=fluid conductivity at freestream stagnation,

γ_o=fluid specific heat ratio at freestream stagnation,

T_o=fluid temperature at freestream stagnation,

g_o=fluid volume specific enthalpy at freestream stagnation, and

ρ_o=fluid density at freestream stagnation.

Furthermore, in order to arrive at the dimensionless form, a transformation of variables is required. Therefore, the following variables are redefined from their original definitions:

$$\boldsymbol{x} \equiv \frac{\boldsymbol{x}}{L}, \boldsymbol{u} \equiv \frac{\boldsymbol{u}}{a_o}, p \equiv \frac{p}{p_o}, \mu \equiv \frac{\mu}{\mu_o}, \rho \equiv \frac{\rho}{\rho_o}, g \equiv \frac{g}{g_o}, k \equiv \frac{k}{k_o}, T \equiv \frac{T}{T_o}, \text{and } t \equiv \frac{t}{\tau}. \tag{B.2}$$

B.2 Conservation of Mass

Substituting the reference fluid conditions and the redefined variables into the continuity equation given by Equation (A.35) and reducing to two dimensions yields

$$\mathcal{L}(\rho) = \frac{\rho_o}{\tau}\frac{\partial \rho}{\partial t} + \frac{\rho_o a_o}{L}\frac{\partial(\rho u_1)}{\partial x_1} + \frac{\rho_o a_o}{L}\frac{\partial(\rho u_2)}{\partial x_2} = 0. \tag{B.3}$$

The Strouhal number, Sr, is defined to be

$$Sr \equiv \frac{L}{a_o \tau}. \tag{B.4}$$

Since a_o and L are defined from the physics of the problem while τ remains arbitrary, specification of the Strouhal number defines the time scale. Hereafter, the Strouhal number is set to unity and is not explicitly included in the equation system.

Substituting these relationships back into the continuity equation, and multiplying through by $\frac{L}{\rho_o a_o}$ yields the final dimensionless form of the continuity equation

$$\mathcal{L}(\rho) = \frac{\partial \rho}{\partial t} + \frac{\partial m_1}{\partial x_1} + \frac{\partial m_2}{\partial x_2} = 0. \tag{B.5}$$

Equation (B.5) is identical in form to Equation (A.35).

B.3 Conservation of Momentum

In a manner similar as the continuity equation, the reference fluid conditions and the redefined variables are substituted in the primary momentum equation given by Equation (A.36) yields

$$\mathcal{L}(m_1)=\frac{\rho_o a_o}{\tau}\frac{\partial m_1}{\partial t}+\frac{\rho_o a_o^2}{L}\left(\frac{\partial(u_1 m_1)}{\partial x_1}+\frac{\partial(u_2 m_1)}{\partial x_2}\right)+\frac{\rho_o p_o}{\rho_o L}\frac{\partial p}{\partial x_1}$$
$$+\frac{(a_o\mu_o)}{L^2}\left(-\frac{4}{3}\frac{\partial}{\partial x_1}\left(\mu\frac{\partial u_1}{\partial x_1}\right)+\frac{2}{3}\frac{\partial}{\partial x_1}\left(\mu\frac{\partial u_2}{\partial x_2}\right)-\frac{\partial}{\partial x_2}\left(\mu\frac{\partial u_1}{\partial x_2}\right)\right.$$
$$\left.-\frac{\partial}{\partial x_2}\left(\mu\frac{\partial u_2}{\partial x_1}\right)\right)-\frac{(\rho_o a_o^2)}{L}\left(\frac{\partial(\overline{u_1' m_1'})}{\partial x}+\frac{\partial(\overline{u_2' m_1'})}{\partial y}\right)=0. \tag{B.6}$$

Similarly, the x_2 momentum equation is written as

$$\mathcal{L}(m_2)=\frac{\rho_o a_o}{\tau}\frac{\partial m_2}{\partial t}+\frac{\rho_o a_o^2}{L}\left(\frac{\partial(u_1 m_2)}{\partial x_1}+\frac{\partial(u_2 m_2)}{\partial x_2}\right)+\frac{\rho_o p_o}{\rho_o L}\frac{\partial p}{\partial x_2}$$
$$+\frac{(a_o\mu_o)}{L^2}\left(-\frac{4}{3}\frac{\partial}{\partial x_2}\left(\mu\frac{\partial u_2}{\partial x_2}\right)+\frac{2}{3}\frac{\partial}{\partial x_2}\left(\mu\frac{\partial u_1}{\partial x_1}\right)-\frac{\partial}{\partial x_1}\left(\mu\frac{\partial u_1}{\partial x_2}\right)\right.$$
$$\left.-\frac{\partial}{\partial x_1}\left(\mu\frac{\partial u_2}{\partial x_1}\right)\right)-\frac{(\rho_o a_o^2)}{L}\left(\frac{\partial(\overline{u_1' m_2'})}{\partial x}+\frac{\partial(\overline{u_2' m_2'})}{\partial y}\right)=0. \tag{B.7}$$

It is now possible to place the momentum equations in a final dimensionless form by defining the Reynolds number as

$$\mathrm{Re}=\frac{\rho_o a_o L}{\mu_o}, \tag{B.8}$$

the Euler number as

$$\mathrm{Eu}=\frac{p_o}{\rho_o a_o^2}, \tag{B.9}$$

specifying a unit Stouhal number, and multiplying through by $\frac{L}{(\rho_o a_o^2)}$ to yield

$$\mathcal{L}(m_1)=\frac{\partial m_1}{\partial t}+\frac{\partial(u_1 m_1)}{\partial x_1}+\frac{\partial(u_2 m_1)}{\partial x_2}+\mathrm{Eu}\frac{\partial p}{\partial x_1}$$
$$-\frac{1}{\mathrm{Re}}\left[\frac{4}{3}\frac{\partial}{\partial x_1}\left(\mu\frac{\partial u_1}{\partial x_1}\right)-\frac{2}{3}\frac{\partial}{\partial x_1}\left(\mu\frac{\partial u_2}{\partial x_2}\right)+\frac{\partial}{\partial x_2}\left(\mu\frac{\partial u_1}{\partial x_2}\right)+\frac{\partial}{\partial x_2}\left(\mu\frac{\partial u_2}{\partial x_1}\right)\right]$$
$$-\left(\frac{\partial(\overline{u_1' m_1'})}{\partial x}+\frac{\partial(\overline{u_2' m_1'})}{\partial y}\right)=0, \tag{B.10}$$

for the x_1 momentum equation, and

$$\mathcal{L}(m_2)=\frac{\partial m_2}{\partial t}+\frac{\partial(u_1 m_2)}{\partial x_1}+\frac{\partial(u_2 m_2)}{\partial x_2}+\mathrm{Eu}\frac{\partial p}{\partial x_2}$$
$$-\frac{1}{\mathrm{Re}}\left[\frac{4}{3}\frac{\partial}{\partial x_2}\left(\mu\frac{\partial u_2}{\partial x_2}\right)-\frac{2}{3}\frac{\partial}{\partial x_2}\left(\mu\frac{\partial u_1}{\partial x_1}\right)+\frac{\partial}{\partial x_1}\left(\mu\frac{\partial u_1}{\partial x_2}\right)+\frac{\partial}{\partial x_1}\left(\mu\frac{\partial u_2}{\partial x_1}\right)\right]$$
$$-\left(\frac{\partial(\overline{u_1' m_2'})}{\partial x}+\frac{\partial(\overline{u_2' m_2'})}{\partial y}\right)=0, \qquad \text{(B.11)}$$

for the x_2 momentum equation. Equations (B.10) and (B.11) are identical to the dimensional forms given by Equation (A.36) with the exception of Reynolds number appearing as a divisor for the viscous terms.

B.4 Conservation of Energy

The final form of the conservation of energy governing equation is given by Equation (A.37). Applying similar variable transformations as the previous sections leads to the following equation

$$\begin{aligned}\mathcal{L}(g) \quad &= \quad \frac{g_o}{\tau}\frac{\partial g}{\partial t}+\frac{g_o a_o}{L}\left(\frac{\partial(u_1 g+\overline{u_1' g'})}{\partial x_1}+\frac{\partial(u_2 g+\overline{u_2' g'})}{\partial x_2}\right)\\ &+ \quad \frac{P_o\rho_o a_o}{\rho_o L}\frac{\partial p}{\partial \rho}\left(\frac{\partial m_1}{\partial x_1}+\frac{\partial m_2}{\partial x_2}\right)\\ &-\frac{\mu_o a_o^2}{L^2}\left\{\frac{\partial}{\partial x_1}\left[u_1\left(\frac{4\mu}{3}\frac{\partial u_1}{\partial x_1}-\frac{2}{3}\mu\frac{\partial u_2}{\partial x_2}\right)+u_2\mu\left(\frac{\partial u_1}{\partial x_2}+\frac{\partial u_2}{\partial x_1}\right)+\overline{u_1'\tau_{11}'}+\overline{u_2'\tau_{12}'}\right]\right.\\ &+ \quad \left.\frac{\partial}{\partial x_2}\left[u_1\mu\left(\frac{\partial u_1}{\partial x_2}+\frac{\partial u_2}{\partial x_1}\right)+u_2\left(\frac{4\mu}{3}\frac{\partial u_2}{\partial x_2}-\frac{2\mu}{3}\frac{\partial u_1}{\partial x_1}\right)+\overline{u_1'\tau_{21}'}+\overline{u_2'\tau_{22}'}\right]\right\}\\ &- \quad \frac{k_o T_o}{L^2}\left(\frac{\partial}{\partial x_1}\left(k\frac{\partial T}{\partial x_1}\right)+\frac{\partial}{\partial x_2}\left(k\frac{\partial T}{\partial x_2}\right)\right)=0. \qquad \text{(B.12)}\end{aligned}$$

Multiplying through by $\frac{L}{a_o g_o}$, recognizing that the Strouhal number (set to unity) appears again in the time derivative term, and performing some rearrangement of terms yields

$$\begin{aligned}\mathcal{L}(g) \quad = \quad & \frac{\partial g}{\partial t} + \left(\frac{\partial(u_1 g + \overline{u_1' g'})}{\partial x_1} + \frac{\partial(u_2 g + \overline{u_2' g'})}{\partial x_2}\right) + \frac{p_o}{g_o}\frac{\partial p}{\partial \rho}\left(\frac{\partial m_1}{\partial x_1} + \frac{\partial m_2}{\partial x_2}\right) \\ & - \frac{\mu_o a_o}{L g_o}\left\{\frac{\partial}{\partial x_1}\left[u_1\left(\frac{4\mu}{3}\frac{\partial u_1}{\partial x_1} - \frac{2}{3}\mu\frac{\partial u_2}{\partial x_2}\right) + u_2\mu\left(\frac{\partial u_1}{\partial x_2} + \frac{\partial u_2}{\partial x_1}\right) + \overline{u_1'\tau_{11}'} + \overline{u_2'\tau_{12}'}\right]\right. \\ & + \left.\frac{\partial}{\partial x_2}\left[u_1\mu\left(\frac{\partial u_1}{\partial x_2} + \frac{\partial u_2}{\partial x_1}\right) + u_2\left(\frac{4\mu}{3}\frac{\partial u_2}{\partial x_2} - \frac{2\mu}{3}\frac{\partial u_1}{\partial x_1}\right) + \overline{u_1'\tau_{21}'} + \overline{u_2'\tau_{22}'}\right]\right\} \\ & - \frac{k_o T_o}{L a_o g_o}\left(\frac{\partial}{\partial x_1}\left(k\frac{\partial T}{\partial x_1}\right) + \frac{\partial}{\partial x_2}\left(k\frac{\partial T}{\partial x_2}\right)\right) = 0. \end{aligned} \tag{B.13}$$

The reference Prandtl number is defined as

$$\mathrm{Pr} = \frac{\mu_o C_{po}}{k_o}, \tag{B.14}$$

and the Eckert number is defined as

$$\mathrm{Ec} = \frac{\rho_o a_o^2}{2 g_o}. \tag{B.15}$$

Substituting these relationships into Equation (B.13) yields a final form of the dimensionless energy equation

$$\begin{aligned}\mathcal{L}(g) = & \frac{\partial g}{\partial t} + \left(\frac{\partial(u_1 g + \overline{u_1' g'})}{\partial x_1} + \frac{\partial(u_2 g + \overline{u_2' g'})}{\partial x_2}\right) \\ & + 2\mathrm{EcEu}\frac{\partial p}{\partial \rho}\left(\frac{\partial m_1}{\partial x_1} + \frac{\partial m_2}{\partial x_2}\right) - \frac{2\mathrm{Ec}}{\mathrm{Re}}\left\{\right. \\ & \frac{\partial}{\partial x_1}\left[u_1\left(\frac{4\mu}{3}\frac{\partial u_1}{\partial x_1} - \frac{2}{3}\mu\frac{\partial u_2}{\partial x_2}\right) + u_2\mu\left(\frac{\partial u_1}{\partial x_2} + \frac{\partial u_2}{\partial x_1}\right) + \overline{u_1'\tau_{11}'} + \overline{u_2'\tau_{12}'}\right] \\ & + \left.\frac{\partial}{\partial x_2}\left[u_1\mu\left(\frac{\partial u_1}{\partial x_2} + \frac{\partial u_2}{\partial x_1}\right) + u_2\left(\frac{4\mu}{3}\frac{\partial u_2}{\partial x_2} - \frac{2\mu}{3}\frac{\partial u_1}{\partial x_1}\right) + \overline{u_1'\tau_{21}'} + \overline{u_2'\tau_{22}'}\right]\right\} \\ & - \frac{1}{\mathrm{RePr}}\left(\frac{\partial}{\partial x_1}\left(k\frac{\partial T}{\partial x_1}\right) + \frac{\partial}{\partial x_2}\left(k\frac{\partial T}{\partial x_2}\right)\right) = 0. \end{aligned} \tag{B.16}$$

B.5 Constitutive Equations

B.5.1 Equations of State

It is also required to transform the closure and related equations to dimensionless form. An equation for pressure as a function of the thermodynamic state is

$$p_o p = z_o z \rho_o \rho R T_o T \tag{B.17}$$

Dividing through by p_o yields

$$p = \frac{z_o \rho_o R T_o}{p_o} \rho T, \tag{B.18}$$

or

$$p = z \rho T, \tag{B.19}$$

since

$$\frac{z_o \rho_o R T_o}{p_o} = 1. \tag{B.20}$$

An equation for pressure as a function of the state variable state (useful in deriving jacobian formulations) is

$$p = \frac{z}{C_p} \left[g - \mathrm{Ec} \left(\frac{m_1^2 + m_2^2}{\rho} \right) \right]. \tag{B.21}$$

B.5.2 Turbulence Closure

In dimensionless form, the Reynolds stress tensor is assumed of the form

$$\overline{u_j' m_i'} = -\frac{\mu_T}{\mathrm{Re}} \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) + \frac{2}{3} \delta_{ij} \left(\frac{\mu_T}{\mathrm{Re}} \frac{\partial u_k}{\partial x_k} + \rho \bar{k} \right). \tag{B.22}$$

The dimensionless form of the averaged Reynolds heat flux term becomes

$$\overline{u_j' g'} + \overline{u_i' \tau_{ij}'} = -\frac{k_T}{\mathrm{RePr}} \frac{\partial T}{\partial x_j}. \tag{B.23}$$

The quantities μ_T and k_T are defined in Appendix A and are transformed to non-dimensional form by dividing by μ_o and k_o respectively.

B.6 Extension to Three Dimensions

After performing a similar variable transformation, and including the turbulence closure, the three dimensional extension given by Equations (A.24) through (A.26) written in concise index notation becomes

$$\mathcal{L}(\rho)=\frac{\partial \rho}{\partial t}+\frac{\partial m_i}{\partial x_i}=0, \tag{B.24a}$$

$$\begin{aligned}\mathcal{L}(m_i)=&\frac{\partial m_i}{\partial t}+\frac{\partial (u_j m_i)}{\partial x_j}+\mathrm{Eu}\frac{\partial p}{\partial x_i}\\&+\frac{\partial}{\partial x_j}\left(\frac{\mu_e}{\mathrm{Re}}\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right)\right.\\&\left.-\frac{2}{3}\frac{\mu_e}{\mathrm{Re}}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right)=0,\end{aligned} \tag{B.24b}$$

$$\begin{aligned}\mathcal{L}(g)=&\frac{\partial g}{\partial t}+\frac{\partial (u_i g)}{\partial x_i}+2\mathrm{EcEu}\frac{\partial p}{\partial \rho}\frac{\partial m_i}{\partial x_i}-\frac{\partial}{\partial x_i}\left(\frac{k_e}{\mathrm{PrRe}}\frac{\partial T}{\partial x_i}\right)\\&+\frac{2\mathrm{Ec}}{\mathrm{Re}}\frac{\partial}{\partial x_i}\left(u_j\mu_e\left[\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}-\frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right]\right)=0.\end{aligned} \tag{B.24c}$$

Appendix C

Generalized Coordinate Transformation

It is desirable to impose a generalized coordinate transformation from (x, y, z) to (ξ, η, ζ) to facilitate arbitrary mesh calculus operations and to handle arbitrary bounding surfaces. This is obtained by first expanding the transformed coordinate variables in terms of the original coordinate variables

$$d\xi = \xi_x dx + \xi_y dy + \xi_z dz, \tag{C.1a}$$

$$d\eta = \eta_x dx + \eta_y dy + \eta_z dz, \text{and} \tag{C.1b}$$

$$d\zeta = \zeta_x dx + \zeta_y dy + \zeta_z dz, \tag{C.1c}$$

where the subscripts denote partial derivatives. In index notation, Equations (C.1a) through (C.1c) possess the alternate form

$$d\eta_i = \frac{\partial \eta_i}{\partial x_j} dx_j, \quad 1 \leq i \leq n, \tag{C.2}$$

where the subscripts in three dimensions ($n = 3$) are $j = 1 \rightarrow 3$ denoting x, y, and z for x_1, x_2, and x_3 respectively, and ξ,η, and ζ for η_1, η_2, and η_3 respectively. In vector notation, Equation C.2 may be written as

$$d\boldsymbol{\eta} = \boldsymbol{\eta}_{\mathbf{x}} d\mathbf{x}, \tag{C.3}$$

where boldface type denotes a vector and $[\boldsymbol{\eta}_{\mathbf{x}}]$ is a matrix of partial derivatives. If Equation C.3 is multiplied by $[\boldsymbol{\eta}_{\mathbf{x}}]^{-1}$ and solved for $d\mathbf{x}$ it yields

$$d\mathbf{x} = [\boldsymbol{\eta}_{\mathbf{x}}]^{-1} d\boldsymbol{\eta}. \tag{C.4}$$

The matrix $[\boldsymbol{\eta}_{\mathbf{x}}]^{-1}$ is called the coordinate transformation jacobian matrix, denoted as $[J]$, and is identical to the matrix $[\mathbf{x}_{\boldsymbol{\eta}}]$. The numerical calculation of a local jacobian is performed utilizing any trial space basis within a finite element solution algorithm. For illustration, over any region Ω_e of Ω, a subset of Re^n, the following equation is true

$$x_j^e = \{N_k^+(\eta_i)\}^T \{XJ\}_e, 1 \leq (j, J) \leq n \tag{C.5}$$

where

$\{N_k^+(\eta)\}$=a finite element tensor product basis function of degree k, and

$\{XJ\}_e$=the array of values taken by the nodal

coordinates$(x, y, z)_{\text{node}}$on a finite element domain Ω_e.

The partial derivative of x_j with respect to η_i on Ω_e may be written as

$$\frac{\partial x_j^e}{\partial \eta_i} = \frac{\partial \{N_k^+\}^T}{\partial \eta_i} \{XJ\}_e, \tag{C.7}$$

since the array $\{XJ\}_e$ does not depend on the η_i system. The partial derivatives within the governing equations on Ω_e may now be written as

$$\frac{\partial}{\partial x_j} = \frac{\partial \eta_i}{\partial x_j} \frac{\partial}{\partial \eta_i} = [\frac{\partial x_j}{\partial \eta_i}]_e^{-1} \frac{\partial}{\partial \eta_i}, \tag{C.8}$$

and the differential volume is written as

$$d\mathbf{x} = det[J]_e d\boldsymbol{\eta}. \tag{C.9}$$

In the derivation of the finite element algorithm, the higher order partial derivatives (greater than first order) are written in terms of first partial derivatives, by either Green's theorem, or integration by parts, or by using higher degree trial space basis functions, or a combination.

As an example of the application of the generalized coordinate transformation, consider the two-dimensional, bilinear ($k = 1$) tensor product trial space basis form. The basis function set is

$$\{N_1^+(\eta)\} = \frac{1}{4}\begin{Bmatrix} (1-\eta_1)(1-\eta_2) \\ (1+\eta_1)(1-\eta_2) \\ (1+\eta_1)(1+\eta_2) \\ (1-\eta_1)(1+\eta_2) \end{Bmatrix}, \tag{C.10}$$

allowing the partial derivatives

$$\frac{\partial\{N_1^+\}}{\partial\eta_1} = \frac{1}{4}\begin{Bmatrix} -(1-\eta_2) \\ +(1-\eta_2) \\ +(1+\eta_2) \\ -(1+\eta_2) \end{Bmatrix}, \tag{C.11}$$

and

$$\frac{\partial\{N_1^+\}}{\partial\eta_2} = \frac{1}{4}\begin{Bmatrix} -(1-\eta_1) \\ -(1+\eta_1) \\ +(1+\eta_1) \\ +(1-\eta_1) \end{Bmatrix} \tag{C.12}$$

to be evaluated on Ω_e. The arrays

$$\{X1\}_e = \{X_1, X_2, X_3, X_4\}_e^T, \text{and} \tag{C.13a}$$

$$\{X2\}_e = \{Y_1, Y_2, Y_3, Y_4\}_e^T \tag{C.13b}$$

contain the physical (x, y) coordinate values of the four vertex nodes of the planar finite element Ω_e. The inverse jacobian matrix is now evaluated as

$$\left[\frac{\partial \boldsymbol{\eta}}{\partial \mathbf{x}}\right]_e = \left[\frac{\partial \mathbf{x}}{\partial \boldsymbol{\eta}}\right]_e^{-1} = \frac{\left[\begin{array}{cc} +\frac{\partial y}{\partial \eta_2} & -\frac{\partial x}{\partial \eta_2} \\ -\frac{\partial y}{\partial \eta_1} & +\frac{\partial x}{\partial \eta_1} \end{array}\right]_e}{\left(\frac{\partial x}{\partial \eta_1}\frac{\partial y}{\partial \eta_2} - \frac{\partial y}{\partial \eta_1}\frac{\partial x}{\partial \eta_2}\right)_e} = \frac{\left[\begin{array}{cc} +\frac{\partial y}{\partial \eta_2} & -\frac{\partial x}{\partial \eta_2} \\ -\frac{\partial y}{\partial \eta_1} & +\frac{\partial x}{\partial \eta_1} \end{array}\right]_e}{det[J]_e}, \tag{C.14}$$

where the partial derivative quantities are evaluated in the following manner

$$\frac{\partial x}{\partial \eta_1} = \frac{1}{4}[(X_2 - X_1)(1 - \eta_2) + (X_3 - X_4)(1 + \eta_2)], \tag{C.15a}$$

$$\frac{\partial y}{\partial \eta_1} = \frac{1}{4}[(Y_2 - Y_1)(1 - \eta_2) + (Y_3 - Y_4)(1 + \eta_2)], \tag{C.15b}$$

$$\frac{\partial x}{\partial \eta_2} = \frac{1}{4}[(X_4 - X_1)(1 - \eta_1) + (X_3 - X_2)(1 + \eta_1)], \text{and} \tag{C.15c}$$

$$\frac{\partial y}{\partial \eta_2} = \frac{1}{4}[(Y_4 - Y_1)(1 - \eta_1) + (Y_3 - Y_2)(1 + \eta_1)]. \tag{C.15d}$$

As an approximation, the $\boldsymbol{\eta}$ dependence may be eliminated by evaluating these expressions at the centroid of Ω_e. For a linear basis function (as shown in Equation C.10), the centroid approximation is identical to the exact integral and is

$$\frac{\partial x}{\partial \eta_1} = \frac{1}{4}[X_2 - X_1 + X_3 - X_4], \tag{C.16a}$$

$$\frac{\partial y}{\partial \eta_1} = \frac{1}{4}[Y_2 - Y_1 + Y_3 - Y_4], \tag{C.16b}$$

$$\frac{\partial x}{\partial \eta_2} = \frac{1}{4}[X_4 - X_1 + X_3 - X_2], \text{and} \tag{C.16c}$$

$$\frac{\partial y}{\partial \eta_2} = \frac{1}{4}[Y_4 - Y_1 + Y_3 - Y_2]. \tag{C.16d}$$

This allows the transformation to be directly incorporated into the finite element algorithm in a cost effective manner.

If the geometry does not remain constant, as in an adaptive grid procedure, the $[\frac{\partial \boldsymbol{\eta}}{\partial \mathbf{x}}]_e$ matrix, Equation (C.14), must be reevaluated at each change in the physical coordinates.

Appendix D

The Pilot Code Input Description

variable	type	length	description
MANUMB	INT	1	Machine number: 1 = VAX under VMS 2 = IBM 3090 at UTCC under MVS 3 = IBM 3081 at UTCC under CMS 4 = IBM 4381 at Cornell under CMS 5 = IBM 3090 at Cornell under CMS 6 = CDC (not tested) 7 = CRAY (not tested)
BINARY	LOG	1	If .true., the input restart file is binary, If .false., the input restart file is ASCII
CPUCHK	LOG	1	Flag for determining the cpu time distribution between major portions of the code. If .true., cpu time distribution will be performed, if .false., it will not.
CPUTMP	REAL	1	CPU time limit of the job. The program will automatically stop when the cpu time is greater than or equal to (CPUTMP-CPUINP), normally halt the program, and write a restart-plot record. (minutes)
CPUINP	REAL	1	CPU time interval over which the job is finished (minutes).
IPROB	INT	1	An integer pointer used by subroutine INITGE to set up initial conditions (1-7)
VISC	LOG	1	If .true., a non-zero value for the fluid viscosity is computed, if .false., the fluid viscosity is always zero.
NUPD	INT	1	If zero, constant fluid properties and ideal gas is utilized, if positive, NUPD time steps are performed between real-gas fluid property updates.
TURB	INT	1	If non-zero, turbulence is simulated normal to the wall number specified as TURB from 1-4. If zero, turbulence is not simulated.
TANG	REAL	1	The transformation angle (degrees) between the physical space input (see below) and the calculational space For example, TANG=45., places equal velocity dissipation for predominant flow direction problems. NEW RECORD
TITLE	CHAR	1	A title up to 80 characters. NEW RECORD

variable	type	length	description
N	INT	1	Dimensionality of the problem equal to the number of spatial dimensions plus time dimension. A 2-D problem requires a value of N=3.
K	INT	1	Degree of the finite element interpolation. A linear tensor product basis is specified by K=1.
NVAR	INT	1	The number of partial differential equations being solved (i.e.,the number of state variables in the equation system).
NALG	INT	1	The number of algebraic dependent variables available for computed output in the ALGFNL routines.
SOLTYP	INT	1	The iterative matrix solution type. If 0, bandwidth=3 and simplified jacobian is used If 1, bandwidth=15 and simplified jacobian is used If 2, bandwidth=15 and full jacobian is used If 10,11, or 12 same as above but IRK method is used
NRST	INT	1	Restart block number. If NRST=0, no restart is attempted. If NRST $>$ 0, a restart is attempted at restart block equal to NRST. If NRST $<$ 0, a restart is attempted at the absolute value of NRST, and print outs are produced at each restart record up to NRST (a time history of the system status is produced).
FACT	LOG	1	If .true., a tensor-product factored quasi-Newton matrix is produced. If .false., only the first direction is swept.
LDETJ	LOG	1	If .true., metric consistency is utilized If .false., the old metric methods are used NEW RECORD
CFDNAM	CHAR	NVAR+NALG	An array of four-character variable names which can be output by the ALGFNL routine. The first NVAR names correspond to the state variable names. The remaining NALG names correspond to the algebraic names of additional dependent variables. NEW RECORD
I		NVAR	The following are entered in a do-loop:
TWSALP(I)	REAL	1	The Taylor-Weak Statement arbitrary parameter α.

variable	type	length	description
TWSBET(I)	REAL	1	The Taylor-Weak Statement arbitrary parameter β.
TWSGAM(I)	REAL	1	The Taylor-Weak Statement arbitrary parameter γ.
TWSMU(I)	REAL	1	The Taylor-Weak Statement arbitrary parameter μ.
			NEW RECORD
			End of the I do loop.
NPR	INT	N	The number of progressions or macro elements in each dimension. NPR(1) specifies the time discretization. NPR(2) to NPR(N) specify the number of macro elements in each spatial direction. If NPR(1) $<$ zero, the execution stops at the end of NEM(1) time steps.
			NEW RECORD
I		N	The following values are entered in a do loop:
XI(I)	REAL	1	The initial value of the macro element in the ith dimension in computational space.
XF(I,J)	REAL	NPR(I)	The sequence of final macro element spacings in the ith dimension in computational space.
PR(I,J)	REAL	NPR(I)	The progression ratio of the ith macro element.
NEM(I,J)	INT	NPR(I)	The number of elements in the ith macro element.
NPROP(I,J)	INT	NPR(I)	The material property flag (not used at this time).
			NEW RECORD
			End of the I do loop.
J		NEM(I,J)	The following input sequence is repeated in a do loop:
I		NEM(I,J)	The following input sequence is repeated in a do loop:
LJ	INT	1	Dummy integer usually to signify the macro element number.
XC1(I,J)	INT	1	X coordinate of Node 1 of the (i,j) macro element.
XC2(I,J)	INT	1	X coordinate of Node 2 of the (i,j) macro element.
XC3(I,J)	INT	1	X coordinate of Node 3 of the (i,j) macro element.
XC4(I,J)	INT	1	X coordinate of Node 4 of the (i,j) macro element.
			NEW RECORD
LJ	INT	1	Dummy integer usually to signify the macro element number.
YC1(I,J)	INT	1	Y coordinate of Node 1 of the (i,j) macro element.
YC2(I,J)	INT	1	Y coordinate of Node 2 of the (i,j) macro element.
YC3(I,J)	INT	1	Y coordinate of Node 3 of the (i,j) macro element.
YC4(I,J)	INT	1	Y coordinate of Node 4 of the (i,j) macro element.
			NEW RECORD

variable	type	length	description
LJ	INT	1	Dummy integer usually to signify the macro element number.
XC5(I,J)	INT	1	X coordinate of Node 5 of the (i,j) macro element.
XC6(I,J)	INT	1	X coordinate of Node 6 of the (i,j) macro element.
XC7(I,J)	INT	1	X coordinate of Node 7 of the (i,j) macro element.
XC8(I,J)	INT	1	X coordinate of Node 8 of the (i,j) macro element.
			NEW RECORD
LJ	INT	1	Dummy integer usually to signify the macro element number.
YC5(I,J)	INT	1	Y coordinate of Node 5 of the (i,j) macro element.
YC6(I,J)	INT	1	Y coordinate of Node 6 of the (i,j) macro element.
YC7(I,J)	INT	1	Y coordinate of Node 7 of the (i,j) macro element.
YC8(I,J)	INT	1	Y coordinate of Node 8 of the (i,j) macro element.
			NEW RECORD
			End of the I loop.
			End of the J loop.
XMACH	REAL	1	Free stream fluid Mach number.
TI	REAL	1	Free stream fluid temperature (F).
PI	REAL	1	Free stream fluid pressure (psia).
FILL	REAL	1	Fraction of the y (or eta) domain which is filled with a linear profile approximating a boundary layer initial condition. This is used in the default initial condition if not overwritten in subroutine INITGE. If FILL < zero, the boundary layer is filled in the entire y space. If FILL > zero, the boundary layer is filled in over half the y space.
TREF	REAL	1	The reference heat sink temperature for convection heat transfer boundary conditions specified in dimensionless form.
HTC	REAL	1	The convection heat transfer coefficient (Btu/hr/ft**2/F)
			NEW RECORD
REYINP	REAL	1	Input Reynolds number. If REYINP=0., the true Reynolds number is calculated. If REYINP > 0., the Reynolds number is set to the value of REYINP. If REYINP < 0., the Reynolds number is initially set to REYINP and then gradually changes to the true Reynolds number (as if REYINP =0.) during the time integration. The Reynolds number will equal the true Reynolds number when the Maximum Courant

variable	type	length	description
			number in the flow field is equal to COURIN (see below).
ERRFAC			Tensor-Product Matrix factorization error reduction factor =1 no error reduction, recommend 10^{-6}.
BFREQ			Digital filter break frequency factor. A value of one filters the time step. A value of ∞ does no filtering.
			NEW RECORD
WEPIL	REAL	1	Minimum residual for all iterative processes.
MITER	INT	1	Maximum number of iterations for all iterative processes. For the system Newton iteration process, a negative value for MITER will cause the time step to advance and the iteration process to proceed if the number of iterations is greater than or equal to the absolute value of MITER. For positive MITER, the program will stop if the iterations exceed MITER.
PMAX	INT	1	The number of time steps between print and restart edits.
AUTOT	LOG	1	If .true., time steps will be determined automatically. If .false., time steps are specified on input.
TEPIL	REAL	1	The maximum fractional change in any nodal degree of freedom during the iteration process.
NITER	INT	1	The number of time steps which occur before the automatic time step control becomes effective.
THETA	REAL	1	The implicitness factor in the Newton iteration procedure. For theta=1., it is fully implicit. For theta=0., it is fully explicit. For Theta $<$ 0., theta will initially be set to its positive value and gradually changes to a value of 0.5 during the time integration. Theta will be equal to 0.5 in this automatic mode when the maximum Courant number in the flow field is equal to COURINP (see below). No effect if IRK is utilized.
NERRQT	INT	1	(not used).
			NEW RECORD
DTMIN	REAL	1	The minimum time step allowed.
DTMAX	REAL	1	The maximum time step allowed.
SMALL	REAL	1	A number typically used to represent a large number when inverted or to replace a division by zero. It is

variable	type	length	description
			recommended that a value equal to 1*10**(-nsd) be used, where nsd equals the number of significant digits realized in the computer being used.
ITPMAX	INT	1	The number of time steps between print and restart edits.
COURIN	REAL	1	The maximum Courant number allowed in the flow field. This number has impact on time step, Reynolds number, and theta.
EPILLO	REAL	1	The minimum value of the fractional change in a nodal degree of freedom which is realized before a time step will change (increase).
EPILHI	REAL	1	The maximum value of the fractional change in a nodal degree of freedom which is realized before a time step will change (decrease).
			NEW RECORD
SURFAC	LOG	2*(N-1)	If .true., the surface number is not a wall and experiences a contribution for $m \cdot \hat{n}$. If .false., the surface is a wall and no contribution for $m \cdot \hat{n}$ is made (i.e., $m \cdot \hat{n}$ equal zero on that surface). The surface numbers 1,2,3,4 correspond to left, right, bottom, and top as seen in the computational space.
			NEW RECORD
FIX	CHAR	$\leq$NVAR	A state variable name as specified in CFDNAM which is desired to be constant (same as the initial conditions) throughout the calculation. A blank entry will not allow any constant state variable (default).
			NEW RECORD
NFIX	INT	$\leq$NVAR	The number of time steps which each state variable specified above remains constant.
			NEW RECORD
LDIR	CHAR	$\leq$NVAR	The state variables which are constant (Dirichlet boundary conditions) on surface 1 (left).
			NEW RECORD
RDIR	CHAR	$\leq$NVAR	The state variables which are constant (Dirichlet boundary conditions) on surface 2 (right).
			NEW RECORD

variable	type	length	description
BDIR	CHAR	$\leq$NVAR	The state variables which are constant (Dirichlet boundary conditions) on surface 3 (bottom). NEW RECORD
TDIR	CHAR	$\leq$NVAR	The state variables which are constant (Dirichlet boundary conditions) on surface 4 (top). NEW RECORD
LNEU	CHAR	$\leq$NVAR	The state variables which experience Neumann boundary conditions on surface 1 (left). NEW RECORD
RNEU	CHAR	$\leq$NVAR	The state variables which experience Neumann boundary conditions on surface 2 (right). NEW RECORD
BNEU	CHAR	$\leq$NVAR	The state variables which experience Neumann boundary conditions on surface 3 (bottom). NEW RECORD
TNEU	CHAR	$\leq$NVAR	The state variables which experience Neumann boundary conditions on surface 4 (top). NEW RECORD
LPRT	CHAR	$\leq$(NVAR+NALG)	The variable names as specified in CFDNAM which will be printed on output (plotting is independent of this). NEW RECORD
NODCON	INT	$\leq$NDF	The node numbers which may be held constant. NDF is the total number of nodal degrees of freedom in the problem. If no entry is made, no nodes are held constant (default). If NODCON(i) is $<$ zero, it is ignored but holds significance in the SPECIA routine. Used to implement special boundary conditions. NEW RECORD
TEST	CHAR	1	If a "Y" appears in column 1, then the initial conditions are specified in subroutine INITGE with input parameter IPROB. If a "Y" does not appear in column 1, default pseudo-boundary layer initial conditions are used or if it is a restart problem, the conditions at restart are used. END OF INPUT

Appendix E

Mini-Investigation of Band-Matrix Direct Solvers

There is a need within our CFD research group for a linear equation solver which utilizes the vector facility of the IBM-3090. The requirements are that the solution utilize a direct method (there are no symmetry properties associated with the matrix) and that the subroutine take advantage of the banded property of the matrix. An existing solver (subroutine SOLVR written by J. Iannelli) efficiently (it is fast and requires a minimum amount of memory) serves this purpose. However, SOLVR does not utilize the vector facility. Therefore, the question arises: Is there a replacement subroutine which is readily available? These notes summarize my conclusions.

The NAG library has several subroutines which can solve banded and tridiagonal matrix systems. However, if you look closely, you will find that all of the NAG routines in this category solve an approximate solution and not an accurate solution. I have quite painfully seen the difference between an approximate (NAG) and accurate (SOLVR) solution. On this basis, I do not recommend using any of the NAG banded matrix routines for our CFD research.

Three other libraries were investigated: (1) in IMSL, the DSLARB subroutine, (2) in ESSL, the DGBF/DGBS subroutines, and (3) in LINPACK the DGBFA/DGBSL. The LINPACK routines were tested and found to be remarkably similar to the ESSL routines (in fact the ESSL routines are probably modeled directly from the LINPACK routines). Because the ESSL routines are tailored by IBM specifically for the IBM-3090VF, no results of the LINPACK testing is included here. The remaining two routines provide accurate results at remarkably different efficiencies, as will be seen. Furthermore, both subroutines require similar matrix structures which are different than the SOLVR structure.

I generated a test subroutine ISOLVR which optionally executes SOLVR, IMSL, and/or ESSL with the same amount of work to be done by all. In this manner, the performance may be compared equally. My matrix is stored in the array SMAT representing the entire physical domain of the problem. Previously, I had been solving the entire matrix at one time with SOLVR. However, because both DSLARB and DGBF/DGBS subroutines require significant additional memory, I restructured the solution procedure to solve for blocks of grid lines rather than the entire domain. This turned out to be good since both routines also require a different storage order for the matrix. Furthermore, I plan to execute this new routine in parallel mode, for which the new structure is well suited for.

The results of my analysis are shown in Table E.1. I found that the IMSL DSLARB routine was remarkably inefficient. Not only did it require more memory but it also produced significantly fewer solutions for the same amount of CPU time. Also, because it uses iterative refinement, DSLARB had problems with Dirichlet data. I put 10^{15} on the diagonal, zero out the off- diagonals, and set the right-hand side equal to zero. In order to get DSLARB to work without printing a bunch of warning messages (that my iterative refinement was only good to 10^{-16}), I had to change the diagonal entries to 10^{6}.

Table E.1: Comparison of Alternative Banded-Matrix Direct Solvers

Category	SOLVR		IMSL		ESSL	
BAND=	3	15	3	15	3	15
CPU (sec)	330	330	330	330	330	330
time steps	286	165	149	72	278	175
iterations per step	1.57	1.55	1.55	1.50	1.57	1.59
F_{eff}	.200	.351	.778	.834	.206	.323
F_{vec}	.328	.186	.225	.180	.326	.206

Notes:

- The total degrees of freedom for this particular test is 1836.
- F_{eff} is defined as the CPU time in seconds divided by [degrees of freedom $\times$ number of time steps $\times$ number of iterations per step] all multiplied by 10^3 and is a measure of the computational efficiency of the work being done.
- F_{vec} is defined as the CPU time on the vector facility divided by the total CPU time and is a measure of the utilization of the vector facility.
- The Implicit Runge-Kutta (IRK) method is employed in the present analysis. Therefore, the number of iterations per step shown in the table must be multiplied by two. This is because IRK solves the equation system two times per global iteration. The number of iterations shown in the table is greater than 1.0 because the time step is being tested for change at each step (the present calculation is in the early part of a transient where the time step is changing considerably).

For the small band-width (3) problems, the SOLVR routine performed a little better than the ESSL DGBF/DGBS subroutine. However, for the larger bandwidth (15), the DGBF/DGBS routines slightly out-performed the scalar SOLVR routine. This is the desired and expected result, although not as dramatic as hoped. This indicates even more that the SOLVR routine is quite powerful!

Addendum Post Parallel FORTRAN

The DGBF/DGBS routines do have a slight speed advantage over the SOLVR routine for the large bandwidth formulation. However, there is significantly more storage requirements and the ordering of the matrix is different from the SOLVR ordering (which is also the same as the solver ordering of the LEARN code used by our research group). I did investigate and test reordering the matrix and splitting up the solver into line groups rather than the entire solution at once, thus saving considerable storage but adding computational overhead. Furthermore, both the ESSL and LINPACK routines did not function correctly as an originated task within the IBM parallel FORTRAN compiler whereas the SOLVR routine functioned correctly. All things considered, I decided that the SOLVR subroutine is still the best choice for all algorithms formulations considered in this research.

Appendix F

A Digital Filter to Reduce Time-Step Oscillations Near Steady-State

[a] This Appendix is a derivation of a digital filter utilized to eliminate the time-step frequency inherent with implicit time integration as the time step nears the stability limit. The digital filter should not be used where time accuracy is important. It is ideal for steady-state digital calculations.

The digital filter is based on an equivalent analog low-pass filter. The analog circuit which describes the low-pass filter is shown in Figure F.1.

An integral equation which describes the output of this circuit is

$$z = -p\int_0^t y\,dt = -p\int_0^t (z - x)dt. \tag{F.1}$$

Differentiating Equation F.1 with respect to time yields

$$\frac{dz}{dt} = -py + py(t = 0) = p(z - x), \tag{F.2}$$

since $y(t = 0) = 0$. Now perform a Laplace transform of Equation F.2 to yield

$$sZ - Z(t = 0) = -pY = -p(Z - X), \tag{F.3}$$

[a] Reference: R. A. Hedrick and J. R. Penland, personal communication.

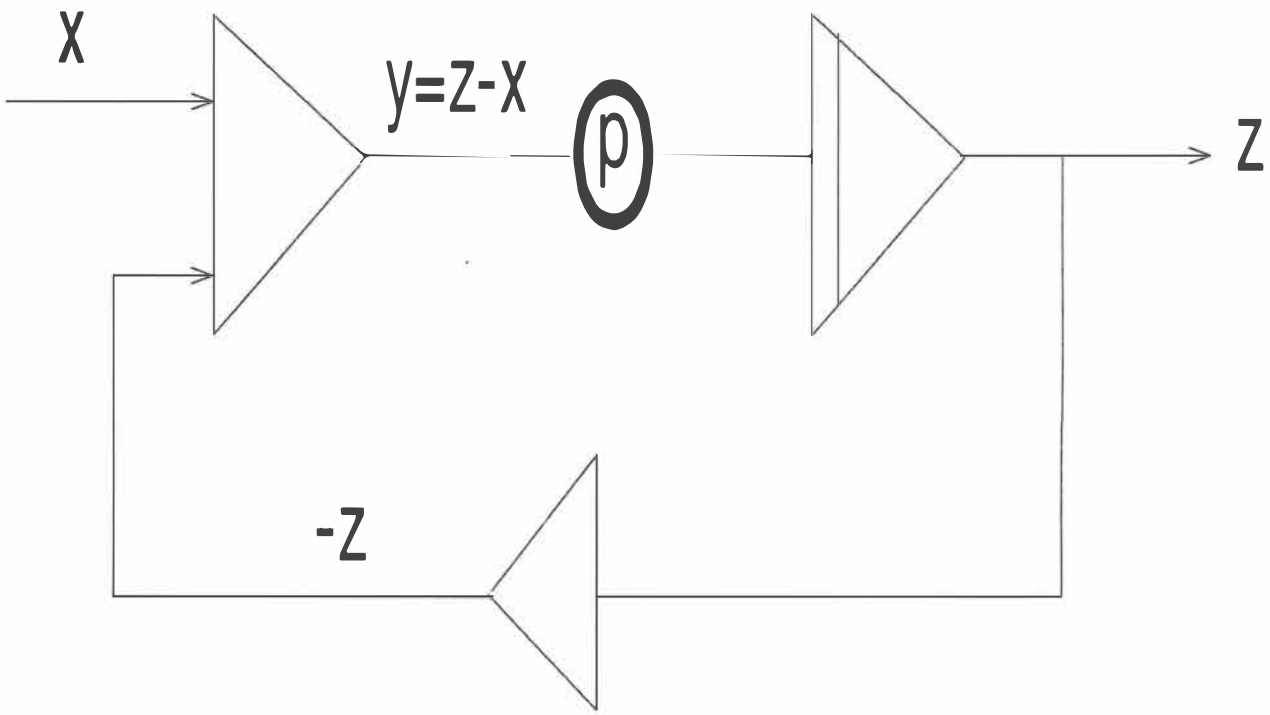


Figure F.1: Analog Circuit for a Low-Pass Filter.

where $Z(t=0)=0, Z=\mathcal{L}[x], Y=\mathcal{L}[y]$, and$X=\mathcal{L}[x]$. Solving Equation F.3 for Z yields

$$Z=\frac{-pY}{s}=-\frac{p}{s}(Z-X)\Rightarrow Z=\frac{pX}{s+p}. \tag{F.4}$$

The frequency response of Equation F.4 follows a classical low-pass filter with the gain p equal to the break frequency.

It is desirable to simulate this analog filter with a digital operation for digital simulation. Recognizing that Equation F.2 is an ordinary differential equation that can be approximated by a Taylor series backward finite difference results in the following equation

$$\frac{z^{n+1}-z^n}{\Delta t}=-p(z^{n+1}-x^{n+1}). \tag{F.5}$$

Solving Equation F.5 for z^{n+1} yields

$$z^{n+1}=\frac{z^n+x^{n+1}p\Delta t}{1+p\Delta t}. \tag{F.6}$$

Let it be assumed that it is desirable to filter out most frequencies above the time-step frequency. This is done by setting the break frequency equal to the inverse of the time step ($p=\frac{1}{\Delta t}$). Substituting this into Equation F.6 yields

$$z^{n+1}=\frac{1}{2}\left[z^n+x^{n+1}\right]. \tag{F.7}$$

Now let it be assumed that it is desirable to remove frequencies even lower than the time-step frequency to virtually guarantee that the time-step frequency is removed. For example, let $p=\frac{1}{2\Delta t}$ to yield

$$z^{n+1}=\frac{2}{3}\left[z^n+\frac{x^{n+1}}{2}\right]. \tag{F.8}$$

This digital filter is very easily implemented into a CFD code by letting the state variable array q equal to the variable z and the state variable array

residual δq equal to the variable x. The normal sequence $q^{n+1} = q^n + \delta q^{n+1}$ is then replaced by the digital filter. The steady-state results shown in this dissertation utilize a break frequency of $p = \frac{1}{\Delta t}$.

VITA

James Downing Freels was born in Morristown, Tennessee, on December 1, 1954 the son of James C. and Patricia M. Freels. While in high school, Jim participated in many sports and other extra-curricular activities including a member/player of the AAA state championship football team in 1969-70. After graduation from Morristown-Hamblen High School East in June 1972, he received his first experience in the engineering field through a summer and part-time job with Culley Engineering and Manufacturing Company (CEMCO) of Whitesburg, Tennessee.

In the fall of 1972, he started his college education as a Mechanical Engineering (ME) major at Virginia Polytechnic Institute and State University (VPI or Virginia Tech) in Blacksburg, Virginia. During his stay at VPI, he participated in the co-op work study program with Tennessee Eastman Company (TEC) of Kingsport, Tennessee. He completed four work quarters as an ME student with TEC.

In the spring of 1975, he transferred to the University of Tennessee in Knoxville (UTK). He obtained the Bachelor of Science degree in Nuclear Engineering (BSNE) on June 10, 1977. During his undergraduate study at UTK, he worked part-time with the Tennessee Valley Authority (TVA) in Knoxville from August 1975 to June 1977 on the design of principal piping systems for the Sequoyah and Watts Bar nuclear plant(s) design projects.

In June, 1977 he was awarded a graduate research assistantship from the NE department of UTK. He remained a full time student for the majority of this period and received the Master of Science degree with a major in Nuclear Engineering (MSNE) in June 1979. His thesis was entitled "An Investigation of High Order and Low Order Dynamic Modeling of a Complete Pressurized Water Reactor Nuclear Power Plant."

He was employed with Science Applications, Inc. (SAI, now Science Applications International Corporation or SAIC) of Oak Ridge, Tennessee on February, 1979. His work with SAI involved various contracts including digital simulation of physical processes and simulation of power plant instrument response using data base management techniques.

In August, 1983 he began employment with Technology for Energy Corporation (TEC) of Knoxville, Tennessee. His work with TEC was primarily through support of the design of new instrumentation systems through digital simulation of physical processes. He also completed several unique and interesting consulting and research projects with TEC.

On September, 1988 he was re-employed with SAIC of Oak Ridge, Tennessee. This work included safety analysis of DOE nuclear facilities, development of a digital feedwater control system for BWR nuclear power plants, and development of CFD numerical algorithms.

In May, 1991 he was employed with the Research Reactors Division of Oak Ridge National Laboratory (ORNL). His principal duties are related to safety and design analyses of the operating High Flux Isotope Reactor (HFIR) and the proposed Advanced Neutron Source (ANS).

Jim is married to the former Elizabeth Gilmore Little of Morristown, Tennessee. Libby and Jim have two sons, Matthew Webster and Michael Little. The Freels family reside in Clinton, Tennessee.