An Unstructured Grid Incompressible Navier-Stokes Algorithm for Convective Heat Transfer Based on Artificial Compressibility

By

Jessica Elaine Kress

Approved:

Lafayette Taylor
Professor of Computational Engineering
(Director of Thesis)

Daniel Hyams
Associate Professor of Computational Engineering
(Committee Member)

William Sutton
Dean of College of Engineering and Computer Science

Timothy Swafford
Research Assistant Professor of Computational Engineering
(Committee Member)

Jerald Ainsworth
Dean of the Graduate School

An Unstructured Grid Incompressible Navier-Stokes Algorithm for Convective Heat Transfer Based on Artificial Compressibility

By

Jessica Elaine Kress

A Thesis
Submitted to the faculty of
The University of Tennessee, Chattanooga
in Partial Fulfillment of the Requirements
for the Degree of Master of Science
in Computational Engineering

The University of Tennessee, Chattanooga
Chattanooga, Tennessee

December 2012

ABSTRACT

The development of an explicit unstructured grid finite-volume scheme for solving the full incompressible Navier-Stokes equations along with the energy equation in a strongly coupled manner is presented. The Boussinesq approximation is utilized to account for thermal buoyancy. The method of artificial compressibility is used to solve the resulting equations in a time marching fashion. Roe's approximate Riemann solver is used for the construction of the numerical flux. An eigensystem is derived for the flux Jacobian matrix, which is used in the evaluation of the numerical flux and the characteristic variable boundary conditions. The resulting algorithm is validated by simulating canonical test cases from the three regimes of convective heat transfer. The computed solutions are in close agreement with analytical solutions and other benchmark computations.

DEDICATION

This is for you, Daddy.

ACKNOWLEDGEMENTS

The author would like to sincerely thank her major professor Dr. Lafayette Taylor for his patience and guidance throughout the course of the research. Further thanks go to Dr. Daniel Hyams and Dr. Kidambi Sreenivas for assistance in understanding of and guidance in the research. Also, the author wishes to thank Dr. Timothy Swafford and all others at the SimCenter for the continued display of kindness and sharing of wisdom.

The parents of the author also deserve a sincerest appreciation for their support and encouragement throughout the earning of this degree. Last, but certainly not least, the author very especially would like to thank Peter Maginnis and Cheryl Kress for their patience and understanding.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

A, flux Jacobian matrix

α_T, coefficient of thermal expansion

$\underline{b}$, body force per unit mass β, artificial compressibility parameter

CFL, Courant number

C_P, pressure coefficient

c_p, specific heat at constant pressure

$\vec{F}$, inviscid flux vector

F, directed invisicd flux vector, $\vec{F} \cdot \vec{n}$

$\vec{F}_v$, viscous flux vector

F_v, directed viscous flux vector, $\vec{F}_v \cdot \vec{n}$

Fr, Froude number

g, acceleration due to gravity

Gr, Grashof number

$\tilde{I}$, identity matrix

$\hat{i}, \hat{j}, \hat{k}$, unit vectors in the Cartesian coordinate directions x, y, z

k, thermal conductivity

L, characteristic length

$\tilde{L}$, left eigenvalue matrix

$\tilde{\Lambda}$, diagonal matrix with eigenvalues as diagonal

μ, viscosity

Nu, Nusselt number

$\vec{n}$, normal vector associated with a control volume face

η, direction normal to boundary

ν, kinematic viscosity

ω, volume domain of integration

p, pressure

$\hat{p}$, combined pressure

Pe, Peclet number

Pr, Prandtl number

Φ, Roe flux approximation for inviscid flux vector

Φ_v, a numerical approximation for the viscous flux vector

Q, solution variable vector

$\tilde{R}$, right eigenvalue matrix

Re, Reynolds number

Ri, Richardson number

ρ, density

$\tilde{\sigma}$, deviatoric part of the Stokes tensor

t, time

T, temperature

T_w, reference wall temperature

τ, shear stress

θ, velocity normal to control volume face, $un_x + vn_y + wn_z$

Υ, temperature vector

$\underline{u}$, inertial velocity

W, characteristic variable vector

$\vec{x}_i$, coordinate location of node i

x, y, z, Cartesian coordinate directions

CHAPTER 1

INTRODUCTION

The primary goal of this work is to implement an explicit, three-dimensional flow solver capable of modeling problems dealing with laminar, incompressible, convective heat transfer flow. The physical conditions leading to incompressible flow occur as a property of the fluid or when the velocity of the fluid is low enough that the variation of density is negligible, thus allowing density to be modeled as a constant. A key aspect of incompressible flow is that of pressure being not a thermodynamic variable, but instead a primitive variable. It becomes a problem of some interest to determine a method with which to correctly model the dependency between pressure and the velocity [1]. One approach that has been developed to deal with this issue is the insertion of an artificial time derivative of pressure in the mass conservation equation. This method, termed artificial compressibility, has several advantages, one being the coupling of pressure and velocity so that the resulting system of equations is hyperbolic and can be solved in the same time marching manner as that of most compressible algorithms [1]. As stated above, an additional consideration in this work is convective heat transfer, which occurs when there is a temperature difference between a surface and the surrounding fluid. Convection has three main subsets which consist of forced, natural, and mixed. Forced convection occurs when fluid flows over a heated or cooled object. In natural or free convection, however, the velocity within the fluid develops naturally from the effects of thermal buoyancy. Mixed convection is the term for the occurrence of both types of convection.

Current work being done in this area can be seen in the implementation done by Koblitz et al. [2]. In this case, the goal was to improve the existing in-house code in order to create better models for wind resource assessment on complex terrain. According to Koblitz et al. [2] few current models for wind related problems show the effects of temperature stratification, which is a drawback since thermal buoyancy can have a significant effect on the flow. A primitive variable approach was used with artificial compressibility, while the Boussinesq approximation is implemented for buoyancy. In addition, the code utilized a finite volume method. Several variations on the lid-driven square cavity test case were investigated in order to validate their work. This case is a prime choice for validation since almost all of the fluid mechanical phenomena are exhibited [2], and a good deal of benchmark data is in existence. All test cases considered were steady, laminar, and two-dimensional. The various dimensionless parameters used are as follows $316 < Re < 5000$, $Gr = 10^6$, and $0.1 < Ri < 10$. The three sub-test cases consisted of a thermally neutral case, a gravitationally stable case, and a gravitationally unstable case. The computed results obtained by Koblitz et al. [2] were excellent, and the first two sub-cases in [2] will be repeated in this work and the results compared.

Other work utilizing artificial-compressibility and heat transfer is being done by Kameyama et al. [3] in order to model the mantle of the earth. This case is much different than that of more common fluids, like water and air, due to the extremely high viscosity of the fluid. Due to this fact, the non-linear terms as well as the time-derivative terms in the momentum equation can be neglected, making the system a set of elliptic equations instead of hyperbolic. Although the method utilized by Kameyama et al. [3] incorporates pseudo-compressibility, a finite-volume cell-centered approach, local time stepping, and treats the Navier-Stokes equations as a whole, the elliptical nature of the equations fits well with the application of a multigrid iteration method. The test cases by Kameyama et al. [3] are similar to those to be examined in this work when the fluid considered has the same viscosity

throughout. One such test case is implemented in a hexahedral shaped domain. The fluid is isoviscous, the bottom and top walls are heated, and the vertical walls are adiabatic. The results obtained by Kameyama et al. [3] compare very well with previous benchmark data.

The work done by Agrawal et al. [4] is more similar in nature to that of Koblitz et al. [2]. According to Agrawal et al. [4] work dealing with mixed convection in a closed cavity is fairly scarce, although it is quite useful in modeling situations such as heat exchange between a container and a laterally flowing stream as well as a lubricating groove between sliding plates. The numerical method used in [4] is also based on artificial compressibility, the reasons for the choice being, robustness in three dimensions (as opposed to the stream-vorticity approach), the relative simplicity in comparison to the pressure method, and an identical approach being possible for both the convective and diffusive terms. Agrawal et al. [4] also considered the effects of turbulence, which is accomplished using an SST model [5]. The code used in the study was finite-volume discretized, cell-centered, and explicit. Previous validation was done for cases of flow including inviscid, viscous, and natural convection. The key test case validated in [4] is that of mixed convection. The first test case presented in [4] is the lid-driven cavity, like that shown in [2] and which will also be repeated in the current work. Agrawal et al. [4] performed this case for $Re = 10^3$ and $Gr = 10^2$ and 10^6, in addition to doing further cases in which turbulence was considered. For the former case the results obtained were excellent relative to previous benchmark data. Other cases, for which no benchmark data existed, were shown to be almost entirely grid independent.

The overall goal for the present work is to study similar problems like those described above by extending the earlier work of Taylor [1] and Er [6]. The artificial compressibility approach was successfully implemented by Taylor [1] in an incompressible flow solver utilizing a primitive variable approach. Taylor [1] obtained excellent results relative to benchmark solutions and theoretical data. The test case of inviscid flow over a cylinder performed in his work will be repeated here, and the results compared. Subsequently, Er [6] successfully

added the energy equation to the existing work of Taylor [1], and this combined approach, especially the use of the Bousinesq approximation, will be followed here as well. The results obtained by Er [6] were also excellent, and the test case of natural buoyancy used in his work will be repeated here as well. An additional goal of this work is to extend previous work, developed for use on structured grids, to a flow solver suitable for unstructured, multi-element grid topologies. Modern computational fluid dynamics has moved in the direction of using unstructured grids as they have several advantages over their structured counterparts, including their ease in implementation for highly complex geometries, and their ease in incorporation with automated grid generation. One drawback to unstructured grids is additional node to node connectivity, thus requiring greater computer memory and necessitating solving a matrix of an unknown sparse structure [7]. However, the ease of grid generation combined with more optimally placed nodes outweighs the disadvantages, and illustrates why the use of unstructured grids is now so widespread.

The contents of this work are summarized below. First, the governing equations of the physical phenomena will be presented, and explanations of various approximations made shall be given, after which the numerical discretization of the equations will be shown. An explanation of the various boundary conditions implemented will then be given, followed by a breakdown of the various test cases investigated for validation. These test cases include inviscid flow over a cylinder, forced convection over a flat plate, natural convection in a cavity, lid-driven cavity with neutral heat transfer, and mixed convection in a lid-driven cavity. The objective is to generate results matching closely with previous benchmark data, and a summary of results will be given in the conclusion of the work.

CHAPTER 2

GOVERNING EQUATIONS

This work deals with laminar convective heat transfer flow. The equations governing this situation are the three-dimensional incompressible Navier-Stokes equations including the energy equation. This chapter will begin by presenting the equations in tensor invariant form. The Boussinesq approximation will then be applied and an artificial compressibility parameter added. After non-dimensionalization, the equations will be presented in final vector, integral form. These equations are also written in expanded form for clarification in Appendix A.

Tensor Invariant Form

Continuity

$$\operatorname{div} \underline{u} = 0 \tag{2.1}$$

Momentum

$$\rho \frac{\partial \underline{u}}{\partial t} + \rho \operatorname{div}(\underline{u}\,\underline{u}) = -\operatorname{div}(p\tilde{I}) + \operatorname{div} \tilde{\sigma} + \rho \underline{b} \tag{2.2}$$

Energy

$$\rho\, c_p \frac{\partial T}{\partial t} + \rho\, c_p \operatorname{div}(\underline{u}\, T) = \operatorname{div}(k \operatorname{grad} T) \tag{2.3}$$

where ρ is density, p is pressure, $\underline{u}$ is inertial velocity, T is temperature, t is time, $\tilde{\sigma}$ is the deviatoric part of the Stokes tensor, $\tilde{I}$ is the identity matrix, $\underline{b}$ is the body force per unit mass, c_p is the specific heat at constant pressure, and k is the thermal conductivity. An underscore below a variable signifies that it is a vector and tilde above indicates it is a

tensor. Also, $\text{div}(\cdot) = \frac{\partial(\cdot)}{\partial x_m} \cdot \underline{i}_m \, , m = 1, 2, 3$ and $\text{grad}(\cdot) = \frac{\partial(\cdot)}{\partial x_m} \underline{i}_m \, , m = 1, 2, 3$. Gravity is the only body force taken into account in this work; therefore, the body force per unit mass is defined below as

$$\underline{b} = -\text{grad}(gx_2) = -\text{div}(gx_2\tilde{I}) \tag{2.4}$$

where g is acceleration due to gravity and x_2 the coordinate direction opposite that of the force of gravity. In this work, the effects of thermal buoyancy are considered, consequently the density in the momentum equation must be carefully approximated. An approach for dealing with this issue is to use the Boussinesq approximation, in which the density will be treated as a constant in all terms except for the body force term. It is known that the change in density due to temperature is indeed very small but still significant to the problem [8]; therefore, the density in the body force term is treated as a function of temperature. Letting ρ_∞ be the density of the fluid at temperature T_∞, then the truncated Taylor series expansion of density is

$$\rho(T) = \rho_\infty + \left(\frac{\partial \rho}{\partial T}\right)_p (T - T_\infty) \tag{2.5}$$

$$= \rho_\infty \left[1 - \alpha_T (T - T_\infty)\right] \tag{2.6}$$

where α_T is the coefficient of thermal expansion for a fluid at constant pressure defined as

$$\alpha_T = -\frac{1}{\rho_\infty}\left(\frac{\partial \rho}{\partial T}\right)_p$$

It should be noted that the effect of pressure is neglected in Equation 2.5, which can be rewritten in the following way

$$\rho = \rho_\infty (1 + \rho') \tag{2.7}$$

where

$$\rho' = -\alpha_T(T - T_\infty)$$

Replacing density in the body force term of the momentum equation with the definition in Equation 2.7 , and rearranging somewhat, gives the following

$$\frac{\partial \underline{u}}{\partial t} + \mathrm{div}(\underline{u}\,\underline{u}) + \frac{1}{\rho_\infty}\mathrm{div}(\hat{p}\tilde{I} - \tilde{\sigma}) = -\rho'\mathrm{div}(gx_2\tilde{I}) \tag{2.8}$$

where $\hat{p} = p + \rho_\infty g x_2$, which simply absorbs hydrostatic pressure into the definition of pressure.

Artificial Compressibility

Since this work deals with the incompressible regime, a numerical approach of modeling incompressibility is needed. Following Taylor [1], among others, the chosen approach here is addiing to the continuity equation a fictitious time derivative of pressure, which then becomes

$$\frac{1}{\beta}\frac{\partial \hat{p}}{\partial t} + \mathrm{div}\,\underline{u} = 0 \tag{2.9}$$

where β is the artificial compressibility parameter and is often set in a range from 5 to 15 [7]. It should be noted that $\hat{p}$, as defined in the previous section, is used as suggested by Beddhu et. al [8] and also by Er [6]. Using $\hat{p}$ allows the body force potential term to be integrated into the eigensystem [6].

Nondimensionalization

The equations are now ready to be non-dimensionalized using the following relations:

$$x^* = \frac{x}{L}\,,\ y^* = \frac{y}{L}\,,\ z^* = \frac{z}{L}\,,\ t^* = t\frac{u_\infty}{L}\,,$$

$$u^* = \frac{u}{u_\infty}\,,\ p^* = \frac{p - p_\infty}{\rho_\infty u_\infty^2}\,,\ T^* = \frac{T - T_\infty}{\Delta T}\,,\ where\ \Delta T = T_w - T_\infty\,,$$

$$\rho^* = \frac{\rho}{\rho_\infty}\,,\ \mu^* = \frac{\mu}{\mu_\infty}\,,\ k^* = \frac{k}{k_\infty}$$

$$\mathrm{div}^*(\cdot) = \frac{1}{L}\frac{\partial(\cdot)}{\partial x_m^*} \cdot \underline{i}_m\,, m = 1,2,3\,,\ \mathrm{grad}^*(\cdot) = \frac{1}{L}\frac{\partial(\cdot)}{\partial x_m^*}\underline{i}_m\,, m = 1,2,3\,, and$$

$$\tilde{\sigma^*} = \frac{L}{\mu_\infty u_\infty}\tilde{\sigma}$$

where L is a characteristic length and T_w is the reference wall temperature. All the quantities subscripted with ∞ refer to the freestream values, while the superscript * signifies that the variable is non-dimensional. Reynolds number (Re_∞), Prandtl number (Pr_∞), Peclet number (Pe_∞), Grashof number (Gr_∞), and Froude number (Fr_∞) are the non-dimensional parameters that appear in the equations. They are defined as follows

$$Re_\infty = \frac{\rho_\infty u_\infty L}{\mu_\infty}\,,\ Pr_\infty = \frac{\mu_\infty c_p}{k_\infty}\,,\ Pe_\infty = Re_\infty Pr_\infty = \frac{\rho_\infty c_p u_\infty L}{k_\infty}\,,$$

$$Gr_\infty = \frac{g\alpha_T \Delta T L^3}{\nu_\infty^2}\,,\ Fr_\infty = \frac{u_\infty}{\sqrt{gL}}$$

After non-dimensionalizing, and dropping the * superscript for clarity, the equations are as follows

Continuity

$$\frac{1}{\beta}\frac{\partial \hat{p}}{\partial t} + \operatorname{div} \underline{u} = 0 \tag{2.10}$$

Momentum

$$\frac{\partial \underline{u}}{\partial t} + \operatorname{div}(\underline{u}\,\underline{u}) + \operatorname{div}(\hat{p}\tilde{I} - \frac{1}{Re_\infty}\tilde{\sigma}) = \frac{\alpha_T(T_w - T_\infty)}{Fr_\infty^2}\, T \operatorname{div}(x_2\tilde{I}) \tag{2.11}$$

Energy

$$\frac{\partial T}{\partial t} + \operatorname{div}(\underline{u}\, T) = \operatorname{div}(\frac{k}{Pe_\infty} \operatorname{grad} T) \tag{2.12}$$

where $\hat{p} = p + \frac{x_2}{Fr_\infty^2}$. It can be shown that the right hand side of Equation 2.11 can be rewritten as

$$\frac{\alpha_T(T_w - T_\infty)}{Fr_\infty^2} T \operatorname{div}(x_2\tilde{I}) = \frac{Gr_\infty}{Re_\infty^2}\, T \operatorname{div}(x_2\tilde{I})$$

Vector, Integral Form

The unsteady three-dimensional nondimensional incompressible Navier-Stokes equations in Cartesian coordinates and in conservative form can be written in integral form as

$$\frac{\partial}{\partial t}\int_\Omega Q\, dV + \int_{\partial\Omega} \vec{F}\cdot\vec{n}\, dA - \int_{\partial\Omega} \vec{F}_v\cdot\vec{n}\, dA - \frac{Gr_\infty}{Re_\infty^2}\int_\Omega \Upsilon\, dV = 0 \tag{2.13}$$

where,

$$Q = \begin{bmatrix} \hat{p} \\ u \\ v \\ w \\ T \end{bmatrix}, F = \vec{F} \cdot \vec{n} \begin{bmatrix} \beta\theta \\ u\theta + \frac{\hat{p}\, n_x}{\rho} \\ v\theta + \frac{\hat{p}\, n_y}{\rho} \\ w\theta + \frac{\hat{p}\, n_z}{\rho} \\ T\theta \end{bmatrix} = \begin{bmatrix} \beta u & \beta v & \beta w \\ u^2 + \frac{\hat{p}}{\rho} & uv & uw \\ uv & v^2 + \frac{\hat{p}}{\rho} & vw \\ uw & vw & w^2 + \frac{\hat{p}}{\rho} \\ Tu & Tv & Tw \end{bmatrix} \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix},$$

$$F_v = \vec{F_v} \cdot \vec{n} = \frac{1}{\rho} \begin{bmatrix} 0 \\ \tau_{xx} n_x + \tau_{xy} n_y + \tau_{xz} n_z \\ \tau_{xy} n_x + \tau_{yy} n_y + \tau_{yz} n_z \\ \tau_{xz} n_x + \tau_{yz} n_y + \tau_{zz} n_z \\ q_x n_x + q_y n_y + q_z n_z \end{bmatrix} = \frac{1}{\rho} \begin{bmatrix} 0 & 0 & 0 \\ \tau_{xx} & \tau_{xy} & \tau_{xz} \\ \tau_{xy} & \tau_{yy} & \tau_{yz} \\ \tau_{xz} & \tau_{yz} & \tau_{zz} \\ q_x & q_y & q_z \end{bmatrix} \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix},$$

$$\Upsilon = \begin{bmatrix} 0 \\ 0 \\ T \\ 0 \\ 0 \end{bmatrix}$$

and

$$\tau_{xx} = \frac{2\mu}{Re_\infty} \frac{\partial u}{\partial x},\ \tau_{yy} = \frac{2\mu}{Re_\infty} \frac{\partial v}{\partial y},\ \tau_{zz} = \frac{2\mu}{Re_\infty} \frac{\partial w}{\partial z},$$

$$\tau_{xy} = \tau_{yx} = \frac{\mu}{Re_\infty}(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}),\ \tau_{xz} = \tau_{zx} = \frac{\mu}{Re_\infty}(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}),\ \tau_{yz} = \tau_{zy} = \frac{\mu}{Re_\infty}(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}),$$

$$q_x = \frac{k}{Pe_\infty}\frac{\partial T}{\partial x}, q_y = \frac{k}{Pe_\infty}\frac{\partial T}{\partial y}, q_z = \frac{k}{Pe_\infty}\frac{\partial T}{\partial z},\ and$$

$$\theta = un_x + vn_y + wn_z$$

CHAPTER 3

NUMERICAL FORMULATION

These nondimensional equations are discretized in a finite volume formulation using an eplicit Euler method and median dual control volumes. Figure shows a cutaway depiction of an example of a median dual control volume surrounding a node.

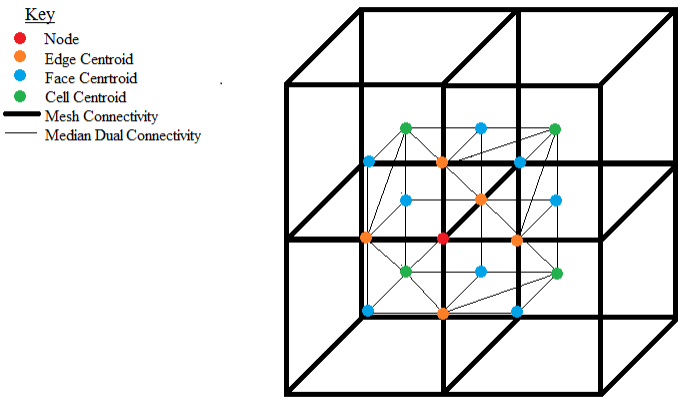

Figure 3.1 Diagram of a Median Dual Control Volume

Although the figure shows all hexahedral elements, the code is capable of handling multi-element grids including tetrahedrons, pyramids, and prisms as well. The method follows the one used in [7] and yields the following discretized equation for a general node i and its corresponding control volume:

$$V_i \frac{\Delta Q_i}{\Delta t_i} + \sum_j (\Phi_{ij} \cdot \hat{n}_{ij}) S_{ij} - \sum_j (\Phi_{v_{ij}} \cdot \hat{n}_{ij}) S_{ij} - \frac{Gr_\infty}{Re_\infty^2} V_i \Upsilon_i = 0$$

where V_i is the volume (assumed stationary and non-deforming), Δt_i is the timestep, and Υ_i is the source term vector, all associated with the particular node i. Φ_{ij} and $\Phi_{v_{ij}}$ are the discretized invsicid and viscous flux contributions respectively, $\hat{n}_{ij}$ is the outward pointing unit normal vector, and S_{ij} is the area vector all associated with the edge between node i and j, where the summation over j includes all the nodes surrounding i.

Inviscid Terms

As indicated in the second term on the left hand side of the discretized formula above, the integration of the inviscid flux is approximated by a summation. This approximation is done by a Riemann flux using the values of the dependent variables at each node on an edge. The midpoints of the edges function as the quadrature points. This method is the Roe flux scheme and has the following formula for the numerical inviscid flux [7]

$$\Phi_{RL} = \frac{1}{2}\left[(F(Q_L) + F(Q_R)) - \tilde{A}(Q_R, Q_L)(Q_R - Q_L)\right]$$

where Q_R and Q_L are the dependent variable vectors evaluated at the node on the right hand side of the edge and the left hand side of the edge, respectively. $F(Q_R)$ and $F(Q_L)$ are the inviscid flux vectors evaluated using the values from the dependent variable vectors. $\tilde{A}$ is a matrix constructed from the flux Jacobian and has the following formula

$$\tilde{A} = \tilde{R}\tilde{\Lambda}\tilde{L}$$

where $\tilde{R}$ and $\tilde{L}$ are the right and left eigenvector matrices, $\tilde{R}\tilde{L} = I$, and $\tilde{\Lambda}$ is a diagonal matrix with the absolute values of the eigenvalues as its diagonal elements. The tilde here denotes that the matrices are constructed using a dependent variable vector which is an averaged vector of the two dependent variable vectors from each node. For an incompressible

flow solver this vector is formed using the simple arithmetic average. The derivation of the eigensystem of the flux Jacobian matrix is contained in Appendix B.

Viscous Terms

There are several methods which can be used for determining the approximation of the viscous flux vector. The method employed here is the directional derivative method which is used in [7]. The viscous terms are linear, thus the numerical approximation has the following formula:

$$\Phi_{v_{ij}} = \frac{1}{Re_\infty} \sum_j C_j (Q_j - Q_i)$$

where C_j is a matrix of coefficients used to emulate the viscous nature of the fluid and j represents the index of a node connected to node i. This coefficient matrix is unique for each node and depends only on the geometry of the mesh. The directional derivative method is used to determine this matrix. The main goal is to use the numerical gradients at each node of the edge along with the edge's local information in order to construct the necessary edge gradient for the viscous calculation [14]. The general formula for the edge gradient is

$$\nabla Q_{ij} = \nabla Q_{ij,norm} + \nabla Q_{ij,tan}$$

where the normal component of the gradient is constructed by a directional derivative and the tangential component is constructed as an average of the gradients at each node. The formula for each of these components is as follows

$$(\nabla Q_{ij} \cdot \hat{s}\,)\hat{s} \approx \frac{Q_j - Q_i}{|\Delta \vec{s}|}\hat{s}$$

$$(\nabla Q_{ij} \cdot \hat{t}\,)\hat{t} \approx \overline{\nabla Q}_{ij} - (\overline{\nabla Q}_{ij} \cdot \hat{s})\hat{s}$$

where $\hat{s}$ is the unit vector in the direction of the edge, $\hat{t}$ is the unit vector perpendicular to the edge, $\overline{\nabla Q}_{ij} = \frac{1}{2}(\nabla Q_i + \nabla Q_j)$, $\Delta\vec{s} = \vec{x}_j - \vec{x}_i$, and $\vec{x}_j$ is the coordinate vector associated with node j. Using the above two approximations the final formula used to approximate the gradient at the edge is

$$\nabla Q_{ij} \approx \overline{\nabla Q}_{ij} + \left(Q_j - Q_i - \overline{\nabla Q}_{ij} \cdot \Delta\vec{s}\right) \frac{\Delta\vec{s}}{|\Delta\vec{s}|^2}$$

A weighted least squares method is commonly used to calculate the numerical approximation of the gradient at each node. This is the method chosen for this work, and the corresponding weight used is the inverse of the distance between the two nodes. Further details on the least squares method are given in Appendix C.

Higher Order Accuracy

In a typical upwind flux formulation the flux terms depend upon the solution variables on the left and right hand sides of the given control volume face. For a first order spatially accurate scheme, the usual formula is to take $Q_L = Q_i$ and $Q_R = Q_j$, where the L and R subscripts refer to the dependent variables on the left and right hand sides of the control volume face respectively and the i and j subscripts refer to the nodes on either end of an edge. In order to achieve higher accuracy for the inviscid terms, the dependent variables can be extrapolated out to the face of the control volume instead of simply using the values at the given nodes. This type of extrapolation is done to create a scheme that is second order in space. These higher order formulas for the dependent variables on the left and right hand side of the control face are:

$$Q_L = Q_i + \nabla Q_i \cdot (\vec{x}_f - \vec{x}_i)$$

$$Q_R = Q_j + \nabla Q_j \cdot (\vec{x}_f - \vec{x}_j)$$

where $\vec{x}_f$ is the coordinate vector of the midpoint of the edge, and $\vec{x}_i$ and $\vec{x}_j$ are the coordinate vectors of each node. As seen in the above formulas, in order to extrapolate the solution, the nodal gradients are needed. Although Green's Theorem is a reasonable method of approximating these gradients, it is shown in [16] that an unweighted least squares method gives a better approximation when using the gradients to extrapolate the solution to the control volume face. The least squares method is shown in further detail in Appendix C.

CHAPTER 4

BOUNDARY CONDITIONS

In most cases of solving for the dependent variables at the boundary node, it is necessary to create an imaginary node outside of the boundary, commonly termed a ghost node. This node is given certain requirements in order to satisfy the necessary boundary conditions. It is not actually a part of the mesh, but is created for the sole purpose of providing information to the node on the boundary. Figure 4.1 shows a depiction of nodes on a two-dimensional boundary with ghost nodes.

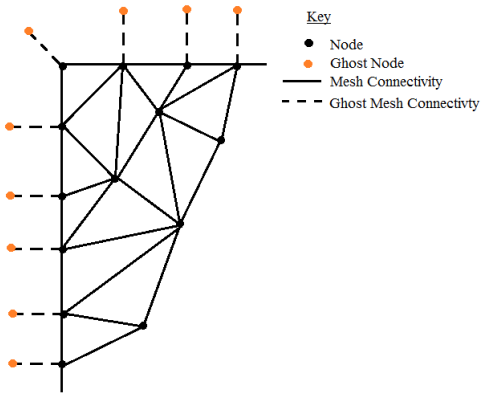

Figure 4.1 Depiction of Ghost Nodes

Characteristic variable boundary conditions are used primarily for farfield boundaries and also for inviscid flow impermeable surfaces. These boundary conditions are formed by beginning with the quasi-linear form of the system and writing the equations in the direction normal to boundary, η:

$$\frac{\partial Q}{\partial t} + A\frac{\partial Q}{\partial \eta} = 0$$

where A is defined above as $A = \frac{\partial F}{\partial Q}$. A can be written as $A = R\Lambda L$ where Λ is the matrix of eigenvalues and R and L are the right and left eigenvector matrices of A respectively. The quasi-linear system then becomes

$$\frac{\partial Q}{\partial t} + R\Lambda L\frac{\partial Q}{\partial \eta} = 0$$

multiplying by L and recalling that $L = R^{-1}$ gives

$$R^{-1}\frac{\partial Q}{\partial t} + \Lambda R^{-1}\frac{\partial Q}{\partial \eta} = 0$$

Evaluating R^{-1} at some reference state (i.e., R_0^{-1} is a constant matrix), it can be placed in the derivative giving

$$\frac{\partial R_0^{-1}Q}{\partial t} + \Lambda\frac{\partial R_0^{-1}Q}{\partial \eta} = 0$$

where the 0 subscript indicates evaluation as a constant. Then, the characteristic variable vector W is defined as $W = R_0^{-1}Q$, and W can be evaluated depending on the sign of the eigenvalues on the boundary to determine the values of Q_b, which are then used in the evaluation of the numerical flux through the boundary.

Inflow

As shown in Appendix B, the eigenvalues for the system of equations are $\lambda_1 = \lambda_2 = \lambda_3 = \theta$, $\lambda_4 = \theta + c$, and $\lambda_5 = \theta - c$. Inflow is characterized by the first three eigenvalues being negative, the fourth being positive, and the fifth being negative. The dependent variables

for the ghost node at such a boundary are then given by

$$\begin{bmatrix} l_0^1 \\ l_0^2 \\ l_0^3 \\ l_0^4 \\ l_0^5 \end{bmatrix} \begin{bmatrix} P \\ u \\ v \\ w \\ T \end{bmatrix}_b = \begin{bmatrix} l_0^1 \cdot Q_a \\ l_0^2 \cdot Q_a \\ l_0^3 \cdot Q_a \\ l_0^4 \cdot Q_l \\ l_0^5 \cdot Q_a \end{bmatrix}$$

where $l_0^1, l_0^2, l_0^3, l_0^4$ and l_0^5 are the left eigenvectors evaluated as constant (Note these are the rows of matrix R_0^{-1}). Vectors Q_a and Q_l are the dependent variable vectors, a subscript of a denotes evaluation from a freestream location and a subscript of l denotes evaluation from directly inside the boundary. The b subscript denotes that these are to be the ghost node values.

Outflow

Outflow is characterized by the first four eigenvalues being positive and the fifth being negative. The dependent variables for the ghost node at this boundary are then given by

$$\begin{bmatrix} l_0^1 \\ l_0^2 \\ l_0^3 \\ l_0^4 \\ l_0^5 \end{bmatrix} \begin{bmatrix} P \\ u \\ v \\ w \\ T \end{bmatrix}_b = \begin{bmatrix} l_0^1 \cdot Q_l \\ l_0^2 \cdot Q_l \\ l_0^3 \cdot Q_l \\ l_0^4 \cdot Q_l \\ l_0^5 \cdot Q_a \end{bmatrix}$$

Impermeable Surface with Inviscid Flow

For an impermeable surface in inviscid flow, the temperature and the pressure are taken to be the same as directly inside the boundary, while the velocity vector is forced to be tangent to the surface. To fulfill this requirement the following formula is used to compute the velocity for the ghost node:

$$u_b = u_l - \theta n_x$$

$$v_b = v_l - \theta n_y$$

$$w_b = w_l - \theta n_z$$

where again the b subscript denotes the ghost node value and the l subscript denotes the value is taken from directly inside the domain. Recall that θ is defined as $\theta = un_x + vn_y + wn_z$.

Symmetry Plane Boundary Conditions

In this work the chosen method of dealing with symmetry planes is to utilize a mirrored boundary condition. In this method phantom nodes are created outside of the symmetry surface which are a mirror image of the nodes directly interior to the surface. The mirrored plane goes one element deep. When this is done, the solver can essentially ignore any boundary conditions that would be associated with the symmetry surface and treat these nodes in the same manner which the interior nodes are treated. Since the interior nodes are mirrored to the outside of the domain, the surface nodes have closed control volumes and are essentially interior nodes. It is critical when implementing this method to ensure that scalars are copied and vectors and tensors mirrored correctly to these fictitious mirrored nodes in order to enforce the requirement that no fluxes penetrate the symmetry plane.

Impermeable Surface with Viscous Flow

The impermeable surfaces in a viscous flow, in this work, are given either Dirichlet boundary conditions, Neumann boundary conditions, or a combination of both. A Dirichlet boundary condition is enforced by the value of the given variable being set a priori and then being kept constant throughout the simulation. This condition is used mainly for nondimensional temperature at the boundary, for cases where a wall will be constantly heated or cooled; for these cases, $T = 0$ or $T = 1$, respectively, throughout the entire simulation. In addition, the velocity is most often given a no-slip condition, which is a subset of the Dirichlet condition. A no-slip condition is fulfilled by forcing the fluid to have a zero velocity at the surface (for a fixed surface) at all times. For example, the following variables could be set and then kept constant

$$\begin{bmatrix} T_l \\ u_l \\ v_l \\ w_l \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

It should be noted that these values are set for the variables of the nodes directly on the boundary. The ghost nodes in this condition simply receive a direct copy of the variables from the boundary.

$$\begin{bmatrix} T_b \\ u_b \\ v_b \\ w_b \end{bmatrix} = \begin{bmatrix} T_l \\ u_l \\ v_l \\ w_l \end{bmatrix}$$

where b subscript again denotes the ghost node variables and the l subscript denotes the boundary node variables.

The second boundary condition used for impermeable surfaces in viscous flow is the Neumann boundary condition, which specifies the value of the gradient for the variable at the surface. The two Neumann boundary conditions used here are one enforcing a zero gradient off the wall, used for both temperature and pressure, and another requiring the pressure gradient off the wall to be equal to the source term when buoyancy is considered; i.e.,

Zero gradient

$$\frac{\partial T}{\partial \eta} = 0\,,\ \frac{\partial P}{\partial \eta} = 0$$

Source term gradient

$$\frac{\partial P}{\partial \eta} = \frac{Gr}{Re^2} T$$

These equations are discretized as

Zero gradient

$$\frac{T_l - T_i}{\eta_l - \eta_i} = 0 \ \Rightarrow\ T_l = T_i$$

$$\frac{P_l - P_i}{\eta_l - \eta_i} = 0 \ \Rightarrow\ P_l = P_i$$

Source term gradient

$$\frac{P_l - P_i}{\eta_l - \eta_i} = \frac{Gr}{Re^2} T_i \ \Rightarrow\ P_l = P_i + \frac{Gr}{Re^2} T_i (\eta_l - \eta_i)$$

where the l subscript again denotes the variables of the node on the boundary and an i subscript now denotes the variables of the node directly interior of the boundary. As above, the values of the ghost node variables in this case are set to be equal to those of the node directly on the boundary. In the case that a node has multiple edges in the direction normal to the boundary, a tolerance is used to select the node with coordinates closest to that of the boundary node, but extruded off the boundary.

CHAPTER 5

RESULTS

Inviscid Flow Over a Cylinder

This case was accomplished to demonstrate that the code can perform a simulation of incompressible, inviscid flow. Therefore, successfully executing this case will show that the mass and momentum equations are being solved correctly without having to solve the temperature equation. For this case an unstructured grid is used with a spacing of 1.0e-3 (distances are expressed in units of cylinder radii) coming off the cylinder wall along the x and y axes. The far field boundary is located at a distance of 10 times the radius of the cylinder. In order to simulate a two-dimensional case, three planes are created in the z-direction, with the outer two having mirror symmetry conditions. For this case, all variables are initialized to 0.0, except for horizontal velocity which is initialized to 1.0. Also, a CFL of 0.6 is used with local time stepping. Characteristic variable boundary conditions are used at the farfield boundary and a no pressure gradient with tangential velocity is enforced on the cylinder. Figure 5.1 shows a diagram of the set up for this case for clarity. As shown in figures 5.2 - 5.5, this test case shows excellent agreement between the present calculations and the results of Taylor [1]. Figures 5.2, 5.3, 5.4, and 5.5 are plots of the pressure, x-direction velocity, y-direction velocity, and surface C_p respectively, each of which show the expected symmetry. The pressure increase on the downstream side of the cylinder is a key feature of this simulation and only develops when higher order accuracy of inviscid flux terms is implemented. In addition, three variations of this case were run with the artificial compressibility parameter, β, set to 5, 10, and 15. Figure 5.5 shows the computed surface

C_P from each β relative to the theoretical answer, and figure 5.6 shows the L2 norm of the residual for each β as well. It is interesting to note that lower values of β tend to converge more quickly and achieve results closer to the theoretical answer, while the higher β values aid in the stability of the algorithm.

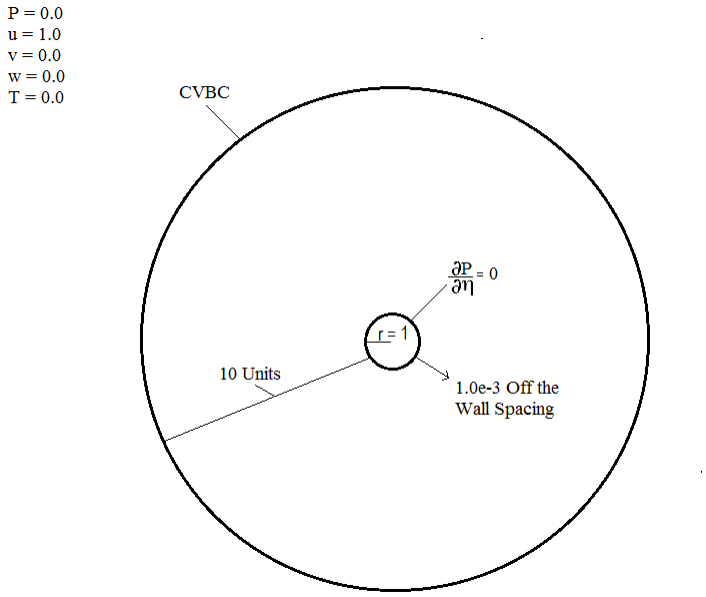

Figure 5.1 Diagram of Grid, Boundary, and Initialization Information

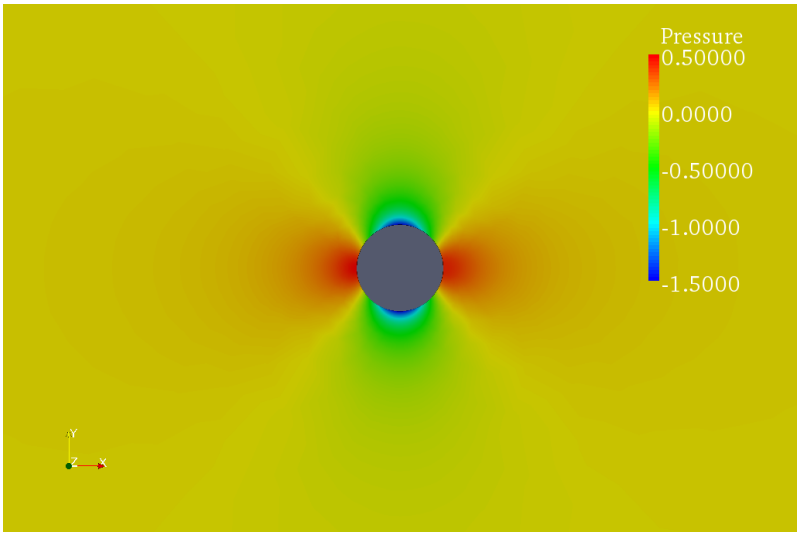

Figure 5.2 Contour Plot of Steady State Pressure

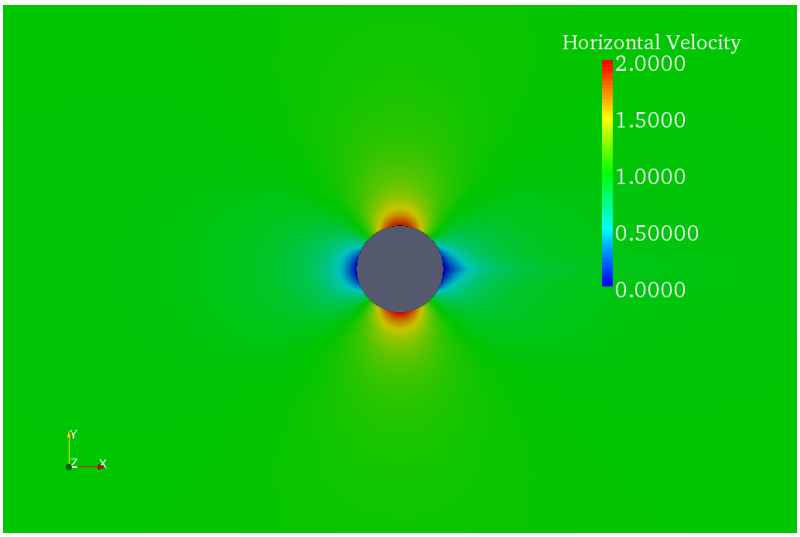

Figure 5.3 Contour Plot of Steady State Horizontal Velocity

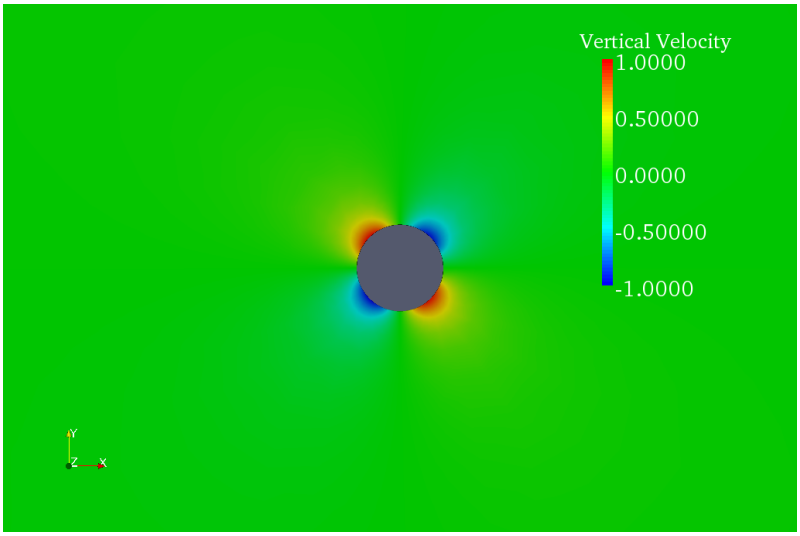

Figure 5.4 Contour Plot of Steady State Vertical Velocity

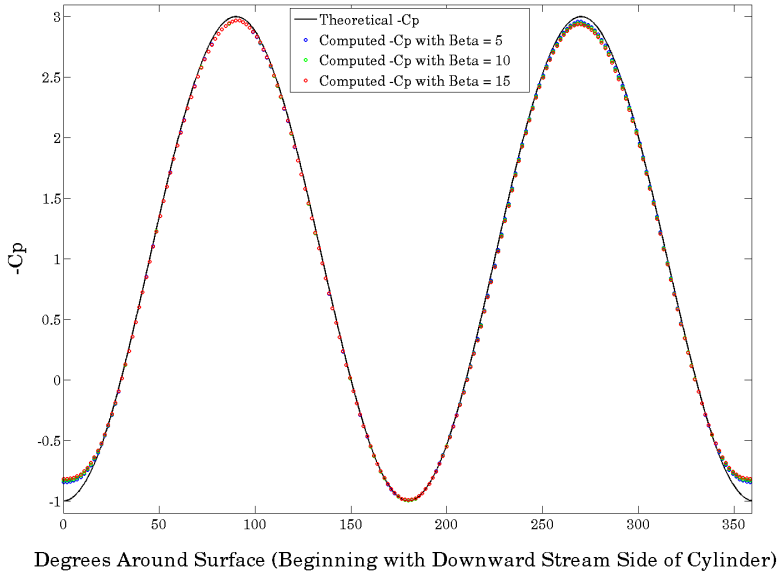

Figure 5.5 Plot of Computed and Theoretical Surface C_P versus Angle around Cylinder

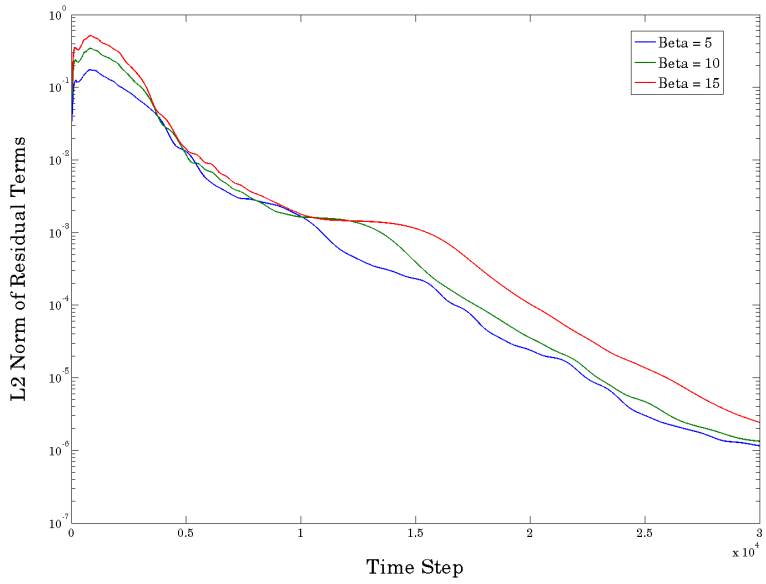

Figure 5.6 Plot of L2 Norm of Residual Terms for $\beta = 5, 10, 15$

Forced Flow Over a Heated Plate

The second case considered is that of laminar forced flow over a heated flat plate. The two main distinctions between this case and the previous is the addition of the viscous terms in the equations (meaning the flow considered is now viscous) and of the temperature equation. Essentially, a successful execution of this case illustrates the capability of the code to perform a simulation of forced convection with viscous flow. A Reynolds number of 10,000 and a Prandtl number of 1.0 are used for this case, along with $\beta = 10.0$, a CFL of 0.6 and local time stepping. The grid created has a flat plate of nondimensional length unity and a farfield boundary a distance of 5 nondimensional units away. The viscous spacing off the plate wall is 1.0e-3. The plate is given a neglible amount of height in order to create a grid that has flow on both sides of the plate. As in the previous case, three grid planes are created in the z-plane with the outer two having mirror symmetry conditions. The flat plate is given no-slip

boundary conditions, and the temperature is kept at 1.0. All other variables are initialized to 0.0 except horizontal velocity, which is initialized to 1.0 everywhere. Figure 5.7 shows a diagram of the grid, boundary, and initialization parameters for clarity. Figures 5.8 and 5.9 show contour plots of the horizontal velocity and the temperature respectively, which show the expected build-up of velocity and thermal boundary layers near the heated plate. The boundary layer velocity profile is shown in figure 5.10 and is compared to the theoretical results from the Blasius equation [9], showing excellent agreement. This plot compares the current results of horizontal velocity versus η, which is a simple linear mapping of the y coordinate values taken at nondimensional $x = 0.2$ distance along the plate, with the known theoretical results of horizontal velocity versus η. The quantity $1.0 - T$ along this line is also plotted, which shows good agreement with the theoretical results as well. Figure 5.11 is a plot of the computed Nusselt number along the plate versus the theoretical Nusselt number along a heated plate; again, there is good agreement. The discrepancy at the end of the plate between the computed and theoretical Nusselt number is due to the fact that the theory is established for an infinite plate and the plate in the simulation is finite. Figure 5.12 shows the convergence rate for this simulation using the L2 norm of the residual terms plotted against time step.

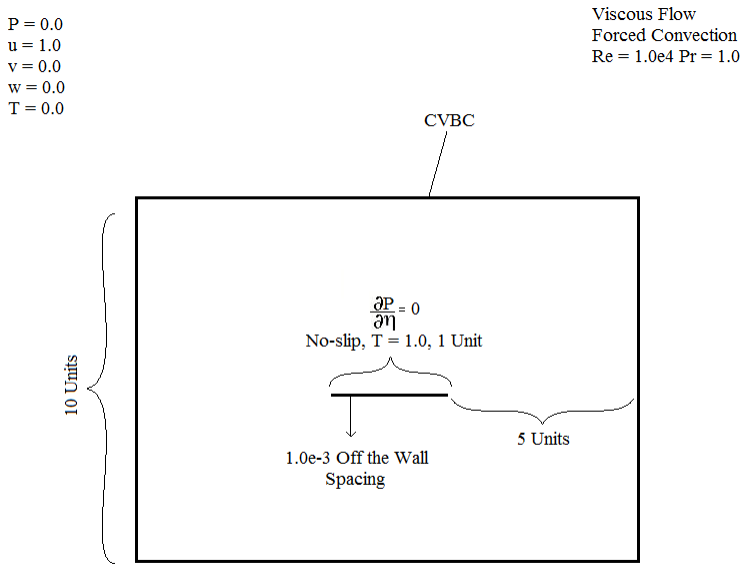

Figure 5.7 Diagram of Grid, Boundary, and Initialization Information

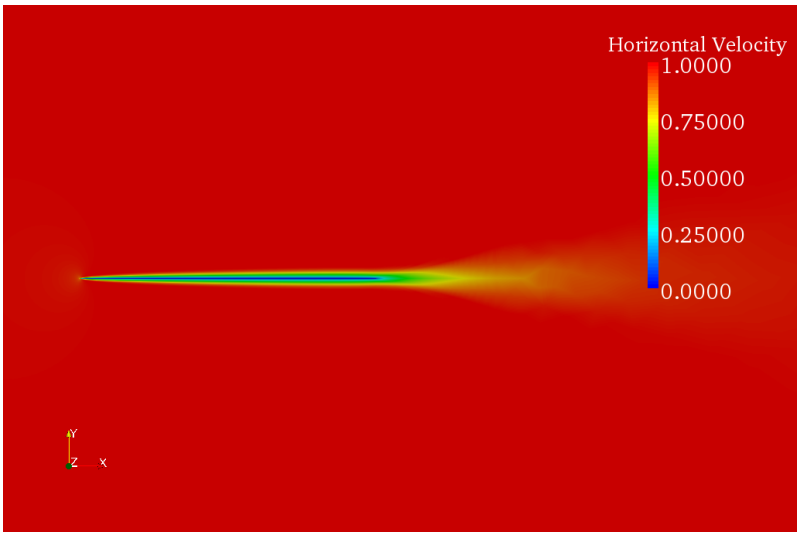

Figure 5.8 Contour Plot of Steady State Horizontal Velocity

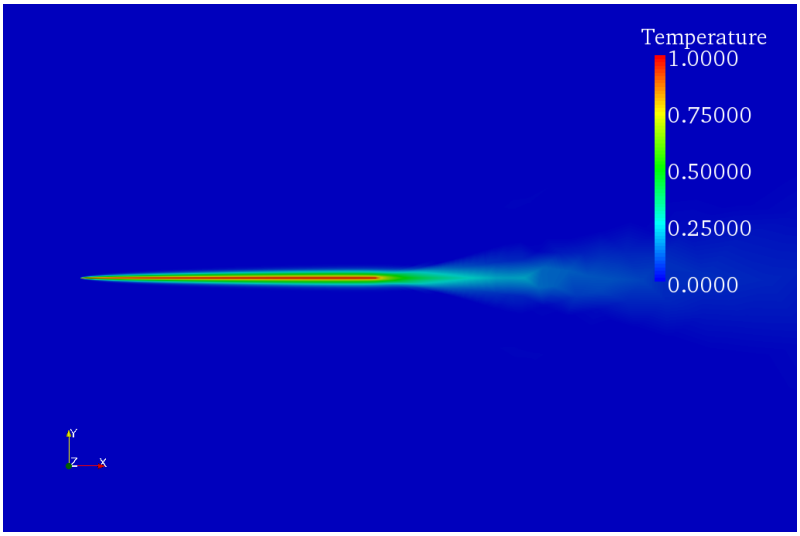

Figure 5.9 Contour Plot of Steady State Temperature

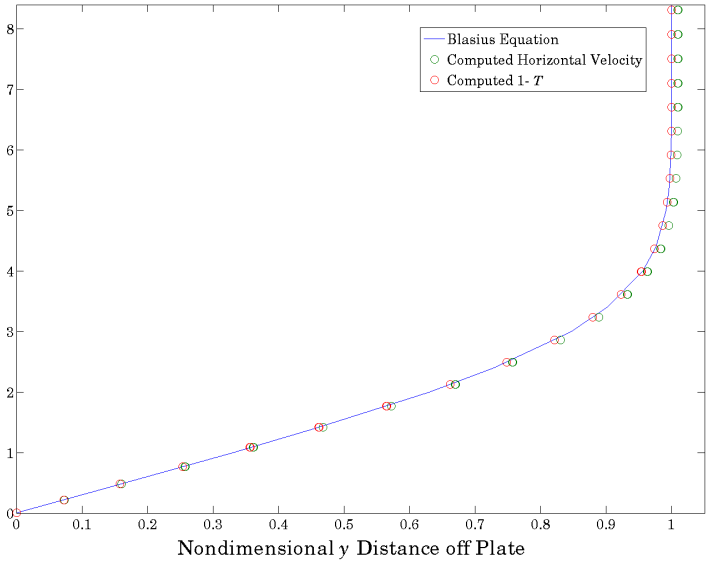

Figure 5.10 Plot of Computed Horizontal Velocity, 1-Temperature, and Blasius Solution

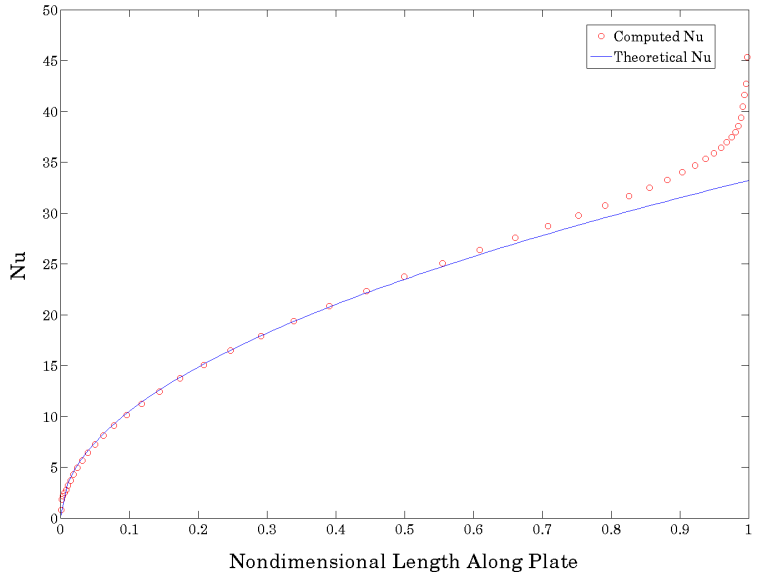

Figure 5.11 Plot of Theoretical and Computed Nusselt Number versus Non-Dimensionalized X Distance along Plate

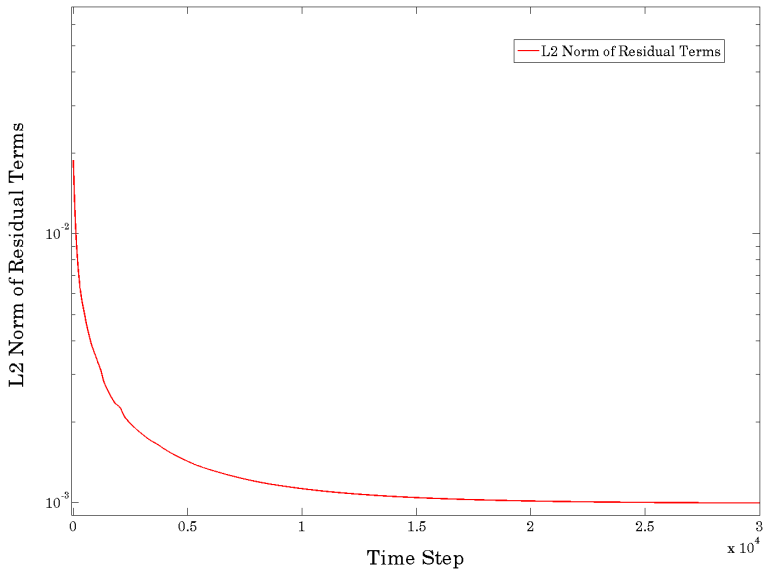

Figure 5.12 Plot of L2 Norm of Residual Terms Versus Time Step

Natural Convection in a Differentially Heated Square Cavity

The purpose of this test case is to validate the treatment of buoyancy in the code. A simple square grid of nondimensional unit length that is 81x81x3 is used, with an off-wall spacing of 1.0e-3. A Raleigh number of $1.0e4$ and a Prandtl number of 0.71 are used. Stability was an issue in this case, so all nodes were given the same time step equal to that of the minimum time step over all the nodes. Every wall in this case has no slip boundary conditions, so velocity is initialized to zero at each point in the grid. The artificial compressibility parameter used is 5. The horizontal walls have an imposed zero normal temperature gradient boundary condition, while the vertical walls have an imposed zero pressure gradient boundary condition. The horizontal walls, however, have the normal pressure gradient equal to the source term in order to properly deal with buoyancy effects. The leftmost vertical wall is kept at a temperature of 1.0, while the rightmost wall is always kept at temperature of 0.0 (both temperatures being nondimensional). The temperature is initialized to 1.0 elsewhere in the grid. Figure 5.13 shows a diagram of the initialization, grid, and boundary information for clarity. Figure 5.14 shows a plot of the steady state temperature, which agrees very well with the results of Er [6] and shows the expected stratification of the fluid when thermal buoyancy is considered. Figure 5.15 shows a plot of the vertical velocity along the y-axis center line compared to the results of Er [6]. Figure 5.16, then has the current results of horizontal velocity along the x-axis center line compared to the results of Er [6] as well. Table 5.1 then compares the values of the maximum components of velocity along the axes for the current results, those of Er [6], and those for the benchmark solution done by Davis [11]. The table illustrates excellent agreement of the current solution to the benchmark solutions. Figure 5.17 shows the convergence plot for this case, with the L2 norm of the residual terms plotted versus the time steps.

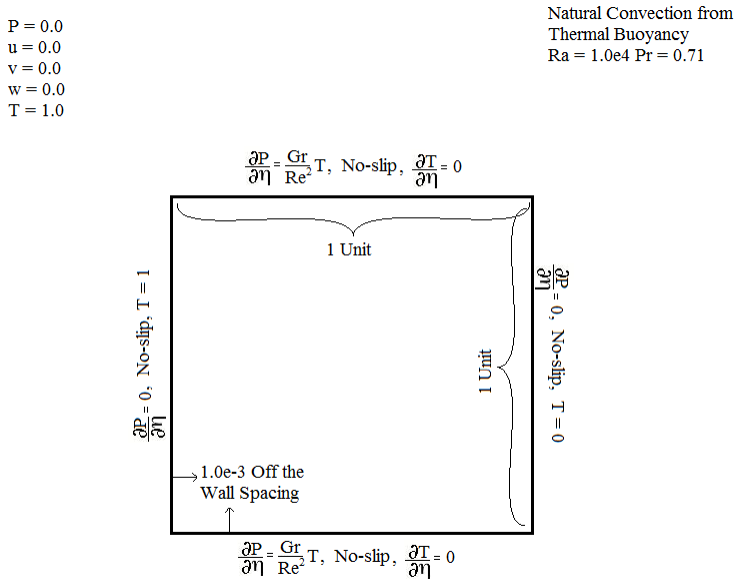

Figure 5.13 Diagram of Grid, Boundary, and Initialization Information

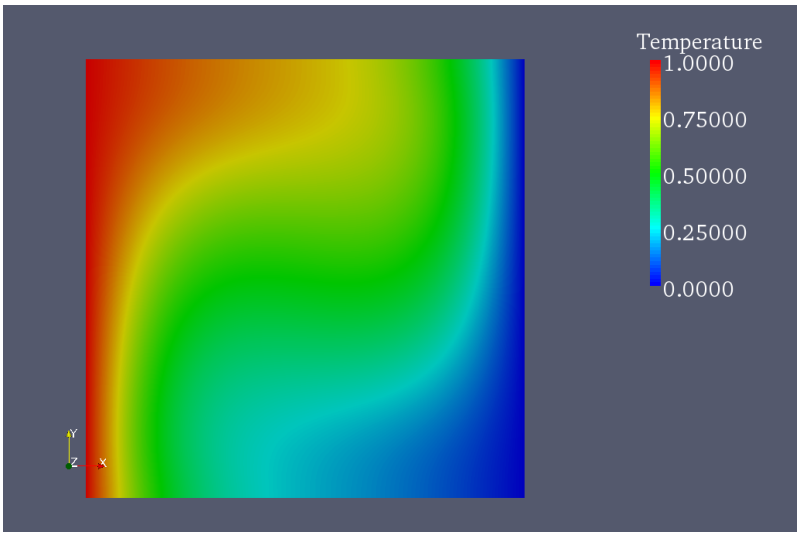

Figure 5.14 Contour Plot of Steady State Temperature

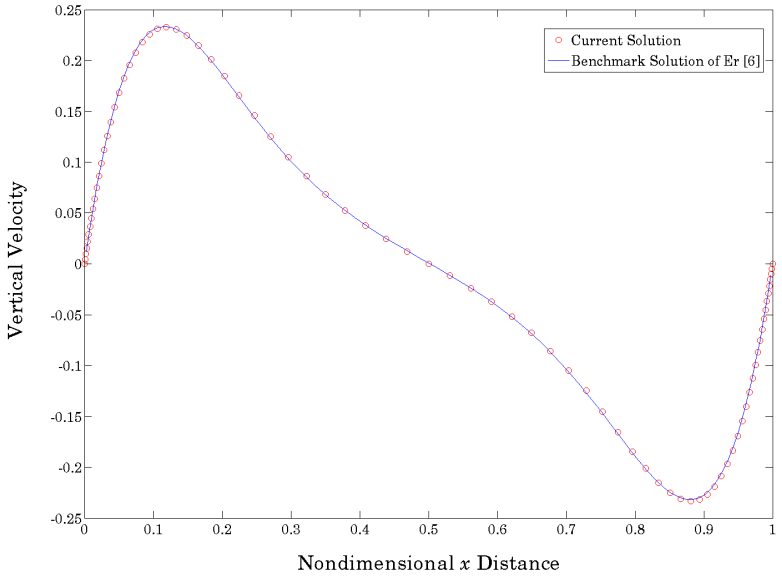

Figure 5.15 Plot of Vertical Velocity along y-Axis Center Line Compared to Results of Er [6]

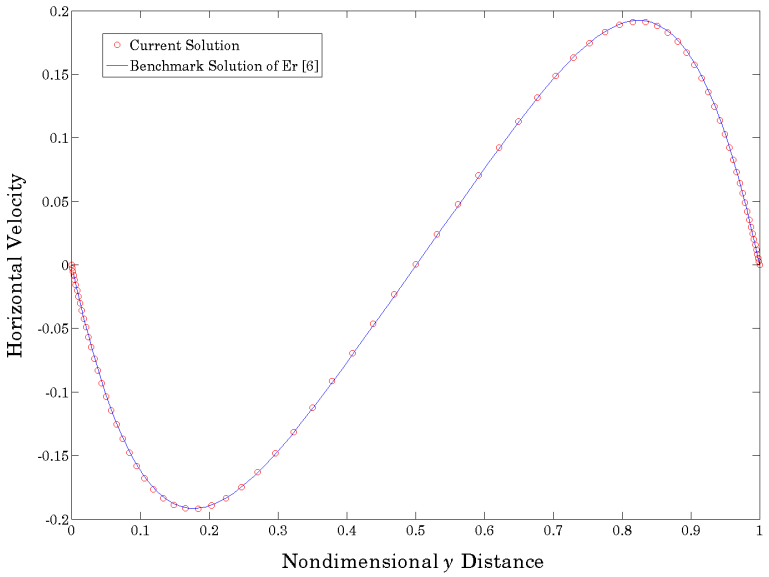

Figure 5.16 Plot of Horizontal Velocity along x-Axis Center Line Compared to Results of Er [6]

Table 5.1 Comparison between Benchmark Solutions and Current Velocity

Variable	Davis [11]	Er [6]	Current
Maximum Horizontal Velocity on Vertical Centerline	0.192	0.192	0.191
Y Location of Maximum Horiztonal Velocity	0.823	0.850	0.816
Maximum Vertical Velocity on Horizontal Centerline	0.233	0.232	0.232
X Location of Maximum Vertical Velocity	0.119	0.125	0.119

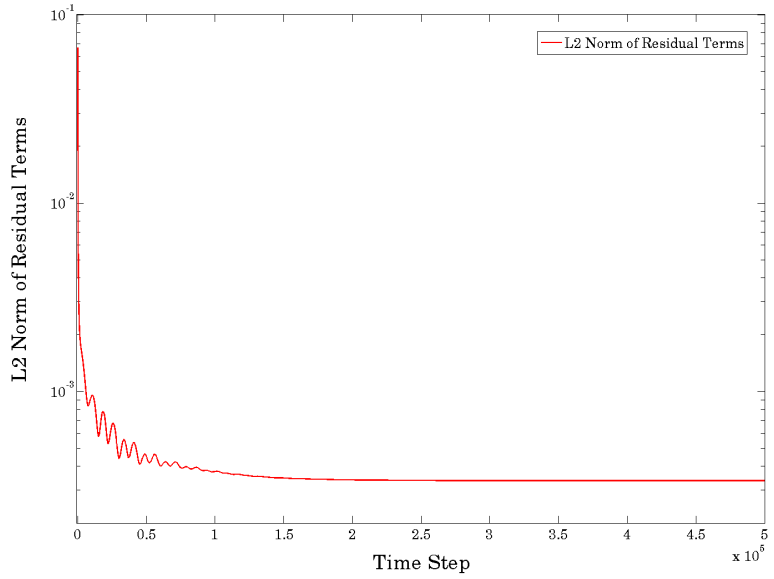

Figure 5.17 Plot of L2 Norm of Residual Terms Versus Time Step

Mixed Convection in a Square Cavity

The objective of this test case is to verify the code's ability to perform a simulation that contains both forced convection and natural convection. A canonical case for testing this is that of a lid driven cavity. The grid from the previous test case was used. The Reynolds number is now $1.0e3$, the Grashof number is now $1.0e2$, and the Richardson number is

$1.0e-4$. A CFL of 0.6 is used with local time stepping, and β is set to 5. First, the code is executed with the temperature set equal to 0.0 everywhere as a check. All walls are set with no slip conditions, except the top horizontal wall which is moving with a horizontal velocity of 1.0. The vertical walls are also adiabatic, i.e., $\frac{\partial T}{\partial \eta} = 0$. Again a diagram is illustrates the initialization, grid, and boundary information in figure 5.18. After running this case, it is seen that the results match well with previous results of the benchmark case done by Ghia [10] (used as the benchmark for both Koblitz et al. [2] and Agrawal et al. [4]). Figures 5.19 and 5.20 show the current results of the vertical velocity along the y-axis centerline and the horizontal velocity along the x-axis centerline compared to the results of Ghia [10]. Figure 5.21 shows an image of the steady state velocity vectors shaded by velocity magnitude, which show the expected vortex throughout the majority of the cavity, as seen in [2] as well. Figure 5.22 shows the convergence plot for this case, where the L2 norm of the residual term is plotted against the time steps.

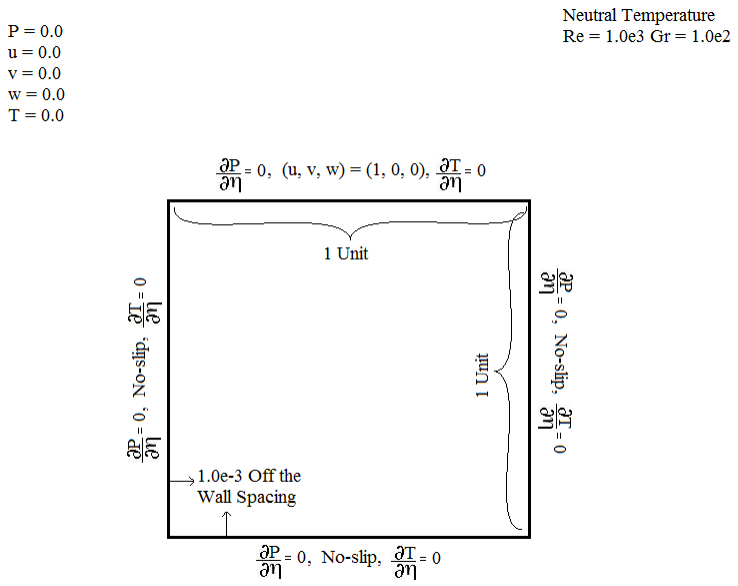

Figure 5.18 Diagram of Grid, Boundary, and Initialization Information

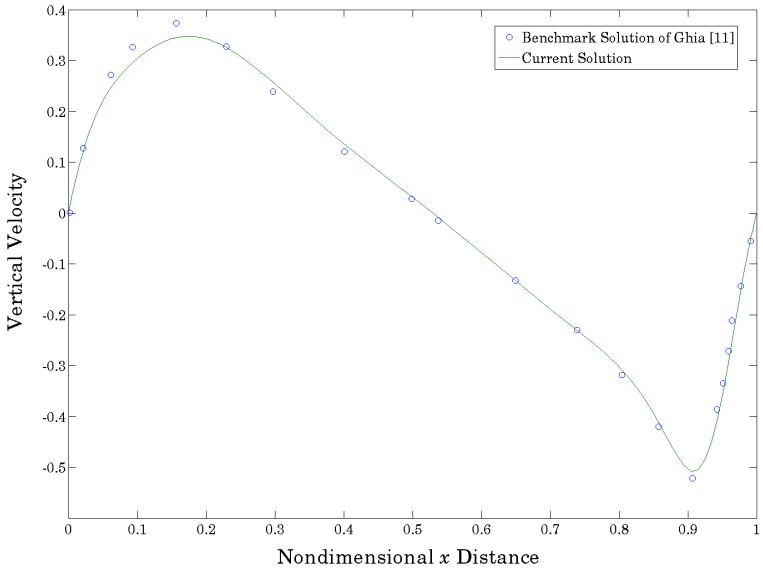

Figure 5.19 Plot of Vertical Velocity along y-Axis Center Line Compared to Results of Ghia [10]

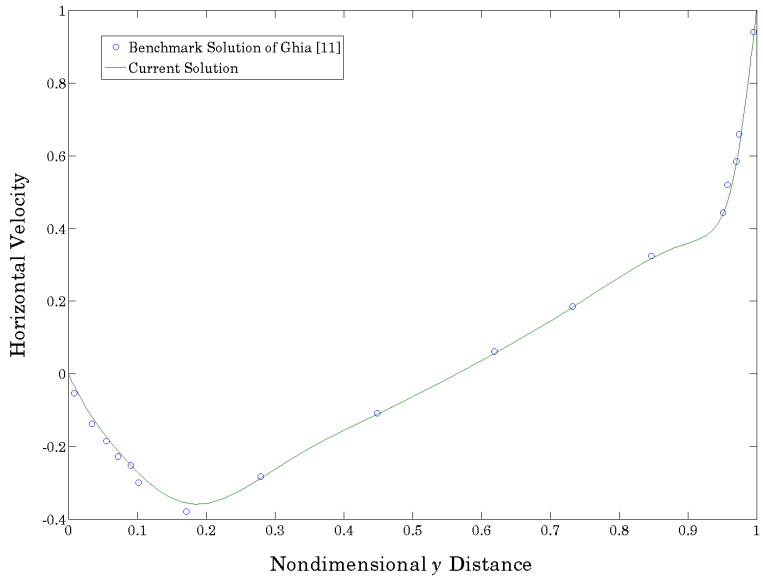

Figure 5.20 Plot of Horizontal Velocity along x-Axis Center Line Compared to Results of Ghia [10]

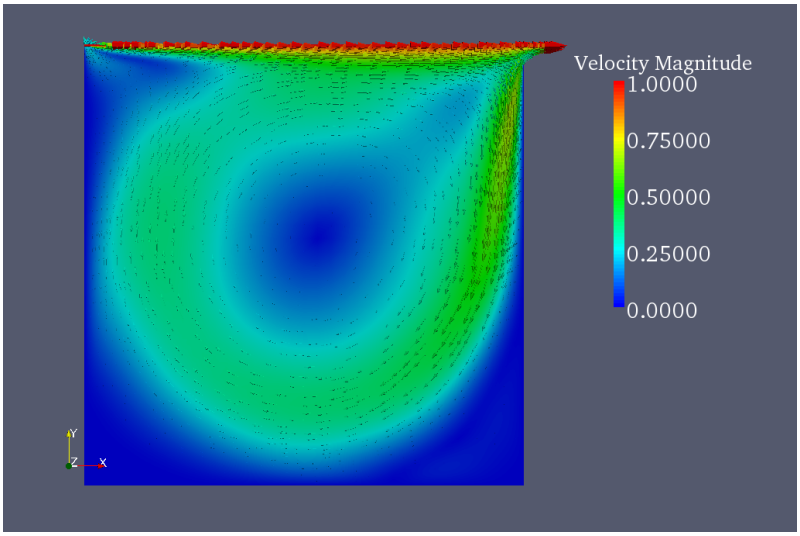

Figure 5.21 Plot of Steady State Velocity Vectors

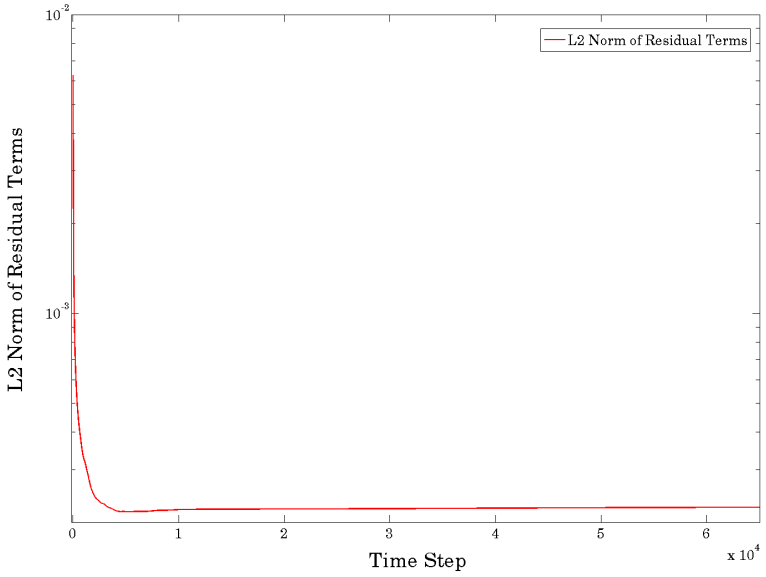

Figure 5.22 Plot of L2 Norm of Residual Terms Versus Time Step

After this initial check was verified, a case is executed to investigate both natural and forced convection. In order to perform this simulation, the same initialization is used as in

the previous case, except the moving lid now is kept at a constant temperature of 1.0. Also, the two horizontal walls include the source term restriction on the gradient of pressure. The Reynolds number is again $1.0e3$, but the Grashof number is now $1.0e6$, and the Richardson number now 1.0. A CFL of 0.6 is used with local time stepping, and β is set to 5. Figure 5.23 shows a diagram of the initialization, grid, and boundary information for clarity. For this case a grid refinement study was also done with three structured grids of dimensions 61x61x3, 81x81x3, and 101x101x3. Figures 5.24 and 5.25 show the current results for each grid of the vertical velocity along the y-axis centerline and the horiztonal velocity along the x-axis centerline compared to the benchmark results of Iwatsu et al [12], (the benchmark data used in both [2] and [4]). These figures show a slight improvement relative to the benchmark solutions due to grid refinement. It should be noted that no experimental data exists for this case, and the slight discrepancy in the current computed velocities and the benchmark velocities of Iwatsu et al, [12] are similar to the discrepancies in the solutions obtained by Koblitz et al. [2]. Figures 5.26 and 5.27 show images of the steady state temperature and velocity vectors for the case with grid size 81x81x3, both of which show good qualitative agreement with the results in [2] and [4]. The diffusion of the temperature from the forced convection of the heated lid is restricted to the upper portion of the cavity due to buoyancy effects, which also affect the thermal stratification in the lower portion of the cavity. Figure 5.28 shows the convergence based on the L2 norm of the residual term for each grid case. Slower convergence occurs as grid size increases as expected.

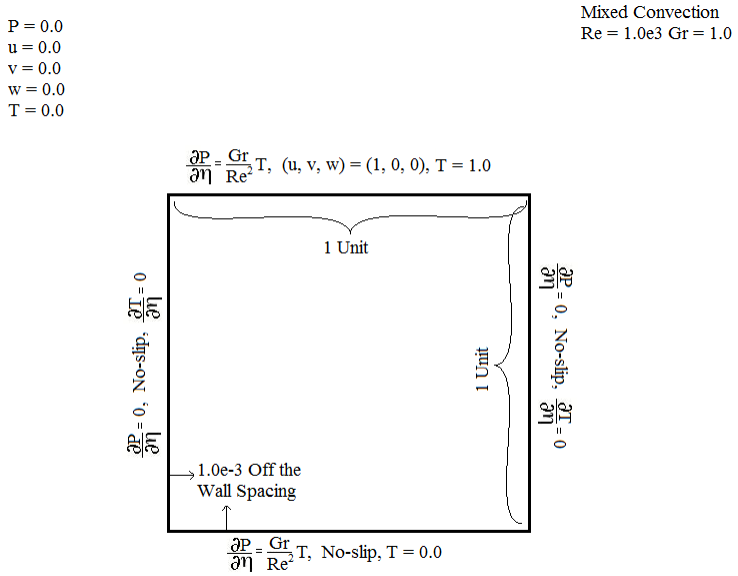

Figure 5.23 Diagram of Grid, Boundary, and Initialization Information

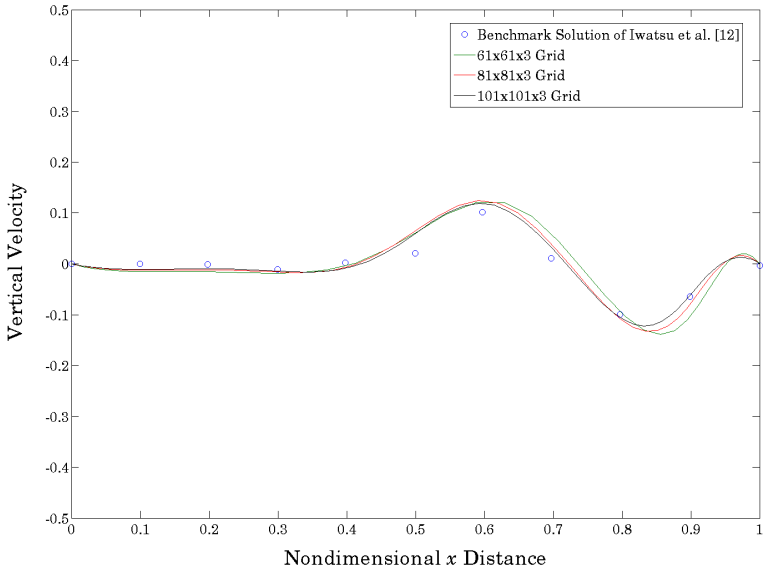

Figure 5.24 Plot of Vertical Velocity along y-Axis Center Line Compared to Results of Iwatsu et al. [12]

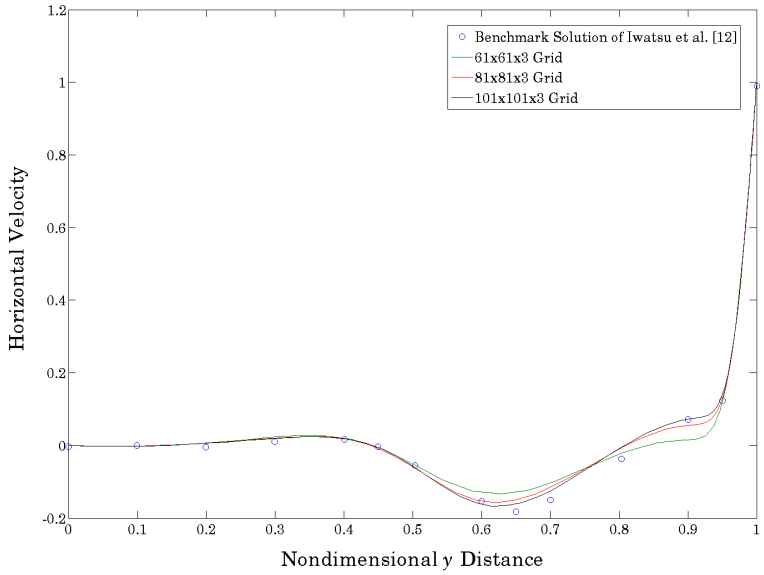

Figure 5.25 Plot of Horizontal Velocity along x-Axis Center Line Compared to Results of Iwatsu et al. [12]

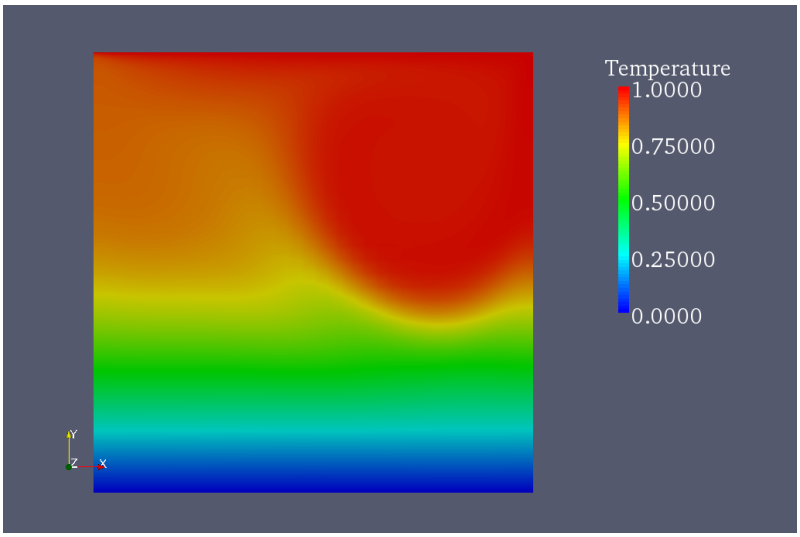

Figure 5.26 Plot of Steady State Temperature

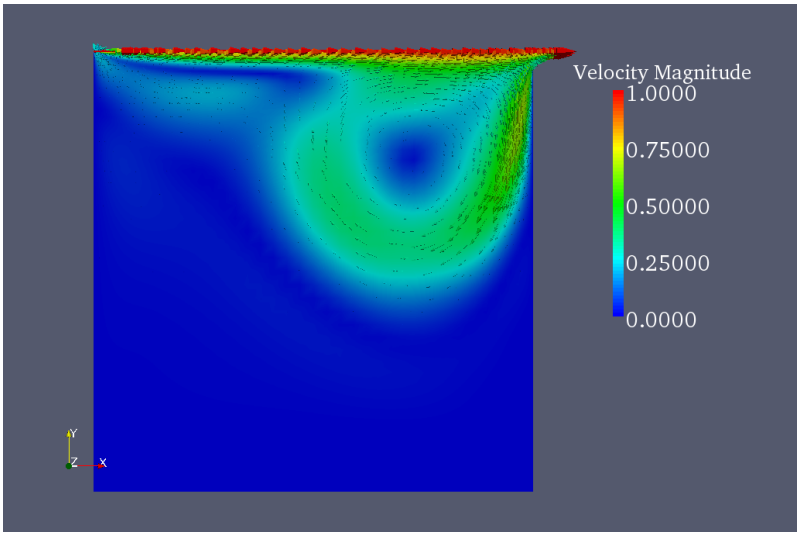

Figure 5.27 Plot of Steady State Velocity Vectors

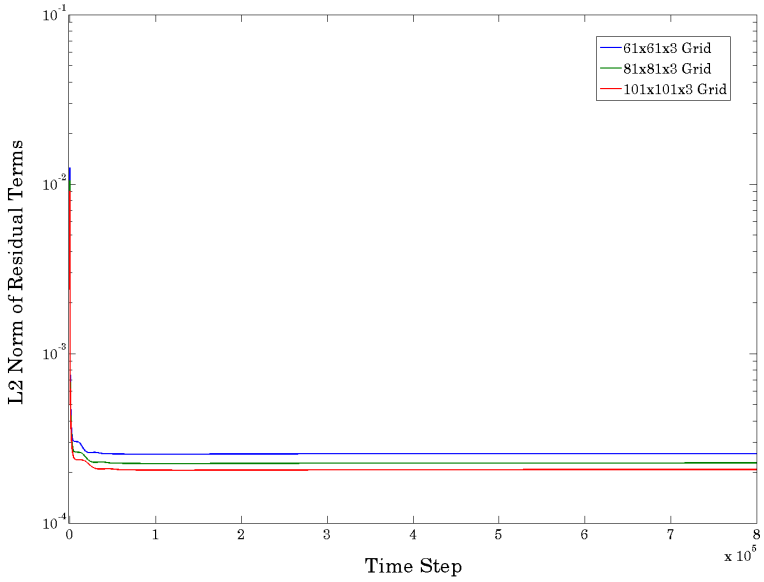

Figure 5.28 Plot of L2 Norm of Residual Terms for Varying Grids Versus Time Step

CHAPTER 6

CONCLUSIONS

The goal of this work was to implement a three-dimensional, explicit, incompressible flow solver that is also capable of dealing with convection and handling multi-element unstructured grids. Based upon comparing the results to previous benchmark solutions and exact theoretical data this goal was successfully met. The constructed numerical formulation and corresponding code can handle a variety of convective flow problems giving accurate results. The limitations of this code are clearly its speed and subsequent inability to handle large amounts of data in a timely fashion. The next step to be taken is to implement this formulation in the existing in house code, Tenasi, which is both implicit, parallel, and includes turbulence models. Hopefully after such an implementation, problems of a significant magnitude dealing with thermal buoyancy can be solved.

APPENDIX A

GOVERNING EQUATIONS

In this appendix the governing equations are written in expanded form for clarity.

Continuity

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0$$

x-direction Momentum

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} + \frac{\partial uv}{\partial y} + \frac{\partial uw}{\partial z} = \frac{-1}{\rho}\frac{\partial p}{\partial x} - \frac{1}{\rho}\frac{\partial \tau_{xx}}{\partial x} - \frac{1}{\rho}\frac{\partial \tau_{xy}}{\partial y} - \frac{1}{\rho}\frac{\partial \tau_{xz}}{\partial z}$$

y-direction Momentum

$$\frac{\partial v}{\partial t} + \frac{\partial uv}{\partial x} + \frac{\partial v^2}{\partial y} + \frac{\partial vw}{\partial z} = \frac{-1}{\rho}\frac{\partial p}{\partial y} - \frac{1}{\rho}\frac{\partial \tau_{xy}}{\partial x} - \frac{1}{\rho}\frac{\partial \tau_{yy}}{\partial y} - \frac{1}{\rho}\frac{\partial \tau_{yz}}{\partial z}$$

z-direction Momentum

$$\frac{\partial w}{\partial t} + \frac{\partial uw}{\partial x} + \frac{\partial vw}{\partial y} + \frac{\partial w^2}{\partial z} = \frac{-1}{\rho}\frac{\partial p}{\partial z} - \frac{1}{\rho}\frac{\partial \tau_{xz}}{\partial x} - \frac{1}{\rho}\frac{\partial \tau_{yz}}{\partial y} - \frac{1}{\rho}\frac{\partial \tau_{zz}}{\partial z}$$

Energy

$$\frac{\partial T}{\partial t} + \frac{\partial uT}{\partial x} + \frac{\partial vT}{\partial y} + \frac{\partial wT}{\partial z} = -\frac{1}{\rho c_p}\frac{\partial q_x}{\partial x} - \frac{1}{\rho c_p}\frac{\partial q_y}{\partial y} - \frac{1}{\rho c_p}\frac{\partial q_z}{\partial z}$$

where

$$\tau_{xx} = -2\mu\frac{\partial u}{\partial x}, \tau_{yy} = -2\mu\frac{\partial v}{\partial y}, \tau_{zz} = -2\mu\frac{\partial w}{\partial z},$$

$$\tau_{xy} = \tau_{yx} = -\mu(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}), \tau_{xz} = \tau_{zx} = -\mu(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}), \tau_{yz} = \tau_{zy} = -\mu(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y})$$

$$q_x = -k\frac{\partial T}{\partial x}, q_y = -k\frac{\partial T}{\partial y}, \, and \, q_z = -k\frac{\partial T}{\partial z}$$

APPENDIX B

EIGENSYSTEM

In order to develop the eigensystem for these equations the flux Jacobian matrix must first be determined. This matrix is defined as $A = \frac{\partial F}{\partial Q}$ where F and Q are as defined in Chapter 2. The entries of A can be calculated by $A_{ij} = \frac{\partial F_i}{\partial Q_j}$. Performing these calculations gives the following:

$$A = \begin{bmatrix} 0 & \beta n_x & \beta n_y & \beta n_z & 0 \\ \frac{n_x}{\rho} & \theta + u n_x & u n_y & u n_z & 0 \\ \frac{n_y}{\rho} & v n_x & \theta + v n_y & v n_z & 0 \\ \frac{n_z}{\rho} & w n_x & w n_y & \theta + w n_z & 0 \\ 0 & T n_x & T n_y & T n_z & \theta \end{bmatrix}$$

In order to facilitate the computation of the eigenvalues a similarity transformation is performed on A by the following matrix

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ \frac{u}{\beta} & 1 & 0 & 0 & 0 \\ \frac{v}{\beta} & 0 & 1 & 0 & 0 \\ \frac{w}{\beta} & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, with\ M^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ -\frac{u}{\beta} & 1 & 0 & 0 & 0 \\ -\frac{v}{\beta} & 0 & 1 & 0 & 0 \\ -\frac{w}{\beta} & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The resulting matrix κ is below:

$$\kappa = M^{-1} A M = \begin{bmatrix} \theta & \beta n_x & \beta n_y & \beta n_z & 0 \\ \frac{n_x}{\rho} + \frac{u\theta}{\beta} & \theta & 0 & 0 & 0 \\ \frac{n_y}{\rho} + \frac{v\theta}{\beta} & 0 & \theta & 0 & 0 \\ \frac{n_z}{\rho} + \frac{w\theta}{\beta} & 0 & 0 & \theta & 0 \\ \frac{T\theta}{\beta} & T n_x & T n_y & T n_z & \theta \end{bmatrix}$$

In order to find the eigenvalues, the characteristic equation, $det(\kappa - I\lambda) = 0$, must be solved:

$$\begin{vmatrix} \theta - \lambda & \beta n_x & \beta n_y & \beta n_z & 0 \\ \frac{n_x}{\rho} + \frac{u\theta}{\beta} & \theta - \lambda & 0 & 0 & 0 \\ \frac{n_y}{\rho} + \frac{v\theta}{\beta} & 0 & \theta - \lambda & 0 & 0 \\ \frac{n_z}{\rho} + \frac{w\theta}{\beta} & 0 & 0 & \theta - \lambda & 0 \\ \frac{T\theta}{\beta} & Tn_x & Tn_y & Tn_z & \theta - \lambda \end{vmatrix} = 0$$

Expanding the determinant along the fourth column gives

$$(\theta - \lambda)^3 \left((\theta - \lambda)^2 - \theta^2 - \frac{\beta}{\rho}({n_x}^2 + {n_y}^2 + {n_z}^2) \right) = 0$$

Solving for λ results in:

$$\lambda_1, \lambda_2, \lambda_3 = \theta$$

$$\lambda_4 = \theta + c$$

$$\lambda_5 = \theta - c$$

where,

$$c = \sqrt{\theta^2 + \frac{\beta}{\rho}(n_x^2 + n_y^2 + n_z^2)}$$

To finish the eigensystem, the right eigenvectors will be solved for first, using $(\kappa - \lambda_i I)R_i = 0$, where R_i is the right eigenvector corresponding to λ_i and has the following form:

$$R_i = \begin{bmatrix} r_i^1 \\ r_i^2 \\ r_i^3 \\ r_i^4 \\ r_i^5 \end{bmatrix}$$

The equation is written out for clarity below:

$$\begin{bmatrix} \theta - \lambda_i & \beta n_x & \beta n_y & \beta n_z & 0 \\ \frac{n_x}{\rho} + \frac{u\theta}{\beta} & \theta - \lambda_i & 0 & 0 & 0 \\ \frac{n_y}{\rho} + \frac{v\theta}{\beta} & 0 & \theta - \lambda_i & 0 & 0 \\ \frac{n_z}{\rho} + \frac{w\theta}{\beta} & 0 & 0 & \theta - \lambda_i & 0 \\ \frac{T\theta}{\beta} & Tn_x & Tn_y & Tn_z & \theta - \lambda_i \end{bmatrix} \begin{bmatrix} r_i^1 \\ r_i^2 \\ r_i^3 \\ r_i^4 \\ r_i^5 \end{bmatrix} = 0$$

Substituting in θ for λ_i and performing the matrix vector multiplication gives the following equations for λ_1, λ_2, and λ_3 (i=1,2, or 3 below):

$$\beta n_x r_i^2 + \beta n_y r_i^3 + \beta n_z r_i^4 = 0 \tag{B.1}$$

$$\left(\frac{n_x}{\rho} + \frac{u\theta}{\beta}\right) r_i^1 = 0 \tag{B.2}$$

$$\left(\frac{n_y}{\rho} + \frac{v\theta}{\beta}\right) r_i^1 = 0 \tag{B.3}$$

$$\left(\frac{n_z}{\rho} + \frac{w\theta}{\beta}\right) r_i^1 = 0 \tag{B.4}$$

$$\frac{T\theta}{\beta} r_i^1 + T n_x r_i^2 + T n_y r_i^3 + T n_z r_i^4 = 0 \tag{B.5}$$

By observing these equations, it is seen that r_i^5 can be arbitrarily assigned. Equation B.1 can be rewritten as

$$n_x r_i^2 + n_y r_i^3 + n_z r_i^4 = 0$$

and equation B.5 can be rewritten as

$$\frac{\theta}{\beta} r_i^1 + n_x r_i^2 + n_y r_i^3 + n_z r_i^4 = 0$$

Subsituting equation B.1 into equation B.5 gives

$$\frac{\theta}{\beta} r_i^1 = 0 \;\Rightarrow\; r_i^1 = 0$$

The only remaining requirement is

$$n_x r_i^2 + n_y r_i^3 + n_z r_i^4 = 0 \;\Rightarrow\; \langle n_x, n_y, n_z \rangle \cdot \left\langle r_i^2, r_i^3, r_i^4 \right\rangle = 0$$

meaning that r_i^2, r_i^3, and r_i^4 can be assigned as desired as long as the vector $\langle r_i^2, r_i^3, r_i^4 \rangle$ is orthogonal to the $\vec{n}$ area vector. This requirement is fairly simple to enforce. For R_1 a vector is created that fulfills this requirement by simply beginning with e_1, e_2, or e_3 and then altering it to force the dot product with $\vec{n}$ to be 0. This first vector will be called v_1. For R_2, v_1 is simply crossed with $\vec{n}$ to form v_2. Since r_i^5 can be arbitrarily assigned, $r_{1,2}^5 = 0$. For R_3, $r_3^{2,3,4} = 0$ and $r_3^5 = 1$ which fulfills the requirement that the eigenvectors be linearly independent and the other requirement that $\langle n_x, n_y, n_z \rangle \cdot \langle r_i^2, r_i^3, r_i^4 \rangle = 0$.

R_4 and R_5 are still needed. Using $\lambda_4 = \theta + c$ in $(\kappa - \lambda_4 I)R_4 = 0$ gives the following equations:

$$-cr_4^1 + \beta n_x r_4^2 + \beta n_y r_4^3 + \beta n_z r_4^4 = 0 \tag{B.6}$$

$$(\frac{n_x}{\rho} + \frac{u\theta}{\beta})r_4^1 - cr_4^2 = 0 \tag{B.7}$$

$$(\frac{n_y}{\rho} + \frac{v\theta}{\beta})r_4^1 - cr_4^3 = 0 \tag{B.8}$$

$$(\frac{n_z}{\rho} + \frac{w\theta}{\beta})r_4^1 - cr_4^4 = 0 \tag{B.9}$$

$$\frac{T\theta}{\beta}r_4^1 + Tn_x r_4^2 + Tn_y r_4^3 + Tn_z r_4^4 - cr_4^5 = 0 \tag{B.10}$$

Through observation, it is noted that equations B.7, B.8, B.9 can be satisfied by the following relations:

$$r_4^1 = c$$

$$r_4^2 = \frac{n_x}{\rho} + \frac{u\theta}{\beta}$$

$$r_4^3 = \frac{n_y}{\rho} + \frac{v\theta}{\beta}$$

$$r_4^4 = \frac{n_z}{\rho} + \frac{w\theta}{\beta}$$

To check these relations, they are plugged into equation B.6, which shows that they hold. The relations are then used in equation B.10 in order to solve for r_4^5, which gives

$$r_4^5 = \frac{T}{c}\left(\frac{1}{\rho}(n_x^2 + n_y^2 + n_z^2) + \frac{\theta}{\beta}(c+\theta)\right) = \frac{T}{c}\left(\frac{1}{\rho}(n_x^2 + n_y^2 + n_z^2) + \frac{\theta}{\beta}\lambda_4\right)$$

Repeating this process with λ_5 gives R_5. The resulting matrix with the right eigenvectors as columns is below:

$$E = \begin{bmatrix} 0 & 0 & 0 & c & -c \\ x_1 & x_2 & 0 & \frac{n_x}{\rho} + \frac{u\theta}{\beta} & \frac{n_x}{\rho} + \frac{u\theta}{\beta} \\ y_1 & y_2 & 0 & \frac{n_y}{\rho} + \frac{v\theta}{\beta} & \frac{n_y}{\rho} + \frac{v\theta}{\beta} \\ z_1 & z_2 & 0 & \frac{n_z}{\rho} + \frac{w\theta}{\beta} & \frac{n_z}{\rho} + \frac{w\theta}{\beta} \\ 0 & 0 & 1 & \frac{T}{c}\phi_1 & \frac{-T}{c}\phi_2 \end{bmatrix}$$

where $v_1 = \langle x_1, y_1, z_1 \rangle$, $v_2 = \langle x_2, y_2, z_2 \rangle$, c is as defined above, and

$$\phi_1 = \frac{1}{\rho}(n_x^2 + n_y^2 + n_z^2) + \frac{\theta}{\beta}\lambda_4$$

$$\phi_2 = \frac{1}{\rho}(n_x^2 + n_y^2 + n_z^2) + \frac{\theta}{\beta}\lambda_5$$

Now that the right eigenvector matrix has been obtained, the left eigenvector matrix can be computed as E^{-1} and is shown below:

$$E^{-1} = \frac{1}{det(E)} \begin{bmatrix} 0 & 2c(y_2\phi_5 - z_2\phi_4) & 2c(z_2\phi_3 - x_2\phi_5) & 2c(x_2\phi_4 - y_2\phi_3) & 0 \\ 0 & 2c(z_1\phi 4 - y_1\phi 5) & 2c(x_1\phi_5 - z_1\phi_3) & 2c(y_1\phi_3 - x_1\phi_4) & 0 \\ \frac{-T}{2c^2}(\phi_1 + \phi_2)det(E) & Tn_x(\phi_2 - \phi_1) & Tn_y(\phi_2 - \phi_1) & Tn_z(\phi_2 - \phi_1) & det(E) \\ \frac{det(E)}{2c} & cn_x & cn_y & cn_z & 0 \\ \frac{-det(E)}{2c} & cn_x & cn_y & cn_z & 0 \end{bmatrix}$$

where variables are defined as above except for ϕ_3, ϕ_4, and ϕ_5, which are defined below:

$$\phi_3 = \frac{n_x}{\rho} + \frac{u\theta}{\beta}, \phi_4 = \frac{n_y}{\rho} + \frac{v\theta}{\beta}, \phi_5 = \frac{n_z}{\rho} + \frac{w\theta}{\beta}, \ and$$

$$det(E) = 2c\left[(x_1y_2 - y_1x_2)\left(\frac{n_z}{\rho} + \frac{w\theta}{\beta}\right) + (z_1x_2 - x_1z_2)\left(\frac{n_y}{\rho} + \frac{v\theta}{\beta}\right) + (y_1z_2 - z_1y_2)\left(\frac{n_x}{\rho} + \frac{u\theta}{\beta}\right)\right]$$

Recall that E and E^{-1} are the eigenvector matrices for κ. R and L, the right and left eigenvector matrices for A respectively, are still needed and can be computed simply using R = M*E and $L = E^{-1}M^{-1}$. These matrices are written out below:

$$R = \begin{bmatrix} 0 & 0 & 0 & c & -c \\ x_1 & x_2 & 0 & \frac{uc}{\beta} + \phi_3 & -\frac{uc}{\beta} + \phi_3 \\ y_1 & y_2 & 0 & \frac{vc}{\beta} + \phi_4 & -\frac{vc}{\beta} + \phi_4 \\ z_1 & z_2 & 0 & \frac{wc}{\beta} + \phi_5 & -\frac{wc}{\beta} + \phi_5 \\ 0 & 0 & 1 & \frac{T}{c}\phi_1 & \frac{-T}{c}\phi_2 \end{bmatrix}$$

$$L = \frac{1}{det(E)} \begin{bmatrix} -\frac{2c}{\beta}(u(y_2\phi_5 - z_2\phi_4) & & & & \\ +v(z_2\phi_3 - x_2\phi_5) & 2c(y_2\phi_5 - z_2\phi_4) & 2c(z_2\phi_3 - x_2\phi_5) & 2c(x_2\phi_4 - y_2\phi_3) & 0 \\ +w(x_2\phi_4 - y_2\phi_3)) & & & & \\ -\frac{2c}{\beta}(u(z_1\phi 4 - y_1\phi 5) & & & & \\ +v(x_1\phi_5 - z_1\phi_3) & 2c(z_1\phi 4 - y_1\phi 5) & 2c(x_1\phi_5 - z_1\phi_3) & 2c(y_1\phi_3 - x_1\phi_4) & 0 \\ +w(y_1\phi_3 - x_1\phi_4)) & & & & \\ -\frac{T}{2c^2}(\phi_1 + \phi_2)det(E) & & & & \\ -\frac{T}{\beta}(\phi_2 - \phi_1)\theta & Tn_x(\phi_2 - \phi_1) & Tn_y(\phi_2 - \phi_1) & Tn_z(\phi_2 - \phi_1) & det(E) \\ \frac{det(E)}{2c} - \frac{c\theta}{\beta} & cn_x & cn_y & cn_z & 0 \\ -\frac{det(E)}{2c} - \frac{c\theta}{\beta} & cn_x & cn_y & cn_z & 0 \end{bmatrix}$$

APPENDIX C

LEAST-SQUARES GRADIENT COMPUTATION

The gradients of the solution at the nodes are needed in order to achieve higher order accuracy for the inviscid flux terms and for the computation of the viscous flux terms when a directional derivative method is used for their computation. In order to approximate these gradients, this work uses a least squares method, following [7]. Below, this method will be formulated and a set of precomputable coeffecients will be determined in order to facilitate the computation of the gradients in code. The reason for determining a set of coeffecients is to allow for the calculation of each nodal gradient by a simple loop over the surrounding edges. These coeffecients depend only upon the geometry and can be computed a priori. They then can be stored at each edge or each node. Storing the coeffecients at each edge is computationally more efficient, but due to the high cost in memory the coeffecients will be stored at each node in this work. The corresponding edge coeffecients can then be recalculated when needed from looping over the nodes surrounding one another. The idea behind the least squares method begins with a Taylor series approximation of the solution vector u at a particular node j. Note that the gradient approximation that is actually desired is for neighboring node i, not j.

$$u_j = u_i + (\vec{x}_j - \vec{x}_i) \cdot \nabla u_i \tag{C.1}$$

where $\vec{x}_j$ is the coordinate vector of node j. The above formula has truncated the higher order terms, making it a linearly accurate approximation. As stated, the gradient at node i is desired. Using the above formula for each node $1, 2, ..., N$ surrounding i the following

linear system is created.

$$\begin{bmatrix} \Delta x_1 & \Delta y_1 & \Delta z_1 \\ \Delta x_2 & \Delta y_2 & \Delta z_2 \\ \vdots & \vdots & \vdots \\ \Delta x_N & \Delta y_N & \Delta z_N \end{bmatrix} \begin{bmatrix} u_{x_i} \\ u_{y_i} \\ u_{z_i} \end{bmatrix} = \begin{bmatrix} u_1 - u_i \\ u_2 - u_i \\ \vdots \\ u_N - u_i \end{bmatrix} \tag{C.2}$$

where $\Delta x_j = x_j - x_i$, $\vec{x}_j = \langle x_j, y_j, z_j \rangle$, and $\nabla u_i = \langle u_{x_i}, u_{y_i}, y_{z_i} \rangle$. If a weighted least squares method is being used then the left and right hand side of the above equations are multiplied by an arbitrary constant, which can be unique for the each equation.

$$\begin{bmatrix} \Delta \bar{x}_1 & \Delta \bar{y}_1 & \Delta \bar{z}_1 \\ \Delta \bar{x}_2 & \Delta \bar{y}_2 & \Delta \bar{z}_2 \\ \vdots & \vdots & \vdots \\ \Delta \bar{x}_N & \Delta \bar{y}_N & \Delta \bar{z}_N \end{bmatrix} \begin{bmatrix} u_{x_i} \\ u_{y_i} \\ u_{z_i} \end{bmatrix} = \begin{bmatrix} \bar{u}_1 - \bar{u}_i \\ \bar{u}_2 - \bar{u}_i \\ \vdots \\ \bar{u}_N - \bar{u}_i \end{bmatrix} \tag{C.3}$$

where $\Delta \bar{x}_j = \alpha_j \Delta x_j$ and α_j is the weight for the equation. If unweighted least squares is used then $\alpha_j = 1$, and if an inverse distance weight is used then $\alpha_j = \frac{1}{\Delta s_j}$ where $\Delta s_j = \vec{x}_j - \vec{x}_i$. In this work, unweighted least squares is used for computing higher order accuracy for the inviscid terms, and the inverse distance weighted least squares is used when computing the viscous terms. Rewriting equation C.3 with the columns of the left and right hand side matrices as vectors gives

$$\begin{bmatrix} L_1 & L_2 & L_3 \end{bmatrix} \begin{bmatrix} u_{x_i} \\ u_{y_i} \\ u_{z_i} \end{bmatrix} = [U] \tag{C.4}$$

This system is over constrained since there are more equations than there are unknowns. Let $A = \begin{bmatrix} L_1 & L_2 & L_3 \end{bmatrix}$. Multiplying both sides of the equation by A^T gives a square system with one solution.

Orthogonalization

In order to solve this new linear system QR factorization will be utilized, and in order to do QR factorization a set of orthonormal vectors needs to be made from the columns of matrix A. Using Gram-Schmidt on the columns of A gives

$$q_1 = \frac{L_1}{||L_1||}$$

$$q_2 = \frac{L_2 - ({q_1}^T L_2)q_1}{||L_2 - ({q_1}^T L_2)q_1||}$$

$$q_3 = \frac{L_3 - ({q_1}^T L_3)q_1 - ({q_2}^T L_3)q_2}{||L_3 - ({q_1}^T L_3)q_1 - ({q_2}^T L_3)q_2||}$$

To simplify these equations let

$$r_{11} = ||L_1||^2 \,, \quad r_{12} = {L_1}^T L_2 \,, \quad r_{13} = {L_1}^T L_3 \,,$$

$$r_{22} = \left|\left|L_2 - ({q_1}^T L_2)q_1\right|\right|^2 \,, \quad r_{23} = (L_2 - \frac{r_{12}}{r_{11}} L_1)^T L_3 \,,$$

$$and \quad r_{33} = \left|\left|L_3 - ({q_1}^T L_3)q_1 - ({q_2}^T L_3)q_2\right|\right|^2$$

then

$$q_1 = \frac{L_1}{\sqrt{r_{11}}}$$

$$q_2 = \frac{L_2 - \frac{r_{12}}{r_{11}} L_1}{\sqrt{r_{22}}}$$

$$q_3 = \frac{L_3 - \frac{r_{13}}{r_{11}} L_1 - \frac{r_{23}}{r_{22}}(L_2 - \frac{r_{12}}{r_{11}} L_1)}{\sqrt{r_{33}}}$$

QR Factorization

Now A must be factored in two matrices Q and R, where Q has columns that are orthonormal to one another and R is upper triangular. Once A is factored into Q and R the system of equations can be easily solved using back substitution. Since the columns of A have been orthogonalized and normalized in the previous section, it remains to find R. Rewriting the columns of A in terms of their orthonormal counterparts gives

$$L_1 = ({q_1}^T L_1) q_1$$

$$L_2 = ({q_1}^T L_2) q_1 + ({q_2}^T L_2) q_2$$

$$L_3 = ({q_1}^T L_3) q_1 + ({q_2}^T L_3) q_2 + ({q_3}^T L_3) q_3$$

Taking the above equations and putting them into matrix form gives

$$\begin{bmatrix} L_1 & L_2 & L_3 \end{bmatrix} = \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix} \begin{bmatrix} {q_1}^T L_1 & {q_1}^T L_2 & {q_1}^T L_3 \\ 0 & {q_2}^T L_2 & {q_2}^T L_3 \\ 0 & 0 & {q_3}^T L_3 \end{bmatrix}$$

thus giving the needed $A = QR$ form. Using QR instead of A makes equation (C.4) have the following form

$$QRx = b$$

and this equation can be solved by doing the following

$$(QR)^T QRx = (QR)^T b$$

$$R^T Q^T Q R x = R^T Q^T b$$

$$R^T I R x = R^T Q^T b$$

$$R x = Q^T b$$

Since R is upper triangular x can now be easily solved for using backward substitution. Using the Q and R matrices that have already been formed and placing them in the above form gives

$$\begin{bmatrix} {q_1}^T L_1 & {q_1}^T L_2 & {q_1}^T L_3 \\ 0 & {q_2}^T L_2 & {q_2}^T L_3 \\ 0 & 0 & {q_3}^T L_3 \end{bmatrix} \begin{bmatrix} u_{x_i} \\ u_{y_i} \\ u_{z_i} \end{bmatrix} = \begin{bmatrix} {q_1}^T U \\ {q_2}^T U \\ {q_3}^T U \end{bmatrix} \tag{C.5}$$

Solution of QR System

Now the QR system in equation C.5 can be solved. The following definitions are made to ease in notation

$$r_{22}^o = {q_2}^T L_2 \sqrt{r_{22}}$$

$$r_{33}^o = {q_3}^T L_3 \sqrt{r_{33}}$$

Using the above definitions and the definitions of r_{11}, r_{12}, r_{13}, r_{22}, r_{23}, and r_{33} from the previous sections equation (C.5) becomes

$$\begin{bmatrix} \frac{r_{11}}{\sqrt{r_{11}}} & \frac{r_{12}}{\sqrt{r_{11}}} & \frac{r_{13}}{\sqrt{r_{11}}} \\ 0 & \frac{r_{22}^o}{\sqrt{r_{22}}} & \frac{r_{23}}{\sqrt{r_{22}}} \\ 0 & 0 & \frac{r_{33}^o}{\sqrt{r_{33}}} \end{bmatrix} \begin{bmatrix} u_{x_i} \\ u_{y_i} \\ u_{z_i} \end{bmatrix} = \begin{bmatrix} {q_1}^T U \\ {q_2}^T U \\ {q_3}^T U \end{bmatrix}$$

Using backward substitution to solve the above system, and replacing q_1, q_2, and q_3 with their definitions gives the following formulas for the components of the solution gradient

$$u_{z_i} = \frac{\left[L_3 - \frac{r_{13}}{r_{11}}L_1 - \frac{r_{23}}{r_{22}}(L_2 - \frac{r_{12}}{r_{11}}L_1)\right]^T}{r_{33}^o}U = (W^z)^T U \tag{C.6}$$

$$u_{y_i} = \frac{\left[L_2 - \frac{r_{12}}{r_{11}}L_1 - r_{23}W^z\right]^T}{r_{22}^o}U = (W^y)^T U \tag{C.7}$$

$$u_{x_i} = \frac{[L_1 - r_{12}W^y - r_{13}W^z]^T}{r_{11}}U = (W^x)^T U \tag{C.8}$$

Implementation

Equations C.6 - C.8 can be rewritten in summation form

$$u_{x_i} = \sum_{e=1}^{N} W_e^x(\bar{u}_{j(e)} - \bar{u}_i)$$

$$u_{y_i} = \sum_{e=1}^{N} W_e^y(\bar{u}_{j(e)} - \bar{u}_i)$$

$$u_{z_i} = \sum_{e=1}^{N} W_e^z(\bar{u}_{j(e)} - \bar{u}_i)$$

where e represents each edge surrounding node i and $j(e)$ is the node j connected to node i by edge e. Also, W_e^x, W_e^y, W_e^z are essentially the same as W^x, W^y, W^z except they contain only a single row of each L_1, L_2, and L_3 vector. They are defined below

$$W_e^x = \frac{1}{r_{11}}\left[\Delta\bar{x}_e - r_{12}W_e^y - r_{13}W_e^z\right]$$

$$W_e^y = \frac{1}{r_{22}^o}\left[\Delta\bar{y}_e - \frac{r_{12}}{r_{11}}\Delta\bar{x}_e - r_{23}W_e^z\right]$$

$$W_e^z = \frac{1}{r_{33}^o}\left[\Delta\bar{z}_e - \frac{r_{13}}{r_{11}}\Delta\bar{x}_e - \frac{r_{23}}{r_{22}}(\Delta\bar{y}_e - \frac{r_{12}}{r_{11}}\Delta\bar{x}_e)\right]$$

The constants in the above equations are rewritten out below in summation form as well

$$r_{11} = \sum_{e=1}^{N}(\Delta\bar{x}_e)^2$$

$$r_{12} = \sum_{e=1}^{N}(\Delta\bar{x}_e\Delta\bar{y}_e)$$

$$r_{13} = \sum_{e=1}^{N}(\Delta\bar{x}_e\Delta\bar{z}_e)$$

$$r_{22} = \sum_{e=1}^{N}\left[\Delta\bar{y}_e - \frac{r_{12}}{r_{11}}\Delta\bar{x}_e\right]^2$$

$$r_{23} = \sum_{e=1}^{N}\left[\Delta\bar{y}_e - \frac{r_{12}}{r_{11}}\Delta\bar{x}_e\right]\Delta\bar{z}_e$$

$$r_{33} = \sum_{e=1}^{N}\left[\Delta\bar{z}_e - \frac{r_{13}}{r_{11}}\Delta\bar{x}_e - \frac{r_{23}}{r_{22}}(\Delta\bar{y}_e - \frac{r_{12}}{r_{11}}\Delta\bar{x}_e)\right]^2$$

$$r_{22}^o = \sum_{e=1}^{N}\left[\Delta\bar{y}_e - \frac{r_{12}}{r_{11}}\Delta\bar{x}_e\right]\Delta\bar{y}_e$$

$$r_{33}^o = \sum_{e=1}^{N}\left[\Delta\bar{z}_e - \frac{r_{13}}{r_{11}}\Delta\bar{x}_e - \frac{r_{23}}{r_{22}}(\Delta\bar{y}_e - \frac{r_{12}}{r_{11}}\Delta\bar{x}_e)\right]\Delta\bar{z}_e$$

Recall that

$$\Delta\bar{x}_e = \alpha_{j(e)}(x_{j(e)} - x_i)$$

$$\Delta\bar{y}_e = \alpha_{j(e)}(y_{j(e)} - y_i)$$

$$\Delta\bar{z}_e = \alpha_{j(e)}(z_{j(e)} - z_i)$$

as defined in a previous section, and $\alpha_{j(e)}$ is the unique weight for each edge.

The formulas are now in a state that precomputable coeffecients can easily be defined for each node. Let these be

$$s_{11} = \sum_{e=1}^{N} (\Delta \bar{x}_e)^2$$

$$s_{12} = \sum_{e=1}^{N} (\Delta \bar{x}_e \Delta \bar{y}_e)$$

$$s_{13} = \sum_{e=1}^{N} (\Delta \bar{x}_e \Delta \bar{z}_e)$$

$$s_{22} = \sum_{e=1}^{N} (\Delta \bar{y}_e)^2$$

$$s_{23} = \sum_{e=1}^{N} (\Delta \bar{y}_e \Delta \bar{z}_e)$$

$$s_{33} = \sum_{e=1}^{N} (\Delta \bar{z}_e)^2$$

This set of data can be calculated and stored at each node before the code enters any solution loop. Then, when the gradients are needed the constants can be computed using these coeffecients as follows

$$r_{11} = s_{11}$$

$$r_{12} = s_{12}$$

$$r_{13} = s_{13}$$

$$r_{22} = s_{22} - \frac{r_{12}^2}{r_{11}}$$

$$r_{23} = s_{23} - \frac{r_{12}}{r_{11}} r_{13}$$

$$r_{33} = s_{33} - \frac{r_{13}^2}{r_{11}} - \frac{r_{23}^2}{r_{22}}$$

Thus, the formulation of precomputable coeffecients is completed, and the gradients can be calculated using a simple loop of nodes surrounding a node.

REFERENCES

[1] Taylor, L. K., *Unsteady Three-Dimensional Incompressible Algorithm Based on Artificial Compressibility*, Ph.D. Dissertation, Mississippi State University, May 1991. 1, 3, 4, 23

[2] Koblitz, T. W., Bechmann, A., Sorenson, N. N., "The 2D Lid-Driven Cavity - Validation of CFD Code to Model Non-Neutral Atmospheric Boundary Layer Conditions," *Proceedings 6th PhD Seminar on Wind Energy in Europe*, 2010, pp. 157-160. 2, 3, 36, 39

[3] Kameyama, M., Kageyama, A., and Sato, T., "Multigrid Iterative Algorithm Using Pseudo-Compressibility for Three-Dimensional Mantle Convection with Strongly Variable Viscosity," *Journal of Computational Physics*, 206, 1, June 2005, pp. 162-181. 2, 3

[4] Agrawal, L., Mandal, J. C., and Marathe, A. G., "Computations of Laminar and Turbulent Mixed Convection in a Driven Cavity using Pseudo-Compressibility Approach," *Computers & Fluids*, 30, 5, June 2001, pp. 607-620. 3, 36, 39

[5] Menter, F. R., "Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications," *AIAA Journal*, 32, 8, pp. 1598-1605. 3

[6] Er, C. S., *Numerical Simulation of Laminar Incompressible Convective Heat Transfer Flow*, M.S. Thesis, Mississippi State University, December 1995. xi, 3, 4, 7, 32, 34, 35

[7] Hyams, D. G., *An Investigation of Parallel Implicit Solution Algorithms for Incompressible Flows on Unstructured Topologies*, Ph.D. Dissertation, Mississippi State University, May 2000. 4, 7, 12, 13, 14, 57

[8] Beddhu, M., Taylor, L. K., and Whitfield, D. L., "A Time Accurate Calculation Procedure for Flows with a Free Surface Using a Modified Artificial Compressibility Formulation," *Applied Mathematics and Computation*, 65, 1994, pp. 33-48. 7

[9] Fox, R. W., and McDonald, A. T., *Introduction to Fluid Mechanics*, New York: Wiley, 1985. 28

[10] Ghia, U., Ghia, K. N., and Shin, C. T., "High-resolutions for Incompressible Navier-Stokes Equation and a Multigrid Method," *Journal of Computational Physics*, 48, 1982, pp. 387-411. xi, 36, 37

[11] Davis, G. D. V., "Natural Convection of Air in a Square Cavity: A Bench Mark Numerical Solution," *International Journal for Numerical Methods in Fluids*, 3, 1983, pp. 249-164. 32, 35

[12] Iwatsu, R., Hyun J. M., "Three-dimensional Driven-cavity Flows with a Vertical Temperature Gradient," *International Journal of Heat Mass Transfer*, 38, 3, 1995, pp.319-328. xi, xii, 39, 40, 41

VITA

Jessica Kress was born in Tucson, Arizona to the parents of Reid and Ann Kress. She moved with her parents to Oak Ridge, Tennessee as an infant, where she later enrolled in public education. After graduation from Oak Ridge High School, she attended the University of Tennessee at Knoxville where she completed her bachelor's degree in Mathematics. Jessica then accepted a graduate research assistantship with the SimCenter at the University of Tennessee at Chattanooga and completed her master's degree in Computational Engineering in December 2012. She is currently employed as a Research Engineer for Illinois Rocstar in Champaign, Illinois.