ABSTRACT

BULLERWELL, LANCE EDWARD. Implementing a Numerically Improved Three-Dimensional Solver in the Residual Formulation of CTF. (Under the direction of Jason Hou).

Nuclear thermal-hydraulic analysis preforms an integral role in enrusing the safe design and operation of nuclear reactors and the main way these analyses are carried out is through the use of thermal-hydraulic software. Much of the software used in nuclear thermal-hydraulics was originally developed for simulation of light water reactors (LWRs) and the majority of the development took place in the 1970's and 1980's. Due to the focus on LWRs and the limited computational capabilities at the time, the thermal-hydraulic software developed leveraged many assumption and approximations to allow these codes to be useful in terms of accuracy and computational efficiency.

With the significant increase in computational capabilities in recent decades, as well as the growth of interest in advanced reactors, a desire and need for improvements to thermal-hydraulic solvers has arisen. One such solver that has seen significant development is CTF, a version of COolant Boiling in Rod Arrays - Three Field (COBRA-TF), which is developed jointly by North Carolina State University and Oak Ridge National Laboratory. One of the developments in CTF has been to develop a residual formulation, CTF-R, which uses residual functions to calculate a Jacobian matrix numerically.

A novel semi-implicit three-dimensional solver is added to the residual formulation of CTF (CTF-R) in this work. This new solver removes the sub-channel approximation from CTF to improve the applicability of the solver to a wider range of thermal-hydraulic systems. To do this, a new discretization scheme was needed, and an overhaul of how the Jacobian matrix is initialized and constructed was needed. This three-dimensional solver was tested on a set of verification cases and compared to the analytic results. In addition to the semi-implicit solver, an implicit solver was developed within the three-dimensional solver which will remove the courant timestep stability restriction. This was done through the implementation of a nonlinear Shamanskii iteration in the solver. The implicit solver was verified over the same test cases as the semi-implicit solver, and the convergence of the Shamanskii method was tested. The ability to solve thermal-hydraulic systems using the conservative form of the Euler equations was implemented in CTF-R which allows for more accurate capturing of flow behavior near sharp gradients. High-order and high-resolution advective schemes were implemented in CTF-R that will decrease the

highly-diffusive behavior of first-order upwinding. Finally, a test of the various linear solver methods available through PETSc was done to show the potential for optimization of the new capabilities developed in this work. The three-dimensional solver in CTF-R and the numerical capabilities implemented alongside it greatly improve the numerical efficiency and accuracy that CTF-R is capable of while also expanding the array of thermal-hydraulic problems the software is applicable to.

Implementing a Numerically Improved Three-Dimensional Solver in the Residual Formulation of CTF

by
Lance Edward Bullerwell

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Nuclear Engineering

Raleigh, North Carolina
2022

APPROVED BY:

Maria Avramova	Agustin Abarca
Xu Wu	Min Kang
Nathan Porter	Jason Hou Chair of Advisory Committee

BIOGRAPHY

The author was born in Raleigh, North Carolina to Martha and Arthur Bullerwell in June 1996. In 2014, he begin attending North Carolina State University. In May 2018 he graduated with a Bachelor of Science in nuclear engineering. In May 2020, he received a Master of Science in nuclear engineering after defending his thesis on a multi-physics coupling scheme. He continued on in pursuit of a Doctor of Philosophy Degree in nuclear engineering focused on thermal-hydraulics and numerical methods.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER

1

INTRODUCTION

For the operation of nuclear reactors, one of the main tasks is understanding and predicting the heating, pressure, and flow related behavior of the fluid and solid structures within the reactor core, which is commonly referred to as thermal-hydraulics. This is important for optimizing the reactor performance as well as ensuring the safe operation of the reactor. Computer simulation tools are one of the main tools for predicting the thermal-hydraulic behavior within nuclear reactors.

Many of the computational thermal-hydraulic tools used in the nuclear industry, referred to as "legacy codes", were initially developed in the 1970's and 1980's and because of this were designed to run on computers of that time. Memory constraints were more severe, necessitating smaller and simpler models for the fluid behavior as well as approximations of the physical model to take advantage of the geometry and flow that will be expected within reactors. Many of these models, while necessary at the time of development, lead to limitations of the accuracy and applicability of the developed tools. With modern computing systems, many of these models can be improved without sacrificing the ability of codes to give results quickly.

In addition to computational advancements, interest in advanced reactors has grown

significantly in the past few decades. Many of the models and approximations implemented in many thermal-hydraulic solvers are valid specifically for light water reactors (LWR) and need to be updated to be applied to advanced reactors. Liquid metal reactors have significant thermal conduction within the fluid, molten salt fast reactors don't have axially dominated flow in the core, and many of the correlations implemented apply only to water are a few of the ways many conventional thermal-hydraulic tools can't be applied to advanced reactor designs.

Due to the improvement of computational capabilities and memory storage, as well as the physics that needs to be captured in advanced reactors, much effort has been made to modernize these "legacy codes" in a variety of ways. One such effort was the development of a residual-based solver for the sub-channel code CTF, referred to as CTF-R. The motivation for this work is to implement improved numerical methods within this residual solver. An array of tasks will be performed to do this, the first of which is to remove the sub-channel approximation and make a three-dimensional Cartesian solver within CTF-R which will allow for the simulation. Next, a nonlinear iteration will be implemented in the solver in the form of a Shamanskii iteration and both an implicit solver and a conservative solver will be implemented using the Shamanskii iteration. The first-order upwinding for the advective derivative will be updated to allow for the option of both high-order and high-resolution upwinding schemes.

The next chapter discusses much of the background and theory that is needed to make these improvements to CTF-R. The implementation of the three-dimensional solver and the initial verification problems used are discussed in chapter 3. The work on implementing and verifying the implicit solver and conservative solver is detailed in chapter 4. The development of high-resolution advective scheme for the new solvers is detailed in chapter 5. The final chapter gives overall conclusions on the work and a discussion of future work for the new three-dimensional solver in CTF-R.

CHAPTER

2

BACKGROUND AND THEORY

In this chapter, much of the necessary background and theory necessary for this work are given. A discussion of nuclear thermal-hydraulics simulation tools, the key differences between the main categories of thermal-hydraulics codes, and a discussion of the residual formulation of CTF are given in section 2.1. The Euler equations which govern fluid flow for inviscid flows are given in both a conservative and non-conservative form in section 2.2. Options for solution of a nonlinear system of equations is given in section 2.3, starting with Newton's method and discussing multiple methods to alter and improve on Newton's method. The main methods implemented in thermal-hydraulics codes for transient iteration control are given in section 2.4. The advective derivative and the difficulties in its discretization is introduced in section 2.5 as are methods for improving the discretization.

2.1 Nuclear Thermal-Hydraulics

One of the main tasks for nuclear engineers is to model and predict the heating and flow of fluid within nuclear reactors, which is referred to as thermal-hydraulic analysis. Predic-

tive thermal-hydraulic analysis can be done in two main ways, through evaluation and comparison to experimental data and through simulation through computational codes. Experimental thermal-hydraulic data on nuclear systems can be difficult to come by, due to a combination of the expense of operating these systems for tests and the difficulty in getting accurate and fine enough data from experiments. In addition, it is extremely difficult and frequently impossible to run predictive experiments for accident cases. Because of this, the main way thermal-hydraulic predictions are done is through the use of thermal-hydraulic codes.

2.1.1 Thermal-Hydraulics Software Overview

There are three main types of codes used for thermal-hydraulic analysis of nuclear reactors: System codes, sub-channel codes, and computational fluid dynamics (CFD) codes. In this section, background info on a variety of nuclear thermal-hydraulic tools will be given with a focus on sub-channel codes and CTF and its residual formulation in particular.

System codes are a frequently used tool in nuclear thermal-hydraulics simulations as they are capable of modeling the behavior of the entire cooling system in the reactor, not just in-core behavior. Due to the scale modeled, system codes frequently rely on coarser models. In addition many system codes have the capability of performing balance of plant (BOP) calculations in conjunction with simulations of the reactor coolant system. There are a wide variety of system codes available including Reactor Excursion and Leak Analysis Program(RELAP7), System Analysis Module (SAM), the TRAC/RELAP Advanced Computational Engine (TRACE), and Methods for Estimation of Leakages and Consequences of Releases (MELCOR) (6; 25; 43; 21).

Sub-channel software are codes designed to preform in-core modeling, with higher-fidelity models than typically present in system codes. Sub-channel codes frequently make assumptions and rely on correlations relating to the geometry, flow paths, and flow behavior inside reactor cores to simplify simulations without losing much fidelity. The defining feature of sub-channel codes is the sub-channel approximation of flow behavior that flow is axially dominated. The ability to couple sub-channel codes to system codes can allow for more accurate in-core analysis while still producing results for the entire coolant system. A few notable sub-channel codes are CTF (52), SUBCHANFLOW (26), and VIPRE (56), and a good overview of the sub-channel codes is given in (41). More detail on sub-channel codes and CTF in particular will be given in Section 2.1.2.

Computational Fluid Dynamics (CFD) codes are codes which use high-fidelity models and very fine meshes to accurately model the turbulent behavior in the fluid through simulation of the Navier-Stokes Equations. CFD codes use minimal assumptions to simplify modeling of the domain which coupled with the turbulent modeling and fine mesh leads to CFD being very expensive to use for modeling purposes. However, the accuracy and fidelity of the results from CFD codes are significantly improved over other methods. In addition to the accuracy of results, CFD codes are capable of more flexibly handling different geometries and problem definitions. CFD codes are frequently used for comparison and verification of other thermal-hydraulic codes and for verification and validation of lower-order models that are used in other codes, such as turbulence and friction. A few of the most commonly used CFD codes in nuclear thermal-hydraulics are Nek5000 (45), STAR-CCM+ (55), and OpenFOAM (59).

A few codes have been developed with the idea of bridging the gap between sub-channel codes and CFD codes. These codes, such as Pronghorn (42) and GOTHIC (17), use turbulence models similar to CFD, however they use wall functions in place of modeling the boundary layer, allowing for significantly coarser and less expensive meshes. The goal of these codes is to offer similar results and flexibility to CFD codes while being significantly more efficient in terms of cost and runtime. Also, a common use of thermal-hydraulics codes is coupling a systems code outside the core with a sub-channel code for in core analysis (57).

2.1.2 Sub-channel Codes and CTF-R

As mentioned in Section 2.1.1, sub-channel codes are thermal-hydraulic tools which are frequently used to provide simulation data for the in-core behavior of nuclear reactors. Sub-channel codes are typically defined by the use of the approximation of axially dominated flow which simplifies the governing equations for the system. Effectively this splits the flow behavior into two directions, the axial direction and the lateral direction (which is any direction orthogonal to the axial). Directional information is not retained for flow leaving or entering a cell in the lateral direction which greatly simplifies the momentum equations for the system (52). While this approximation greatly simplifies the system being solved, it also greatly limits the applicability of the solver to systems that satisfy this assumption.

One of the most widely-used families of sub-channel code is the COolant Boiling in Rod Arrays (COBRA) family of codes. COBRA-I was the original version of COBRA and was

developed at Batelle Pacific Northwest Laboratory in the late 1960's (50). COBRA development was continued in the form of COBRA-II, -III, and -IV until the late 1970's (41; 60). COBRA-Three Field (COBRA-TF) was developed at Pacific Northwest Laboratory in the early 1980's and descending versions of COBRA-TF have been developed and maintained by many different institutions today (57). COBRA-IE developed by Bettis Atomic Laboratory, COBRA-FLX developed by AREVA, CTF developed jointly by North Carolina State University and Oak Ridge National Laboratory, are a few examples of codes based upon COBRA-TF (4; 2; 52).

Most of the development of COBRA-based codes, and in fact most other sub-channel codes, occurred in the 1960's-1980's, or are based heavily on codes developed in that window (41; 57; 50). Because of this, most sub-channel codes are designed to be run on computer architecture from that era, when memory and processor capabilities were many orders of magnitude smaller and more expensive. This lead to methods and approximations being implemented to greatly reduce the storage and computational power requirements to get meaningful results. The sub-channel approximation is one such example, as are the use of the semi-implicit method as discussed in (35; 23), coarse discretizations, ignoring fluid conduction, and ignoring or simplifying turbulence effects, among others.

Since the initial development of these codes, computational capabilities have skyrocketed, allowing for many of these approximations and simplifications to be removed without an excessive computational burden . In addition to computational capabilities improving, there has been a large growth in interest to advanced reactor designs. Many of the assumptions made in these codes leveraged the geometry and design of light water reactors (LWRs) and many of these assumptions are invalid in advanced reactors meaning these codes cannot be used in the modeling of those systems. These two factors have lead to many of the assumptions and approximations being revisited and further development work being done to modernize these codes (48; 12; 36).

One specific modernization effort is in the residual formulation of CTF (CTF-R) which was developed initially at Pennsylvania State University by C Dances and subsequently at North Carolina State University (12; 11; 48; 47). CTF-R is a single-phase sub-channel solver that directly calculates the Jacobian matrix numerically for use in solving the system of equations

$$F(x) = 0 \tag{2.1}$$

which is different than the analytic pressure matrix method in COBRA-TF based codes (52; 4;

12; 48). This direct numerical computation of the Jacobian allows for a degree of flexibility in the solution that is not offered, both in the method of solution through nonlinear methods and time discretization methods as discussed in section 2.3 and section 2.4 and in the ability to alter terms in the governing equations for the system without significant alterations to the code.

CTF-R uses PETSc (5) to solve the matrix system currently. CTF-R is capable of running transient and null transient simulations, has gravitational, friction and form loss, and heat transfer source terms and has much of the same capabilities available to it as CTF with the exclusion of multi-phase flow. More detail on CTF-R is given in section 3.1.

2.2 Euler Equations for Fluid Flow

In Fluid Mechanics, there are two main types of the governing equations that are typically used, the Navier-Stokes Equations, and the Euler Equations. The main difference between the two is that the Navier-Stokes Equations models the behavior of viscous flow and the Euler Equations model the behavior of inviscid flow (14; 1). This difference leads to the Euler equations not including the frictional effects of the stress tensor or thermal conduction in the fluid.

In Thermal-Hydraulics, the Euler equations are typically the equations of interest and the Euler equations, or some variation on them, are what is implemented in most system and sub-channel codes. The reason for this is that the inertial forces overwhelm the viscous forces in the fluid which allows them to be neglected without significantly changing the result of the analysis.

Mathematically, there are two main ways to represent the Euler equations, the conservative form and the non-conservative form. The conservative form is simply the form of the Euler equations, or any differential equation, written in terms that conserve a quantity exactly. The general form for the conservation form of an equation is that which has a time derivative of the property and the flux of that property through a surface boundary. In terms of fluid flow, the conservative form of an equation is typically written in terms of a change in a fluid property and the advection of that property due to the fluid velocity,

$$\frac{\partial}{\partial t}(x)+\nabla\cdot(\vec{V}x)=S, \tag{2.2}$$

where x is some property or variable of the system, $\vec{V}$ is the fluid velocity, and S is a source term that affects that property (1).

The non-conservative form of an equation is simply the form of a differential equation not written in conservative form. In fluid flow, the conservative and non-conservative forms of the governing equations are derived by taking two differing approaches. To derive the conservative form, a control volume that is fixed in space is modeled while for deriving the non-conservative form, a control volume that flows with or is advected by the fluid flow is typically used. The conservative and non-conservative form of the Euler equations are discussed in further detail in this section.

2.2.1 Conservative Form of the Euler Equations

There are three main governing equations in thermal-hydraulics, the continuity, or conservation of mass, equation, the vector momentum equation, and the conservation of energy equation. In this subsection, the conservative form of these equations are introduced and analyzed for a three-dimensional Cartesian geometry.

The general form of the continuity equation is

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{V}) = S_{mass}, \tag{2.3}$$

where ρ is the fluid density, $\vec{V}$ is the fluid velocity vector with components (u, v, w), and S_{mass} is the source of mass in the system. The first term, $\frac{\partial \rho}{\partial t}$, represents the change in the fluid density with respect to time. The second term, $\nabla \cdot (\rho \vec{V})$, represents the advection of mass in the system due to the velocity field. The mass source term is typically only due to phase change, however the analysis here will be focused on a single-phase fluid. Nonetheless, the mass source term will be included in further derivation and equations. In three-dimensions, the continuity equation becomes,

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} + \frac{\partial \rho v}{\partial y} + \frac{\partial \rho w}{\partial z} = S_{mass}. \tag{2.4}$$

The generic form of the conservation of momentum equation is

$$\frac{\partial \rho \vec{V}}{\partial t} + \nabla \cdot (\rho \vec{V} \vec{V}) + \nabla P = \vec{F}, \tag{2.5}$$

where P is the pressure acting on the fluid and $\vec{F}$ is the force terms acting on the fluid. The first term, $\frac{\partial \rho \vec{V}}{\partial t}$, represents the change in specific momentum with respect to time. The second term, $\nabla \cdot (\rho \vec{V} \vec{V})$, represents the advection of momentum in the system. The last term on the left-hand side, ∇P, represent the pressure gradient. The pressure gradient has been separated from the rest of the force terms.

In three-dimensions, Equation 2.5 is a vector equation with three components, one in each dimension. The conservation of momentum in the x, y, and z-directions respectively are

$$\frac{\partial \rho u}{\partial t} + \frac{\partial \rho uu}{\partial x} + \frac{\partial \rho vu}{\partial y} + \frac{\partial \rho wu}{\partial z} + \frac{\partial P}{\partial x} = F_x \tag{2.6a}$$

$$\frac{\partial \rho v}{\partial t} + \frac{\partial \rho uv}{\partial x} + \frac{\partial \rho vv}{\partial y} + \frac{\partial \rho wv}{\partial z} + \frac{\partial P}{\partial y} = F_y \tag{2.6b}$$

$$\frac{\partial \rho w}{\partial t} + \frac{\partial \rho uw}{\partial x} + \frac{\partial \rho vw}{\partial y} + \frac{\partial \rho ww}{\partial z} + \frac{\partial P}{\partial z} = F_z. \tag{2.6c}$$

The conservation of energy equation is

$$\frac{\partial \rho E}{\partial t} + \nabla \cdot (\rho E \vec{V}) + \nabla \cdot (P \vec{V}) = S_{energy}, \tag{2.7}$$

where E is the total energy and S_{energy} is the energy source term in the fluid. The first term, $\frac{\partial \rho E}{\partial t}$, represents the change in the total energy of the system with respect to time. The second term, $\nabla \cdot (\rho E \vec{V})$, represents the advection of total energy in the fluid and the third term, $\nabla \cdot (P \vec{V})$, represents the pressure work on the fluid. In three-dimensions, Equation 2.7 becomes,

$$\frac{\partial \rho E}{\partial t} + \frac{\partial \rho Eu}{\partial x} + \frac{\partial \rho Ev}{\partial y} + \frac{\partial \rho Ew}{\partial z} + \frac{\partial uP}{\partial x} + \frac{\partial vP}{\partial y} + \frac{\partial wP}{\partial z} = S_{energy}. \tag{2.8}$$

The conservation of energy equation written in terms in internal energy is,

$$\frac{\partial \rho(e + \frac{\vec{V}^2}{2})}{\partial t} + \frac{\partial \rho(e + \frac{\vec{V}^2}{2})u}{\partial x} + \frac{\partial \rho(e + \frac{\vec{V}^2}{2})v}{\partial y} + \frac{\partial \rho(e + \frac{\vec{V}^2}{2})w}{\partial z} + \frac{\partial uP}{\partial x} + \frac{\partial vP}{\partial y} + \frac{\partial wP}{\partial z} = S_{energy}, \tag{2.9}$$

where e is the internal energy defined as $e = E - \frac{\vec{V}^2}{2}$. This equation can be changed once

more to be in terms of the specific enthalpy of the fluid, $h = e + \frac{P}{\rho}$,

$$\frac{\partial \rho(h - \frac{P}{\rho} + \frac{\vec{V}^2}{2})}{\partial t} + \frac{\partial \rho(h - \frac{P}{\rho} + \frac{\vec{V}^2}{2})u}{\partial x} + \frac{\partial \rho(h - \frac{P}{\rho} + \frac{\vec{V}^2}{2})v}{\partial y} + \frac{\partial \rho(h - \frac{P}{\rho} + \frac{\vec{V}^2}{2})w}{\partial z} + \frac{\partial uP}{\partial x} + \frac{\partial vP}{\partial y} + \frac{\partial wP}{\partial z} = S_{energy}. \quad (2.10)$$

This equation will be simplified in the Section 2.2.2

2.2.2 Non-Conservative Form of the Euler Equations

In this subsection, the conservative form of the Euler equations introduced in Section 2.2.1 will be used to derive the non-conservative form of the Euler equations. The continuity equation shown in Equation 2.3 is used in both the conservative and non-conservative form of the Euler equations. The continuity equation in both the general and three-dimensional form are shown again here for completeness.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{V}) = S_{mass}, \quad (2.11)$$

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} + \frac{\partial \rho v}{\partial y} + \frac{\partial \rho w}{\partial z} = S_{mass}. \quad (2.12)$$

Conservation of Momentum

The three-dimensional conservation of momentum equation in vector form is,

$$\frac{\partial \rho \vec{V}}{\partial t} + \nabla \cdot (\rho \vec{V} \vec{V}) + \nabla P = \vec{F}, \quad (2.13)$$

as shown in Equation 2.5. Expanding the first two terms in this equation by the product rule yields

$$\rho \frac{\partial \vec{V}}{\partial t} + \vec{V} \frac{\partial \rho}{\partial t} + \rho \vec{V} \nabla \cdot \vec{V} + \vec{V} \nabla \cdot (\rho \vec{V}) + \nabla P = \vec{F}. \quad (2.14)$$

This equation can be rearranged and part of it can be seen as equal to the left-hand side of the continuity equation in Equation 2.3. This term can be set equal to the right-hand side

of the continuity equation, which is just the mass source term.

$$\rho\frac{\partial\vec{V}}{\partial t}+\rho\vec{V}\nabla\cdot\vec{V}+\vec{V}\left(\frac{\partial\rho}{\partial t}+\nabla\cdot(\rho\vec{V})\right)+\nabla P=\vec{F} \tag{2.15}$$

$$\rho\frac{\partial\vec{V}}{\partial t}+\rho\vec{V}\nabla\cdot\vec{V}+\vec{V}S_{mass}+\nabla P=\vec{F} \tag{2.16}$$

Rearranging to move the mass source term to the right-hand side, the non-conservative form of the vector momentum equation is shown to be

$$\rho\frac{\partial\vec{V}}{\partial t}+\rho\vec{V}\nabla\cdot\vec{V}+\nabla P=\vec{F}-\vec{V}S_{mass}. \tag{2.17}$$

Expanding the vector form of the momentum equation to show the three component equations yields,

$$\rho\frac{\partial u}{\partial t}+\rho u\frac{\partial u}{\partial x}+\rho v\frac{\partial u}{\partial y}+\rho w\frac{\partial u}{\partial z}+\frac{\partial P}{\partial x}=F_x-uS_{mass} \tag{2.18a}$$

$$\rho\frac{\partial v}{\partial t}+\rho u\frac{\partial v}{\partial x}+\rho v\frac{\partial v}{\partial y}+\rho w\frac{\partial v}{\partial z}+\frac{\partial P}{\partial y}=F_y-vS_{mass} \tag{2.18b}$$

$$\rho\frac{\partial w}{\partial t}+\rho u\frac{\partial w}{\partial x}+\rho v\frac{\partial w}{\partial y}+\rho w\frac{\partial w}{\partial z}+\frac{\partial P}{\partial z}=F_z-wS_{mass} \tag{2.18c}$$

Conservative Conservation of Enthalpy

The conservation of energy equation written in terms of the enthalpy introduced in Equation 2.10, in general form, is

$$\frac{\partial\rho(h-\frac{P}{\rho}+\frac{\vec{V}^2}{2})}{\partial t}+\nabla\cdot\left(\rho(h-\frac{P}{\rho}+\frac{\vec{V}^2}{2})\vec{V}\right)+\nabla\cdot(P\vec{V})=S_{energy}. \tag{2.19}$$

Rearranging this equation yields,

$$\frac{\partial\rho(h-\frac{P}{\rho})}{\partial t}+\nabla\cdot\left(\rho(h-\frac{P}{\rho})\vec{V}\right)+\frac{1}{2}\left(\frac{\partial\rho\vec{V}^2}{\partial t}+\nabla\cdot(\rho\vec{V}^2\vec{V})\right)+\nabla\cdot(P\vec{V})=S_{energy}. \tag{2.20}$$

Analyzing the term in parentheses multiplied by one half, we can expand the derivatives with the product rule to give

$$\frac{\partial \rho \vec{V}^2}{\partial t} + \nabla \cdot (\rho \vec{V}^2 \vec{V}) = \rho \vec{v} \frac{\partial \vec{V}}{\partial t} + \vec{V} \frac{\partial \rho \vec{V}}{\partial t} + \rho \vec{V}^2 \nabla \cdot \vec{V} + \vec{V} \nabla \cdot (\rho \vec{V}^2). \tag{2.21}$$

The right-hand side can be rearranged to show the left-hand side of both the conservative and the non-conservative momentum equations minus the pressure gradient terms,

$$\frac{\partial \rho \vec{V}^2}{\partial t} + \nabla \cdot (\rho \vec{V}^2 \vec{V}) = \vec{V} \left(\rho \frac{\partial \vec{V}}{\partial t} + \rho \vec{V} \nabla \cdot \vec{V} \right) + \vec{V} \left(\frac{\partial \rho \vec{V}}{\partial t} + \nabla \cdot (\rho \vec{V}^2) \right). \tag{2.22}$$

Substituting the right-hand side of the two momentum equations shown in Equation 2.5 and Equation 2.17 yields

$$\frac{\partial \rho \vec{V}^2}{\partial t} + \nabla \cdot (\rho \vec{V}^2 \vec{V}) = \vec{V} \left(\vec{F} - \vec{V} S_{mass} - \nabla P \right) + \vec{V} \left(\vec{F} - \nabla P \right). \tag{2.23}$$

Substituting the relation derived in Equation 2.23 back into Equation 2.20 yields

$$\frac{\partial \rho (h - \frac{P}{\rho})}{\partial t} + \nabla \cdot \left(\rho (h - \frac{P}{\rho}) \vec{V} \right) + \frac{1}{2} \vec{V} \left(\vec{F} - \vec{V} S_{mass} - \nabla P + \vec{F} - \nabla P \right) + \nabla \cdot (P \vec{V}) = S_{energy}. \tag{2.24}$$

Combining like terms and moving all source terms to the right-hand side gives

$$\frac{\partial \rho (h - \frac{P}{\rho})}{\partial t} + \nabla \cdot \left(\rho (h - \frac{P}{\rho}) \vec{V} \right) - \vec{V} \cdot \nabla P + \nabla \cdot (P \vec{V}) = S_{energy} - \vec{V} \cdot \vec{F} + \frac{1}{2} \vec{V}^2 S_{mass}. \tag{2.25}$$

Next, the time and advective derivatives are separated out to yield

$$\frac{\partial \rho h}{\partial t} - \frac{\partial P}{\partial t} + \nabla \cdot (\rho h \vec{V}) - \nabla \cdot (P \vec{V}) - \vec{V} \cdot \nabla P + \nabla \cdot (P \vec{V}) = S_{energy} - \vec{V} \cdot \vec{F} + \frac{1}{2} \vec{V}^2 S_{mass}. \tag{2.26}$$

which, when the pressure work terms are canceled out, yields the conservative form of the conservation of enthalpy equation which is

$$\frac{\partial \rho h}{\partial t} - \frac{\partial P}{\partial t} + \nabla \cdot (\rho h \vec{V}) - \vec{V} \cdot \nabla P = S_{energy} - \vec{V} \cdot \vec{F} + \frac{1}{2} \vec{V}^2 S_{mass}. \tag{2.27}$$

In three-dimensions, this equation becomes

$$\frac{\partial \rho h}{\partial t} - \frac{\partial P}{\partial t} + \frac{\partial \rho h u}{\partial x} + \frac{\partial \rho h v}{\partial y} + \frac{\partial \rho h w}{\partial z} - u\frac{\partial P}{\partial x} - v\frac{\partial P}{\partial y} - w\frac{\partial P}{\partial z} = S_{energy} - \vec{V}\cdot\vec{F} + \frac{1}{2}\vec{V}^2 S_{mass}. \tag{2.28}$$

Non-Conservative Conservation of Enthalpy

Starting from the conservative form of the conservation of enthalpy equation in Equation 2.27 and expanding the time and advective derivatives with the product rule results in

$$\rho\frac{\partial h}{\partial t} + h\frac{\partial \rho}{\partial t} - \frac{\partial P}{\partial t} + h\nabla\cdot(\rho\vec{V}) + \rho\vec{V}\cdot\nabla h - \vec{V}\cdot\nabla P = S_{energy} - \vec{V}\cdot\vec{F} + \frac{1}{2}\vec{V}^2 S_{mass}. \tag{2.29}$$

Rearranging the expanded terms leaves the left-hand side of the continuity equation, Equation 2.3, multiplied by the enthalpy. This term can be replaced with the mass source multiplied by the fluid enthalpy.

$$\rho\frac{\partial h}{\partial t} - \frac{\partial P}{\partial t} + \rho\vec{V}\cdot\nabla h + h\left(\frac{\partial \rho}{\partial t} + \nabla\cdot(\rho\vec{V})\right) - \vec{V}\cdot\nabla P = S_{energy} - \vec{V}\cdot\vec{F} + \frac{1}{2}\vec{V}^2 S_{mass}. \tag{2.30}$$

$$\rho\frac{\partial h}{\partial t} - \frac{\partial P}{\partial t} + \rho\vec{V}\cdot\nabla h + hS_{mass} - \vec{V}\cdot\nabla P = S_{energy} - \vec{V}\cdot\vec{F} + \frac{1}{2}\vec{V}^2 S_{mass}. \tag{2.31}$$

Moving the enthalpy multiplied by the mass source term to the right-hand side leaves the non-conservative form of the conservation of enthalpy equation.

$$\rho\frac{\partial h}{\partial t} - \frac{\partial P}{\partial t} + \rho\vec{V}\cdot\nabla h - \vec{V}\cdot\nabla P = S_{energy} - \vec{V}\cdot\vec{F} + S_{mass}\left(\frac{1}{2}\vec{V}^2 - h\right). \tag{2.32}$$

It is important to note that while the mass source term in the momentum and enthalpy equations and the inclusion of the force vector in the enthalpy equation are included in the derivation of the non-conservative equations and the enthalpy equations, they are frequently neglected in thermal-hydraulic codes. In addition to these source terms from higher equations, the product of the velocity and the pressure gradient in the conservation of enthalpy equation is neglected in codes like CTF, which is discussed in Appendix A of (48).

2.3 Nonlinear Iteration Strategies

The system that is created by the governing equations discussed in section 2.2 is a nonlinear system of equations. There are many ways to solve a nonlinear system of equations, but the most widely used way is to approximate the system of equations with a two term Taylor expansion such as

$$x^* = x^k + F'(x)^{-1}F(x), \tag{2.33}$$

where the system of equations is defined as $F(x) = 0$ (27). The matrix, $F'(x)$, is referred to as the Jacobian matrix and is frequently denoted $J(x)$. The equation in Equation 2.33 represents a linear system of equations that is much simpler to solve than a nonlinear system, however the nonlinearities in the initial system must still be resolved. To do this, a fixed point iteration is frequently used.

2.3.1 Newton's Method

The most common way to solve a nonlinear system of equations is through an iteration that uses a Newton method. Newton's method is a nonlinear iteration method that reconstructs the Jacobian matrix after each update to the solution is carried out according to Equation 2.33. The basic algorithm of Newton's method is shown below.

Algorithm 1 Newton's Method

1: **for** $k = 1 \rightarrow K$ **do**
2: construct $J(\boldsymbol{x}^k)$
3: $\delta \boldsymbol{x}^k = J^{-1}(\boldsymbol{x}^k)F(\boldsymbol{x}^k)$
4: $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \delta \boldsymbol{x}^k$
5: Check Convergence
6: **end for**

Newton's method is referred to as locally convergent on some domain, β, about the true solution of the system, $\boldsymbol{x}^*$. The domain β is defined as the region within a radius, r, of the solution, $\boldsymbol{x}^*$. This means that if the current iterate, $\boldsymbol{x}^k$ is within r of $\boldsymbol{x}^*$, then Newton's Method will converge. The convergence of Newton's method to $\boldsymbol{x}^*$ can be shown to be q-quadratic, which means that for a current iterate, $\boldsymbol{x}^k$ there exists a positive real number,

K, such that the following inequality is true,

$$\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^*\| \leq K\|\boldsymbol{x}^k - \boldsymbol{x}^*\|^2. \tag{2.34}$$

The error of the current solution vector against the true solution vector can be described by the variable e^k by

$$e^k = \boldsymbol{x}^k - \boldsymbol{x}^*. \tag{2.35}$$

This leads to Equation 2.34 being rewritten as

$$\|e^{k+1}\| \leq K\|e^k\|^2. \tag{2.36}$$

There are two main drawbacks for Newton's method, the first of which is the locally convergent behavior of Newton's method. This means that the initial iterate, $\boldsymbol{x}^0$, must be sufficiently close to the true root of the problem, $\boldsymbol{x}^*$, to converge, otherwise the iteration will diverge. There are a variety of ways to improve the initial iterate of Newton's method, a few of which are detailed here. A simpler or lower-order model of the same problem can be used to improve the initial guess of the simulation. An example of this would be to use the pressure gradient due to gravity $\frac{\partial P}{\partial x} = -\rho g$ to initialize the pressure gradient, or using a constant heat source to calculate the enthalpy rise in a heated channel (40).

For more complex problems and solvers, it can become more difficult to get a sufficiently close initial iterate to lead Newton's method to converge, even with improvements to the initial iterate. This leads to the need for a class of methods defines as global Newton methods. There are a variety of global Newton methods, and one of the more widely used methods of enabling global convergence is a line search algorithm. One of the main reasons that line searchs are used so commonly is that it is relatively trivial to add a line search algorithm to an existing Newton method. Specifically detailed here will be a line search which also uses the Armijo rule (27). The values of $[\sigma_0, \sigma_1]$ are selected on the range $[0, 1]$ and chosen to ensure sufficient decrease in the the norm of the residual vector in a reasonable number of iterations and α is a number on the range $(0, 1)$, intended to ensure a sufficient decrease in the norm of the residual vector during the search. The algorithm for a line search within Newton's method is shown below.

Algorithm 2 Line Search with Newton's Method

for $k = 1 \rightarrow K$ **do**
    construct $J(\boldsymbol{x}^k)$
    $d = -J^{-1}(\boldsymbol{x}^k)F(\boldsymbol{x}^k)$
    $\lambda = 1$
    $\boldsymbol{x}^t = \boldsymbol{x}^k + \lambda d$
    while $|F(\boldsymbol{x}^t)| < (1-\alpha\lambda)|F(\boldsymbol{x}^k)|$ **do**
        $\lambda = \frac{\lambda}{2}$
        $\sigma \epsilon [\sigma_0, \sigma_1]$
        $\lambda = \sigma\lambda$
        $\boldsymbol{x}^t = \boldsymbol{x}^k + \lambda d$
    end while
    $\delta\boldsymbol{x}^k = \lambda d$
    $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \delta\boldsymbol{x}^k$
    Check Convergence
end for

The other main drawback of Newton's method lies in the construction of the Jacobian matrix. For larger problem domains and for more complex solvers, the construction of the Jacobian matrix can become a significant, sometimes overwhelmingly so, portion of software runtime. Because of this, methods that allow for the cost associated with the Jacobian to be reduced are desired. A few of the ways to do this, decreasing the frequency with which the Jacobian is constructed, such as decreasing the cost of the Jacobian construction, or removing the Jacobian calculation altogether, are detailed in the following subsections.

2.3.2 Chord Method

The most simple adjustment to Newton's method that can be implemented is the chord method, which performs the calculation of the Jacobian matrix outside of the nonlinear iteration loop. This means that the Jacobian matrix for iterates of the nonlinear loop other than the first is only using an approximation of the Jacobian matrix of the current solution vector. The algorithm for the chord method is shown below. The main difference between the chord method and Newton's method is the fact that the Jacobian matrix is not reconstructed in each iteration and is only constructed from the initial iterate.

Algorithm 3 Chord Method

1: construct $J(\boldsymbol{x}^0)$ and $F(\boldsymbol{x}^0)$
2: **for** $k = 1 \rightarrow K$ **do**
3: $\quad \delta \boldsymbol{x}^k = J^{-1}(\boldsymbol{x}^0)F(\boldsymbol{x}^k)$
4: $\quad \boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \delta \boldsymbol{x}^k$
5: $\quad$ Construct $F(\boldsymbol{x}^k)$
6: $\quad$ Check Convergence
7: **end for**

Similar to Newton's method, the chord method is locally convergent within some radius, r, about the true problem solution, $\boldsymbol{x}^*$. The convergence of the chord method is described as q-linearly convergent, which means that for a current iterate, $\boldsymbol{x}^k$, and initial iterate, $\boldsymbol{x}^0$, there exists a positive real number, K_c, such that the inequality,

$$\|e^{k+1}\| \leq K_C \|e^k\| \|e^0\|, \tag{2.37}$$

is true (27).

As shown in Equation 2.37, the chord method is slower to converge than Newton's method, meaning it will take more iterations to converge. However, in many problems the construction of the Jacobian matrix can be extremely expensive and in these cases, the chord method can be more efficient in terms of finding the solution than Newton's method. This is due to significantly decreasing the costs of constructing the Jacobian.

2.3.3 Shamanskii Method

Newton's method and the chord method can be shown to be special cases of broader class of higher-order nonlinear methods referred to as Shamanskii methods (53). In a Shamanskii method, the Jacobian is reconstructed every m iterations. This method is a higher-order method which converges q-superlinearly to the solution with q-order $m+1$, meaning that the inequality,

$$\|e^{k+1}\| \leq K_S \|e^k\|^{m+1} \tag{2.38}$$

is satisfied (27; 7). The algorithm for the Shamanskii Method is shown below.

Algorithm 4 Shamanskii Method

for $k = 1 \rightarrow K$ **do**
    construct $J(\boldsymbol{x}^k)$
    for $j = 1 \rightarrow m$ **do**
        $\delta \boldsymbol{x}^j = J^{-1}(\boldsymbol{x}^k)F(\boldsymbol{x}^j)$
        $\boldsymbol{x}^{j+1} = \boldsymbol{x}^j + \delta \boldsymbol{x}^j$
        Check Convergence
    end for
    $\boldsymbol{x}^{k+1} = \boldsymbol{x}^{j+1}$
    Check Convergence
end for

It can be seen that for the case of $m = 1$, the Shamanskii method will reduce to the Newton method, with the Jacobian reconstructed every iteration, and it reduces to the chord method for $m = \infty$ where the Jacobian will never be reconstructed. The main benefit of the Shamanskii method is that it offers a greater flexibility in determining an optimum balance point between the increased cost of solving the matrix system for a larger number of iterations and the cost of reconstructing the Jacobian matrix. The selection of the m value for the Shamanskii method will be based on the trade-off in computational cost between reconstructing the Jacobian matrix more often with the decreased convergence rate leading to an increase in the number of iterations required.

2.3.4 Further Considerations on Nonlinear Methods

In the discussion of further options from Newton's method, the chord and Shamanskii methods were introduced as methods that do not recalculate the Jacobian every iteration as a way to improve performance of Newton's method. In this subsection, methods which aim to improve the efficiency of Newton's method via other means: Quasi-Newton methods, Jacobian-free methods, and inexact Newton methods, are introduced.

Quasi-Newton Methods

Quasi-Newton methods are a class of nonlinear methods which are similar to Newton's method except an approximation of the current solution and of the Jacobian matrix are stored at each iteration, denoted x^c and B_c, instead of the true Jacobian matrix. These

approximated terms are used to calculate an updated solution according to

$$x_t = x_c - B_c(x_c)^{-1}F(x_c), \tag{2.39}$$

where x^t is the updated estimate of the solution vector and $F(x_c)$ is the residual vector of the previous approximation of the solution vector.

Broyden's method (8) and the Broyden family of methods are the most widely used examples of quasi-Newton methods (3). Broyden's method and is an example of a secant update and works by calculating an updated approximation to the Jacobian via

$$B_t(x) = B_c(x) + \frac{F(x^t)\delta x_c^T}{\delta x_c^T \delta x_c}, \tag{2.40}$$

where $B_t(x)$ is the update to the approximate Jacobian matrix, δx_c is the update to the approximation to the solution and the superscript T denotes the transpose of a vector. For a general iterative Quasi-Newton method it can be shown the method converges q-superlinearly to the solution, x^*, if the Dennis-Moré Condition is satisfied (16). The Dennis-Moré Condition is,

$$\lim_{n\to\infty} \frac{\|(B_c^n(x) - F'(x^*))\delta x_c^n\|}{\|\delta x_c^n\|} = 0 \tag{2.41}$$

Quasi-Newton methods can be desirable to implement as they reduce the cost associated with the calculation of the Jacobian matrix, with the performance of the method being able to approximate that of the chord method on a per iteration basis, while also exhibiting q-superlinear convergence (27).

Inexact Newton Methods

Inexact Newton methods are nonlinear methods based on Newton's method in which the solution of the system,

$$\delta x = J^{-1}(x)F(x), \tag{2.42}$$

is done approximately, frequently through an iterative method such as a Krylov subspace solver. The inequality,

$$|J(x)\delta x + F(x)| \leq \eta |F(x)|, \tag{2.43}$$

is used to check for a sufficient reduction in the norm of the residual vector. The value η is referred to as the forcing factor and is a value between zero and one. Equation 2.43

is convenient as this condition is similar to those present in iterative solvers that can be used to solve the linear step in Newton's method. It has been shown that inexact Newton methods will have q-linear convergence to the root, x^*, when used. If the sequence of forcing terms used in the inexact Newton method approached zero, the convergence has been shown to be q-superlinear (15; 27).

Krylov methods, which are frequently used in the solution of inexact newton methods, are methods which solve the linear system

$$Ax = b \tag{2.44}$$

with an iterative method. This method works to minimize the error in the space described by

$$x_0 + \kappa_k, \tag{2.45}$$

where x_0 is the initial iterate and κ_k is the k^{th} Krylov subspace defined as

$$\kappa_k = span(r_0, Ar_0, ..., A^k r_0). \tag{2.46}$$

Here, r_0 is the linear residual defined by

$$r_0 = b - Ax_0. \tag{2.47}$$

Krylov methods include those such as the Generalized Minimal Residual (GMRES) method, the Conjugate Gradient (CG) method, Bi-Conjugate Gradient Stabilized (BiCGSTAB), Transpose-Free Quasi-Minimal Residual (TFQMR) (18; 58; 24; 51; 27).

Jacobian-Free Methods

After discussing Quasi-Newton and inexact Newton methods, it is important to introduce Jacobian-free Newton Krylov (JFNK) methods, which do not use the Jacobian matrix directly in the solution. JFNK methods function by using a Newton method-like nonlinear iteration which contains a linear iteration over the Krylov subspace. The advantage of JFNK methods comes as a result of the fact the Krylov methods rely on the action of the matrix vector product in the Krylov subspace defined in Equation 2.46 as opposed to the individual elements in the matrix A. JFNK methods offer a significant improvement to the computational time of nonlinear solvers. For a more detailed discussion of JFNK methods, see (28).

2.4 Numerical Iteration Strategies and Time Discretization

There are many methods for the iteration strategy and time discretization used in thermal-hydraulic codes, a few of which will be discussed in this section. The decision between the methods presented below comes down to a variety of choices, the main of which is the time scale of physical phenomena that is being observed compared to the stability criterion of the method (1). A few other consideration are the difficulty of implementing the method, how well the method matches up with the governing equations and any existing structures within the code, and the efficiency and performance of the method. Most existing thermal-hydraulic codes implement either the semi-implicit method or the Stability Enhancing Two Step (SETS) method (6; 43; 52)

2.4.1 Explicit Method

The method that is the most efficient for a single timestep is the fully explicit method. The fully explicit method takes all variables in the conservation equations at the current timestep, n, with the exception of the next timestep value in the rate of change term. The algorithm for the fully explicit method is shown below.

Algorithm 5 Fully Explicit Method

1: **for** $n = 1 \rightarrow N$ **do**
2: $t^{n+1} = t^n + \delta t$
3: calculate $J(\boldsymbol{x}^n)$ and $F(\boldsymbol{x}^n)$
4: $\delta \boldsymbol{x}^n = -J^{-1}(\boldsymbol{x}^n)F(\boldsymbol{x}^n)$
5: $\boldsymbol{x}^{n+1} = \boldsymbol{x}^n + \delta \boldsymbol{x}^n$
6: **end for**

The fully explicit method is very efficient on a timestep basis. This is mainly due to the fact that only one next timestep variable is present in each conservation equation. This allows the next timestep values to be evaluated independently of each other. In addition, explicit methods causes little numerical diffusion. However, explicit methods introduce timestep stability criterion in the form of Courant-Friedrichs-Lewy (CFL) limits (1; 10; 34).

For the fully explicit method, the CFL limit is the sonic courant limit which is

$$\frac{\Delta t(|\vec{V}|+c)}{\Delta x} \leq 1. \tag{2.48}$$

In the sonic courant limit, c is the speed of sound in the fluid and $|\vec{V}|$ is the magnitude of the fluid velocity. The fully explicit method is very useful in tracking the behavior of pressure wave propagation (43). However, in most simulations of interest in nuclear thermal-hydraulics, the fluid velocity is orders of magnitude smaller than the speed of sound in the fluid so the explicit method leads to extremely short timesteps being required. This means that while the explicit method is the most efficient on a per-timestep basis, the timestep stability criterion leads to timesteps that are short enough that the fully explicit method is significantly less efficient than other methods overall. For nuclear thermal-hydraulics this means that a more implicit method is needed.

2.4.2 Semi-Implicit Method

There are a variety of methods that can be described as semi-implicit methods, but commonly it refers to methods similar to the Newton Block Gauss Seidel (NBGS) method discussed in (35). In this method, the velocity terms in the mass and energy convection terms as well as the pressure gradient in the momentum equation and the compressible work term in the energy equation are evaluated at the next timestep. This allows for the system of equations to be solved as a linear system in a single iteration, as conservation equations are cast in terms of a variable at the next timestep as a function of current timestep variables and next timestep values of other variables. While this process is less efficient per timestep than the fully explicit method, it is much more efficient both in runtime per timestep and memory requirements than the more implicit methods as it does not require a nonlinear iteration (35; 36).

Algorithm 6 Semi-Implicit Method

1: **for** $n = 1 \rightarrow N$ **do**
2: $t^{n+1} = t^n + \delta t$
3: calculate $J(\boldsymbol{x}^n, \boldsymbol{x}^{n+1})$ and $F(\boldsymbol{x}^n, \boldsymbol{x}^{n+1})$
4: $\delta \boldsymbol{x}^n = -J^{-1}(\boldsymbol{x}^n, \boldsymbol{x}^{n+1}) F(\boldsymbol{x}^n, \boldsymbol{x}^{n+1})$
5: $\boldsymbol{x}^{n+1} = \boldsymbol{x}^n + \delta \boldsymbol{x}^n$
6: **end for**

The main benefit of the semi-implicit method over the fully explicit method is the fact that the CFL limit that exists is the material courant limit and not the sonic courant limit from Equation 2.48. The material courant limit is similar to the sonic courant limit however the speed of sound effect is removed as shown in Equation 2.49 (43; 35; 36). For modeling flows where the flow velocity is significantly slower than the speed of sound in the material the semi-implicit method allows for significantly longer timesteps than the fully explicit method.

$$\frac{\Delta t |\vec{V}|}{\Delta x} \leq 1 \tag{2.49}$$

There are variations of the semi-implicit method that take the same form of the finite difference equations with the same level if implicitness. However the solution method implemented is different. An widely-used example of this is the implicit continuous-fluid Eulerian (ICE) method, which was originally developed as a numerical method intended to be capable of modeling flows of any mach number (22). Currently, the method implemented in CTF-R is based upon the semi-implicit method discussed here.

2.4.3 SETS Method

The ability of the semi-implicit method discussed above to soften the timestep restriction of the CFL limit makes it widely useful in nuclear simulations. However, in some cases it can be useful to have timesteps that are not restricted by convective timescales in the way the semi-implicit method is. The stability-enhancing two-step (SETS) method was developed as an extension of the semi-implicit method, specifically ICE, to remove the material courant timestep limit without needing the computational expense of fully implicit or alternating-direction implicit (ADI) (46) schemes (37).

The SETS method uses a combination of solution of the basic governing equations with solution of the stabilizer equations. The stabilizer equations provide info on how mass,

momentum, and energy are transported across cell boundaries, which allows for forms of the convective divergence operator that remove the material courant limit. The ordering of the stabilized and basic equation solutions can be ordered in many ways and one ordering that was found to be stable for all problems is described here. First, a stabilizer step on the motion equations, which are derived from the non-conservative momentum equations divided by density, is performed. Next the basic equations are solved with a nonlinear iteration using info from the motion stabilizer step, and then a stabilizer step for the mass and energy equations is performed after the nonlinear iteration (37).

The general algorithm of the SETS method is shown below. Subscripts on the Jacobian matrix, J, and the residual vector, F, indicate which equations are included in the Jacobian matrix and residual vector at that step. Stabilizer steps are indicated with an s. The nonlinear iteration index is represent by k and the maximum number of nonlinear iterations is given by K. The nonlinear iteration is assumed to be a Newton iteration.

Algorithm 7 Stability-Enhancing Two-Step Method

for $n = 1 \rightarrow N$ **do**
  $t^{n+1} = t^n + \delta t$
  calculate $J_{s,mom}(\boldsymbol{x}^n)$ and $F_{s,mom}(\boldsymbol{x}^n)$
  $\delta \vec{V}^* = -J_{s,mom}^{-1}(\boldsymbol{x}^n) F_{s,mom}(\boldsymbol{x}^n)$
  $\vec{V}^* = \vec{V}^n + \delta \vec{V}^*$
  for $k = 1 \rightarrow K$ **do**
    calculate $J(\boldsymbol{x}^n, \vec{V}^*)$ and $F(\boldsymbol{x}^n, \vec{V}^*)$
    $\delta \tilde{\boldsymbol{x}}^{n+1} = -J^{-1}(\boldsymbol{x}^n, \vec{V}^*) F(\boldsymbol{x}^n, \vec{V}^*)$
    $\tilde{\boldsymbol{x}}^{n+1} = \boldsymbol{x}^n + \delta \tilde{\boldsymbol{x}}^{n+1}$
    Check Convergence
  end for
  calculate $\boldsymbol{x}^{n+1}$ from $\tilde{\boldsymbol{x}}^{n+1}$
end for

The nonlinear iteration in the SETS method is designed to be solved in much the same way as the semi-implicit method. This allows codes that use the semi-implicit method to be adjusted to use the SETS method within this framework. It has been shown that while the SETS method does remove the material courant limit in the source-less equations, it does not exhibit the stability of the fully implicit method. Also, as the timestep size of the

SETS method decreases, the numerical diffusion of the method approaches that of the fully implicit method (38).

2.4.4 Implicit Method

The fully implicit method involves setting all the terms in the governing equations to the next timestep value, except for the second term in the rate of change terms. The algorithm for the fully implicit method is shown below. The nonlinear iteration index is given by k. The solution vector at a nonlinear iteration step, $\boldsymbol{x}^k$ represents the current guess of the solution vector at the next timestep, $\boldsymbol{x}^{n+1}$.

Algorithm 8 Fully Implicit Method

for $n = 1 \rightarrow N$ **do**
  $t^{n+1} = t^n + \delta t$
  $\boldsymbol{x}^k = \boldsymbol{x}^n$
  for $k = 1 \rightarrow K$ **do**
    calculate $J(\boldsymbol{x}^k, , \boldsymbol{x}^n)$ and $F(\boldsymbol{x}^k, \boldsymbol{x}^n)$
    $\delta \boldsymbol{x}^{k+1} = -J^{-1}(\boldsymbol{x}^k, \boldsymbol{x}^n) F(\boldsymbol{x}^k, \boldsymbol{x}^n)$
    $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \delta \boldsymbol{x}^{k+1}$
    Check Convergence
  end for
  $\boldsymbol{x}^{n+1} = \boldsymbol{x}^{k+1}$
end for

The implicit method does not have any stability-based restrictions on the timestep length, which makes it an attractive method to use. However, the implicit method has a higher amount of diffusion that is introduced into the solution than the methods discussed previously. Also, the implicit method requires a nonlinear iteration which is costly. Also, unlike the SETS method which has a nonlinear iteration that solves the same equations as the semi-implicit method, the implicit method requires solution of a different set of equations. Due to the high development effort to implement the implicit method in an existing code, and the computational cost of the method, the implicit method is rarely used in nuclear thermal-hydraulic codes.

2.5 Discretization of the Advective Derivative

An important part of the numerical solution of the Euler equations as presented in section 2.2 is the discretization of the advective derivative. The advective derivative is the derivative of the form

$$\nabla \cdot (\phi \vec{V}), \tag{2.50}$$

where ϕ is some property of the fluid, such as enthalpy, density, or momentum, that is advected by the fluid velocity. In one-dimension, the advective derivative of a property ϕ, which will be used in this section as a demonstrative case, is

$$\frac{\partial \phi u}{\partial x}. \tag{2.51}$$

Discretizing this equation about a staggered grid yields

$$\frac{\phi_{i+\frac{1}{2}} u_{i+\frac{1}{2}} - \phi_{i-\frac{1}{2}} u_{i-\frac{1}{2}}}{\Delta x}. \tag{2.52}$$

For a staggered grid, the use of which is discussed in (44), the velocity is stored at the cell face values, $(i \pm \frac{1}{2})$ but the density, enthalpy, and pressure are stored at cell center values. Due to this, a method for determining the value of these variables at cell faces is needed.

At first glance, the choice for discretization of this derivative would be to evaluate it by central differencing,

$$\dot{\phi}_{i+\frac{1}{2}} = \frac{\phi_i + \phi_{i+1}}{2}. \tag{2.53}$$

However, it has been shown that this differencing scheme is unstable in cases where convective terms are more significant than diffusive terms and allows for highly nonphysical results (44; 14). In terms of the Euler equation, the diffusive terms such as friction and conduction are neglected, meaning a central difference method as shown here will be unconditionally unstable.

2.5.1 First-Order Upwinding

To remedy this stability issue caused by central-differencing, first-order upwinding is widely-used. First-order upwinding accounts for the highly-directional behavior of convection, taking the variable ϕ from the direction from which the flow is coming. This is done ac-

cording to

$$\dot{\phi}_{i+\frac{1}{2}} = \begin{cases} \phi_i & u_{i+\frac{1}{2}} > 0 \\ \phi_{i+1} & u_{i+\frac{1}{2}} \leq 0, \end{cases} \tag{2.54}$$

where if the velocity is positive the advected variable is taken from the backwards cell-center and if the velocity is negative the advected variable is taken from the forwards cell-center. The advected variable is taken in these cases from the "upwind" direction in the fluid, thus how the method is named.

The use of first-order upwinding greatly increases the stability of convection-based equations. However, first-order upwinding is known to be highly artificially diffusive. For this reason, alternative options have been heavily researched. A class of exponential-based differencing schemes are frequently used to improve the behavior of first-order upwinding (44). These schemes utilize a mixture of central differencing and upwinding depending on the Peclet number, which is the ratio of the convective terms to the diffusive terms in a system. However, there are issues with these exponential based schemes, mainly in cases where the underlying assumptions of single-dimensional steady-state flow are violated (33). Also, for the Euler equations, where the diffusive terms are neglected, these schemes will always reduce to first-order upwinding and will not improve the diffusivity of the method.

2.5.2 High-Order Schemes

To improve upon the diffusivity of the first-order upwinding schemes without the issues and assumptions present in the Peclet number based exponential schemes mentioned, high order (HO) upwind methods were proposed. These methods implement second-or-higher order methods to the advected variable upwinding process. One of the simplest HO methods is the second-order upwind scheme,

$$\phi_{i+\frac{1}{2}} = \begin{cases} \frac{3}{2}\phi_i - \frac{1}{2}\phi_{i-1} & u_{i+\frac{1}{2}} > 0 \\ \frac{3}{2}\phi_{i+1} - \frac{1}{2}\phi_{i+2} & u_{i+\frac{1}{2}} \leq 0. \end{cases} \tag{2.55}$$

In this scheme, a linear profile is fitted between the upwind value used in Equation 2.54 and the "far-upwind" node which is the cell center value for the advected variable one node further upwind from the upwind node (54). Other HO schemes introduced here are the scheme introduced by Fromm in (19) and the Quadratic Upstream Interpolation for

Convective Kinematics (32).

Fromm's scheme,

$$\phi_{i+\frac{1}{2}} = \begin{cases} \phi_i + \frac{\phi_{i+1}-\phi_{i-1}}{4} & u_{i+\frac{1}{2}} > 0 \\ \phi_{i+1} + \frac{\phi_i-\phi_{i+2}}{4} & u_{i+\frac{1}{2}} \leq 0, \end{cases} \tag{2.56}$$

performs a linear interpolation similar to the SOU scheme, however it fits between the far upwind node and the downwind node.

The quadratic upstream interpolation for convective kinematics (QUICK) scheme,

$$\phi_{i+\frac{1}{2}} = \begin{cases} \frac{3}{4}\phi_i + \frac{3}{8}\phi_{i+1} - \frac{1}{8}\phi_{i-1} & u_{i+\frac{1}{2}} > 0 \\ \frac{3}{4}\phi_{i+1} + \frac{3}{8}\phi_i - \frac{1}{8}\phi_{i+2} & u_{i+\frac{1}{2}} \leq 0, \end{cases} \tag{2.57}$$

is a third-order method based on interpolating the value of ϕ based on a quadratic polynomial that is biased towards the upstream direction (32).

While these methods have been shown to be significantly less-diffusive than the first-order upwind scheme, they are not as stable as first-order upwinding. In addition, these schemes can all be shown to except exhibit overshooting or undershooting of the desired result in cases with convective-dominated flow and can exhibit oscillatory behavior. For these reasons using HO methods as described in this section is not done widely (14; 1; 33).

2.5.3 High-Resolution Schemes

To fix the stability/unboundedness issues and the associated oscillatory and overshoot/undershoot problems exhibited by HO schemes, high-resolution (HR) upwind schemes are used. HR schemes rely on a HO scheme implementation along with a convection boundedness criterion (CBC),

$$min(\phi_i, \phi_{i+1}) \leq \phi_{i+\frac{1}{2}} \leq max(\phi_i, \phi_{i+1}). \tag{2.58}$$

The CBC acts to limit the advection of the variable of interest across boundaries which acts to preserve monotonicity. In effect, the CBC prevents artificial creation of extrema, which greatly improves the stability issues exhibited by HO schemes. This methodology is frequently referred to as flux limiting.

The derivation of HR schemes typically relies on the idea of a normalized variable

formulation (NVF) for the variable ϕ which is defined as $\tilde{\phi}$ which is defined as

$$\tilde{\phi} = \begin{cases} \frac{\phi - \phi_{i-1}}{\phi_{i+1} - \phi_{i-1}} & u_{i+\frac{1}{2}} > 0 \\ \frac{\phi - \phi_{i+1}}{\phi_{i-1} - \phi_{i+1}} & u_{i+\frac{1}{2}} \leq 0. \end{cases} \tag{2.59}$$

The NVF of ϕ uses the downwind and far upwind nodes to determine a normalized value.

This method allows the face values of ϕ to be written solely in terms of the NVF form of the upwind node (14; 13). The general form of this relation is,

$$\tilde{\phi}_{i+\frac{1}{2}} = \begin{cases} f(\tilde{\phi}_i) & u_{i+\frac{1}{2}} > 0 \\ f(\tilde{\phi}_{i+1}) & u_{i+\frac{1}{2}} \leq 0 \end{cases}. \tag{2.60}$$

The methods introduced previously: first-order upwinding, second-order upwinding, Fromm's scheme, and the QUICK scheme, are shown below in the NVF form. Note the velocity is assumed positive for each. If the velocity were negative, the NVF at node i, $\bar{\phi}_i$ would be changed to the NVF at node $i+1$, $\bar{\phi}_{i+1}$.

First-order upwinding:

$$\bar{\phi}_{i+\frac{1}{2}} = \bar{\phi}_i \tag{2.61}$$

Second-order upwinding:

$$\bar{\phi}_{i+\frac{1}{2}} = \frac{3}{2}\bar{\phi}_i \tag{2.62}$$

Fromm's scheme:

$$\bar{\phi}_{i+\frac{1}{2}} = \bar{\phi}_i + \frac{1}{4} \tag{2.63}$$

QUICK scheme:

$$\bar{\phi}_{i+\frac{1}{2}} = \frac{3}{4}\bar{\phi}_i + \frac{3}{8} \tag{2.64}$$

An HR scheme based on the QUICK scheme introduced previously which is referred to as the modified SMART scheme (14; 13), is

$$\tilde{\phi}_{i+\frac{1}{2}} = \begin{cases} 3\tilde{\phi}_i & 0 < \tilde{\phi}_i < \frac{1}{6} \\ \frac{3}{4}\tilde{\phi}_i + \frac{3}{8} & \frac{1}{6} \leq \tilde{\phi}_i < \frac{7}{10} \\ \frac{1}{3}\tilde{\phi}_i + \frac{2}{3} & \frac{7}{10} \leq \tilde{\phi}_i < 1 \\ \tilde{\phi}_i & \textit{elsewhere.} \end{cases} \tag{2.65}$$

The effect of the CBC can be seen in the last term, where first-order upwinding is used anywhere that $\tilde{\phi}_i$ is not between zero and one. This ensures that the CBC is always satisfied and that the resulting method is stable.

It is important to note that HR schemes typically require some method such as the downwind weighting factor (DWF), the variable curvature factor (VCF), or normalized weighting factor (NWF) method to improve the stability of the system. Another option is to implement a deferred correction of the HO or HR model where the difference between the upwind and HO or HR model result is included as a source term (13; 39; 14). The implementation of these methods frequently cause a significant increase in the computational time required and can affect the rate of convergence of the solver. In addition, to increasing the simulation time, it frequently takes significant development work to implement the weighting factor methods required for HR and HO methods. This is why, in spite of the knowledge that first-order upwind schemes and exponential-based schemes have severe flaws, they are still widely-used.

CHAPTER

3

THREE DIMENSIONAL SOLVER IN THE RESIDUAL FORMULATION OF CTF

In this chapter, the development and initial verification of the novel three-dimensional solver in the Residual Formulation of CTF is detailed. The existing sub-channel form of the residual formulation for CTF is presented in section 3.1. The spatial discretization of the new solver and the discretized equations implemented are discussed in section 3.2 and section 3.3. The alterations to the process of constructing the Jacobian matrix to allow for the three-dimensional solver are given in section 3.4. Initial verification of the new solver is performed over a series of test cases that are compared to analytic results in section 3.5.

3.1 Residual Formulation of CTF

The residual formulation of CTF (CTF-R) is a solver that is contained within CTF that started development at Pennsylvania State University and continues development at North

Carolina State University. CTF-R solves the system of fluid governing equations,

$$F(x) = 0, \tag{3.1}$$

via a linearized Newton step in a transient solver. This linear step,

$$\boldsymbol{x}^{n+1} = \boldsymbol{x}^n - J(\boldsymbol{x})F(\boldsymbol{x}^n), \tag{3.2}$$

is used to calculate the next timestep values, $\boldsymbol{x}^{n+1}$ based upon the solution vector at the previous step, $\boldsymbol{x}^n$ and the residual vector of the system of governing equations for the previous step, $F(x^n)$. The solution algorithm in CTF-R is detailed below. This method is similar to the methods implemented in (35; 9).

Algorithm 9 CTF-R Algorithm

for $n = 1 \rightarrow N$ **do**
  while not converged **do**
    $t^{n+1} = t^n + \delta t$
    construct $J(\boldsymbol{x})$
    $\delta \boldsymbol{x}^n = J^{-1}(\boldsymbol{x})F(\boldsymbol{x}^n)$
    if not accepted **then**
      decrease δt
    else
      $\boldsymbol{x}^{n+1} = \boldsymbol{x}^n + \delta \boldsymbol{x}^n$
    end if
  end while
end for

In CTF-R, the timestep for step n is selected following the CTF timestep selection criteria (52), and from that timestep size, the Jacobian matrix and residual vector are constructed. Using the Jacobian matrix and residual vector, the solution update, $\delta \boldsymbol{x}$ for timestep n is determined using PETSc (5). This update is then checked for convergence parameters as in CTF, and if the update is acceptable, the solution is updated and the simulation moves to the next timestep. If the update is not accepted, the timestep size is decreased and the process is repeated until the update is accepted.

The Jacobian matrix, $J(\boldsymbol{x})$, is approximated numerically in CTF-R as

$$J_{i,j}(\boldsymbol{x}) = \frac{\partial F_j(\boldsymbol{x})}{\partial x_i} \approx \frac{F_j(\boldsymbol{x} + \epsilon \vec{e}_i) - F_j(\boldsymbol{x})}{\epsilon}, \tag{3.3}$$

where ϵ is a small perturbation parameter, currently taken to be 0.001, and $\vec{e}_i$ is the i^{th} unit vector.

The mesh in CTF-R is setup, similar to CTF, into a series of axial channels in the x-direction connected by gaps between channels in the "lateral" directions. The axial nodes are indexed over i and the gaps on node i in a channel is indexed over g. CTF-R uses a staggered grid, meaning the velocity and momentum equation are staggered between nodes of the mass and energy equations. A discussion of the reason for the mesh being staggered in thermal-hydraulic codes is given in Chapter 6 of (44). The figures below show the axial and lateral mesh discretization for CTF-R. A further description of the mesh in CTF-R can be found in (48) and the CTF mesh is further detailed in (52).

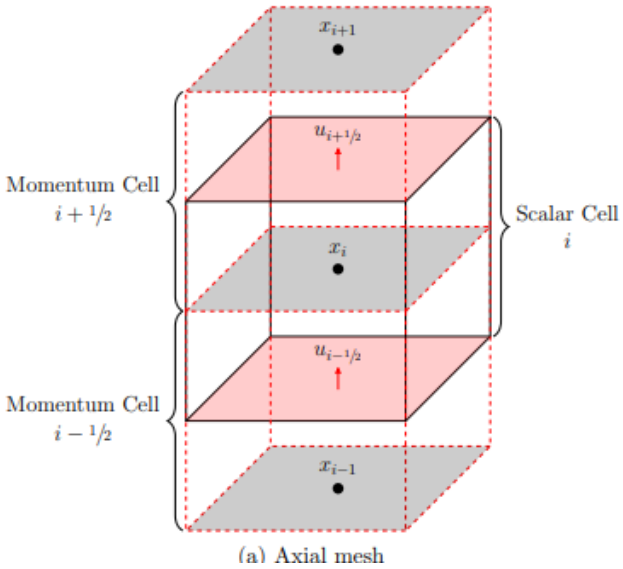
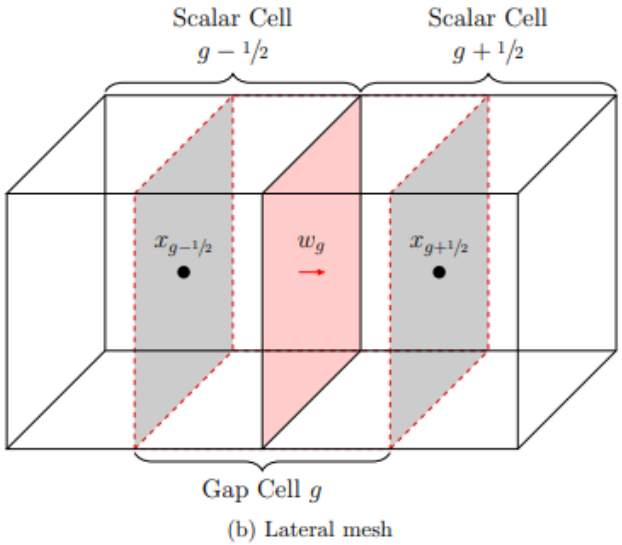


Figure 3.1: CTF-R Axial Mesh



(b) Lateral mesh

Figure 3.2: CTF-R Lateral Mesh

CTF-R is currently capable of simulating single-phase simulations with 4 conservation equations, the continuity equation, an axial momentum equation, a lateral momentum equation, and an energy equation in the form of an enthalpy equation. A more detailed discussion of these equations is given in (48).

The continuity, or mass, equation in CTF-R is

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} + \frac{\partial \rho w}{\partial z} - \kappa_1 \frac{\partial^2 \rho}{\partial z^2} = 0. \tag{3.4}$$

The terms in this equation are the rate of change of density, the axial advection of mass, the lateral advection of mass, and an artificial diffusion term meant for smoothing density in the lateral direction. This artificial diffusion term was discussed originally in (30) and discussed further in (29) to improve the stability of methods for solution of certain types of partial differential equations (PDEs). It has been referred in many ways, as a viscosity term, a surface tension term (49), and as a turbulent mixing term (52). Effectively, this term is used as a method to handle instabilities in the equations when discontinuities occur (31). κ_1 is a tuning parameter used to ensure stability with the diffusion operator.

The discretized form of the continuity equation in CTF-R is

$$\frac{\rho_i^{n+1}-\rho_i^n}{\Delta t}+\frac{\dot{\rho}_{i+\frac{1}{2}}^n u_{i+\frac{1}{2}}^{n+1}-\dot{\rho}_{i-\frac{1}{2}}^n u_{i-\frac{1}{2}}^{n+1}}{\Delta x}+\sum_g^{N_g}\left[\frac{\dot{\rho}_g w_g}{L_g}-\kappa_1\frac{\rho_{g+\frac{1}{2}}^n-\rho_{g-\frac{1}{2}}^n}{L_g^2}\right]=0. \tag{3.5}$$

In the mass advection term, the dotted density values are first-order upwinded on the staggered spatial mesh according to

$$\dot{\rho}_{i+\frac{1}{2}}^n=\begin{cases}\rho_i^n & u_{i+\frac{1}{2}}^n>0\\ \rho_{i+1}^n & u_{i+\frac{1}{2}}^n\leq 0.\end{cases} \tag{3.6}$$

This is done to improve the stability of the system, introducing the Courant-Friedrichs-Lewy (CFL) condition for ensuring stability of the explicitly time-differentiated scheme (10). The CFL condition, referred to as the material Courant limit, that this implies is

$$\left|\frac{u_0\Delta t}{\Delta x}\right|\leq 1, \tag{3.7}$$

where u_0 is a given velocity profile. Note that the lateral terms are summed over all of the lateral gaps to the current cell. Also, the velocity in the advection term is taken at the next timestep. This is done to remove the classic stability criterion relating to the speed of sound and to allow the maximum level of implicitness while allowing for a system of equations that can be rearranged and solved linearly in a single step (35).

The axial momentum equation implemented in CTF-R is

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+\frac{1}{\rho}\frac{\partial P}{\partial x}+\frac{1}{\rho}\frac{\partial \rho u w}{\partial z}-\frac{\kappa_3}{\rho}\frac{\partial^2 u}{\partial z^2}=g+S_f. \tag{3.8}$$

The terms in this axial momentum equation are the change in momentum in time, the advection of momentum axially, the pressure gradient force, the advection of axial momentum laterally, and an artificial momentum diffusion term similar to the one described above for the mass equation. The RHS source terms are gravity effects and any other body forces. Note that this equation is a form of the non-conservative form of the momentum equation, where the equation has been divided by a density term. A detailed derivation of the non-conservative form is shown in Section 2.2.2.

The discretized form of the axial momentum equation in CTF-R is

$$\frac{u^{n+1}_{i+\frac{1}{2}}-u^{n}_{i+\frac{1}{2}}}{\Delta t}+u^{n}_{i+\frac{1}{2}}\frac{\dot{\partial u}}{\partial x}\bigg|^{n+1}_{i+\frac{1}{2}}+\frac{1}{\bar{\rho}^{n}_{i+\frac{1}{2}}}\frac{P^{n+1}_{i+1}-P^{n+1}_{i}}{\Delta x}+\frac{1}{\bar{\rho}^{n}_{i+\frac{1}{2}}}\sum_{g}^{N_g}\left[\frac{(\dot{\rho u})^{n}_{g}w^{n}_{g}}{L_g}-\kappa_3\frac{u^{n}_{g+\frac{1}{2}}-u^{n}_{g-\frac{1}{2}}}{L^2_g}\right]=g+S_f. \tag{3.9}$$

The dotted velocity derivative in the axial advection term and the dotted density and velocity in the advection of axial momentum laterally are first-order upwinded in space, similar to the discussion on the continuity equation, according to

$$\frac{\dot{\partial u}}{\partial x}\bigg|^{n+1}_{i+\frac{1}{2}}=\begin{cases}\frac{u^{n}_{i+\frac{1}{2}}-u^{n}_{i-\frac{1}{2}}}{\Delta x} & u^{n}_{i+\frac{1}{2}}>0\\ \frac{u^{n}_{i+\frac{3}{2}}-u^{n}_{i+\frac{1}{2}}}{\Delta x} & u^{n}_{i+\frac{1}{2}}\leq 0,\end{cases} \tag{3.10}$$

$$(\dot{\rho u})^{n}_{g}=\begin{cases}(\rho u)^{n}_{g-\frac{1}{2}} & w^{n}_{g}>0\\ (\rho u)^{n}_{g+\frac{1}{2}} & w^{n}_{g}\leq 0.\end{cases} \tag{3.11}$$

Similar to the velocity terms in the mass advection term in the continuity equation, the pressure gradient term is taken at the next time step value (35).

The momentum cell densities are not currently averaged in CTF-R, $\bar{\rho}^{n}_{i+\frac{1}{2}}=\rho^{n}_{i}$. The density terms in the momentum cell gaps are averaged $\rho_{g+\frac{1}{2}}{}^{n}=\frac{\rho^{n}_{i,g+\frac{1}{2}}+\rho^{n}_{i+1,g+\frac{1}{2}}}{2}$.

The lateral momentum equation in CTF-R is

$$\frac{\partial w}{\partial t}+\frac{1}{\rho}\frac{\partial P}{\partial z}=H|w|w. \tag{3.12}$$

The change in the lateral velocity in gaps is caused by lateral pressure gradients between subchannels. There is a tunable form loss in the model implemented to model the advection and lateral friction in the lateral directions. The term H is a tunable parameter for the form loss.

The lateral momentum equation in CTF-R is discretized according to

$$\frac{w^{n+1}_{g}-w^{n}_{g}}{\Delta t}+\frac{1}{\rho^{n}_{g}}\frac{P^{n+1}_{g+\frac{1}{2}}-P^{n+1}_{g-\frac{1}{2}}}{L_g}=H|w^{n}_{g}|w^{n}_{g}. \tag{3.13}$$

The gap density is an arithmetic average of the channel density in the neighboring scalar

cells $\bar{\rho}_g^n = \frac{\rho^n_{g-\frac{1}{2}} + \rho^n_{g+\frac{1}{2}}}{2}$.

The energy equation in CTF-R is in enthalpy form and is

$$\rho \frac{\partial h}{\partial t} + \rho u \frac{\partial h}{\partial x} - \frac{\partial P}{\partial t} + \frac{\partial \rho h w}{\partial z} - \kappa_2 \frac{\partial^2 h}{\partial z^2} = \frac{S_{ht}}{V}. \tag{3.14}$$

The terms in this equation are the rate of change in the enthalpy, the axial advection of the enthalpy, the pressure work term discussed more in Section 2.2.2, the lateral enthalpy advection, and an artificial diffusion term of a similar form to that discussed in the mass and axial momentum equations. The source term on the RHS is the heat transfer from solids to the fluid. This is the non-conservative form of the enthalpy equation, and also omits a pressure gradient term that is deemed negligible (48).

The discretized form of the enthalpy equation in CTF-R is

$$\rho_i^n \frac{h_i^{n+1} - h_i^n}{\Delta t} + \rho_i^n \bar{u}_i^{n+1} \dot{\frac{\partial h}{\partial x}}_i^n - \frac{P_i^{n+1} - P_i^n}{\Delta t} + \sum_g^{N_g} \left[\frac{(\dot{\rho h})_g^n w_g^n}{L_g} - \kappa_2 \frac{h^n_{g+\frac{1}{2}} - h^n_{g-\frac{1}{2}}}{L_g^2} \right] = \frac{S_{ht}}{V}. \tag{3.15}$$

The dotted velocity derivative and the dotted density and enthalpy product are first-order upwinded in space according to

$$\dot{\frac{\partial h}{\partial x}}_i^n = \begin{cases} \frac{h_i^n - h_{i-1}^n}{\Delta x} & \bar{u}_i^n > 0 \\ \frac{h_{i+1}^n - h_i^n}{\Delta x} & \bar{u}_i^n \leq 0, \end{cases} \tag{3.16}$$

$$(\dot{\rho h})_g^n = \begin{cases} (\rho h)^n_{g-\frac{1}{2}} & w_g^n > 0 \\ (\rho h)^n_{g+\frac{1}{2}} & w_g^n \leq 0. \end{cases} \tag{3.17}$$

The barred velocity term is averaged in space and is made implicit by

$$\bar{u}_i^n = \frac{u^n_{i+\frac{1}{2}} + u^n_{i-\frac{1}{2}}}{2}. \tag{3.18}$$

3.2 Spatial Discretization

To Implement a three-dimensional solver in the residual formulation of CTF detailed in section 3.1, many changes to the code were needed. A main adjustment was the change from

the channel-based mesh in CTF that indexes in the axial and lateral directions to a mesh that is a general three-dimensional Cartesian mesh. In this section the spatial discretization in the three-dimensional form of CTF-R is discussed, as well as the alterations to the data structures in CTF-R that were required for the development of this new solver.

For the mesh in the three-dimensional form of CTF-R, a staggered grid will be used for the momentum equations and variables. The reason for this is discussed in Chapter 6 of (44). In the staggered grid, the scalar equations (continuity and enthalpy) and variables (density, pressure, and enthalpy) are stored at cell center values, (i, j, k). The momentum values are stored on cell faces, $(i+\frac{1}{2}, j, k)$ for the x-momentum equation and velocity, $(i, j+\frac{1}{2}, k)$ for the y-momentum equation and velocity, and $(i, j, k+\frac{1}{2})$ for the z-momentum equation and velocity. This means that the there must be four separate grids in a three-dimensional solver in CTF-R, one for the scalar equations and one each for the three momentum equations. The mesh for the three-dimensional scalar cell is shown in Figure 3.3. The cell center, r, at (i, j, k) is shown in black. The cell faces in the x-direction, where the u-velocity and x-momentum equation are stored and solved, are shown in green. The cell faces in the y-direction, where the v-velocity and y-momentum equation are stored and solved, are shown in red. The cell faces in the z-direction, where the w-velocity and z-momentum equation are stored and solved, are shown in blue.

The description of the new mesh that is used in the three-dimensional form of CTF-R is detailed in Table 3.1. The mesh is currently restricted to constant mesh spacings in each direction. This means there is only one value of Δx for a problem, one value of Δy, and one value of Δz. The maximum number of cells in each direction, input by the user, is given by N_I, N_J, and N_K for the x-, y-, and z-directions respectively.

Table 3.1: Computational Domain for the Three-dimensional Solver

	X range	Y-range	z-range
Scalar variables (ρ, h, P)	$(1, N_I)$	$(1, N_J)$	$(1, N_K)$
x-Momentum variables (u)	$(0, N_I)$	$(1, N_J)$	$(1, N_K)$
y-Momentum variables (v)	$(1, N_I)$	$(0, N_J)$	$(1, N_K)$
z-Momentum variables (w)	$(1, N_I)$	$(1, N_J)$	$(0, N_K)$

The boundary conditions for this solver will be done through the definition of ghost cells outside the computational domain shown in Table 3.1, similar to how boundaries

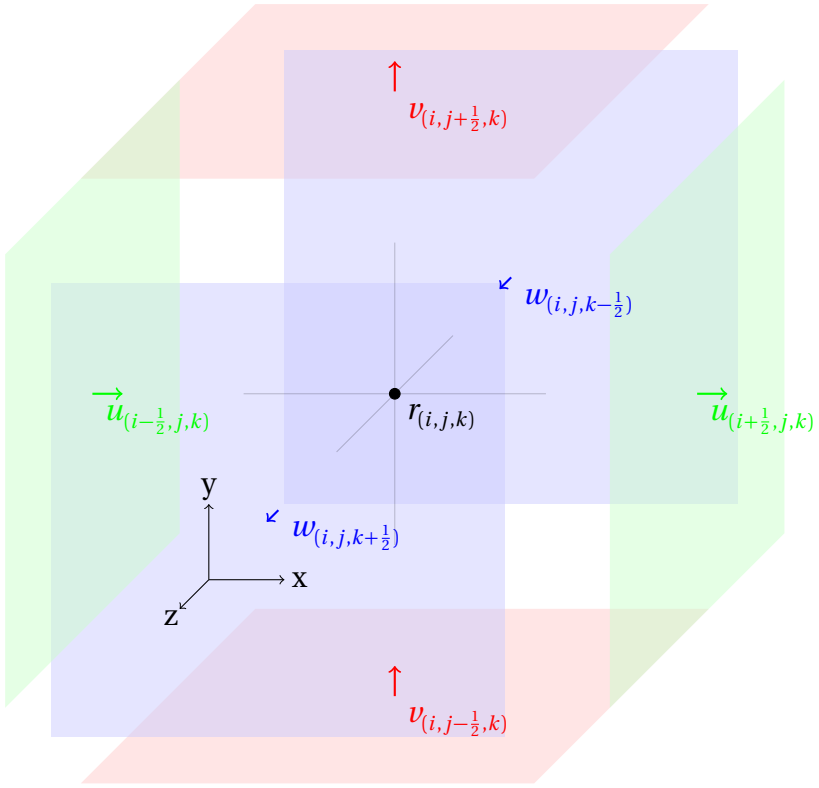


Figure 3.3: Mesh for the Three-Dimensional Cell

are defined in CTF. The ghost cells occur for each variable on each face, and the resulting domain of variables in the system are shown in Table 3.2. The ghost cells simply expand the domain by one scalar and momentum cell in each direction.

Initially, only three types of boundary conditions are implemented. The value of a ghost cell can be set to a value by the user, such as an inlet velocity, a zero gradient condition can be set, and a linear extrapolation from the nearest two cells in the computational domain can be performed.

To implement the three-dimensional discretization process detailed here within CTF-R, new data structures needed to be implemented. The first step was to implement new values in the residual solver for the number of cells in each direction in the computational domain, N_I, N_J, and N_k. Also, the mesh spacing values Δx, Δy, and Δz needed to be defined. These values were setup to be read from input.

The sub-channel form of CTF-R utilizes the same data structures for storing fluid variables as CTF, however this would not be possible in the three-dimensional form of CTF-R.

Table 3.2: Computational Domain and Ghost Cells for the Three-dimensional Solver

	X range	Y-range	z-range
Scalar variables (ρ, h, P)	$(0, N_I+1)$	$(0, N_J+1)$	$(0, N_K+1)$
x-Momentum variables (u)	$(-1, N_I+1)$	$(0, N_J+1)$	$(0, N_K+1)$
y-Momentum variables (v)	$(0, N_I+1)$	$(-1, N_J+1)$	$(0, N_K+1)$
z-Momentum variables (w)	$(0, N_I+1)$	$(0, N_J+1)$	$(-1, N_K+1)$

Therefore, new arrays for storing the current and previous timestep values of the fluid density, enthalpy, pressure, and velocity were implemented. In addition, the object `fluid_state_variables` is used by the sub-channel form of CTF-R to store the local, perturbable variables about the current cell. The values stored in this object related to the sub-channel mesh and thus a new object `fluid_state_3d_variables` was implemented. The new mesh also requires a new definition for the boundary conditions, as well as a new method to input them. To handle this, a set of five arrays of six elements were implemented to handle the boundaries on each surface of the mesh for the enthalpy, pressure, and three velocities.

3.3 Discretized Semi-Implicit Governing Equations

The governing equations in the three-dimensional form of CTF-R are given in this section. These equations are based upon the derivation of the non-conservative governing equations given in section 2.2. The time discretization is semi-implicit as presented in section 2.4, similar to the method for CTF-R which was discussed in section 3.1. The three-dimensional staggered mesh that these equations are discretized over was presented in section 3.2. In this section the time discretization of the source term is not handled. Currently, time-dependent behavior of the source terms is not implemented.

3.3.1 Discretized Continuity Equation

The three-dimensional form of the continuity equation, as presented in Equation 2.4, is

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} + \frac{\partial \rho v}{\partial y} + \frac{\partial \rho w}{\partial z} = S_{mass}. \tag{3.19}$$

Discretizing this equation about the three-dimensional mesh with the semi-implicit method as presented in Section 2.4.2 yields

$$\frac{\rho^{n+1}_{i,j,k}-\rho^{n}_{i,j,k}}{\Delta t^{n}}+\frac{\dot{\rho}^{n}_{i+\frac{1}{2},j,k}u^{n+1}_{i+\frac{1}{2},j,k}-\dot{\rho}^{n}_{i-\frac{1}{2},j,k}u^{n+1}_{i-\frac{1}{2},j,k}}{\Delta x}+\frac{\dot{\rho}^{n}_{i,j+\frac{1}{2},k}v^{n+1}_{i,j+\frac{1}{2},k}-\dot{\rho}^{n}_{i,j-\frac{1}{2},k}v^{n+1}_{i,j-\frac{1}{2},k}}{\Delta y}+$$
$$\frac{\dot{\rho}^{n}_{i,j,k+\frac{1}{2}}w^{n+1}_{i,j,k+\frac{1}{2}}-\dot{\rho}^{n}_{i,j,k-\frac{1}{2}}w^{n+1}_{i,j,k-\frac{1}{2}}}{\Delta z}=(S_{mass})_{i,j,k}. \quad (3.20)$$

The velocity terms are handled implicitly to remove the speed of sound term from the CFL limit as discussed in section 2.4. The dotted density terms are first-order upwinded in space,

$$\rho^{n}_{i+\frac{1}{2},j,k}=\begin{cases}\rho^{n}_{i,j,k} & u^{n+1}_{i+\frac{1}{2},j,k}>0\\ \rho^{n}_{i+1,j,k} & u^{n+1}_{i+\frac{1}{2},j,k}<0\end{cases} \quad (3.21a)$$

$$\rho^{n}_{i-\frac{1}{2},j,k}=\begin{cases}\rho^{n}_{i-1,j,k} & u^{n+1}_{i-\frac{1}{2},j,k}>0\\ \rho^{n}_{i,j,k} & u^{n+1}_{i-\frac{1}{2},j,k}<0\end{cases} \quad (3.21b)$$

$$\rho^{n}_{i,j+\frac{1}{2},k}=\begin{cases}\rho^{n}_{i,j,k} & v^{n+1}_{i,j+\frac{1}{2},k}>0\\ \rho^{n}_{i,j+1,k} & v^{n+1}_{i,j+\frac{1}{2},k}<0\end{cases} \quad (3.21c)$$

$$\rho^{n}_{i,j-\frac{1}{2},k}=\begin{cases}\rho^{n}_{i,j-1,k} & v^{n+1}_{i,j-\frac{1}{2},k}>0\\ \rho^{n}_{i,j,k} & v^{n+1}_{i,j-\frac{1}{2},k}<0\end{cases} \quad (3.21d)$$

$$\rho^{n}_{i,j,k+\frac{1}{2}}=\begin{cases}\rho^{n}_{i,j,k} & w^{n+1}_{i,j,k+\frac{1}{2}}>0\\ \rho^{n}_{i,j,k+1} & w^{n+1}_{i,j,k+\frac{1}{2}}<0\end{cases} \quad (3.21e)$$

$$\rho^{n}_{i,j,k-\frac{1}{2}}=\begin{cases}\rho^{n}_{i,j-1,k} & v^{n+1}_{i,j,k-\frac{1}{2}}>0\\ \rho^{n}_{i,j,k} & v^{n+1}_{i,j,k-\frac{1}{2}}<0\end{cases} \quad (3.21f)$$

3.3.2 Discretized Momentum Equations

The three-dimensional form of the non-conservative momentum equation, as given in Equation 2.18, is

$$\rho\frac{\partial u}{\partial t}+\rho u\frac{\partial u}{\partial x}+\rho v\frac{\partial u}{\partial y}+\rho w\frac{\partial u}{\partial z}+\frac{\partial P}{\partial x}=F_x \tag{3.22a}$$

$$\rho\frac{\partial v}{\partial t}+\rho u\frac{\partial v}{\partial x}+\rho v\frac{\partial v}{\partial y}+\rho w\frac{\partial v}{\partial z}+\frac{\partial P}{\partial y}=F_y \tag{3.22b}$$

$$\rho\frac{\partial w}{\partial t}+\rho u\frac{\partial w}{\partial x}+\rho v\frac{\partial w}{\partial y}+\rho w\frac{\partial w}{\partial z}+\frac{\partial P}{\partial z}=F_z. \tag{3.22c}$$

There is no mass source term implemented in the three-dimensional form of CTF-R yet, and for simplicity the discretization of this term is not included.

X-momentum Equation

The non-conservative momentum equation in the x-direction is

$$\rho\frac{\partial u}{\partial t}+\rho u\frac{\partial u}{\partial x}+\rho v\frac{\partial u}{\partial y}+\rho w\frac{\partial u}{\partial z}+\frac{\partial P}{\partial x}=F_x \tag{3.23}$$

Discretizing this equation yields

$$\begin{aligned}\bar{\rho}^n_{i+\frac{1}{2},j,k}\frac{u^{n+1}_{i+\frac{1}{2},j,k}-u^n_{i+\frac{1}{2},j,k}}{\Delta t^n}+\bar{\rho}^n_{i+\frac{1}{2},j,k}u^n_{i+\frac{1}{2},j,k}\frac{\dot{u}^n_{i+1,j,k}-\dot{u}^n_{i,j,k}}{\Delta x}+\\ \bar{\rho}^n_{i+\frac{1}{2},j,k}\bar{v}^n_{i+\frac{1}{2},j,k}\frac{\dot{u}^n_{i+\frac{1}{2},j+\frac{1}{2},k}-\dot{u}^n_{i+\frac{1}{2},j-\frac{1}{2},k}}{\Delta y}+\bar{\rho}^n_{i+\frac{1}{2},j,k}\bar{w}^n_{i+\frac{1}{2},j,k}\frac{\dot{u}^n_{i+\frac{1}{2},j,k+\frac{1}{2}}-\dot{u}^n_{i+\frac{1}{2},j,k-\frac{1}{2}}}{\Delta z}+\\ \frac{P^{n+1}_{i+1,j,k}-P^{n+1}_{i,j,k}}{\Delta x}=(F_x)_{i+\frac{1}{2},j,k}.\end{aligned} \tag{3.24}$$

The pressure terms are handled implicitly as discussed in section 2.4. The barred density and off-axis velocity terms are arithmetically averaged according to

$$\bar{\rho}^n_{i+\frac{1}{2},j,k}=\frac{\rho^n_{i,j,k}+\rho^n_{i+1,j,k}}{2} \tag{3.25a}$$

$$\bar{v}^n_{i+\frac{1}{2},j,k}=\frac{v^n_{i,j+\frac{1}{2},k}+v^n_{i,j-\frac{1}{2},k}+v^n_{i+1,j+\frac{1}{2},k}+v^n_{i+1,j-\frac{1}{2},k}}{4} \tag{3.25b}$$

$$\bar{w}^n_{i+\frac{1}{2},j,k} = \frac{w^n_{i,j,k+\frac{1}{2}} + w^n_{i,j,k-\frac{1}{2}} + w^n_{i+1,j,k+\frac{1}{2}} + w^n_{i+1,j,k-\frac{1}{2}}}{4}. \tag{3.25c}$$

The dotted velocity terms are first-order upwinded according to

$$\dot{u}^n_{i+1,j,k} - \dot{u}^n_{i,j,k} = \begin{cases} u^n_{i+\frac{1}{2},j,k} - u^n_{i-\frac{1}{2},j,k} & u^n_{i+\frac{1}{2},j,k} > 0 \\ u^n_{i+\frac{3}{2},j,k} - u^n_{i+\frac{1}{2},j,k} & u^n_{i+\frac{1}{2},j,k} < 0 \end{cases} \tag{3.26a}$$

$$\dot{u}^n_{i+\frac{1}{2},j+\frac{1}{2},k} - \dot{u}^n_{i+\frac{1}{2},j-\frac{1}{2},k} = \begin{cases} u^n_{i+\frac{1}{2},j,k} - u^n_{i+\frac{1}{2},j-1,k} & \bar{v}^n_{i+\frac{1}{2},j,k} > 0 \\ u^n_{i+\frac{1}{2},j+1,k} - u^n_{i+\frac{1}{2},j,k} & \bar{v}^n_{i+\frac{1}{2},j,k} < 0 \end{cases} \tag{3.26b}$$

$$\dot{u}^n_{i+\frac{1}{2},j,k+\frac{1}{2}} - \dot{u}^n_{i+\frac{1}{2},j,k-\frac{1}{2}} = \begin{cases} u^n_{i+\frac{1}{2},j,k} - u^n_{i+\frac{1}{2},j,k-1} & \bar{w}^n_{i+\frac{1}{2},j,k} > 0 \\ u^n_{i+\frac{1}{2},j,k+1} - u^n_{i+\frac{1}{2},j,k} & \bar{w}^n_{i+\frac{1}{2},j,k} < 0 \end{cases} \tag{3.26c}$$

Y-momentum Equation

The non-conservative momentum equation in the y-direction is

$$\rho \frac{\partial v}{\partial t} + \rho u \frac{\partial v}{\partial x} + \rho v \frac{\partial v}{\partial y} + \rho w \frac{\partial v}{\partial z} + \frac{\partial P}{\partial y} = F_y. \tag{3.27}$$

Discretizing this equation yields

$$\begin{aligned} \bar{\rho}^n_{i,j+\frac{1}{2},k} \frac{v^{n+1}_{i,j+\frac{1}{2},k} - v^n_{i,j+\frac{1}{2},k}}{\Delta t^n} + \bar{\rho}^n_{i,j+\frac{1}{2},k} \bar{u}^n_{i,j+\frac{1}{2},k} \frac{\dot{v}^n_{i+\frac{1}{2},j+\frac{1}{2},k} - \dot{v}^n_{i+\frac{1}{2},j-\frac{1}{2},k}}{\Delta x} + \\ \bar{\rho}^n_{i,j+\frac{1}{2},k} v^n_{i,j+\frac{1}{2},k} \frac{\dot{v}^n_{i,j+1,k} - \dot{v}^n_{i,j,k}}{\Delta y} + \bar{\rho}^n_{i,j+\frac{1}{2},k} \bar{w}^n_{i,j+\frac{1}{2},k} \frac{\dot{v}^n_{i,j+\frac{1}{2},k+\frac{1}{2}} - \dot{v}^n_{i,j+\frac{1}{2},k-\frac{1}{2}}}{\Delta z} + \\ \frac{P^{n+1}_{i,j+1,k} - P^{n+1}_{i,j,k}}{\Delta y} = (F_y)_{i,j+\frac{1}{2},k}. \end{aligned} \tag{3.28}$$

The pressure terms are handled implicitly as discussed in section 2.4. The barred density and off-axis velocity terms are arithmetically averaged according to

$$\bar{\rho}^n_{i,j+\frac{1}{2},k} = \frac{\rho^n_{i,j,k} + \rho^n_{i,j+1,k}}{2} \tag{3.29a}$$

$$\bar{u}^n_{i,j+\frac{1}{2},k} = \frac{u^n_{i+\frac{1}{2},j,k} + u^n_{i-\frac{1}{2},j,k} + u^n_{i+\frac{1}{2},j+1,k} + u^n_{i-\frac{1}{2},j+1,k}}{4} \tag{3.29b}$$

$$\bar{w}^n_{i+\frac{1}{2},j,k} = \frac{w^n_{i,j,k+\frac{1}{2}} + w^n_{i,j,k-\frac{1}{2}} + w^n_{i,j+1,,k+\frac{1}{2}} + w^n_{i,j+1,k-\frac{1}{2}}}{4}. \tag{3.29c}$$

The dotted velocity terms are first-order upwinded according to

$$\dot{v}^n_{i+1,j,k} - \dot{v}^n_{i,j,k} = \begin{cases} v^n_{i,j+\frac{1}{2},k} - v^n_{i,j-\frac{1}{2},k} & v^n_{i,j+\frac{1}{2},k} > 0 \\ v^n_{i,j+\frac{3}{2},k} - v^n_{i,j+\frac{1}{2},k} & v^n_{i,j+\frac{1}{2},k} < 0 \end{cases} \tag{3.30a}$$

$$\dot{v}^n_{i+\frac{1}{2},j+\frac{1}{2},k} - \dot{v}^n_{i+\frac{1}{2},j+\frac{1}{2},k} = \begin{cases} v^n_{i,j+\frac{1}{2},k} - v^n_{i-1,j+\frac{1}{2},k} & \bar{u}^n_{i+\frac{1}{2},j,k} > 0 \\ v^n_{i+1,j+\frac{1}{2},k} - v^n_{i,j+\frac{1}{2},k} & \bar{u}^n_{i+\frac{1}{2},j,k} < 0 \end{cases} \tag{3.30b}$$

$$\dot{v}^n_{i,j+\frac{1}{2},k+\frac{1}{2}} - \dot{v}^n_{i,j+\frac{1}{2},k+\frac{1}{2}} = \begin{cases} v^n_{i,j+\frac{1}{2},k} - v^n_{i,j+\frac{1}{2},k-1} & \bar{w}^n_{i+\frac{1}{2},j,k} > 0 \\ v^n_{i,j+\frac{1}{2},k+1} - v^n_{i,j+\frac{1}{2},k} & \bar{w}^n_{i+\frac{1}{2},j,k} < 0 \end{cases} \tag{3.30c}$$

Z-momentum Equation

The non-conservative momentum equation in the z-direction is

$$\rho \frac{\partial w}{\partial t} + \rho u \frac{\partial w}{\partial x} + \rho v \frac{\partial w}{\partial y} + \rho w \frac{\partial w}{\partial z} + \frac{\partial P}{\partial z} = F_z. \tag{3.31}$$

Discretizing this equation yields

$$\begin{aligned} \bar{\rho}^n_{i,j,k+\frac{1}{2}} \frac{w^{n+1}_{i,j,k+\frac{1}{2}} - w^n_{i,j,k+\frac{1}{2}}}{\Delta t^n} + \bar{\rho}^n_{i,j,k+\frac{1}{2}} \bar{u}^n_{i,j,k+\frac{1}{2}} \frac{\dot{w}^n_{i+\frac{1}{2},j,k+\frac{1}{2}} - \dot{w}^n_{i-\frac{1}{2},j,k+\frac{1}{2}}}{\Delta x} + \\ \bar{\rho}^n_{i,j,k+\frac{1}{2}} \bar{v}^n_{i,j,k+\frac{1}{2}} \frac{\dot{w}^n_{i,j+\frac{1}{2},k+\frac{1}{2}} - \dot{w}^n_{i,j-\frac{1}{2},k+\frac{1}{2}}}{\Delta y} + \bar{\rho}^n_{i,j,k+\frac{1}{2}} \bar{w}^n_{i,j,k+\frac{1}{2}} \frac{\dot{w}^n_{i,j,k+1} - \dot{w}^n_{i,j,k}}{\Delta z} + \\ \frac{P^{n+1}_{i,j,k+1} - P^{n+1}_{i,j,k}}{\Delta z} = (F_z)_{i,j,k+\frac{1}{2}}. \end{aligned} \tag{3.32}$$

The pressure terms are handled implicitly as discussed in section 2.4. The barred density and off-axis velocity terms are arithmetically averaged according to

$$\bar{\rho}^n_{i,j,k+\frac{1}{2}} = \frac{\rho^n_{i,j,k} + \rho^n_{i,j,k+1}}{2} \tag{3.33a}$$

$$\bar{v}^n_{i+\frac{1}{2},j,k} = \frac{v^n_{i,j+\frac{1}{2},k} + v^n_{i,j-\frac{1}{2},k} + v^n_{i,j+\frac{1}{2},k+1} + v^n_{i,j-\frac{1}{2},k+1}}{4} \tag{3.33b}$$

$$\bar{u}^n_{i+\frac{1}{2},j,k} = \frac{u^n_{i+\frac{1}{2},j,k} + u^n_{i-\frac{1}{2},j,k} + u^n_{i+\frac{1}{2},j,k+1} + u^n_{i-\frac{1}{2},j,k+1}}{4}. \tag{3.33c}$$

The dotted velocity terms are first-order upwinded according to

$$\dot{w}^n_{i+\frac{1}{2},j,k+\frac{1}{2}} - \dot{w}^n_{i-\frac{1}{2},j,k+\frac{1}{2}} = \begin{cases} w^n_{i,j,k+\frac{1}{2}} - w^n_{i-1,j,k+\frac{1}{2}} & \bar{u}^n_{i,j,k+\frac{1}{2}} > 0 \\ w^n_{i+1,j,k+\frac{1}{2}} - w^n_{i,j,k+\frac{1}{2}} & \bar{u}^n_{i,j,k+\frac{1}{2}} < 0 \end{cases} \tag{3.34a}$$

$$\dot{w}^n_{i,j+\frac{1}{2},k+\frac{1}{2}} - \dot{w}^n_{i,j-\frac{1}{2},k+\frac{1}{2}} = \begin{cases} w^n_{i,j,k+\frac{1}{2}} - u^n_{i,j-1,k+\frac{1}{2}} & \bar{v}^n_{i,j,k+\frac{1}{2}} > 0 \\ w^n_{i,j+1,k+\frac{1}{2}} - u^n_{i,j,k+\frac{1}{2}} & \bar{v}^n_{i,j,k+\frac{1}{2}} < 0 \end{cases} \tag{3.34b}$$

$$\dot{w}^n_{i,j,k+1} - \dot{w}^n_{i,j,k} = \begin{cases} w^n_{i,j,k+\frac{3}{2}} - w^n_{i,j,k+\frac{1}{2}} & w^n_{i+\frac{1}{2},j,k} > 0 \\ w^n_{i,j,k+\frac{1}{2}} - u^n_{i,j,k-\frac{1}{2}} & w^n_{i+\frac{1}{2},j,k} < 0 \end{cases} \tag{3.34c}$$

3.3.3 Discretized Enthalpy Equation

The three-dimensional form of the non-conservative enthalpy equation, as given in Equation 2.32, is

$$\rho \frac{\partial h}{\partial t} - \frac{\partial P}{\partial t} + \rho \vec{V} \cdot \nabla h - \vec{V} \cdot \nabla P = S_{energy} - \vec{V} \cdot \vec{F}. \tag{3.35}$$

Expanding this equation yields

$$\rho \frac{\partial h}{\partial t} - \frac{\partial P}{\partial t} + \rho u \frac{\partial h}{\partial x} + \rho v \frac{\partial h}{\partial y} + \rho w \frac{\partial h}{\partial z} - u \frac{\partial P}{\partial x} - v \frac{\partial P}{\partial y} - q \frac{\partial P}{\partial z} = S_{energy} - u F_x - v F_y - w F_z \tag{3.36}$$

The discretized form of this equation is

$$\rho^n_{i,j,k}\frac{h^{n+1}_{i,j,k}-h^n_{i,j,k}}{\Delta t^n}-\frac{P^{n+1}_{i,j,k}-P^n_{i,j,k}}{\Delta t^n}+\rho^n_{i,j,k}\bar{u}^{n+1}_{i,j,k}\frac{\dot{h}^n_{i+\frac{1}{2},j,k}-\dot{h}^n_{i-\frac{1}{2},j,k}}{\Delta x}$$
$$+\rho^n_{i,j,k}\bar{v}^{n+1}_{i,j,k}\frac{\dot{h}^n_{i,j+\frac{1}{2},k}-\dot{h}^n_{i,j-\frac{1}{2},k}}{\Delta y}+\rho^n_{i,j,k}\bar{w}^{n+1}_{i,j,k}\frac{\dot{h}^n_{i,j,k+\frac{1}{2}}-\dot{h}^n_{i,j,k-\frac{1}{2}}}{\Delta z}+$$
$$\bar{u}^n_{i,j,k}\frac{P^{n+1}_{i+1,j,k}-P^{n+1}_{i-1,j,k}}{2\Delta x}+\bar{v}^n_{i,j,k}\frac{P^{n+1}_{i,j+1,k}-P^{n+1}_{i,j-1,k}}{2\Delta y}+\bar{w}^n_{i,j,k}\frac{P^{n+1}_{i,j,k+1}-P^{n+1}_{i,j,k-1}}{2\Delta z}=$$
$$(S_{energy})_{i,j,k}-\bar{u}_{i,j,k}(F_x)_{i,j,k}-\bar{v}_{i,j,k}(F_y)_{i,j,k}-\bar{w}_{i,j,k}(F_z)_{i,j,k} \quad (3.37)$$

The advecting velocities and the pressure gradient terms are taken at the next timestep as discussed in section 2.4. The pressure gradient terms are arithmetically averaged and then simplified to the form shown. The barred velocity terms and pressure terms are arithmetically averaged in space according to

$$\bar{u}^{n+1}_{i,j,k}=\frac{u^{n+1}_{i+\frac{1}{2},j,k}+u^{n+1}_{i-\frac{1}{2},j,k}}{2} \quad (3.38a)$$

$$\bar{v}^{n+1}_{i,j,k}=\frac{v^{n+1}_{i,j+\frac{1}{2},k}+v^{n+1}_{i,j-\frac{1}{2},k}}{2} \quad (3.38b)$$

$$\bar{w}^{n+1}_{i,j,k}=\frac{u^{n+1}_{i,j,k+\frac{1}{2}}+u^{n+1}_{i,j,k-\frac{1}{2}}}{2} \quad (3.38c)$$

The dotted enthalpy terms are first-order upwinded in space according to

$$\dot{h}^n_{i+\frac{1}{2},j,k}-\dot{h}^n_{i-\frac{1}{2},j,k}=\begin{cases}h^n_{i,j,k}-h^n_{i-1,j,k} & \bar{u}^{n+1}_{i,j,k}>0\\ h^n_{i+1,j,k}-h^n_{i,j,k} & \bar{u}^{n+1}_{i,j,k}<0\end{cases} \quad (3.39a)$$

$$\dot{h}^n_{i,j+\frac{1}{2},k}-\dot{h}^n_{i,j-\frac{1}{2},k}=\begin{cases}h^n_{i,j,k}-h^n_{i,j-1,k} & \bar{v}^{n+1}_{i,j,k}>0\\ h^n_{i,j+1,k}-h^n_{i,j,k} & \bar{v}^{n+1}_{i,j,k}<0\end{cases} \quad (3.39b)$$

$$\dot{h}^n_{i,j,k+\frac{1}{2}}-\dot{h}^n_{i,j-,k\frac{1}{2}}=\begin{cases}h^n_{i,j,k}-h^n_{i,j,k-1} & \bar{w}^{n+1}_{i,j,k}>0\\ h^n_{i,j,k+1}-h^n_{i,j,k} & \bar{w}^{n+1}_{i,j,k}<0\end{cases} \quad (3.39c)$$

3.4 Solver Methods

In this section, the methods used in the solution process are discussed, including how the Jacobian matrix is constructed, how convergence of the simulation is checked, and an overview of changes to the solution control made to allow for the new solver without affecting the existing capabilities of CTF and CTF-R.

To allow for the three-dimensional solver to be run without affecting the behavior of CTF or the sub-channel form of CTF-R, changes to how solution control is handled within the code had to be made. The first step in this process was the addition of an input flag in the CTF input deck, `cartsolve`, which was used to define whether or not the code run was that of a sub-channel or three-dimensional case. Also, a new input file, *res_3d.inp*, and new output file, *"inputfile".ctf.cartesian.out*, were created that are used in the three-dimensional form of the residual solver.

Multiple times in the code, a logic check was used to see if the run was three-dimensional or sub-channel and if it was three-dimensional, control was passed to the residual solver. These cases were: checking the convergence of a null transient, calculation of the CFL limit, constructing and solving the Jacobian matrix, updating the system variables, and initializing the Jacobian in PETSc.

Changes were needed for many of the subroutines in the residual solver, and to do this new versions of the subroutines were implemented. This was done for changing how the system variables were perturbed, how the system variables were updated after system solution, and the subroutine for constructing the Jacobian matrix. A new subroutine, `residualConvergence` was implemented to check the convergence of the system. To check convergence of the solver for null transient cases, two values are used. The first is the l^2-norm of the residual vector calculated and the second is the l^∞-norm of the update vector which are calculated according to

$$|F|_2 = \sqrt{\sum_i F_i^2} \tag{3.40}$$

$$|\delta \boldsymbol{x}|_\infty = max_i(\delta \boldsymbol{x}_i). \tag{3.41}$$

For convergence of a pseudo-steady-state case to be reached these values must both lie below some user-input values.

The boundary conditions in the new solver are done by a new subroutine, `setBCs`.

Currently, boundary conditions can be set to one of three types for each face and variable in the domain. A set value can be specified, a zero gradient can be specified, which is done currently by setting the ghost value equal to the neighboring value in the computational domain. Also, a linear extrapolation from the first two values in the computational domain can be used. This is typically only used as a pressure inlet condition.

An overview of the system simulation process is given in Algorithm 10. This process assumes that the simulation is a three-dimensional residual run.

Algorithm 10 Simulation Overview in 3-D CTF-R

Simulation Start
Read CTF input
Read 3-D input from *res_3d.inp* and initialize residual solver
Initialize Jacobian matrix in CTF-R and PETSc
for $n = 1, N_{time}$ **do**
    calculate Δt^n based on CFL limit
    Call `timestep_init` subroutine
    Calculate the 3-D Cartesian residual vector $F(\boldsymbol{x})$
    Calculate the 3-D Cartesian Jacobian matrix $J(\boldsymbol{x})$
    Solve $\delta \boldsymbol{x}^n = J^{-1}(\boldsymbol{x})F(\boldsymbol{x})$
    Update 3-D variables
    Calculate the residual convergence
    Check if transient is complete
end for

The numerical approximation of the Jacobian matrix being used is

$$J_{i,j}(\boldsymbol{x}) = \frac{\partial F_j(\boldsymbol{x})}{\partial x_i} \approx \frac{F_j(\boldsymbol{x} + \epsilon \vec{e}_i) - F_j(\boldsymbol{x})}{\epsilon}, \tag{3.42}$$

as introduced in Equation 3.3. Performing this calculation for every variable in the problem domain for each conservation equation in the domain is a prohibitively expensive process even for fairly small domains. However, many of the terms in the Jacobian matrix will always be zero, due to the discretization of the conservation equations. Only certain variables and nodes can affect the equations of a node.

This means that values in the Jacobian matrix for a single conservation equation can be calculated for only the variables that can affect that equation, which will allow for a

significant decrease in the cost of construction of the Jacobian matrix. While this could be done for singular variables and singular equations, the coding strategy for that would be difficult to maintain and adjust due to the large number of exceptions. Instead, this selection was done on matching variables to all the equations in a cell. This means that if a variable in cell n can affect any equation in cell m, then that variable in cell n will be checked in the construction of the Jacobian for all equations in cell m. The way velocity values are "owned" by cells in this process is that the forward velocity face of cell n, $n+\frac{1}{2}$ is part of cell n. The backwards boundary velocities are "owned" by the first cell value, e.g. $u_{\frac{1}{2},j,k}$ is owned by cell $(1, j, k)$.

The algorithm for the process of constructing the Jacobian in three-dimensions is outlined below. The outer-most two loops are over the number of cells in the domain and then the number of rows, or equations, in that cell. The value of the residual vector for that cell is set and then a loop over the cells that are within a $3\times3\times3$ cube centered at the cell N is started. If this cell, noted here as cell r, is within the stencil, then a loop over the number of coupled variables from that cell is done. The number of coupled variables, denoted as L, varies from cell to cell in the stencil. Once in the loop over the coupled variables in cell r, the specified coupled variable is perturbed by ϵ and used to recalculate the residual of the current equation. The column that this variable corresponds to in the Jacobian is determined and then the perturbed value is used to calculate the entry in the Jacobian matrix for the equation and variable. This process is repeated for each coupled variable in the stencil for an equation, and then repeated for each equation and cell in the domain.

Algorithm 11 Jacobian Construction Process in 3-D Semi-Implicit Solver

for $N = 1, N_{cells}$ **do**
  for $row = 1, N_{rows}$ **do**
    Set $F_{row}(x)$
    for $r = 1, 27$ **do**
      if r in stencil **then**
        for $l = 1, L$ **do**
          Perturb variable l in cell r
          Calculate $F_{row}(x + \epsilon x_{l,r})$
          Determine the column in the Jacobian that corresponds to variable $x_{l,r}$
          $J_{row,column} = \frac{F_{row}(x+\epsilon x_{l,r}) - F_{row}(x)}{\epsilon}$
        end for
      end if
    end for
  end for
end for

The discretized equations presented in section 3.3 are implemented in the form of functions within the residual solver. These functions are used to calculate the residual vector, $F(\boldsymbol{x})$ and Jacobian matrix, $J(\boldsymbol{x})$.

3.5 Method Verification

To verify the new three-dimensional solver, a set of three simple test cases were set up and compared to an analytic result. These three test cases were a case with no sources with boundary conditions different than the initial condition, a test case with a gravity source opposite the flow direction and a flat initial pressure distribution, and a test with a volumetric heat generation source and a flat initial enthalpy distribution. For each of these cases, the solver was run with six orientations: inlet flow in the negative x-direction, inlet flow in the positive x-direction, inlet flow in the negative y-direction, inlet flow in the positive y-direction, inlet flow in the negative z-direction, and inlet flow in the positive z-direction.

3.5.1 Sourceless Verification Case

The problem specifications for the sourceless test case is shown in Table 3.3. These specifications presented are for inlet flow in the x-direction.

Table 3.3: Initial Problem Specification

Channel Height (m)	0.5	Maximum Time steps	5000
Channel Length (m)	0.5	Timestep Length (s)	0.001
Channel Width (m)	0.5	Outlet Pressure (Pa)	0
x-direction Mesh Spacing (m)	0.025	Inlet Velocity Vector (m/s)	(5.0, 0.0, 0.0)
y-direction Mesh Spacing (m)	0.05	Inlet Enthalpy (kJ/kg)	530
z-direction Mesh Spacing (m)	0.05	Initial Pressure (Pa)	10
Steady-State Convergence Criterion	0.0001	Initial Enthalpy (kJ/kg)	520

The analytic form of this case can be analyzed by first removing the off-axis terms from the governing equations to yield the one-dimensional form of the governing equations with no source terms,

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} = 0 \tag{3.43a}$$

$$\rho \frac{\partial u}{\partial t} + \rho u \frac{\partial u}{\partial x} + \frac{\partial P}{\partial x} = 0 \tag{3.43b}$$

$$\rho \frac{\partial h}{\partial t} - \frac{\partial P}{\partial t} + \rho u \frac{\partial h}{\partial x} - u \frac{\partial P}{\partial x} = 0. \tag{3.43c}$$

For a steady-state simulation, the time terms will be equal to zero, leaving

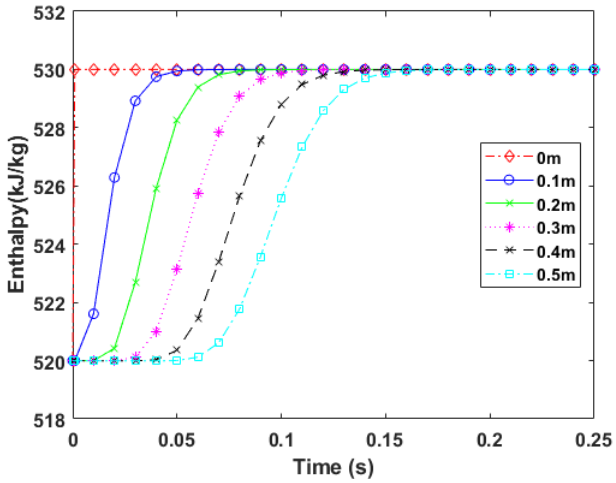
$$\frac{\partial \rho u}{\partial x} = 0 \tag{3.44a}$$

$$\rho u \frac{\partial u}{\partial x} + \frac{\partial P}{\partial x} = 0 \tag{3.44b}$$

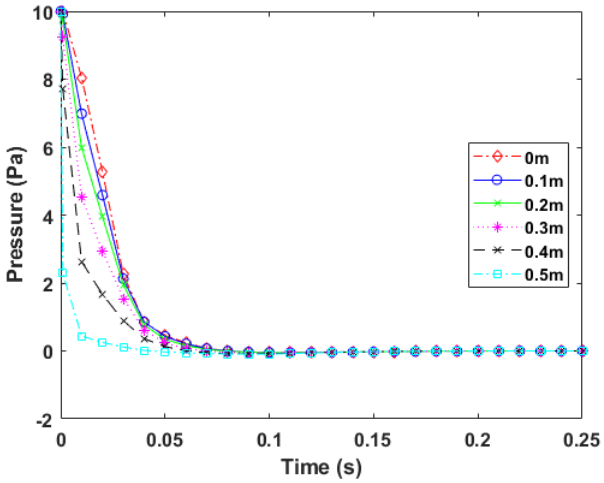
$$\rho \frac{\partial h}{\partial x} - \frac{\partial P}{\partial x} = 0, \tag{3.44c}$$

and this system will result in the gradients all being equal to zero and the conditions in the problem domain being equal to the values at the boundaries when steady-state is reached. This means the pressure should be equal to 0 Pa throughout the domain and the enthalpy should be equal to 530 kJ/kg throughout the domain. The plots in Figure 3.4a

and Figure 3.4b. It can be seen that both the enthalpy and the pressure distribution rapidly converge to the analytic values. It can be seen that there is some numerical diffusion in the enthalpy distribution over the domain, and the diffusion is more pronounced later in the channel. This is expected from the use of first-order upwinding in this solver.



(a) Enthalpy Distribution in the Sourceless Test of the Semi-Implicit Solver



(b) Pressure Distribution in the Sourceless Test of the Semi-Implicit Solver

Figure 3.4: Sourceless Test Case for the Semi-Implicit Solver

From these results, tt can be seen that the semi-implicit solver converges well to the

analytic result for a fluid flow with no source terms present.

3.5.2 Heat Source Verification Case

The second test case was designed to verify the behavior of the heat source term in the enthalpy equation. The problem specifications are given in Table 3.4. A uniform volumetric heat generation of 800,000 W/m^3 is set throughout the fluid domain.

Table 3.4: Initial Problem Specification

Channel Height (m)	0.5	Maximum Time steps	5000
Channel Length (m)	0.5	Timestep Length (s)	0.001
Channel Width (m)	0.5	Outlet Pressure (Pa)	0
x-direction Mesh Spacing (m)	0.025	Inlet Velocity Vector (m/s)	(5.0, 0.0, 0.0)
y-direction Mesh Spacing (m)	0.05	Inlet Enthalpy (kJ/kg)	530
z-direction Mesh Spacing (m)	0.05	Initial Pressure (Pa)	0
Steady-State Convergence Criterion	0.0001	Initial Enthalpy (kJ/kg)	530

In this problem, assuming the compressibility of the fluid to be negligible, the momentum and mass equations are trivial. The enthalpy equation becomes one-dimensional and there is no pressure gradients or changes, leaving

$$\rho \frac{\partial h}{\partial t} + \rho u \frac{\partial h}{\partial x} = S_{ht}. \tag{3.45}$$

For the steady-state case, the rate of change of the enthalpy will be zero, leaving

$$\rho u \frac{\partial h}{\partial x} = S_{ht}. \tag{3.46}$$

The expected enthalpy gradient will then be

$$\frac{\partial h}{\partial x} = \frac{S_{ht}}{\rho u}. \tag{3.47}$$

For a density of 1000 kg/m^3 and a velocity of 5.0 m/s, the enthalpy gradient will be

$$\frac{\partial h}{\partial x} = \frac{800000 \frac{W}{m^3}}{1000 \frac{kg}{m^3} * 5 \frac{m}{s}} = 160 \frac{kJ/kg}{m} \tag{3.48}$$

The plot in Figure 3.5 shows the enthalpy distribution from the new solver for this case. The solid lines represent the analytic steady-state value at each height. As can be seen from the figure, the enthalpy at each height converges to the analytic value. Also, the enthalpy rate of change when there is no enthalpy gradient should be

$$\frac{\partial h}{\partial t} = \frac{S_{ht}}{\rho} = \frac{800000}{1000} = 800 \frac{kJ/kg}{s} \tag{3.49}$$

and the slope of the linear enthalpy rise for the early timesteps is approximately 800 $\frac{kJ/kg}{s}$.

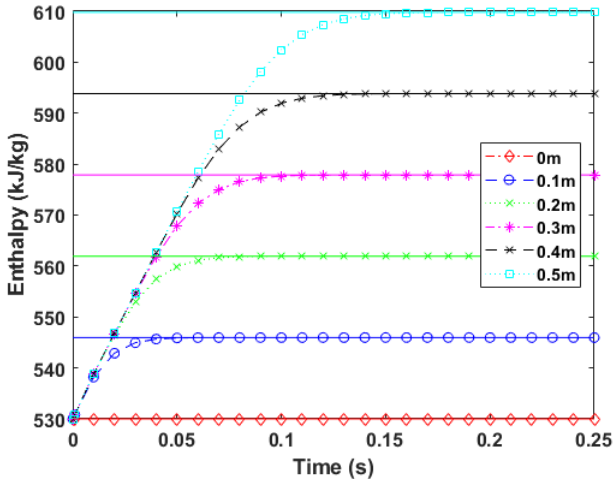


Figure 3.5: Enthalpy Distribution in the Heat Source Case for the Semi-Implicit Solver

These results show that the behavior of the new three-dimensional solver function as expected in the case of a heat source in the problem domain.

3.5.3 Gravity Verification Case

The third test case was designed to verify the behavior of the gravitational source term in the pressure and enthalpy equations. The problem specifications are given in Table 3.5. A gravitational term of 10 m/s^2 is present in the negative x-direction.

Table 3.5: Initial Problem Specification

Channel Height (m)	0.5	Maximum Time steps	5000
Channel Length (m)	0.5	Timestep Length (s)	0.001
Channel Width (m)	0.5	Outlet Pressure (Pa)	0
x-direction Mesh Spacing (m)	0.025	Inlet Velocity Vector (m/s)	(5.0, 0.0, 0.0)
y-direction Mesh Spacing (m)	0.05	Inlet Enthalpy (kJ/kg)	530
z-direction Mesh Spacing (m)	0.05	Initial Pressure (Pa)	0
Steady-State Convergence Criterion	0.0001	Initial Enthalpy (kJ/kg)	530

In the analytic case, there is no enthalpy change and the fluid compressibility is assumed negligible, which means that there will be no velocity gradient in the fluid. Based on these statements, the governing equations simplify to

$$\rho \frac{\partial u}{\partial x} = 0 \tag{3.50a}$$

$$\frac{\partial P}{\partial x} = \rho g_x \tag{3.50b}$$

$$-\frac{\partial P}{\partial t} - u\frac{\partial P}{\partial x} = u\rho g_x \tag{3.50c}$$

For the steady-state case, the rate of change of the pressure will be zero, which means the enthalpy equation simplifies to the momentum equation and the pressure gradient at steady-state will be

$$\frac{\partial P}{\partial x} = \rho g_x \tag{3.51}$$

For a density of 1000 kg/m^3, the pressure gradient will be 10,000 Pa/m. The plot in Figure 3.6 shows the pressure distribution for this case compared to the analytic result, which is shown by the solid horizontal lines at each height. It should be noted that the boundary condition of 0 Pa is set in a ghost node, so it is specified at 0.525m. The value at each node can be seen to converge to the analytic result. In that process, there is an overshoot in the pressure terms as it reaches the analytic value, and then a slight undershoot after. This behavior is discussed more in Section 4.3.4.

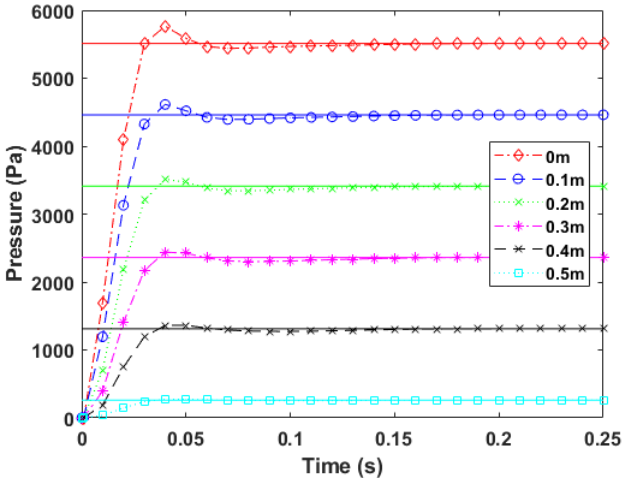


Figure 3.6: Pressure Distribution in the Gravitational Source Case for the Semi-Implicit Solver

These results show that the behavior of the new three-dimensional solver functions as expected in the case of a gravitational source in the problem domain.

In this section, multiple terms in the governing equations are tested, however some remain untested. All terms in the continuity and enthalpy equations shown in Equation 2.4 and Equation 2.32 are tested. In the momentum equations, the only untested terms are the off-axis momentum flux terms. This can be seen in the below equations where the terms that are tested are highlighted and the untested terms are not.

$$\rho\frac{\partial u}{\partial t}+\rho u\frac{\partial u}{\partial x}+\rho v\frac{\partial u}{\partial y}+\rho w\frac{\partial u}{\partial z}+\frac{\partial P}{\partial x}=F_x \tag{3.52a}$$

$$\rho\frac{\partial v}{\partial t}+\rho u\frac{\partial v}{\partial x}+\rho v\frac{\partial v}{\partial y}+\rho w\frac{\partial v}{\partial z}+\frac{\partial P}{\partial y}=F_y \tag{3.52b}$$

$$\rho\frac{\partial w}{\partial t}+\rho u\frac{\partial w}{\partial x}+\rho v\frac{\partial w}{\partial y}+\rho w\frac{\partial w}{\partial z}+\frac{\partial P}{\partial z}=F_z \tag{3.52c}$$

CHAPTER

4

DEVELOPMENT OF AN IMPLICIT AND A CONSERVATIVE SOLVER

In this chapter, the three-dimensional Cartesian solver introduced in chapter 3 is expanded to include a nonlinear iteration. This iteration allows the implementation of both an implicit solver and a conservative solver which is shown in section 4.1 and section 4.2. Verification problems for the nonlinear iteration, as well as the implicit and conservative solvers that use it, are shown in section 4.3.

4.1 Implicit Solver

The three-dimensional solver implemented in CTF-R in chapter 3 used a semi-implicit time discretization. The semi-implicit method is widely used in thermal-hydraulics, but a large part of this use is the fact that, as discussed in (35), the semi-implicit method is the most implicit method that can be implemented to approximate the solution of the system in a single linear Newton step without a nonlinear iteration. An implicit method offers

multiple features not available in semi-implicit or linearized methods, two of which are the removal of CFL timestep constraints on the stability of the solver and the ability to solve the conservative form of the Euler equations. The main drawbacks of an implicit method is the requirement of implementing a nonlinear iteration, which is computationally expensive, and increased artificial diffusion in the resulting solver. A nonlinear iteration is required in the implicit method to allow the effect of the implicit variables to effect the simulation. Without a nonlinear solution process, the implicit method will simplify to the semi-implicit method, as the effect of using solution variables at the next timestep is not captured.

4.1.1 Implementation of a Nonlinear Iteration

The first step in implementing an implicit solver is to implement a nonlinear iteration to allow for the resolution of the product of values taken at the next timestep in the governing equations. For implementing a nonlinear iteration, a Shamanskii method was implemented as introduced in Section 2.3.3. It should be noted that through the use of Krylov subspace methods in PETSc to solve the linear system this implementation will be classified as an inexact Newton method since the solution to the linear system is approximate.

This scheme was implemented in the *solve_GovEqSolve* subroutine, which handles the solution process for the system matrix. The process implemented is shown below in Algorithm 12, with the assumption of the three-dimensional residual solver being run. Note that this implementation of a nonlinear iteration is not available in CTF or the sub-channel form of CTF-R currently. When the solve step is begun and the *solve_GovEqSolve* subroutine is entered, the max number of iterations is set based upon a user-input value in *res_3d.inp*. Next, the residual vector and Jacobian matrix is calculated for the 0^{th} nonlinear solution vector. A loop is entered while the number of iterations is less than the maximum number of iterations. If the remainder of the number of iterations taken and the m value for the Shamanskii iteration is zero, the Jacobian matrix is recalculated. The most recently calculated Jacobian is then used with the residual vector of the current solution to calculate the solution update, $\delta \boldsymbol{x}^k$ and update the solution vector to the next iterate, $\boldsymbol{x}^{k+1}$. This new solution value is used to update the residual vector and then the l^2-norm of the updated residual vector, as well as the l∞-norm of the solution update, is used to check the convergence of the nonlinear iteration. The values ϵ_{l2} and $\epsilon_{l\infty}$ are convergence criteria input by the user. If the iteration has converged it is exited and otherwise, the iteration index is incremented and the iteration is repeated until either convergence is reached or

the maximum number of iterations is reached.

Algorithm 12 System Solution Overview in Shamanskii Iteration in 3-D CTF-R

Begin solve step
Set max number of iterations, N_{iter} from *res_3d.inp*
Calculate $F(\boldsymbol{x_0})$ and $J(\boldsymbol{x}_0)$
Set $k = 1$
while $k < N_{iter}$ **do** *Shamanskii Iteration*
    if $Mod(k,m) = 0$ **then**
        Calculate $J(\boldsymbol{x_k})$
    end if
    calculate $\delta \boldsymbol{x}^k = -J^{-1}(\boldsymbol{x}^{k-Mod(k,m)})F(\boldsymbol{x}^k)$
    calculate $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \delta \boldsymbol{x}^k$
    calculate $F(\boldsymbol{x}^{k+1})$
    calculate $|F(\boldsymbol{x}^{k+1})|_2$ and $|\delta \boldsymbol{x}^k|_\infty$
    if $|F(\boldsymbol{x}^{k+1})|_2 > \epsilon_{l2}$ and $|\delta \boldsymbol{x}^k|_\infty < \epsilon_{l\infty}$ **then**
        Exit *Shamanskii Iteration*
    else
        $k = k + 1$
    end if
end while

It is important to note that while the residual vector and Jacobian matrix are denoted as functions of the solution for a value of the iteration index, they are also functions of the initial iterate, $\boldsymbol{x}_0$ for transient simulations.

To handle including the capabilities of this iteration without affecting the behavior of CTF, the sub-channel form of CTF-R, and the semi-implicit form of CTF-R presented in chapter 3, certain flags had to be added to the solver input. A flag was created, `AlgFlag` to indicate which time algorithm would be used. A value of zero represents that the linear solution method will be used, so the $While$ statement in Algorithm 12 will not be looped over. A value of one or higher represents that the nonlinear iteration should be used and the value of the flag will correspond to the m value of the iteration. A flag for specifying whether the system will be semi-implicit or implicit, `ImpFlag`, was implemented as well. If this flag is set to zero, the semi-implicit solver will be used and if it is set to one the implicit solver will be used. For `ImpFlag` to be set to one, the *AlgFlag* must be greater than zero.

The Equations presented in Section 4.1.2 are implemented as functions within the residual solver. When the `ImpFlag` is set to one, those functions are used to calculate the residual vector and the Jacobian matrix. When `ImpFlag` is set to zero, the residual functions that correspond to the discretized equations in section 3.3 are used.

4.1.2 Implicit Discretized Governing Equations

The governing equations for the implicit solver are presented here. These equations are based on the equations presented in section 2.2 and are discretized spatially in the same manner as the semi-implicit equations presented in section 3.3.

Implicit Continuity Equation

The three-dimensional form of the continuity equation, as presented in Equation 2.4, is

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} + \frac{\partial \rho v}{\partial y} + \frac{\partial \rho w}{\partial z} = S_{mass}. \tag{4.1}$$

Discretizing this equation about the three-dimensional mesh with the implicit method, neglecting the mass source term, gives

$$\frac{\rho^{n+1}_{i,j,k} - \rho^{n}_{i,j,k}}{\Delta t^{n}} + \frac{\dot{\rho}^{n+1}_{i+\frac{1}{2},j,k} u^{n+1}_{i+\frac{1}{2},j,k} - \dot{\rho}^{n+1}_{i-\frac{1}{2},j,k} u^{n+1}_{i-\frac{1}{2},j,k}}{\Delta x} + \frac{\dot{\rho}^{n+1}_{i,j+\frac{1}{2},k} v^{n+1}_{i,j+\frac{1}{2},k} - \dot{\rho}^{n+1}_{i,j-\frac{1}{2},k} v^{n+1}_{i,j-\frac{1}{2},k}}{\Delta y} + \frac{\dot{\rho}^{n+1}_{i,j,k+\frac{1}{2}} w^{n+1}_{i,j,k+\frac{1}{2}} - \dot{\rho}^{n+1}_{i,j,k-\frac{1}{2}} w^{n+1}_{i,j,k-\frac{1}{2}}}{\Delta z} = 0. \tag{4.2}$$

The dotted density terms are first-order upwinded in space as shown in Equation 3.21.

The three-dimensional form of the non-conservative momentum equation, as given in Equation 2.18, is

$$\rho \frac{\partial u}{\partial t} + \rho u \frac{\partial u}{\partial x} + \rho v \frac{\partial u}{\partial y} + \rho w \frac{\partial u}{\partial z} + \frac{\partial P}{\partial x} = F_x \tag{4.3a}$$

$$\rho \frac{\partial v}{\partial t} + \rho u \frac{\partial v}{\partial x} + \rho v \frac{\partial v}{\partial y} + \rho w \frac{\partial v}{\partial z} + \frac{\partial P}{\partial y} = F_y \tag{4.3b}$$

$$\rho \frac{\partial w}{\partial t} + \rho u \frac{\partial w}{\partial x} + \rho v \frac{\partial w}{\partial y} + \rho w \frac{\partial w}{\partial z} + \frac{\partial P}{\partial z} = F_z. \tag{4.3c}$$

Implicit X-Momentum Equation

The discretization of the x-momentum equation with the implicit method gives

$$\bar{\rho}^{n+1}_{i+\frac{1}{2},j,k}\frac{u^{n+1}_{i+\frac{1}{2},j,k}-u^{n}_{i+\frac{1}{2},j,k}}{\Delta t^{n}}+\bar{\rho}^{n+1}_{i+\frac{1}{2},j,k}u^{n+1}_{i+\frac{1}{2},j,k}\frac{\dot{u}^{n+1}_{i+1,j,k}-\dot{u}^{n+1}_{i,j,k}}{\Delta x}+$$
$$\bar{\rho}^{n+1}_{i+\frac{1}{2},j,k}\bar{v}^{n+1}_{i+\frac{1}{2},j,k}\frac{\dot{u}^{n+1}_{i+\frac{1}{2},j+\frac{1}{2},k}-\dot{u}^{n+1}_{i+\frac{1}{2},j-\frac{1}{2},k}}{\Delta y}+\bar{\rho}^{n+1}_{i+\frac{1}{2},j,k}\bar{w}^{n+1}_{i+\frac{1}{2},j,k}\frac{\dot{u}^{n+1}_{i+\frac{1}{2},j,k+\frac{1}{2}}-\dot{u}^{n+1}_{i+\frac{1}{2},j,k-\frac{1}{2}}}{\Delta z}+$$
$$\frac{P^{n+1}_{i+1,j,k}-P^{n+1}_{i,j,k}}{\Delta x}=(F_x)_{i+\frac{1}{2},j,k}. \quad (4.4)$$

The barred density and velocity terms are arithmetically averaged in space according to

$$\bar{\rho}^{n+1}_{i+\frac{1}{2},j,k}=\frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i+1,j,k}}{2} \quad (4.5a)$$

$$\bar{v}^{n+1}_{i+\frac{1}{2},j,k}=\frac{v^{n+1}_{i,j+\frac{1}{2},k}+v^{n+1}_{i,j-\frac{1}{2},k}+v^{n+1}_{i+1,j+\frac{1}{2},k}+v^{n+1}_{i+1,j-\frac{1}{2},k}}{4} \quad (4.5b)$$

$$\bar{w}^{n+1}_{i+\frac{1}{2},j,k}=\frac{w^{n+1}_{i,j,k+\frac{1}{2}}+w^{n+1}_{i,j,k-\frac{1}{2}}+w^{n+1}_{i+1,j,k+\frac{1}{2}}+w^{n+1}_{i+1,j,k-\frac{1}{2}}}{4}. \quad (4.5c)$$

The dotted velocity terms are first-order upwinded according to

$$\dot{u}^{n+1}_{i+1,j,k}-\dot{u}^{n+1}_{i,j,k}=\begin{cases}u^{n+1}_{i+\frac{1}{2},j,k}-u^{n+1}_{i-\frac{1}{2},j,k} & u^{n+1}_{i+\frac{1}{2},j,k}>0\\ u^{n+1}_{i+\frac{3}{2},j,k}-u^{n+1}_{i+\frac{1}{2},j,k} & u^{n+1}_{i+\frac{1}{2},j,k}<0\end{cases} \quad (4.6a)$$

$$\dot{u}^{n+1}_{i+\frac{1}{2},j+\frac{1}{2},k}-\dot{u}^{n+1}_{i+\frac{1}{2},j-\frac{1}{2},k}=\begin{cases}u^{n+1}_{i+\frac{1}{2},j,k}-u^{n+1}_{i+\frac{1}{2},j-1,k} & \bar{v}^{n+1}_{i+\frac{1}{2},j,k}>0\\ u^{n+1}_{i+\frac{1}{2},j+1,k}-u^{n+1}_{i+\frac{1}{2},j,k} & \bar{v}^{n+1}_{i+\frac{1}{2},j,k}<0\end{cases} \quad (4.6b)$$

$$\dot{u}^{n+1}_{i+\frac{1}{2},j,k+\frac{1}{2}}-\dot{u}^{n+1}_{i+\frac{1}{2},j,k-\frac{1}{2}}=\begin{cases}u^{n+1}_{i+\frac{1}{2},j,k}-u^{n+1}_{i+\frac{1}{2},j,k-1} & \bar{w}^{n+1}_{i+\frac{1}{2},j,k}>0\\ u^{n+1}_{i+\frac{1}{2},j,k+1}-u^{n+1}_{i+\frac{1}{2},j,k} & \bar{w}^{n+1}_{i+\frac{1}{2},j,k}<0.\end{cases} \quad (4.6c)$$

Implicit Y-Momentum Equation

The discretization of the y-momentum equation with the implicit method gives

$$\bar{\rho}^{n+1}_{i,j+\frac{1}{2},k}\frac{v^{n+1}_{i,j+\frac{1}{2},k}-v^{n}_{i,j+\frac{1}{2},k}}{\Delta t^{n}}+\bar{\rho}^{n+1}_{i,j+\frac{1}{2},k}\bar{u}^{n+1}_{i,j+\frac{1}{2},k}\frac{\dot{v}^{n+1}_{i+\frac{1}{2},j+\frac{1}{2},k}-\dot{v}^{n+1}_{i+\frac{1}{2},j-\frac{1}{2},k}}{\Delta x}+$$
$$\bar{\rho}^{n+1}_{i,j+\frac{1}{2},k}v^{n+1}_{i,j+\frac{1}{2},k}\frac{\dot{v}^{n+1}_{i,j+1,k}-\dot{v}^{n+1}_{i,j,k}}{\Delta y}+\bar{\rho}^{n+1}_{i,j+\frac{1}{2},k}\bar{w}^{n+1}_{i,j+\frac{1}{2},k}\frac{\dot{v}^{n+1}_{i,j+\frac{1}{2},k+\frac{1}{2}}-\dot{v}^{n+1}_{i,j+\frac{1}{2},k-\frac{1}{2}}}{\Delta z}+$$
$$\frac{P^{n+1}_{i,j+1,k}-P^{n+1}_{i,j,k}}{\Delta y}=(F_y)_{i,j+\frac{1}{2},k}. \quad (4.7)$$

The barred density and off-axis velocity terms are arithmetically averaged according to

$$\bar{\rho}^{n+1}_{i,j+\frac{1}{2},k}=\frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i,j+1,k}}{2} \quad (4.8a)$$

$$\bar{u}^{n+1}_{i,j+\frac{1}{2},k}=\frac{u^{n+1}_{i+\frac{1}{2},j,k}+u^{n+1}_{i,j-\frac{1}{2},k}+u^{n+1}_{i+\frac{1}{2},j+1,k}+u^{n+1}_{i-\frac{1}{2},j+1,k}}{4} \quad (4.8b)$$

$$\bar{w}^{n+1}_{i+\frac{1}{2},j,k}=\frac{w^{n+1}_{i,j,k+\frac{1}{2}}+w^{n+1}_{i,j,k-\frac{1}{2}}+w^{n+1}_{i,j+1,,k+\frac{1}{2}}+w^{n+1}_{i,j+1,k-\frac{1}{2}}}{4}. \quad (4.8c)$$

The dotted velocity terms are first-order upwinded according to

$$\dot{v}^{n+1}_{i+1,j,k}-\dot{v}^{n+1}_{i,j,k}=\begin{cases}v^{n+1}_{i,j+\frac{1}{2},k}-v^{n+1}_{i,j-\frac{1}{2},k} & v^{n+1}_{i,j+\frac{1}{2},k}>0\\ v^{n+1}_{i,j+\frac{3}{2},k}-v^{n+1}_{i,j+\frac{1}{2},k} & v^{n+1}_{i,j+\frac{1}{2},k}<0\end{cases} \quad (4.9a)$$

$$\dot{v}^{n+1}_{i+\frac{1}{2},j+\frac{1}{2},k}-\dot{v}^{n+1}_{i+\frac{1}{2},j+\frac{1}{2},k}=\begin{cases}v^{n+1}_{i,j+\frac{1}{2},k}-v^{n+1}_{i-1,j+\frac{1}{2},k} & \bar{u}^{n+1}_{i+\frac{1}{2},j,k}>0\\ v^{n+1}_{i+1,j+\frac{1}{2},k}-v^{n+1}_{i,j+\frac{1}{2},k} & \bar{u}^{n+1}_{i+\frac{1}{2},j,k}<0\end{cases} \quad (4.9b)$$

$$\dot{v}^{n+1}_{i,j+\frac{1}{2},k+\frac{1}{2}}-\dot{v}^{n+1}_{i,j+\frac{1}{2},k+\frac{1}{2}}=\begin{cases}v^{n+1}_{i,j+\frac{1}{2},k}-v^{n+1}_{i,j+\frac{1}{2},k-1} & \bar{w}^{n+1}_{i+\frac{1}{2},j,k}>0\\ v^{n+1}_{i,j+\frac{1}{2},k+1}-v^{n+1}_{i,j+\frac{1}{2},k} & \bar{w}^{n+1}_{i+\frac{1}{2},j,k}<0.\end{cases} \quad (4.9c)$$

Implicit Z-Momentum Equation

The discretization of the z-momentum equation with the implicit method gives

$$\bar{\rho}^{n+1}_{i,j,k+\frac{1}{2}} \frac{w^{n+1}_{i,j,k+\frac{1}{2}} - w^{n}_{i,j,k+\frac{1}{2}}}{\Delta t^n} + \bar{\rho}^{n+1}_{i,j,k+\frac{1}{2}} \bar{u}^{n+1}_{i,j,k+\frac{1}{2}} \frac{\dot{w}^{n+1}_{i+\frac{1}{2},j,k+\frac{1}{2}} - \dot{w}^{n+1}_{i-\frac{1}{2},j,k+\frac{1}{2}}}{\Delta x} + \bar{\rho}^{n+1}_{i,j,k+\frac{1}{2}} \bar{v}^{n+1}_{i,j,k+\frac{1}{2}} \frac{\dot{w}^{n+1}_{i,j+\frac{1}{2},k+\frac{1}{2}} - \dot{w}^{n+1}_{i,j-\frac{1}{2},k+\frac{1}{2}}}{\Delta y} + \bar{\rho}^{n+1}_{i,j,k+\frac{1}{2}} \bar{w}^{n+1}_{i,j,k+\frac{1}{2}} \frac{\dot{w}^{n+1}_{i,j,k+1} - \dot{w}^{n+1}_{i,j,k}}{\Delta z} + \frac{P^{n+1}_{i,j,k+1} - P^{n+1}_{i,j,k}}{\Delta z} = (F_z)_{i,j,k+\frac{1}{2}}. \quad (4.10)$$

The pressure terms are handled implicitly as discussed in section 2.4. The barred density and off-axis velocity terms are arithmetically averaged according to

$$\bar{\rho}^{n+1}_{i,j,k+\frac{1}{2}} = \frac{\rho^{n+1}_{i,j,k} + \rho^{n+1}_{i,j,k+1}}{2} \quad (4.11a)$$

$$\bar{v}^{n+1}_{i+\frac{1}{2},j,k} = \frac{v^{n+1}_{i,j+\frac{1}{2},k} + v^{n+1}_{i,j-\frac{1}{2},k} + v^{n+1}_{i,j+\frac{1}{2},k+1} + v^{n+1}_{i,j-\frac{1}{2},k+1}}{4} \quad (4.11b)$$

$$\bar{u}^{n+1}_{i+\frac{1}{2},j,k} = \frac{u^{n+1}_{i+\frac{1}{2},j,k} + u^{n+1}_{i-\frac{1}{2},j,k} + u^{n+1}_{i+\frac{1}{2},j,k+1} + u^{n+1}_{i-\frac{1}{2},j,k+1}}{4}. \quad (4.11c)$$

The dotted velocity terms are first-order upwinded according to

$$\dot{w}^{n+1}_{i+\frac{1}{2},j,k+\frac{1}{2}} - \dot{w}^{n+1}_{i-\frac{1}{2},j,k+\frac{1}{2}} = \begin{cases} w^{n+1}_{i,j,k+\frac{1}{2}} - w^{n+1}_{i-1,j,k+\frac{1}{2}} & \bar{u}^{n+1}_{i,j,k+\frac{1}{2}} > 0 \\ w^{n+1}_{i+1,j,k+\frac{1}{2}} - w^{n+1}_{i,j,k+\frac{1}{2}} & \bar{u}^{n+1}_{i,j,k+\frac{1}{2}} < 0 \end{cases} \quad (4.12a)$$

$$\dot{w}^{n+1}_{i,j+\frac{1}{2},k+\frac{1}{2}} - \dot{w}^{n+1}_{i,j-\frac{1}{2},k+\frac{1}{2}} = \begin{cases} w^{n+1}_{i,j,k+\frac{1}{2}} - u^{n+1}_{i,j-1,k+\frac{1}{2}} & \bar{v}^{n+1}_{i,j,k+\frac{1}{2}} > 0 \\ w^{n+1}_{i,j+1,k+\frac{1}{2}} - u^{n+1}_{i,j,k+\frac{1}{2}} & \bar{v}^{n+1}_{i,j,k+\frac{1}{2}} < 0 \end{cases} \quad (4.12b)$$

$$\dot{w}^{n+1}_{i,j,k+1} - \dot{w}^{n+1}_{i,j,k} = \begin{cases} w^{n+1}_{i,j,k+\frac{3}{2}} - w^{n+1}_{i,j,k+\frac{1}{2}} & w^{n+1}_{i+\frac{1}{2},j,k} > 0 \\ w^{n+1}_{i,j,k+\frac{1}{2}} - u^{n+1}_{i,j,k-\frac{1}{2}} & w^{n+1}_{i+\frac{1}{2},j,k} < 0. \end{cases} \quad (4.12c)$$

Implicit Enthalpy Equation

The non-conservative form of the enthalpy equation in three-dimensions is,

$$\rho\frac{\partial h}{\partial t}-\frac{\partial P}{\partial t}+\rho u\frac{\partial h}{\partial x}+\rho v\frac{\partial h}{\partial y}+\rho w\frac{\partial h}{\partial z}-u\frac{\partial P}{\partial x}-v\frac{\partial P}{\partial y}-q\frac{\partial P}{\partial z}=S_{energy}-uF_x-vF_y-wF_z. \tag{4.13}$$

The discretization of this equation with the implicit method gives

$$\begin{aligned}\rho^{n+1}_{i,j,k}\frac{h^{n+1}_{i,j,k}-h^{n}_{i,j,k}}{\Delta t^n}-\frac{P^{n+1}_{i,j,k}-P^{n}_{i,j,k}}{\Delta t^n}+\rho^{n+1}_{i,j,k}\bar{u}^{n+1}_{i,j,k}\frac{\dot{h}^{n+1}_{i+\frac{1}{2},j,k}-\dot{h}^{n+1}_{i-\frac{1}{2},j,k}}{\Delta x}\\ +\rho^{n+1}_{i,j,k}\bar{v}^{n+1}_{i,j,k}\frac{\dot{h}^{n+1}_{i,j+\frac{1}{2},k}-\dot{h}^{n+1}_{i,j-\frac{1}{2},k}}{\Delta y}+\rho^{n+1}_{i,j,k}\bar{w}^{n+1}_{i,j,k}\frac{\dot{h}^{n+1}_{i,j,k+\frac{1}{2}}-\dot{h}^{n+1}_{i,j,k-\frac{1}{2}}}{\Delta z}+\\ \bar{u}^{n+1}_{i,j,k}\frac{P^{n+1}_{i+1,j,k}-P^{n+1}_{i-1,j,k}}{2\Delta x}+\bar{v}^{n+1}_{i,j,k}\frac{P^{n+1}_{i,j+1,k}-P^{n+1}_{i,j-1,k}}{2\Delta y}+\bar{w}^{n+1}_{i,j,k}\frac{P^{n+1}_{i,j,k+1}-P^{n+1}_{i,j,k-1}}{2\Delta z}=\\ (S_{energy})_{i,j,k}-\bar{u}_{i,j,k}(F_x)_{i,j,k}-\bar{v}_{i,j,k}(F_y)_{i,j,k}-\bar{w}_{i,j,k}(F_z)_{i,j,k}.\end{aligned} \tag{4.14}$$

The advecting velocities and the pressure gradient terms are taken at the next timestep as discussed in section 2.4. The pressure gradient terms are arithmetically averaged and then simplified to the form shown. The barred velocity terms and pressure terms are arithmetically averaged in space according to

$$\bar{u}^{n+1}_{i,j,k}=\frac{u^{n+1}_{i+\frac{1}{2},j,k}+u^{n+1}_{i-\frac{1}{2},j,k}}{2} \tag{4.15a}$$

$$\bar{v}^{n+1}_{i,j,k}=\frac{v^{n+1}_{i,j+\frac{1}{2},k}+v^{n+1}_{i,j-\frac{1}{2},k}}{2} \tag{4.15b}$$

$$\bar{w}^{n+1}_{i,j,k}=\frac{u^{n+1}_{i,j,k+\frac{1}{2}}+u^{n+1}_{i,j,k-\frac{1}{2}}}{2}. \tag{4.15c}$$

The dotted enthalpy terms are first-order upwinded in space according to

$$\dot{h}^{n+1}_{i+\frac{1}{2},j,k}-\dot{h}^{n+1}_{i-\frac{1}{2},j,k}=\begin{cases}h^{n+1}_{i,j,k}-h^{n+1}_{i-1,j,k} & \bar{u}^{n+1}_{i,j,k}>0\\ h^{n+1}_{i+1,j,k}-h^{n+1}_{i,j,k} & \bar{u}^{n+1}_{i,j,k}<0\end{cases} \tag{4.16a}$$

$$\dot{h}^{n+1}_{i,j+\frac{1}{2},k} - \dot{h}^{n+1}_{i,j-\frac{1}{2},k} = \begin{cases} h^{n+1}_{i,j,k} - h^{n+1}_{i,j-1,k} & \bar{v}^{n+1}_{i,j,k} > 0 \\ h^{n+1}_{i,j+1,k} - h^{n+1}_{i,j,k} & \bar{v}^{n+1}_{i,j,k} < 0 \end{cases} \tag{4.16b}$$

$$\dot{h}^{n+1}_{i,j,k+\frac{1}{2}} - \dot{h}^{n+1}_{i,j-,k\frac{1}{2}} = \begin{cases} h^{n+1}_{i,j,k} - h^{n+1}_{i,j,k-1} & \bar{w}^{n+1}_{i,j,k} > 0 \\ h^{n+1}_{i,j,k+1} - h^{n+1}_{i,j,k} & \bar{w}^{n+1}_{i,j,k} < 0. \end{cases} \tag{4.16c}$$

4.2 Conservative Solver

In addition to the implementation of the implicit solver shown in Section 4.1.2, a conservative solver was implemented that utilizes the nonlinear Shamanskii iteration presented in Section 4.1.1. This solver follows the same setup process as shown in Algorithm 12. A flag, `ConsFlag`, was implemented to select whether or not the conservative solver is run. If it is set to zero, the implicit solver is run and if it is set to one, the conservative solver is run. Note that for this flag to be set to one, the flag for the implicit solver must be set to one, and thus the flag for the nonlinear iteration must be equal to or greater than one.

4.2.1 Discretized Governing Equations

The discretized continuity equation for the conservative solver is the same as the equation presented for the implicit solver in Equation 4.2,

$$\frac{\rho^{n+1}_{i,j,k} - \rho^{n}_{i,j,k}}{\Delta t^n} + \frac{\dot{\rho}^{n+1}_{i+\frac{1}{2},j,k} u^{n+1}_{i+\frac{1}{2},j,k} - \dot{\rho}^{n+1}_{i-\frac{1}{2},j,k} u^{n+1}_{i-\frac{1}{2},j,k}}{\Delta x} + \frac{\dot{\rho}^{n+1}_{i,j+\frac{1}{2},k} v^{n+1}_{i,j+\frac{1}{2},k} - \dot{\rho}^{n+1}_{i,j-\frac{1}{2},k} v^{n+1}_{i,j-\frac{1}{2},k}}{\Delta y} + \frac{\dot{\rho}^{n+1}_{i,j,k+\frac{1}{2}} w^{n+1}_{i,j,k+\frac{1}{2}} - \dot{\rho}^{n+1}_{i,j,k-\frac{1}{2}} w^{n+1}_{i,j,k-\frac{1}{2}}}{\Delta z} = 0. \tag{4.17}$$

The dotted density terms are first-order upwinded in space as shown in Equation 3.21.

Conservation of Momentum

The conservative form of the conservation of momentum in three-dimensions, as shown in Equation 2.6, is

$$\frac{\partial \rho u}{\partial t} + \frac{\partial \rho u u}{\partial x} + \frac{\partial \rho v u}{\partial y} + \frac{\partial \rho w u}{\partial z} + \frac{\partial P}{\partial x} = F_x \tag{4.18a}$$

$$\frac{\partial \rho v}{\partial t} + \frac{\partial \rho u v}{\partial x} + \frac{\partial \rho v v}{\partial y} + \frac{\partial \rho w v}{\partial z} + \frac{\partial P}{\partial y} = F_y \tag{4.18b}$$

$$\frac{\partial \rho w}{\partial t} + \frac{\partial \rho u w}{\partial x} + \frac{\partial \rho v w}{\partial y} + \frac{\partial \rho w w}{\partial z} + \frac{\partial P}{\partial z} = F_z. \tag{4.18c}$$

Discretizing the x-momentum equation yields

$$\frac{(\bar{\rho}\, u)^{n+1}_{i+\frac{1}{2},j,k} - (\bar{\rho}\, u)^{n}_{i+\frac{1}{2},j,k}}{\Delta t^n} + \frac{(\rho\, \bar{u}\, \dot{u})^{n+1}_{i+1,j,k} - (\rho\, \bar{u}\, \dot{u}^{n+1}_{i,j,k})}{\Delta x} + \frac{(\bar{\rho}\, \bar{v}\, \dot{u})^{n+1}_{i+\frac{1}{2},j+\frac{1}{2},k} - (\bar{\rho}\, \bar{v}\, \dot{u})^{n+1}_{i+\frac{1}{2},j-\frac{1}{2},k}}{\Delta y} + \frac{(\bar{\rho}\, \bar{w}\, \dot{u})^{n+1}_{i+\frac{1}{2},j,k+\frac{1}{2}} - (\bar{\rho}\, \bar{w}\, \dot{u})^{n+1}_{i+\frac{1}{2},j,k-\frac{1}{2}}}{\Delta z} + \frac{P^{n+1}_{i+1,j,k} - P^{n+1}_{i,j,k}}{\Delta x} = (F_x)_{i+\frac{1}{2},j,k}. \tag{4.19}$$

In the momentum rate of change term, the densities are arithmetically averaged resulting in

$$(\bar{\rho}\, u)^{n+1}_{i+\frac{1}{2},j,k} = \frac{\rho^{n+1}_{i+1,j,k} - \rho^{n+1}_{i,j,k}}{2}\, u^{n+1}_{i+\frac{1}{2},j,k} \tag{4.20a}$$

$$(\bar{\rho}\, u)^{n}_{i+\frac{1}{2},j,k} = \frac{\rho^{n}_{i+1,j,k} - \rho^{n+1}_{i,j,k}}{2}\, u^{n}_{i+\frac{1}{2},j,k}. \tag{4.20b}$$

In the momentum flux terms, the barred velocities are arithmetically averaged in space and the dotted terms are first-order upwinded in space as follows

$$(\rho\, \bar{u}\, \dot{u})^{n+1}_{i+1,j,k} = \begin{cases} \rho^{n+1}_{i+1,j,k} \frac{u^{n+1}_{i+\frac{3}{2},j,k} + u^{n+1}_{i+\frac{1}{2},j,k}}{2}\, u^{n+1}_{i+\frac{1}{2},j,k} & \frac{u^{n+1}_{i+\frac{3}{2},j,k} + u^{n+1}_{i+\frac{1}{2},j,k}}{2} > 0 \\ \rho^{n+1}_{i,j,k} \frac{u^{n+1}_{i+\frac{3}{2},j,k} + u^{n+1}_{i+\frac{1}{2},j,k}}{2}\, u^{n+1}_{i+\frac{3}{2},j,k} & \frac{u^{n+1}_{i+\frac{3}{2},j,k} + u^{n+1}_{i+\frac{1}{2},j,k}}{2} < 0 \end{cases} \tag{4.21a}$$

$$(\rho\, \bar{u}\, \dot{u})^{n+1}_{i,j,k} = \begin{cases} \rho^{n+1}_{i,j,k} \frac{u^{n+1}_{i+\frac{1}{2},j,k} + u^{n+1}_{i-\frac{1}{2},j,k}}{2}\, u^{n+1}_{i-\frac{1}{2},j,k} & \frac{u^{n+1}_{i+\frac{1}{2},j,k} + u^{n+1}_{i-\frac{1}{2},j,k}}{2} > 0 \\ \rho^{n+1}_{i,j,k} \frac{u^{n+1}_{i+\frac{1}{2},j,k} + u^{n+1}_{i-\frac{1}{2},j,k}}{2}\, u^{n+1}_{i+\frac{1}{2},j,k} & \frac{u^{n+1}_{i+\frac{1}{2},j,k} + u^{n+1}_{i-\frac{1}{2},j,k}}{2} < 0 \end{cases} \tag{4.21b}$$

$$(\bar{\rho}\,\bar{v}\,\dot{u})^{n+1}_{i+\frac{1}{2},j+\frac{1}{2},k} = \begin{cases} \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i+1,j,k}+\rho^{n+1}_{i,j+1,k}+\rho^{n+1}_{i+1,j+1,k}}{4}\,\frac{v^{n+1}_{i,j+\frac{1}{2},k}+v^{n+1}_{i+1,j+\frac{1}{2},k}}{2}\,u_{i+\frac{1}{2},j,k} & \frac{v^{n+1}_{i,j+\frac{1}{2},k}+v^{n+1}_{i+1,j+\frac{1}{2},k}}{2} > 0 \\ \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i+1,j,k}+\rho^{n+1}_{i,j+1,k}+\rho^{n+1}_{i+1,j+1,k}}{4}\,\frac{v^{n+1}_{i,j+\frac{1}{2},k}+v^{n+1}_{i+1,j+\frac{1}{2},k}}{2}\,u_{i+\frac{1}{2},j+1,k} & \frac{v^{n+1}_{i,j+\frac{1}{2},k}+v^{n+1}_{i+1,j+\frac{1}{2},k}}{2} < 0 \end{cases} \tag{4.21c}$$

$$(\bar{\rho}\,\bar{v}\,\dot{u})^{n+1}_{i+\frac{1}{2},j+\frac{1}{2},k} = \begin{cases} \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i+1,j,k}+\rho^{n+1}_{i,j-1,k}+\rho^{n+1}_{i+1,j-1,k}}{4}\,\frac{v^{n+1}_{i,j-\frac{1}{2},k}+v^{n+1}_{i+1,j-\frac{1}{2},k}}{2}\,u_{i+\frac{1}{2},j-1,k} & \frac{v^{n+1}_{i,j-\frac{1}{2},k}+v^{n+1}_{i+1,j-\frac{1}{2},k}}{2} > 0 \\ \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i+1,j,k}+\rho^{n+1}_{i,j+1,k}+\rho^{n+1}_{i+1,j+1,k}}{4}\,\frac{v^{n+1}_{i,j+\frac{1}{2},k}+v^{n+1}_{i+1,j+\frac{1}{2},k}}{2}\,u_{i+\frac{1}{2},j,k} & \frac{v^{n+1}_{i,j-\frac{1}{2},k}+v^{n+1}_{i+1,j-\frac{1}{2},k}}{2} < 0 \end{cases} \tag{4.21d}$$

$$(\bar{\rho}\,\bar{w}\,\dot{u})^{n+1}_{i+\frac{1}{2},j,k+\frac{1}{2}} = \begin{cases} \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i+1,j,k}+\rho^{n+1}_{i,j,k+1}+\rho^{n+1}_{i+1,j,k+1}}{4}\,\frac{w^{n+1}_{i,j,k+\frac{1}{2}}+w^{n+1}_{i+1,j,k+\frac{1}{2}}}{2}\,u_{i+\frac{1}{2},j,k} & \frac{w^{n+1}_{i,j,k+\frac{1}{2}}+w^{n+1}_{i+1,j,k+\frac{1}{2}}}{2} > 0 \\ \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i+1,j,k}+\rho^{n+1}_{i,j,k+1}+\rho^{n+1}_{i+1,j,k+1}}{4}\,\frac{w^{n+1}_{i,j,k+\frac{1}{2}}+w^{n+1}_{i+1,j,k+\frac{1}{2}}}{2}\,u_{i+\frac{1}{2},j,k+1} & \frac{w^{n+1}_{i,j,k+\frac{1}{2}}+w^{n+1}_{i+1,j,k+\frac{1}{2}}}{2} > 0 \end{cases} \tag{4.21e}$$

$$(\bar{\rho}\,\bar{w}\,\dot{u})^{n+1}_{i+\frac{1}{2},j,k-\frac{1}{2}} = \begin{cases} \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i+1,j,k}+\rho^{n+1}_{i,j,k-1}+\rho^{n+1}_{i+1,j,k-1}}{4}\,\frac{w^{n+1}_{i,j,k-\frac{1}{2}}+w^{n+1}_{i+1,j,k-\frac{1}{2}}}{2}\,u_{i+\frac{1}{2},j,k-1} & \frac{w^{n+1}_{i,j,k-\frac{1}{2}}+w^{n+1}_{i+1,j,k-\frac{1}{2}}}{2} > 0 \\ \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i+1,j,k}+\rho^{n+1}_{i,j,k-1}+\rho^{n+1}_{i+1,j,k-1}}{4}\,\frac{w^{n+1}_{i,j,k-\frac{1}{2}}+w^{n+1}_{i+1,j,k-\frac{1}{2}}}{2}\,u_{i+\frac{1}{2},j,k} & \frac{w^{n+1}_{i,j,k-\frac{1}{2}}+w^{n+1}_{i+1,j,k-\frac{1}{2}}}{2} > 0. \end{cases} \tag{4.21f}$$

Discretizing the y-momentum equation yields

$$\frac{(\bar{\rho}\,v)^{n+1}_{i,j+\frac{1}{2},k}-(\bar{\rho}\,v)^{n}_{i,j+\frac{1}{2},k}}{\Delta t^{n}} + \frac{(\bar{\rho}\,\bar{u}\,\dot{v})^{n+1}_{i+\frac{1}{2},j+\frac{1}{2},k}-(\bar{\rho}\,\bar{u}\,\dot{v})^{n+1}_{i-\frac{1}{2},j+\frac{1}{2},k})}{\Delta x} + \frac{(\rho\,\bar{v}\,\dot{v})^{n+1}_{i,j+1,k}-(\rho\,\bar{v}\,\dot{v})^{n+1}_{i,j,k}}{\Delta y} + \frac{(\bar{\rho}\,\bar{w}\,\dot{v})^{n+1}_{i,j+\frac{1}{2},k+\frac{1}{2}}-(\bar{\rho}\,\bar{w}\,\dot{v})^{n+1}_{i,j+\frac{1}{2},k-\frac{1}{2}}}{\Delta z} + \frac{P^{n+1}_{i,j+1,k}-P^{n+1}_{i,j,k}}{\Delta y} = (F_y)_{i,j+\frac{1}{2},k}. \tag{4.22}$$

In the momentum rate of change term, the densities are arithmetically averaged resulting in

$$(\bar{\rho}\,v)^{n+1}_{i,j+\frac{1}{2},j,k} = \frac{\rho^{n+1}_{i,j+1,k}-\rho^{n+1}_{i,j,k}}{2}\,v^{n+1}_{i,j+\frac{1}{2},k} \tag{4.23a}$$

$$(\bar{\rho}\,v)^{n}_{i,j+\frac{1}{2},k} = \frac{\rho^{n}_{i,j+1,k}-\rho^{n+1}_{i,j,k}}{2}\,v^{n}_{i,j+\frac{1}{2},k}. \tag{4.23b}$$

In the momentum flux terms, the barred velocities are arithmetically averaged in space and the dotted terms are first-order upwinded in space according to

$$(\bar{\rho}\,\bar{u}\,\dot{v})^{n+1}_{i+\frac{1}{2},j+\frac{1}{2},k} = \begin{cases} \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i+1,j,k}+\rho^{n+1}_{i,j+1,k}+\rho^{n+1}_{i+1,j+1,k}}{4}\,\frac{u^{n+1}_{i+\frac{1}{2},j+1,k}+u^{n+1}_{i+\frac{1}{2},j,k}}{2}\,v^{n+1}_{i,j+\frac{1}{2},k} & \frac{u^{n+1}_{i+\frac{1}{2},j+1,k}+u^{n+1}_{i+\frac{1}{2},j,k}}{2} > 0 \\ \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i+1,j,k}+\rho^{n+1}_{i,j+1,k}+\rho^{n+1}_{i+1,j+1,k}}{4}\,\frac{u^{n+1}_{i+\frac{1}{2},j+1,k}+u^{n+1}_{i+\frac{1}{2},j,k}}{2}\,v^{n+1}_{i+1,j+\frac{1}{2},k} & \frac{u^{n+1}_{i+\frac{1}{2},j+1,k}+u^{n+1}_{i+\frac{1}{2},j,k}}{2} < 0 \end{cases} \tag{4.24a}$$

$$(\bar{\rho}\,\bar{u}\,\dot{v})^{n+1}_{i-\frac{1}{2},j+\frac{1}{2},k} = \begin{cases} \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i-1,j,k}+\rho^{n+1}_{i,j+1,k}+\rho^{n+1}_{i-1,j+1,k}}{4}\,\frac{u^{n+1}_{i-\frac{1}{2},j+1,k}+u^{n+1}_{i-\frac{1}{2},j,k}}{2}\,v^{n+1}_{i-1,j+\frac{1}{2},k} & \frac{u^{n+1}_{i-\frac{1}{2},j+1,k}+u^{n+1}_{i-\frac{1}{2},j,k}}{2} > 0 \\ \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i-1,j,k}+\rho^{n+1}_{i,j+1,k}+\rho^{n+1}_{i-1,j+1,k}}{4}\,\frac{u^{n+1}_{i-\frac{1}{2},j+1,k}+u^{n+1}_{i-\frac{1}{2},j,k}}{2}\,v^{n+1}_{i,j+\frac{1}{2},k} & \frac{u^{n+1}_{i-\frac{1}{2},j+1,k}+u^{n+1}_{i-\frac{1}{2},j,k}}{2} < 0 \end{cases} \tag{4.24b}$$

$$(\rho\,\bar{v}\,\dot{v}_{i,j+1,k})^{n+1} = \begin{cases} \rho^{n+1}_{i,j+1,k}\,\frac{v^{n+1}_{i,j+\frac{3}{2},k}+v^{n+1}_{i,j+\frac{1}{2},k}}{2}\,v^{n+1}_{i,j+\frac{1}{2},k} & \frac{v^{n+1}_{i,j+\frac{3}{2},k}+v^{n+1}_{i,j+\frac{1}{2},k}}{2} > 0 \\ \rho^{n+1}_{i,j+1,k}\,\frac{v^{n+1}_{i,j+\frac{3}{2},k}+v^{n+1}_{i,j+\frac{1}{2},k}}{2}\,v^{n+1}_{i,j+\frac{3}{2},k} & \frac{v^{n+1}_{i,j+\frac{3}{2},k}+v^{n+1}_{i,j+\frac{1}{2},k}}{2} < 0 \end{cases} \tag{4.24c}$$

$$(\rho\,\bar{v}\,\dot{v}_{i,j,k})^{n+1} = \begin{cases} \rho^{n+1}_{i,j,k}\,\frac{v^{n+1}_{i,j+\frac{1}{2},k}+v^{n+1}_{i,j-\frac{1}{2},k}}{2}\,v^{n+1}_{i,j-\frac{1}{2},k} & \frac{v^{n+1}_{i,j+\frac{1}{2},k}+v^{n+1}_{i,j-\frac{1}{2},k}}{2} > 0 \\ \rho^{n+1}_{i,j,k}\,\frac{v^{n+1}_{i,j+\frac{1}{2},k}+v^{n+1}_{i,j-\frac{1}{2},k}}{2}\,v^{n+1}_{i,j+\frac{1}{2},k} & \frac{v^{n+1}_{i,j+\frac{1}{2},k}+v^{n+1}_{i,j-\frac{1}{2},k}}{2} < 0 \end{cases} \tag{4.24d}$$

$$(\bar{\rho}\,\bar{w}\,\dot{v})^{n+1}_{i,j+\frac{1}{2},k+\frac{1}{2}} = \begin{cases} \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i,j,k+1}+\rho^{n+1}_{i,j+1,k}+\rho^{n+1}_{i,j+1,k+1}}{4}\,\frac{w^{n+1}_{i,j+1,k+\frac{1}{2}}+w^{n+1}_{i,j,k+\frac{1}{2}}}{2}\,v^{n+1}_{i,j+\frac{1}{2},k} & \frac{w^{n+1}_{i,j+1,k+\frac{1}{2}}+w^{n+1}_{i,j,k+\frac{1}{2}}}{2} < 0 \\ \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i,j,k+1}+\rho^{n+1}_{i,j+1,k}+\rho^{n+1}_{i,j+1,k+1}}{4}\,\frac{w^{n+1}_{i,j+1,k-\frac{1}{2}}+w^{n+1}_{i,j,k-\frac{1}{2}}}{2}\,v^{n+1}_{i,j+\frac{1}{2},k+1} & \frac{w^{n+1}_{i,j+1,k-\frac{1}{2}}+w^{n+1}_{i,j,k-\frac{1}{2}}}{2} < 0 \end{cases} \tag{4.24e}$$

$$(\bar{\rho}\,\bar{w}\,\dot{v})^{n+1}_{i,j+\frac{1}{2},k-\frac{1}{2}} = \begin{cases} \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i,j,k-1}+\rho^{n+1}_{i,j+1,k}+\rho^{n+1}_{i,j+1,k-1}}{4}\,\frac{w^{n+1}_{i,j+1,k-\frac{1}{2}}+w^{n+1}_{i,j,k-\frac{1}{2}}}{2}\,v^{n+1}_{i,j+\frac{1}{2},k-1} & \frac{w^{n+1}_{i,j+1,k-\frac{1}{2}}+w^{n+1}_{i,j,k-\frac{1}{2}}}{2} < 0 \\ \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i,j,k-1}+\rho^{n+1}_{i,j+1,k}+\rho^{n+1}_{i,j+1,k-1}}{4}\,\frac{w^{n+1}_{i,j+1,k-\frac{1}{2}}+w^{n+1}_{i,j,k-\frac{1}{2}}}{2}\,v^{n+1}_{i,j+\frac{1}{2},k} & \frac{w^{n+1}_{i,j+1,k-\frac{1}{2}}+w^{n+1}_{i,j,k-\frac{1}{2}}}{2} < 0. \end{cases} \tag{4.24f}$$

Discretizing the z-momentum equation yields

$$\frac{(\bar{\rho}\,w)^{n+1}_{i,j,k+\frac{1}{2}}-(\bar{\rho}\,w)^{n}_{i,j,k+\frac{1}{2}}}{\Delta t^n}+\frac{(\bar{\rho}\,\bar{u}\,\dot{w})^{n+1}_{i+\frac{1}{2},j.k+\frac{1}{2}}-(\bar{\rho}\,\bar{u}\,\dot{w})^{n+1}_{i-\frac{1}{2},j,k+\frac{1}{2}})}{\Delta x}+\frac{(\bar{\rho}\,\bar{v}\,\dot{w})^{n+1}_{i,j+\frac{1}{2},k+\frac{1}{2}}-(\bar{\rho}\,\bar{v}\,\dot{w})^{n+1}_{i,j-\frac{1}{2},k+\frac{1}{2}}}{\Delta y}+$$
$$\frac{(\rho\,\bar{w}\,\dot{w})^{n+1}_{i,j,k+1}-(\rho\,\bar{w}\,\dot{w})^{n+1}_{i,j,k}}{\Delta z}+\frac{P^{n+1}_{i,j,k+1}-P^{n+1}_{i,j,k}}{\Delta z}=(F_z)_{i,j,k+\frac{1}{2}}. \tag{4.25}$$

In the momentum rate of change term, the densities are arithmetically averaged resulting

in

$$(\bar{\rho}\, w)^{n+1}_{i,j,k+\frac{1}{2},j} = \frac{\rho^{n+1}_{i,j,k+1} - \rho^{n+1}_{i,j,k}}{2}\, w^{n+1}_{i,j,k+\frac{1}{2}} \tag{4.26a}$$

$$(\bar{\rho}\, w)^{n}_{i,j,k+\frac{1}{2}} = \frac{\rho^{n}_{i,j,k+1} - \rho^{n+1}_{i,j,k}}{2}\, w^{n}_{i,j,k+\frac{1}{2}}. \tag{4.26b}$$

In the momentum flux terms, the barred velocities are arithmetically averaged in space and the dotted terms are first-order upwinded in space according to

$$(\bar{\rho}\,\bar{u}\,\dot{w})^{n+1}_{i+\frac{1}{2},j,k+\frac{1}{2}} = \begin{cases} \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i+1,j,k}+\rho^{n+1}_{i,j,k+1}+\rho^{n+1}_{i+1,j,k+1}}{4}\, \frac{u^{n+1}_{i+\frac{1}{2},j,k+1}+u^{n+1}_{i+\frac{1}{2},j,k}}{2}\, w^{n+1}_{i,j,k+\frac{1}{2}} & \frac{u^{n+1}_{i+\frac{1}{2},j,k+1}+u^{n+1}_{i+\frac{1}{2},j,k}}{2} > 0 \\ \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i+1,j,k}+\rho^{n+1}_{i,j,k+1}+\rho^{n+1}_{i+1,j,k+1}}{4}\, \frac{u^{n+1}_{i+\frac{1}{2},j,k+1}+u^{n+1}_{i+\frac{1}{2},j,k}}{2}\, w^{n+1}_{i+1,j,k+\frac{1}{2}} & \frac{u^{n+1}_{i+\frac{1}{2},j,k+1}+u^{n+1}_{i+\frac{1}{2},j,k}}{2} < 0 \end{cases} \tag{4.27a}$$

$$(\bar{\rho}\,\bar{u}\,\dot{w})^{n+1}_{i-\frac{1}{2},j,k+\frac{1}{2}} = \begin{cases} \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i-1,j,k}+\rho^{n+1}_{i,j,k+1}+\rho^{n+1}_{i-1,j,k+1}}{4}\, \frac{u^{n+1}_{i-\frac{1}{2},j,k+1}+u^{n+1}_{i-\frac{1}{2},j,k}}{2}\, w^{n+1}_{i-1,j,k+\frac{1}{2}} & \frac{u^{n+1}_{i-\frac{1}{2},j,k+1}+u^{n+1}_{i-\frac{1}{2},j,k}}{2} > 0 \\ \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i-1,j,k}+\rho^{n+1}_{i,j,k+1}+\rho^{n+1}_{i-1,j,k+1}}{4}\, \frac{u^{n+1}_{i-\frac{1}{2},j,k+1}+u^{n+1}_{i-\frac{1}{2},j,k}}{2}\, w^{n+1}_{i,j,k+\frac{1}{2}} & \frac{u^{n+1}_{i-\frac{1}{2},j,k+1}+u^{n+1}_{i-\frac{1}{2},j,k}}{2} < 0 \end{cases} \tag{4.27b}$$

$$(\bar{\rho}\,\bar{v}\,\dot{w})^{n+1}_{i,j+\frac{1}{2},k+\frac{1}{2}} = \begin{cases} \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i,j+1,k}+\rho^{n+1}_{i,j,k+1}+\rho^{n+1}_{i,j+1,k+1}}{4}\, \frac{v^{n+1}_{i,j+\frac{1}{2},k+1}+v^{n+1}_{i,j+\frac{1}{2},k}}{2}\, w^{n+1}_{i,j,k+\frac{1}{2}} & \frac{v^{n+1}_{i,j+\frac{1}{2},k+1}+v^{n+1}_{i,j+\frac{1}{2},k}}{2} > 0 \\ \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i,j+1,k}+\rho^{n+1}_{i,j,k+1}+\rho^{n+1}_{i,j+1,k+1}}{4}\, \frac{v^{n+1}_{i,j+\frac{1}{2},k+1}+v^{n+1}_{i,j+\frac{1}{2},k}}{2}\, w^{n+1}_{i,j+1,k+\frac{1}{2}} & \frac{v^{n+1}_{i,j+\frac{1}{2},k+1}+v^{n+1}_{i,j+\frac{1}{2},k}}{2} < 0 \end{cases} \tag{4.27c}$$

$$(\bar{\rho}\,\bar{v}\,\dot{w})^{n+1}_{i,j-\frac{1}{2},k+\frac{1}{2}} = \begin{cases} \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i,j-1,k}+\rho^{n+1}_{i,j,k+1}+\rho^{n+1}_{i,j-1,k+1}}{4}\, \frac{v^{n+1}_{i,j-\frac{1}{2},k+1}+v^{n+1}_{i,j-\frac{1}{2},k}}{2}\, w^{n+1}_{i,j-1,k+\frac{1}{2}} & \frac{v^{n+1}_{i,j-\frac{1}{2},k+1}+v^{n+1}_{i,j-\frac{1}{2},k}}{2} > 0 \\ \frac{\rho^{n+1}_{i,j,k}+\rho^{n+1}_{i,j-1,k}+\rho^{n+1}_{i,j,k+1}+\rho^{n+1}_{i,j-1,k+1}}{4}\, \frac{v^{n+1}_{i,j-\frac{1}{2},k+1}+v^{n+1}_{i,j-\frac{1}{2},k}}{2}\, w^{n+1}_{i,j,k+\frac{1}{2}} & \frac{v^{n+1}_{i,j-\frac{1}{2},k+1}+v^{n+1}_{i,j-\frac{1}{2},k}}{2} < 0 \end{cases} \tag{4.27d}$$

$$(\bar{\rho}\,\bar{v}\,\dot{w})^{n+1}_{i,j,k+1} = \begin{cases} \rho^{n+1}_{i,j,k+1}\, \frac{w^{n+1}_{i,j,k+\frac{3}{2}}+w^{n+1}_{i,j,k+\frac{1}{2},k}}{2}\, w^{n+1}_{i,j,k+\frac{1}{2}} & \frac{w^{n+1}_{i,j,k+\frac{3}{2}}+w^{n+1}_{i,j,k+\frac{1}{2},k}}{2} > 0 \\ \rho^{n+1}_{i,j,k+1}\, \frac{w^{n+1}_{i,j,k+\frac{3}{2}}+w^{n+1}_{i,j,k+\frac{1}{2},k}}{2}\, w^{n+1}_{i,j,k+\frac{3}{2}} & \frac{w^{n+1}_{i,j,k+\frac{3}{2}}+w^{n+1}_{i,j,k+\frac{1}{2},k}}{2} < 0 \end{cases} \tag{4.27e}$$

$$(\bar{\rho}\,\bar{v}\,\dot{w})^{n+1}_{i,j,k} = \begin{cases} \rho^{n+1}_{i,j,k+1}\, \frac{w^{n+1}_{i,j,k+\frac{1}{2}}+w^{n+1}_{i,j,k-\frac{1}{2},k}}{2}\, w^{n+1}_{i,j,k-\frac{1}{2}} & \frac{w^{n+1}_{i,j,k+\frac{1}{2}}+w^{n+1}_{i,j,k-\frac{1}{2},k}}{2} > 0 \\ \rho^{n+1}_{i,j,k+1}\, \frac{w^{n+1}_{i,j,k+\frac{1}{2}}+w^{n+1}_{i,j,k-\frac{1}{2},k}}{2}\, w^{n+1}_{i,j,k+\frac{1}{2}} & \frac{w^{n+1}_{i,j,k+\frac{1}{2}}+w^{n+1}_{i,j,k-\frac{1}{2},k}}{2} < 0. \end{cases} \tag{4.27f}$$

Conservation of Enthalpy

The conservative form of the conservation of enthalpy equation in three dimensions, as presented in Equation 2.28, is

$$\frac{\partial \rho h}{\partial t} - \frac{\partial P}{\partial t} + \frac{\partial \rho h u}{\partial x} + \frac{\partial \rho h v}{\partial y} + \frac{\partial \rho h w}{\partial z} - u\frac{\partial P}{\partial x} - v\frac{\partial P}{\partial y} - w\frac{\partial P}{\partial z} = S_{energy} - \vec{V}\cdot\vec{F}. \quad (4.28)$$

Discretizing this equation results in

$$\begin{aligned}
&\frac{(\rho h)^{n+1}_{i,j,k} - (\rho h)^{n}_{i,j,k}}{\Delta t^n} - \frac{P^{n+1}_{i,j,k} - P^{n}_{i,j,k}}{\Delta t^n} + \frac{\left((\rho\dot{h})u\right)^{n+1}_{i+\frac{1}{2},j,k} - \left((\rho\dot{h})u\right)^{n+1}_{i-\frac{1}{2},j,k}}{\Delta x} + \\
&\frac{\left((\rho\dot{h})v\right)^{n+1}_{i,j+\frac{1}{2},k} - \left((\rho\dot{h})v\right)^{n+1}_{i,j-\frac{1}{2},k}}{\Delta y} + \frac{\left((\rho\dot{h})w\right)^{n+1}_{i,j,k+\frac{1}{2}} - \left((\rho\dot{h})w\right)^{n+1}_{i,j,k-\frac{1}{2}}}{\Delta z} - \bar{u}^{n+1}_{i,j,k}\frac{P^{n+1}_{i+1,j,k} - P^{n+1}_{i-1,j,k}}{2\Delta x} \\
&\qquad - \bar{v}^{n+1}_{i,j,k}\frac{P^{n+1}_{i,j+1,k} - P^{n+1}_{i,j-1,k}}{2\Delta y} - \bar{w}^{n+1}_{i,j,k}\frac{P^{n+1}_{i,j+1,k} - P^{n+1}_{i,j-1,k}}{2\Delta z} = \\
&\qquad (S_{energy})_{i,j,k} + \bar{u}^{n+1}_{i,j,k}(F_x)_{i,j,k} + \bar{v}^{n+1}_{i,j,k}(F_y)_{i,j,k} + \bar{w}^{n+1}_{i,j,k}(F_z)_{i,j,k}. \quad (4.29)
\end{aligned}$$

The rate of change of the enthalpy term is discretized according to

$$(\rho h)^{n+1}_{i,j,k} = \rho^{n+1}_{i,j,k} h^{n+1}_{i,j,k} \quad (4.30a)$$

$$(\rho h)^{n}_{i,j,k} = \rho^{n+1}_{i,j,k} h^{n}_{i,j,k}. \quad (4.30b)$$

The barred velocity terms are arithmetically averaged in space by

$$\bar{u}^{n+1}_{i,j,k} = \frac{u^{n+1}_{i+\frac{1}{2},j,k} + u^{n+1}_{i-\frac{1}{2},j,k}}{2} \quad (4.31a)$$

$$\bar{v}^{n+1}_{i,j,k} = \frac{v^{n+1}_{i,j+\frac{1}{2},k} + v^{n+1}_{i,j-\frac{1}{2},k}}{2} \quad (4.31b)$$

$$\bar{w}^{n+1}_{i,j,k} = \frac{w^{n+1}_{i,j,k+\frac{1}{2}} + w^{n+1}_{i,j,k-\frac{1}{2}}}{2}. \quad (4.31c)$$

The enthalpy advection terms are discretized according to

$$\left((\dot{\rho h})u\right)^{n+1}_{i+\frac{1}{2},j,k} = \begin{cases} \rho^{n+1}_{i,j,k} h^{n+1}_{i,j,k} u^{n+1}_{i+\frac{1}{2},j,k} & u^{n+1}_{i+\frac{1}{2},j,k} > 0 \\ \rho^{n+1}_{i+1,j,k} h^{n+1}_{i+1,j,k} u^{n+1}_{i+\frac{1}{2},j,k} & u^{n+1}_{i+\frac{1}{2},j,k} > 0 \end{cases} \tag{4.32a}$$

$$\left((\dot{\rho h})u\right)^{n+1}_{i-\frac{1}{2},j,k} = \begin{cases} \rho^{n+1}_{i-1,j,k} h^{n+1}_{i-1,j,k} u^{n+1}_{i-\frac{1}{2},j,k} & u^{n+1}_{i-\frac{1}{2},j,k} > 0 \\ \rho^{n+1}_{i,j,k} h^{n+1}_{i,j,k} u^{n+1}_{i-\frac{1}{2},j,k} & u^{n+1}_{i-\frac{1}{2},j,k} > 0 \end{cases} \tag{4.32b}$$

$$\left((\dot{\rho h})v\right)^{n+1}_{i,j+\frac{1}{2},k} = \begin{cases} \rho^{n+1}_{i,j,k} h^{n+1}_{i,j,k} v^{n+1}_{i,j+\frac{1}{2},k} & v^{n+1}_{i,j+\frac{1}{2},k} > 0 \\ \rho^{n+1}_{i,j+1,k} h^{n+1}_{i,j+1,k} v^{n+1}_{i,j+\frac{1}{2},k} & v^{n+1}_{i,j+\frac{1}{2},k} > 0 \end{cases} \tag{4.32c}$$

$$\left((\dot{\rho h})v\right)^{n+1}_{i,j-\frac{1}{2},k} = \begin{cases} \rho^{n+1}_{i,j-1,k} h^{n+1}_{i,j-1,k} v^{n+1}_{i,j-\frac{1}{2},k} & v^{n+1}_{i,j-\frac{1}{2},k} > 0 \\ \rho^{n+1}_{i,j,k} h^{n+1}_{i,j,k} v^{n+1}_{i,j-\frac{1}{2},k} & v^{n+1}_{i,j-\frac{1}{2},k} > 0 \end{cases} \tag{4.32d}$$

$$\left((\dot{\rho h})w\right)^{n+1}_{i,j,k+\frac{1}{2}} = \begin{cases} \rho^{n+1}_{i,j,k} h^{n+1}_{i,j,k} w^{n+1}_{i,j,k+\frac{1}{2}} & w^{n+1}_{i,j,k+\frac{1}{2}} > 0 \\ \rho^{n+1}_{i,j,k+1} h^{n+1}_{i,j,k+1} w^{n+1}_{i,j,k+\frac{1}{2}} & w^{n+1}_{i,j,k+\frac{1}{2}} > 0 \end{cases} \tag{4.32e}$$

$$\left((\dot{\rho h})w\right)^{n+1}_{i,j,k-\frac{1}{2}} = \begin{cases} \rho^{n+1}_{i,j,k-1} h^{n+1}_{i,j,k-1} w^{n+1}_{i,j,k-\frac{1}{2}} & w^{n+1}_{i,j,k-\frac{1}{2}} > 0 \\ \rho^{n+1}_{i,j,k} h^{n+1}_{i,j,k} w^{n+1}_{i,j,k\frac{1}{2},k} & w^{n+1}_{i,j,k-\frac{1}{2}} > 0. \end{cases} \tag{4.32f}$$

4.2.2 Discussion of Conservative and Non-Conservative Solvers

In most cases, results from a solver for the conservative form of a set of governing equations will be very similar to the results from a solver for the non-conservative form of the same equations. The main difference between the two forms comes in simulating systems with a shock or physical discontinuity, where non-conservative solvers struggle (14). The reason that simulating a flow that contains a physical discontinuity is difficult with the non-conservative form of the equations can be seen from the derivation in Section 2.2.2. The product rule is applied in the derivation of the non-conservative momentum and enthalpy equations to derivative terms to expand the derivatives to allow for the derivation. However, the application of the product rule requires both functions to be continuous and differentiable over the domain the product rule is being applied on. In flows with shocks, pressure boundaries, or other physical discontinuities, this assumption fails and

the non-conservative form will not result in an accurate depiction of the flow.

Another way of describing the difference between the conservative form and the non-conservative form is that the conservative form acts to conserve the quantity of interest while this is not necessarily true for the non-conservative form. For example, the momentum in a domain is conserved exactly by the conservative form of the equation, as the advective terms represent the net flux out of the domain surface. The momentum in the non-conservative form is not conserved exactly.

As an example to visualize this effect in numerical terms, the discretized form of the momentum flux in the x-direction in the x-momentum equation, in the conservative and non-conservative form respectively, are

$$\frac{(\rho\,\bar{u}\,\dot{u})_{i+1,j,k}-(\rho\,\bar{u}\,\dot{u}_{i,j,k})}{\Delta x} \tag{4.33a}$$

$$\bar{\rho}_{i+\frac{1}{2},j,k}\,u_{i+\frac{1}{2},j,k}\frac{\dot{u}_{i+1,j,k}-\dot{u}_{i,j,k}}{\Delta x}. \tag{4.33b}$$

The timestep at which the values are taken is neglected. Two momentum cells will be analyzed, the momentum cell centered at $(i+\frac{1}{2},j,k)$ and the momentum cell centered at $(i+\frac{3}{2},j,k)$. Specifically the shared surface between the two momentum cells, $(i+1,j,k)$ and the momentum that is advected across this boundary is analyzed. First, for the conservative solver, the momentum flux out of cell $(i+\frac{1}{2},j,k)$ is

$$\rho\,\bar{u}\,\dot{u}_{i,j,k}. \tag{4.34}$$

Analyzing the discretized momentum flux for cell $(i+\frac{3}{2},j,k)$ yields

$$\frac{(\rho\,\bar{u}\,\dot{u})_{i+2,j,k}-(\rho\,\bar{u}\,\dot{u}_{i+1,j,k})}{\Delta x} \tag{4.35}$$

and the momentum flux into this cell from cell $(i+\frac{1}{2},j,k)$ is

$$\rho\,\bar{u}\,\dot{u}_{i,j,k}. \tag{4.36}$$

Thus, the momentum flux between cells is conserved exactly.

Looking at the non-conservative form, the momentum flux out of cell $(i+\frac{1}{2},j,k)$ is

$$\bar{\rho}_{i+\frac{1}{2},j,k}\,u_{i+\frac{1}{2},j,k}\,\dot{u}_{i+1,j,k}. \tag{4.37}$$

Analyzing the discretized momentum flux for cell $(i+\frac{3}{2}, j, k)$ yields

$$\bar{\rho}_{i+\frac{3}{2},j,k} u_{i+\frac{3}{2},j,k} \frac{\dot{u}_{i+2,j,k} - \dot{u}_{i+1,j,k}}{\Delta x}, \tag{4.38}$$

and the momentum flux into this cell from cell $(i+\frac{1}{2}, j, k)$ is

$$\bar{\rho}_{i+\frac{3}{2},j,k} u_{i+\frac{3}{2},j,k} \dot{u}_{i+1,j,k}. \tag{4.39}$$

The index on the density and the advecting velocity are not the same in Equation 4.37 and Equation 4.39, so it can be seen that the momentum is not conserved across the domain, even for an arbitrary action of the barred and dotted variables. If there is a sharp density or velocity gradient, the momentum will either be lost or generated artificially between cells or domains. In most scenarios, the gradient in the momentum will be small, as will the velocity gradient, so the artificial change in the momentum is small. However, when these gradients are larger, such as near the physical discontinuities mentioned above, the solution will be affected significantly by this artificial source and loss of momentum. The example here focuses on the momentum flux in the x-direction in the x-momentum equation, but it is also true for all the momentum flux terms and for the enthalpy advection terms as well.

4.3 Verification of the Implicit and Conservative Solvers

The preliminary verification work for the nonlinear iteration introduced in Section 4.1.1, as well as the implicit solver and conservative solver that utilizes it, is presented in this section. The verification work for the nonlinear iteration is presented in Section 4.3.1 and the verification of the implicit solver and conservative solver is presented in the following subsections.

4.3.1 Verification of the Nonlinear Iteration

To verify the function of the nonlinear iteration, a test case was run with a strict convergence criterion for a channel with gravity and heat source terms. The implicit solver was used for this case with a chord method, $m = \infty$. The two metrics used to check the convergence of the method were the l^2-norm of the residual vector, $|F|_2$ and the l^2-norm of the error in

the solution vector, $|e|_2$. These norms are defined as,

$$|e|_2 = \sqrt{\sum_i |x_i - x_i^*|^2}, \tag{4.40}$$

$$|F|_2 = \sqrt{\sum_i F_i^2}, \tag{4.41}$$

where x_i is the i^{th} term in the solution vector, x^* is the root of the defined problem, and F_i is the i^{th} term in the residual vector.

The results of the convergence of the chord method are shown in the Figure 4.1 and Figure 4.2 on a semi-log plot. The expected behavior of the chord method is to appear as a straight line on a semi-log plot and both the norm of the error and the norm of the residuals to appear to match that behavior. The calculated order from the two plots, using the equation

$$\alpha = \frac{log(\frac{e_{n+1}}{e_n})}{log(\frac{e_n}{e_{n-1}})}, \tag{4.42}$$

is 1.0245 for the order of convergence of the norm of the error and 0.947 for the order of convergence of the norm of the residuals. Both of these values agree well with the expected order of convergence, $\alpha = 1$.

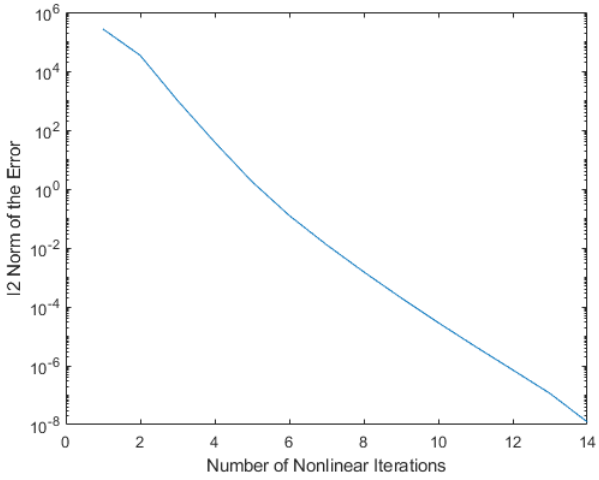


Figure 4.1: l2 Norm of the Error in the Solution Vector in the Chord Method

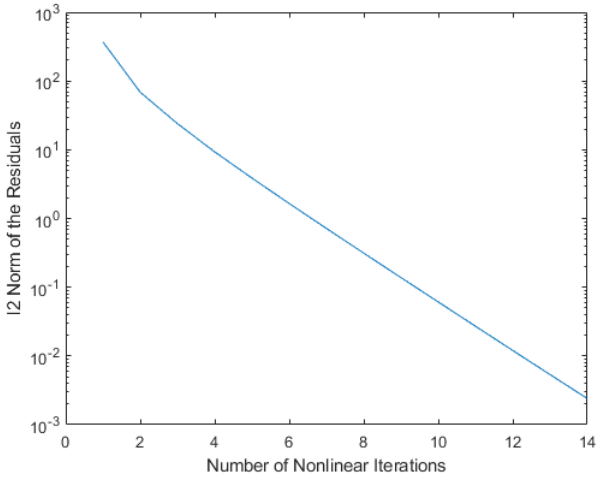


Figure 4.2: l2 Norm of the Residual Vector in the Chord Method

The results for the convergence discussed here are for the chord method only. This is due to the fact that inexact Newton methods, which the nonlinear iteration implemented here is, only guarantee linear convergence for a sequence of forcing factors that do not converge to zero. The forcing factors in this solver are not being altered in PETSc so superlinear convergence is not expected. Therefore the analysis of the convergence of the Shamanskii method gives the same linear convergence for any value of m. Once forcing factors that converge to zero are implemented the nonlinear iteration should be tested again for q-superlinear convergence behavior.

4.3.2 Sourceless Verification Case

The verification cases for the implicit and conservative solvers are the same as the cases for the semi-implicit solver as shown in section 3.5. The first case is the sourceless test case

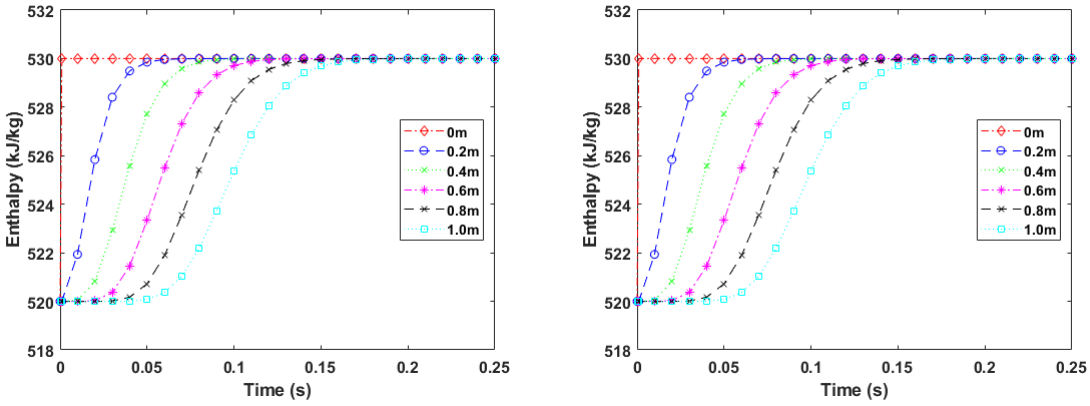
The problem specifications for the sourceless test case is shown in the table below which is the same as presented in Table 3.3. These specifications presented are for inlet flow in the x-direction.

For this case at steady-state, the pressure should be equal to 0 Pa throughout the domain and the enthalpy should be equal to 530 kJ/kg throughout the domain. The plots in Figure 4.3a and Figure 4.3b show the enthalpy distribution for the sourceless case and the plots in Figure 4.4a and Figure 4.4b show the pressure distribution for the sourceless case. It can

Table 4.1: Initial Problem Specification

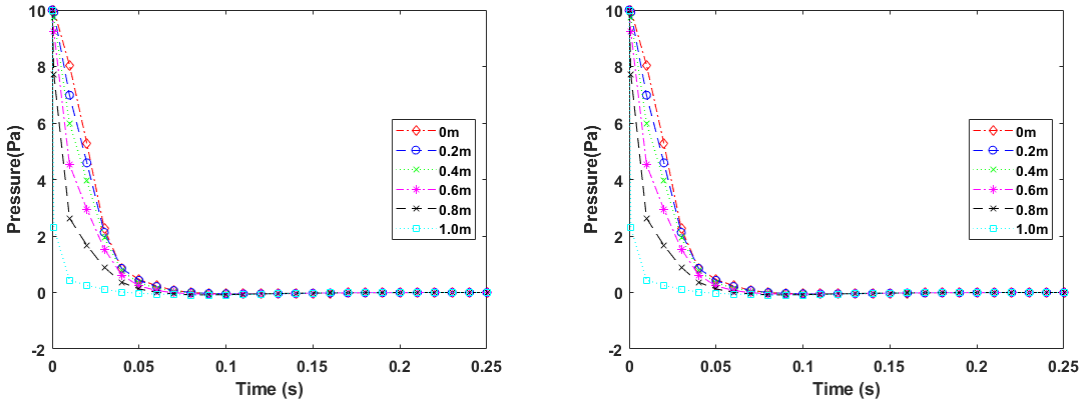
Channel Height (m)	0.5	Maximum Time steps	5000
Channel Length (m)	0.5	Timestep Length (s)	0.001
Channel Width (m)	0.5	Outlet Pressure (Pa)	0
x-direction Mesh Spacing (m)	0.025	Inlet Velocity Vector (m/s)	(5.0, 0.0, 0.0)
y-direction Mesh Spacing (m)	0.05	Inlet Enthalpy (kJ/kg)	530
z-direction Mesh Spacing (m)	0.05	Initial Pressure (Pa)	10
Steady-State Convergence Criterion	0.0001	Initial Enthalpy (kJ/kg)	520

be seen that both the enthalpy and the pressure distribution rapidly converge to the analytic values in both the implicit and the conservative case. It can be seen that there is some numerical diffusion in the enthalpy distribution over the domain, and the diffusion is more pronounced later in the channel. This is expected from the use of first-order upwinding in this solver. When comparing to the semi-implicit solver, there is slightly more diffusion present in these cases. This is expected as it is known that the more implicit the solver the more diffusive the results are expected to be.



(a) Implicit Case (b) Conservative Case

Figure 4.3: Enthalpy Distribution in the Sourceless Case



(a) Implicit Case (b) Conservative Case

Figure 4.4: Enthalpy Distribution in the Sourceless Case

4.3.3 Heat Source Verification Case

The second test case was designed to verify the behavior of the heat source term in the enthalpy equation. The problem specifications are given in the table below, which is the same as presented in Table 3.4. A uniform volumetric heat generation of 800,000 W/m^3 is set throughout the fluid domain.

Table 4.2: Initial Problem Specification

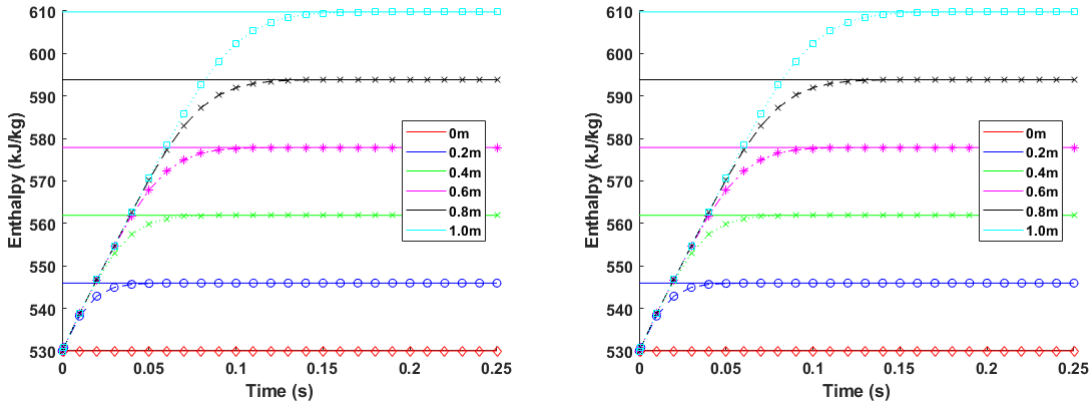
Channel Height (m)	0.5	Maximum Time steps	5000
Channel Length (m)	0.5	Timestep Length (s)	0.001
Channel Width (m)	0.5	Outlet Pressure (Pa)	0
x-direction Mesh Spacing (m)	0.025	Inlet Velocity Vector (m/s)	(5.0, 0.0, 0.0)
y-direction Mesh Spacing (m)	0.05	Inlet Enthalpy (kJ/kg)	530
z-direction Mesh Spacing (m)	0.05	Initial Pressure (Pa)	0
Steady-State Convergence Criterion	0.0001	Initial Enthalpy (kJ/kg)	530

The plots in Figure 4.5a and Figure 4.5b show the enthalpy distribution from the implicit and conservative solver, respectively, for this case. The solid lines represent the analytic steady-state value at each height. As can be seen from the figure, the enthalpy at each height converges to the analytic value. Also, the enthalpy rate of change when there is no enthalpy

gradient should be

$$\frac{\partial h}{\partial t} = \frac{S_{ht}}{\rho} = \frac{800000}{1000} = 800 \frac{kJ/kg}{s} \tag{4.43}$$

and the slope of the linear enthalpy rise for the early timesteps is approximately 800 $\frac{kJ/kg}{s}$ for both solvers.



(a) Implicit Case (b) Conservative Case

Figure 4.5: Enthalpy Distribution in the Heat Source Case

4.3.4 Gravity Verification Case

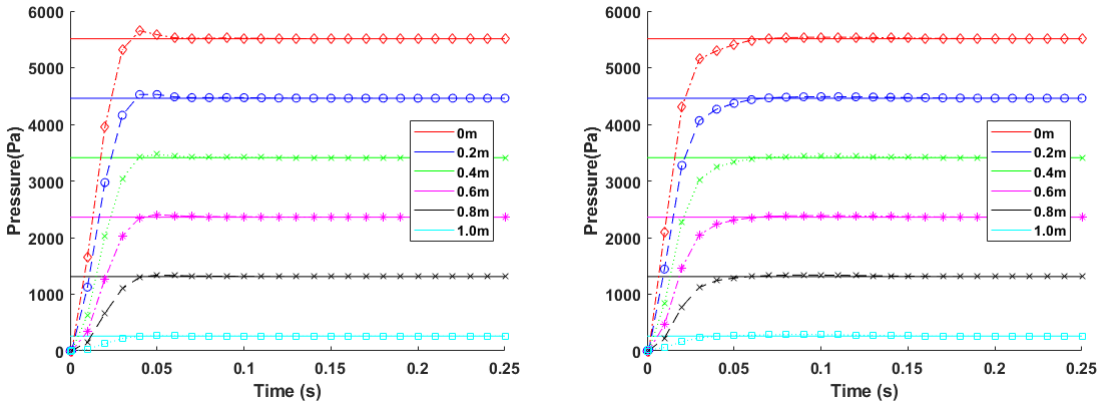
The third test case was designed to verify the behavior of the gravitational source term in the pressure and enthalpy equations. The problem specifications are given in the table below which is the same as presented in Table 3.5. A gravitational term of 10 m/s^2 is present in the negative x-direction.

For a density of 1000 kg/m^3, the pressure gradient will be 10,000 Pa/m. The plots in Figure 4.6a and Figure 4.6b show the pressure distribution for this case compared to the analytic result, which is shown by the solid horizontal lines at each height. It should be noted that the boundary condition of 0 Pa is set in a ghost/boundary node, so it is specified at 0.525m. The value at each node can be seen to converge to the analytic result. It can be seen that the pressure overshoots the analytic result, similar to the results from the semi-implicit case in Figure 3.6. This overshoot is slightly less severe in the implicit case

Table 4.3: Initial Problem Specification

Channel Height (m)	0.5	Maximum Time steps	5000
Channel Length (m)	0.5	Timestep Length (s)	0.001
Channel Width (m)	0.5	Outlet Pressure (Pa)	0
x-direction Mesh Spacing (m)	0.025	Inlet Velocity Vector (m/s)	(5.0, 0.0, 0.0)
y-direction Mesh Spacing (m)	0.05	Inlet Enthalpy (kJ/kg)	530
z-direction Mesh Spacing (m)	0.05	Initial Pressure (Pa)	0
Steady-State Convergence Criterion	0.0001	Initial Enthalpy (kJ/kg)	530

than it was in the semi-implicit case. The overshoot in the conservative case is extremely small, especially when compared to the implicit and semi-implicit results. It should also be noted that the conservative solver approaches the analytic result quicker in the first two data points presented, which correspond to the first 20 timesteps or first 0.02 seconds of the simulation. The reason for this overshoot is not certain at this time, nor the reason for the differences in the overshoot between the different solvers. A likely cause of the difference between the implicit and semi-implicit cases is the more diffusive behavior of the implicit solver smoothing the overshoot out somewhat. For the conservative solver, it is likely that the process of solving for the pressure rise in this case leads to a physical discontinuity or shock which conservative solvers are much better at handling (14).



(a) Implicit Case (b) Conservative Case

Figure 4.6: Pressure Distribution in the Gravitational Source Case

CHAPTER

5

THE ADVECTIVE DERIVATIVE AND SOLVER ANALYSIS

In this chapter, information on the discretization of the advective derivative in the three-dimensional form of CTF-R is given, particularly on the implementation of both high-order (HO) and high-resolution (HR) advective schemes as introduced in section 2.5. The implementation and verification of a high-order upwind scheme is presented in section 5.1 and the implementation and verification of a high-resolution scheme based on the implemented high-order scheme is presented in section 5.2. Additional results pertaining to tests on the solver method implemented for the matrix-vector system are given in section 5.3.

5.1 High-Order Upwind Schemes

In CTF-R, both in the sub-channel form and the three-dimensional solvers presented in chapter 3 and chapter 4, the advective derivative terms are first-order upwinded in space

according to

$$\phi_{i+\frac{1}{2}} = \begin{cases} \phi_i & u_{i+\frac{1}{2}} > 0 \\ \phi_{i+1} & u_{i+\frac{1}{2}} > 0 \end{cases} \tag{5.1}$$

where ϕ is the advected variable and u is the advecting velocity. As discussed in section 2.5, this method is very stable, but also results in significant artificial diffusion of the advected variable. This diffusion can be seen in the enthalpy distribution in Figure 3.4a, Figure 4.3a, and Figure 4.3b. The enthalpy should be advected as a sharp boundary between the initial values and the inlet value, however the enthalpy is smeared between these values.

5.1.1 Implementation of High-Order Schemes

To fix the issue presented by the diffusivity of first-order upwinding, a high-order method has been implemented in the three-dimensional form of CTF-R in the form of the second-order upwind scheme (SOU),

$$\phi_{i+\frac{1}{2}} = \begin{cases} \frac{3}{2}\phi_i - \frac{1}{2}\phi_{i-1} & u_{i+\frac{1}{2}} > 0 \\ \frac{3}{2}\phi_{i+1} - \frac{1}{2}\phi_{i+2} & u_{i+\frac{1}{2}} \leq 0. \end{cases} \tag{5.2}$$

To implement a high-order method in CTF-R, changes needed to be made to how the Jacobian matrix is initialized and constructed, how the variables are perturbed in calculating the Jacobian, and how the upwind is applied in the functions for the governing equations. The first step in implementing this method was to alter the process shown in Algorithm 11. An updated form of this process is shown in Algorithm 13. The main addition is to include the effect of the variables, density, enthalpy, and the three velocities, in cells that are two away from the current cell along one of the coordinate axes: $(i+2, j, k), (i, j+2, k), (i, j, k+2), (i-2, j, k), (i, j-2, k), (i, j, k-2)$. This addition to the Jacobian matrix is only included if the flag for the advective derivative, `convFlag`, is greater than zero. Otherwise, first-order upwinding is used and the additions relating to high-order methods are ignored by the simulation.

Algorithm 13 Jacobian Construction Process in 3-D Semi-Implicit Solver

for $N = 1, N_{cells}$ **do**
  for $row = 1, N_{rows}$ **do**
    Set $F_{row}(x)$
    for $r = 1, 27$ **do**
      if r in stencil **then**
        for $l = 1, L$ **do**
          Perturb variable l in cell r
          Calculate $F_{row}(x + \epsilon x_{l,r})$
          Determine the column in the Jacobian that corresponds to variable $x_{l,r}$
          $J_{row,column} = \frac{F_{row}(x+\epsilon x_{l,r}) - F_{row}(x)}{\epsilon}$
        end for
      end if
    end for
    if `convFlag` > 0 **then**
      for $s = 1, 6$ **do**
        for $l = 1, 5$ **do**
          Perturb variable l in cell s
          Calculate $F_{row}(x + \epsilon x_{l,r})$
          Determine the column in the Jacobian that corresponds to variable $x_{l,s}$
          $J_{row,column} = \frac{F_{row}(x+\epsilon x_{l,s}) - F_{row}(x)}{\epsilon}$
        end for
      end for
    end if
  end for
end for

After this new process for constructing the Jacobian was implemented, the routine that perturbs the variables in the solution vector to allow the Jacobian to be numerically calculated was altered to include the variables from the additional cells necessary for HO schemes. Then, the upwind function was updated to use Equation 5.2 when the `convFlag` is set to 1, as well as the functions for the governing equations being updated to call the upwind with the relevant cells for the HO scheme.

In addition to SOU being implemented, the quadratic upstream interpolation for convective kinematics (QUICK) scheme was implemented as well. The QUICK scheme is based

upon

$$\phi_{i+\frac{1}{2}} = \begin{cases} \frac{3}{4}\phi_i + \frac{3}{8}\phi_{i+1} - \frac{1}{8}\phi_{i-1} & u_{i+\frac{1}{2}} > 0 \\ \frac{3}{4}\phi_{i+1} + \frac{3}{8}\phi_i - \frac{1}{8}\phi_{i+2} & u_{i+\frac{1}{2}} \leq 0, \end{cases} \tag{5.3}$$

and is a third-order method based on interpolating the value of ϕ based on a quadratic polynomial that is biased towards the upstream direction (32). The QUICK scheme is used when the `convFlag` is set to three.

5.1.2 Verification of High-Order Schemes

To verify the HO scheme implemented in Section 5.1.1, a square wave advection problem was first used. The problem specifications for this problem are given in Table 5.1, which is similar to the sourceless test problem used in section 3.5 and section 4.3. This problem was run using the semi-implicit form of the governing equaiotns

Table 5.1: Square Wave Advection Problem Specification

Channel Height (m)	2.0	Maximum Time steps	5000
Channel Length (m)	0.4	Timestep Length (s)	0.001
Channel Width (m)	0.4	Outlet Pressure (Pa)	0
x-direction Mesh Spacing (m)	0.05	Inlet Velocity Vector (m/s)	(5.0, 0.0, 0.0)
y-direction Mesh Spacing (m)	0.05	Inlet Enthalpy (kJ/kg)	530
z-direction Mesh Spacing (m)	0.05	Initial Pressure (Pa)	0
Steady-State Convergence Criterion	0.0001	Initial Enthalpy (kJ/kg)	525

This problem was run with three different options for the calculation of the advective derivative, first-order upwinding (FOU), second-order upwinding (SOU), and QUICK. A plot of the results for this case is shown in Figure 5.1. The analytic results for the wave advection are shown with solid lines. The results for the FOU are shown in the dotted-and-dashed line, the SOU results are shown with a dotted line, the QUICK results are shown with the dashed lines, and the solid lines represent the analytic result. The FOU results clearly show the diffusivity expected from FOU, and shows the diffusion increasing as the flow goes along the channel length. In the SOU and QUICK results, the diffusion of the results is significantly less than the FOU case. However, the unboundness of the high-order schemes discussed in section 2.5 can be seen near the wave boundary. The high-order schemes are

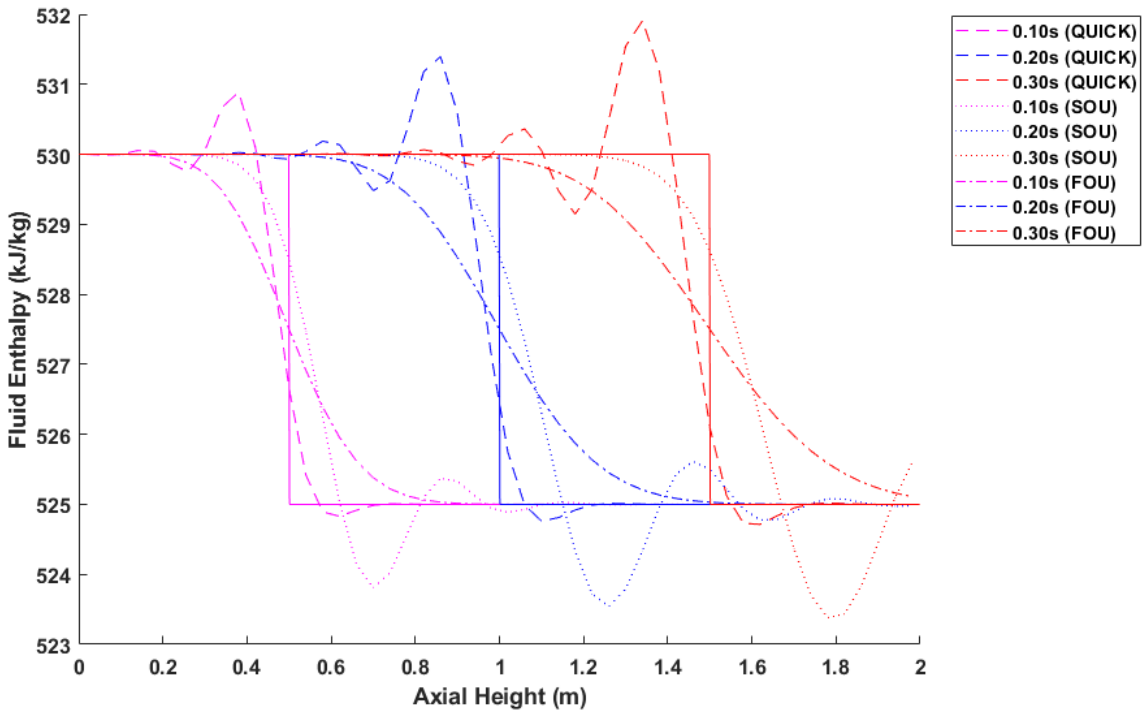


Figure 5.1: Comparison of High-Order Methods to First-Order

unbounded which can be seen in the oscillatory behavior of the SOU and QUICK results. The SOU results have more oscillatory behavior after the wave interface and the QUICK results show the oscillatory behavior more before the wave interface.

In addition to showing the results for a set mesh spacing, the case was run for both of the HO methods with mesh spacings of $0.1m$, $0.05m$, and $0.025m$. The results for this are shown in Figure 5.2 and Figure 5.3. The analytic results for the wave advection are shown with solid lines on these plots. In these plots, it can be seen that as the mesh is refined, the diffusion near the boundary is decreased and the estimate of the flat wave interface improves significantly. However, the mesh refinement also leads to the oscillatory behavior near the interface increasing, leading to inaccurate and meaningless results as can be seen in the large sine-like waves after the interface in SOU and before the interface in QUICK.

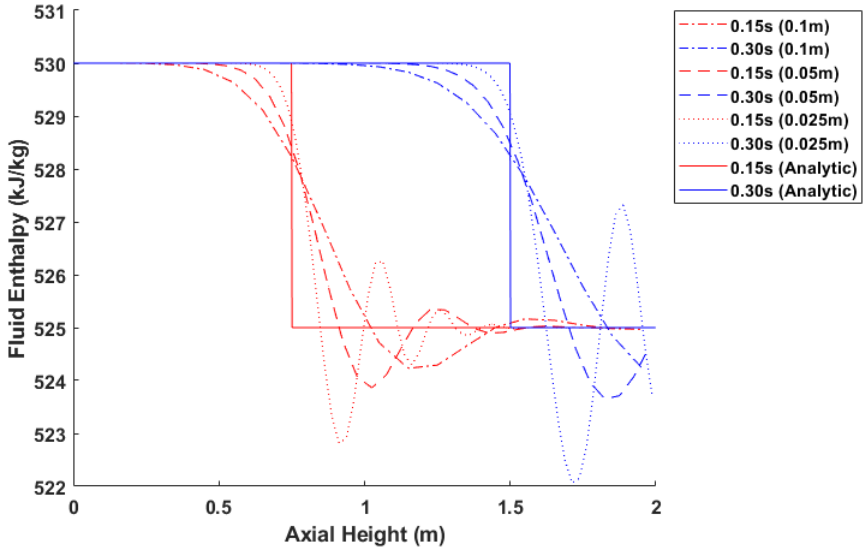


Figure 5.2: Effect of Spatial Discretization in Square Wave Advection with SOU

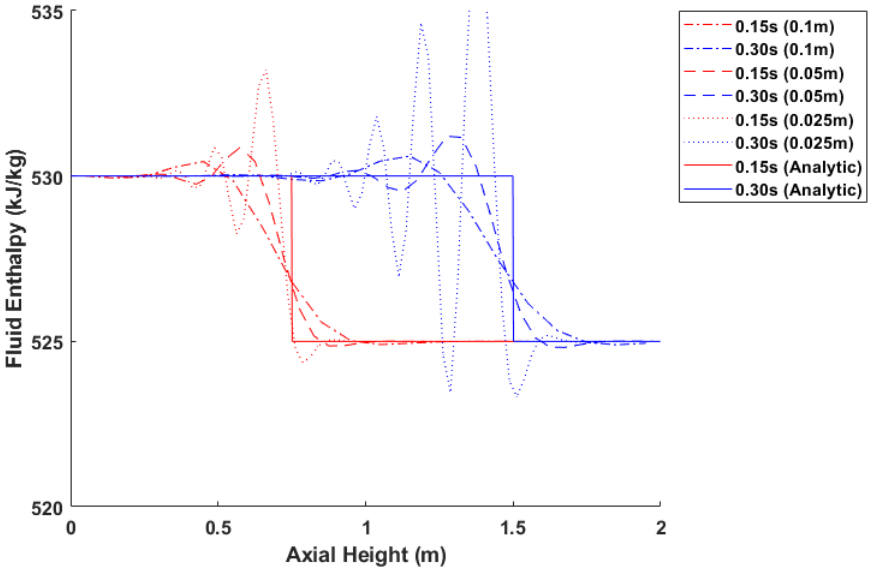


Figure 5.3: Effect of Spatial Discretization in Square Wave Advection with QUICK

In terms of the location of the oscillatory behavior, the reason for SOU to have oscillations after the interface and QUICK to have them before the interface lies in the formulations for each. In Equation 5.2, the advected quantity is taken from the upwind and far upwind nodes. This means that flow behavior downstream are more affected by the far upwind node. The QUICK scheme, as shown in Equation 2.57, has effects from both the far upwind and the downwind nodes. The effect of the downwind node is larger than that of the far upwind node, leading to larger oscillations occurring before the wave interface and a smaller undershoot existing after the wave interface.

From the results for the HO schemes implemented in this section, we can see that the wave interface is resolved much better by the HO schemes than the first-order upwinding, and less diffusion is seen. However, the behavior either side of the wave interface exhibits strong oscillatory behavior, and becomes unstable as the simulation progresses further. These methods as shown here do not offer an attractive alternative to the first-order upwinding previously used.

5.2 High-Resolution Upwind Schemes

While the high-order schemes implemented in Section 5.1.1 improve upon the diffusivity of first-order upwinding, they are unbounded and therefore do not preserve extrema. This behavior can lead to some oscillatory behavior, as seen in the results shown in Section 5.1.2. To prevent the unbounded behavior of the high-order upwind schemes that were implemented, a flux limiting scheme was implemented in the three-dimensional form of CTF-R to form a high-resolution scheme.

5.2.1 Implementation of High-Resolution Schemes

The two high-resolution schemes implemented are the MINMOD scheme (14),

$$\tilde{\phi}_{i+\frac{1}{2}} = \begin{cases} \frac{3}{2}\tilde{\phi}_i & 0 \leq \tilde{\phi}_i < \frac{1}{2} \\ \frac{1}{2}\tilde{\phi}_i + \frac{1}{2} & \frac{1}{2} \leq \tilde{\phi}_i < 1 \\ \tilde{\phi}_i & elsewhere, \end{cases} \tag{5.4}$$

which is an extension of the SOU scheme and the Modified SMART scheme (20; 14),

$$\tilde{\phi}_{i+\frac{1}{2}} = \begin{cases} 3\tilde{\phi}_i & 0 \leq \tilde{\phi}_i < \frac{1}{6} \\ \frac{3}{4}\tilde{\phi}_i + \frac{3}{8} & \frac{1}{6} \leq \tilde{\phi}_i < \frac{7}{10} \\ \frac{1}{3}\tilde{\phi}_i + \frac{2}{3} & \frac{7}{10} \leq \tilde{\phi}_i \leq 1 \\ \tilde{\phi}_i & elsewhere, \end{cases} \tag{5.5}$$

which is an extension of the QUICK scheme. These schemes are shown here in terms of the normalized variable formulation (NVF) for a positive advecting velocity. For a negative advecting velocity, the NVF at $i+1$ is used instead of the NVF i. The NVF of a variable, as presented in Equation 2.59, is

$$\tilde{\phi} = \begin{cases} \frac{\phi - \phi_{i-1}}{\phi_{i+1} - \phi_{i-1}} & u_{i+\frac{1}{2}} > 0 \\ \frac{\phi - \phi_{i+1}}{\phi_{i-1} - \phi_{i+1}} & u_{i+\frac{1}{2}} \leq 0. \end{cases} \tag{5.6}$$

Changing from the NVF formulation of the high-resolution schemes back to an explicit form of the value in the upwind node yields

$$\phi_{i+\frac{1}{2}} = \begin{cases} \frac{3}{2}\phi_i - \frac{1}{2}\phi_{i-1} & 0 \leq \tilde{\phi}_i < \frac{1}{2} \\ \frac{1}{2}\phi_i + \frac{1}{2}\phi_{i+1} & \frac{1}{2} \leq \tilde{\phi}_i < 1 \\ \phi_i & elsewhere, \end{cases} \tag{5.7}$$

for the MINMOD scheme, assuming a positive advecting velocity. The SMART scheme becomes

$$\phi_{i+\frac{1}{2}} = \begin{cases} 3\phi_i - 2\phi_{i-1} & 0 \leq \tilde{\phi}_i < \frac{1}{6} \\ \frac{3}{4}\phi_i + \frac{3}{8}\phi_{i+1} - \frac{1}{8}\phi_{i-1} & \frac{1}{6} \leq \tilde{\phi}_i < \frac{7}{10} \\ \frac{1}{3}\phi_i + \frac{2}{3}\phi_{i+1} & \frac{7}{10} \leq \tilde{\phi}_i \leq 1 \\ \phi_i & elsewhere, \end{cases} \tag{5.8}$$

for a positive advecting velocity.

To implement a high-resolution scheme in CTF-R, the main alteration made was to add the calculation of $\tilde{\phi}_i$ for positive velocities and the calculation of $\tilde{\phi}_{i+1}$ for negative velocities. Once this was implemented for the case of `convFlag` of 3 and 4, for MINMOD and the modified SMART scheme respectively.

5.2.2 Verification of High-Resolution Schemes

To verify the HR scheme implemented in Section 5.2.1, a square wave advection problem was used identical to the one used in Section 5.1.2. The problem specifications for this problem are given in Table 5.1.

A comparison of the results of the SOU scheme with the MINMOD scheme which is a flux limited version of a SOU scheme, is shown in Figure 5.4. The analytic results for the wave advection are shown with solid lines. It can be seen that the high-resolution scheme has similar behavior before the wave interface and a similar level of diffusion at the interface. However, the boundedness of the scheme leads to the oscillations seen after the interface in the SOU, which were discussed in Section 5.2.2, are removed, leading to a much smoother approximation of the wave interface.

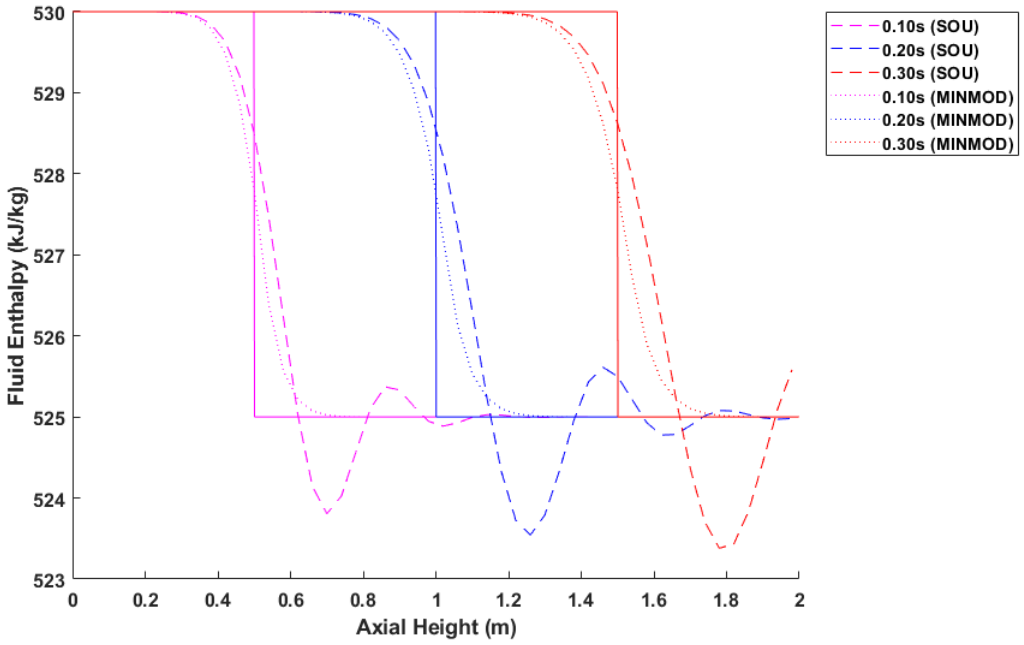


Figure 5.4: Comparison of Square Wave Advection Results between SOU and MINMOD

A comparison of the results of the QUICK scheme with the Modified SMART scheme, which is a flux limited version of the QUICK scheme, is shown in Figure 5.5. The analytic results for the wave advection are shown with solid lines. It can be seen that the high-

resolution SMART scheme does not exhibit the overshoots and undershoots of the wave exhibited by the QUICK scheme while maintaining the level of accuracy in defining the wave interface.

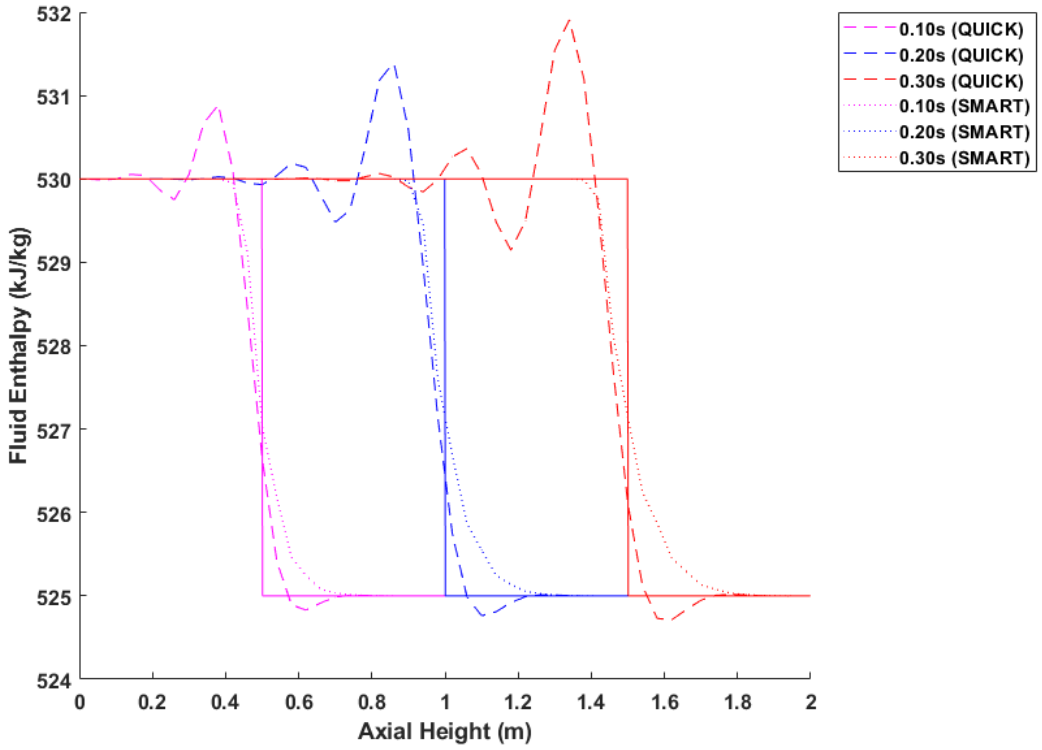


Figure 5.5: Comparison of Square Wave Advection Results between QUICK and SMART

The Results of the MINMOD scheme are compared with the results of the modified SMART scheme in Figure 5.6. The analytic results for the wave advection are shown with solid lines. It can be seen that both methods remove any overshoots or oscillations in the result near the wave interface. On the upwind side of the wave interface, the SMART scheme shows a much closer prediction to the analytic result. This is due to the bounded form of the QUICK scheme offering a higher-order accuracy in this region compared to the bounded second-order upwind scheme.

Downwind of the boundary, the two solvers give similar accuracy and diffusion. This is due to the fact that the NVF of the enthalpy will be slightly under one or will be larger than

one in this region. In Equation 5.7 and Equation 5.8, the MINMOD and SMART schemes can be seen to have a similar form, for NVF values above $\frac{7}{10}$. This leads to the similar prediction of the shape of the enthalpy distribution downwind of the wave interface.

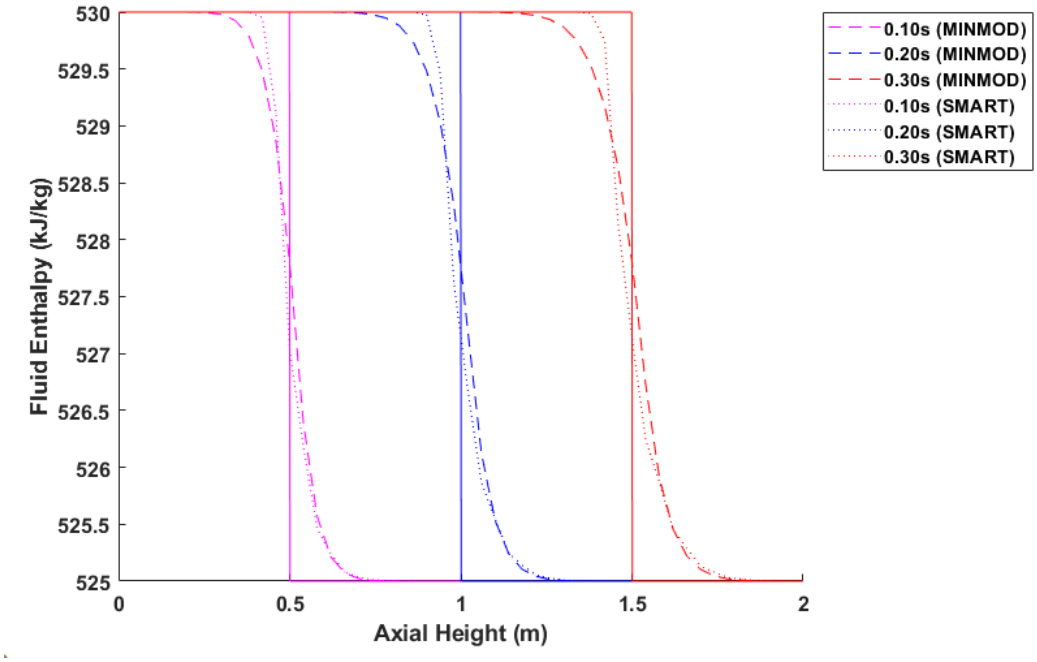


Figure 5.6: Comparison of Square Wave Advection Results between MINMOD and SMART

In addition to showing the results for a set mesh spacing for comparison to the other HO and HR methods, the wave advection case was run for both of the HR methods with mesh spacings of $0.1m$, $0.05m$, and $0.025m$. The results for this are shown in Figure 5.7 and Figure 5.8. The analytic results for the wave advection are shown with solid lines on these plots. In these plots, it can be seen that as the mesh is refined, the prediction of the wave interface improves greatly, and that there is no issue with the oscillations seen in Figure 5.2 and Figure 5.3.

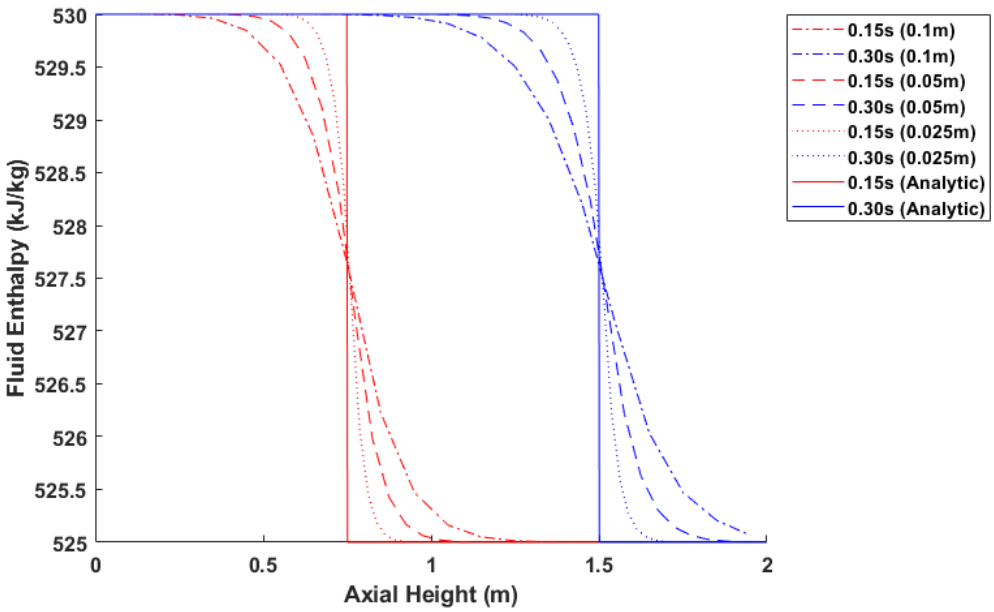


Figure 5.7: Effect of Spatial Discretization in Square Wave Advection with MINMOD

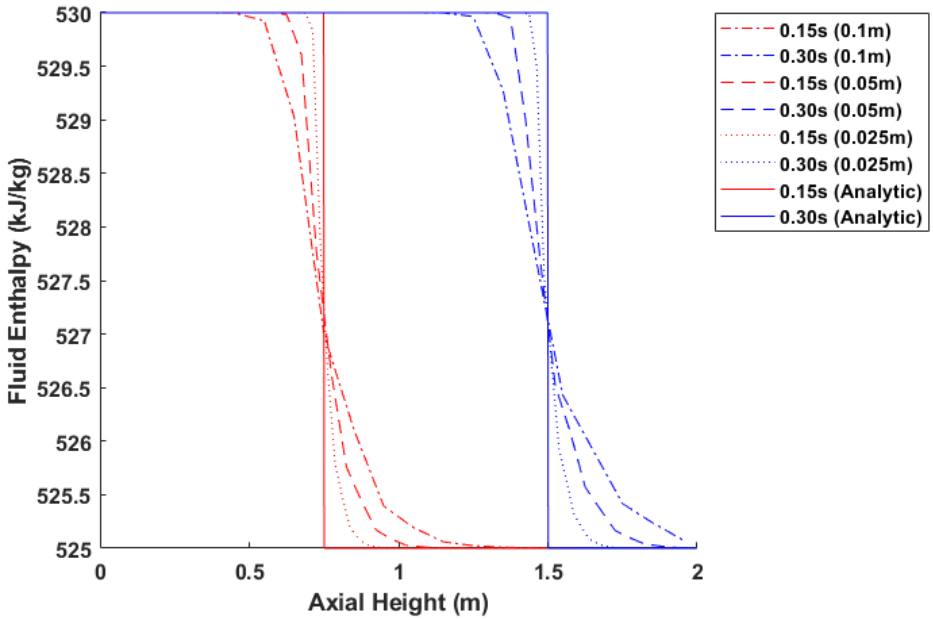


Figure 5.8: Effect of Spatial Discretization in Square Wave Advection with SMART

5.3 Performance of Solver Methods in PETSc

The matrix-vector system, presented in Equation 2.33, that is solved to update the solution vector is solved using PETSc (5). Specifically, the KSP linear solver capabilities are used. Within this, there are a variety of different solver methods that can be implemented. The default that is used by CTF and CTF-R is the Bi-Conjugate Gradient STABilized (BiCGSTAB) method (58). In previous tests using the sub-channel form of CTF-R, the effect of changing the solver method used by PETSc did not have a measurable effect on the simulation runtime, as the matrices generated were typically small and simple to solve. With the implementation of the three-dimensional solver, as well as the implicit and conservative formulations of the three-dimensional solver, a study of the various solver methods available and their effect on the simulation accuracy and efficiency was analyzed.

To perform this study, a sample of linear solvers available in PETSc was taken, with a focus on some of the more popular and well known methods being selected. An option was introduced in the input file, *res_3d.inp*, to select each of these solvers. The solvers tested were: the Generalized Conjugate Residual (GCR) method, the Bi-Conjugate Gradient Stabilized (BiCGSTAB) method, the Improved Bi-Conjugate Gradient Stabilized (IBiCGSTAB) method, the Generalize Minimal Residual (GMRES) method, the Flexible Generalized Minimal Residual (FGMRES) method, the Generalized Minimal Residual Method with Accelerated Restart (LGMRES), and the Transpose-Free Quasi-Minimal Residual (TFQMR) method.

The test case for this study was for a system with positive flow in the x-direction, a volumetric heat generation source term, and a gravitational source term that acts opposite to the flow direction. This case was chosen as the resulting matrix will have similar properties to a typical single-phase LWR simulation problem. This test case was run for each of the solvers listed above, using the RDFMG high-performance computing cluster at North Carolina State University. Each case was run on the same computing node concurrently and was run multiple times to yield a more accurate estimate of the runtime. This process was done using the semi-implicit solver, the implicit solver, and the conservative solver.

The results for the study of the performance of the semi-implicit solver with a variety of solver methods specified in PETSc are shown in Table 5.2. The average amount of time spent in solution of the matrix-vector system and the average percent of the overall simulation runtime spent in solution were the two metrics used to measure the performance of the different methods. FGMRES performed the best of the seven methods tested, with 7.426%

of the simulation spent in solving for the solution update vector. The other GMRES method that were tested (GMRES and LGMRES) were the next best performing options, along with GCR, with around 8.5% of simulation time spent on the solution process. BiCGSTAB, IBiCGSTAB and TFQMR showed significantly worse performance than the other methods.

Table 5.2: Table of Runtime Results for different PETSc Solvers with the Semi-Implicit Solver

	GCR	BiCGSTAB	iBiCGSTAB	GMRES
Average Simulation Time (s)	592.2	632.0	635.2	609.1
Average Solution Time (s)	50.38	72.67	76.59	52.11
Percent of Runtime in Solution	8.509	11.50	12.06	8.554
	FGMRES	LGMRES	TFQMR	
Average Simulation Time (s)	601.1	582.6	610.7	
Average Solution Time (s)	44.64	50.27	78.70	
Percent of Runtime in Solution	7.426	8.628	12.89	

The l2-norms of the residual vector and of the error in the solution were tracked to test the convergence of the simulation for each test case. These values were the same, to 6 significant figures for each case throughout the simulation, and showed identical convergence to the solution for each solver. There does not appear to be a significant effect on the accuracy of the simulation for any of the tested linear solvers.

The results for the study of the performance of the implicit solver with a variety of solver methods specified in PETSc are shown in Table 5.3. The average amount of time spent in solution of the matrix-vector system and the average percent of the overall simulation runtime spent in solution were the two metrics used to measure the performance of the different methods. FGMRES and GCR performed the best of the seven methods tested, with 29.19% and 29.63% of the simulation spent in solving for the solution update vector respectively. The other GMRES method that were tested (GMRES and LGMRES) were the next best performing options with around 31% of simulation time spent on the solution process. BiCGSTAB, IBiCGSTAB and TFQMR showed significantly worse performance than the other methods.

Table 5.3: Table of Runtime Results for different PETSc Solvers with the Implicit Solver

	GCR	BiCGSTAB	iBiCGSTAB	GMRES
Average Simulation Time (s)	884.9	939.0	947.2	899.6
Average Solution Time (s)	262.2	320.6	327.0	278.4
Percent of Runtime in Solution	29.63	34.15	34.52	30.95
	FGMRES	LGMRES	TFQMR	
Average Simulation Time (s)	875.9	899.4	976.4	
Average Solution Time (s)	255.7	279.3	356.6	
Percent of Runtime in Solution	29.19	31.06	36.52	

The results for the study of the performance of the conservative solver with a variety of solver methods specified in PETSc are shown in Table 5.4. The average amount of time spent in solution of the matrix-vector system and the average percent of the overall simulation runtime spent in solution were the two metrics used to measure the performance of the different methods. FGMRES and GCR performed the best of the seven methods tested, with 30.00% and 30.07% of the simulation spent in solving for the solution update vector respectively. The other GMRES method that were tested (GMRES and LGMRES) were the next best performing options with 31.55% and 31.76% of simulation time spent on the solution process respectively. BiCGSTAB, IBiCGSTAB and TFQMR showed significantly worse performance than the other methods.

Table 5.4: Table of Runtime Results for different PETSc Solvers with the Conservative Solver

	GCR	BiCGSTAB	iBiCGSTAB	GMRES
Average Simulation Time (s)	894.7	962.7	964.6	919.9
Average Solution Time (s)	262.2	320.6	327.0	278.4
Percent of Runtime in Solution	30.07	35.03	35.00	31.55
	FGMRES	LGMRES	TFQMR	
Average Simulation Time (s)	896.2	916.4	990.6	
Average Solution Time (s)	255.7	279.3	367.7	
Percent of Runtime in Solution	30.00	31.76	37.11	

From these results, it appears that the Generalized Conjugate Residual (GCR) method and the Flexible Generalize Minimal Residual (FGMRES) method perform the best for all three solvers, with the other GMRES-based solver methods showing the next best performance. There was no effect on the accuracy of the results seen based on the selection of the solver method.

CHAPTER

6

CONCLUSIONS

In this work, multiple numerical improvements were made in the residual formulation of CTF. Overall, these improvements represent a significant improvement to the numerical capabilities of CTF-R. A new solver in which the sub-channel approximation has been removed has been implemented. The option to adjust order of discretizing the advective derivative has been increased as well as implementing flux limiting to ensure stability of the derivative. Control over the degree of implicitness of the simulation has been implemented, as well as the option of solving the conservative form of the governing equations.

A three-dimensional solver was implemented in CTF-R in addition to the existing sub-channel form of equations. To implement this solver, the process of initializing the matrix system and constructing the matrix system had to be updated to allow for the new governing equations. A new mesh logic was implemented to allow for the storage and solution of information for true three-dimensional systems. The semi-implicit governing equations were implemented in non-conservative form to be used in constructing the new matrix system. This new solver was tested against a set of analytic cases to ensure accurate simulation and performed well on each case. The ability to simulate a true three-dimensional system as opposed to a sub-channel system will allow for a greater array of potential thermal-hydraulic

problems to be simulated in CTF-R. In addition to this, this new solver will allow for the implementation of further capabilities such as thermal conduction and fluid viscosity with more ease than most thermal-hydraulic solvers. This is due to both the use of residual functions to construct a numerical Jacobian, and the care taken in the development of the solver to allow for additional capabilities to be implemented.

This new three-dimensional semi-implicit solver was extended to allow for the simulation of fluid flows with fully implicit non-conservative governing equations as well as with the conservative form of the governing equations. Both of these new solvers were tested against the same test cases for the semi-implicit solver, and again the results converged to the distribution for the analytic case. To allow the implicit and conservative solvers to function, a nonlinear iteration was implemented in the form of a Shamanskii iteration. The convergence of the Shamanskii iteration was verified and shown to converge q-linearly. The reason only q-linear convergence was seen was due to the fact that the solver being tested is an inexact Newton method due to the use of Krylov methods to solve the linear system. These new capabilities that allow for simulation of the fully implicit equations and the conservative form of the equations represent a novel capability in the field of nuclear thermal-hydraulics, particularly the ability to simulate the conservative form.

In CTF-R and other thermal-hydraulic codes, numerical diffusion is seen due to the use of first-order upwinding. To imrpove upon this, the capability to use a high-order (HO) convective scheme was implemented in the three-dimensional solver in CTF-R. To do this, the process of constructing the Jacobian matrix was altered to extend the cells which can affect the simulation, as well as altering the process of calculating the advected variables in the code. The Second-Order Upwind (SOU) and the Quadratic Upstream Interpolated for Convective Kinematics (QUICK) schemes were implemented and shown to reduce the diffusivity of the solver at the cost of the scheme being unbounded leading to oscillations in the result. To improve this, high-resolution (HR) convective schemes were implemented in the code in the form of the MINMOD scheme and the modified SMART scheme. These high-resolution methods were shown to greatly improve the diffusivity of the first-order method while removing the oscillatory behavior seen in the high-order methods.

The implementation of HO and HR convective schemes is not widely done in fluid mechanics except for CFD codes, and is never done in nuclear thermal-hydraulics codes. This is due to the fact that accounting for the far-upwind terms used in these methods requires weighting factor methods as discussed in Section 2.5.3 due to the way most fluid solvers are set up. In the three-dimensional form of CTF-R, the fact that the residual functions

are used to calculate the Jacobian matrix numerically means that the far-upwind terms can be accounted for directly without the use of weighting factor methods. This leads to the implementation of HR methods in this work causing a significantly smaller increase in computational time compared to the typical HR methods. In addition to this, the stability and accuracy concerns of some of the methods used to implement HR advective schemes are not present either.

A study was performed on the linear solver method from PETSc that was used in the solution of the matrix-vector system in the three-dimensional form of CTF-R. A survey of seven of the available methods was performed to determine the effect of the solver method on the efficiency and accuracy of the solver. There was no significant effect on the accuracy seen from the selection of solver method and the runtime for the solver was best when run with the Flexible Generalized Minimal Residual method or the Generalized Conjugate Residual method. This work showed that noticeable computational improvements can be realized by changing from the BiCGSTAB solver that was used by default previously.

The developments presented in this work represent a new and numerically improved solver in CTF-R. Still, there is an array of future work to be done for this solver. Significant improvement to the flexibility of the solver to simulate more fluid problems can be achieved by expanding the physics that it is capable of modeling. The effect of solid structures, both in terms of heat generation and conduction through fuel as well as friction effects across solid structures need to be implemented. This is necessary for modeling of many reactor designs, such as LWRs and liquid metal reactors. In addition to modeling solid structures, the implementation of a multi-phase flow models, similar to many system and sub-channel codes, is necessary for the accurate simulation of LWR cases. For liquid metal reactors, the effects of thermal conduction within the fluid, which is neglected in the Euler equations, is necessary to accurately predict the temperature distribution within the coolant. Implementing the Navier-Stokes equations, specifically the terms in the stress tensor governing viscosity and turbulence, is necessary for simulation of pool-type reactors such as molten salt fast reactor designs. The development of the three-dimensional solver was performed with forethought for the future implementation of these capabilities.

While preliminary verification work is done here, further work is required in both verification and validation of the new capabilities presented. As discussed at the end of section 3.5, the off-axis momentum flux terms in the momentum equations must be tested, which can be done in a case where the system velocity is not orthogonal to one of the system axes. Testing of the cases here should be expanded to analyze convergence behavior of the problems,

in terms of refining the mesh and timestep size of the problems. As the additional modeling capabilities for solid structure, thermal conduction and viscosity, and multi-phase flow are implemented, each of these new capabilities will need to be tested as well. In addition to improving and expanding the verification cases, Cases to display the improved range of applicability of the solver implemented should be performed. The removal of the CFL limit in the implicit solver and the effect on timestep size selection should be demonstrated. The improved behavior around sharp gradients in the conservative solver should be displayed further beyond the gravitational verification case presented here. The tests on the solver performed in section 5.3 show the initial performance of different solver methods with the new solvers implemented in this work, but this process should be expanded. A variety of test cases, with different numerical structure should be performed to see the effect of different problems on the solver choice. Additionally, the effect of the selection of the tolerances in PETSc on efficiency and convergence, particularly for the implicit and conservative solvers, should be performed.

The method implemented for nonlinear solution presented here is a noticeable improvement upon the common Newton's method, but there is still much room for improvement. The inexact Newton method presented can be adjusted to converge the forcing factor to zero to yield superlinear convergence of the method. Additionally, due to the high cost of calculating the Jacobian matrix for the system, which will only rise with the addition of the planned additional modeling capabilities, methods which remove the calculation of the Jacobian entirely should be pursued. Either a quasi-Newton method, such as Broyden's method, or a Jacobian-Free method such as JFNK would be attractive alternatives to the current method. Due to the numerical approximation of the matrix-vector product needed in JFNK being similar to the existing method for calculating the numerical Jacobian matrix and the Krylov-based solvers being used only need the action of that matrix-vector product, JFNK is the suggested process to be pursued.

REFERENCES

[1] Anderson, J. D. and Wendt, J. (1995). *Computational fluid dynamics*, volume 206. Springer.

[2] AREVA NP Inc. (2010). *COBRA-FLX: A Core Thermal-Hydraulic Analysis Code*.

[3] Ariyawansa, K. and Lau, D. (1992). Local and q-superlinear convergence of a class of collinear scaling algorithms that extends quasi-newton methods with broyden's bounded class of updates. *Optimization*, 23(4):323–339.

[4] Aumiller, D. et al. (2015). COBRA-IE: A new sub-channel analysis code. In *Proceedings of the 16th International Topical Meeting on Nuclear Reactor Thermal Hydraulics (NURETH-16), Chicago, IL*.

[5] Balay, S. et al. (2021). PETSc/TAO users manual. Technical Report ANL-21/39 - Revision 3.16, Argonne National Laboratory.

[6] Berry, R. A. et al. (2018). *RELAP-7 theory manual*.

[7] Brent, R. P. (1973). Some efficient algorithms for solving systems of nonlinear equations. *SIAM Journal on Numerical Analysis*, 10:327–344.

[8] Broyden, C. G., Dennis Jr, J. E., and Moré, J. J. (1973). On the local and superlinear convergence of quasi-newton methods. *IMA Journal of Applied Mathematics*, 12(3):223–245.

[9] Casulli, V. and Greenspan, D. (1984). Pressure method for the numerical solution of transient, compressible fluid flows. *International Journal for Numerical Methods in Fluids*, 4:1001–1012.

[10] Courant, R., Friedrichs, K., and Lewy, H. (1928). Über die partiellen differzengleichungen der mathematischen physik. *Mathematische Annalen*, 100:32–74.

[11] Dances, C., Mousseau, V., and Avramova, M. (2015). Ctf residual formulation of solid liquid coupling. In *Proceedings of the 16th International Topical Meeting on Nuclear Reactor Thermal Hydraulics (NURETH-16), Chicago, IL*, pages 3690–3702.

[12] Dances, C. A. (2015). Initial residual formulation of ctf. Master's thesis, Pennsylvania State University.

[13] Darwish, M. and Moukalled, F. (1996). The normalized weighting factor method: a novel technique for accelerating the convergence of high-resolution convective schemes. *Numerical Heat Transfer*, 30(2):217–237.

[14] Darwish, M. and Moukalled, F. (2021). *The finite volume method in computational fluid dynamics: an advanced introduction with OpenFOAM® and Matlab®*. Springer.

[15] Dembo, R. S., Eisenstat, S. C., and Steihaug, T. (1982). Inexact newton methods. *SIAM Journal on Numerical analysis*, 19(2):400–408.

[16] Dennis, J. E. and Moré, J. J. (1974). A characterization of superlinear convergence and its application to quasi-newton methods. *Mathematics of computation*, 28(126):549–560.

[17] EPRI (2014). *GOTHIC Thermal Hydraulic Analysis Package*.

[18] Freund, R. W. (1993). A transpose-free quasi-minimal residual algorithm for non-hermitian linear systems. *SIAM journal on scientific computing*, 14(2):470–482.

[19] Fromm, J. E. (1968). A method for reducing dispersion in convective difference schemes. *Journal of Computational Physics*, 3:176–189.

[20] Gaskell, P. and Lau, A. (1988). Curvature-compensated convective transport: Smart, a new boundedness-preserving transport algorithm. *International Journal for numerical methods in fluids*, 8(6):617–641.

[21] Gauntt, R. et al. (2000). *MELCOR Computer Code Manuals*. Sandia National Laboratories, NUREG/CR.

[22] Harlow, F. H. and Amsden, A. A. (1968). Numerical calculation of almost incompressible flow. *Journal of Computational Physics*, 3:80–93.

[23] Harlow, F. H. and Amsden, A. A. (1971). A numerical fluid dynamics calculation method for all flow speeds. *Journal of Compuatational Physics*, 8:197–213.

[24] Hestenes, M. R. and Steifel, E. (1952). Methods of conjugate gradient for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49:409.

[25] Hu, R., Zou, L., Hu, G., and Mui, T. (2021). *SAM user's guide*.

[26] Imke, U. and Sanchez, V. H. (2012). Validation of the subchannel code subchanflow using the nupec pwr tests (psbt). *Science and Technology of Nuclear Installations*, 2012.

[27] Kelley, C. T. (1995). *Iterative methods for linear and nonlinear equations*. SIAM.

[28] Knoll, D. A. and Keyes, D. E. (2004). Jacobian-free newton–krylov methods: a survey of approaches and applications. *Journal of Computational Physics*, 193(2):357–397.

[29] Lax, P. and Wendroff, B. (1960). Systems of conservation laws. *Communications on Pure and Applied Mathematics*, 13:217–237.

[30] Lax, P. D. (1954). Weak solutions of nonlinear hyperbolic equations and their numerical computation. *Communications on pure and applied mathematics*, 7(1):159–193.

[31] Lax, P. D. (1957). Hyberbolic systems of conservation laws ii. *Communications on Pure and Applied Mathematics*, 10:537–566.

[32] Leonard, B. P. (1979). A stable and accurate convective modelling procedure based on quadratic upstream interpolation. *Computer methods in applied mechanics and engineering*, 19(1):59–98.

[33] Leonard, B. P. and Drummond, J. E. (1995). Why you should not use hybrid, power-law, or related exponential schemes for convective modelling - there are much better alternatives. *International Journal for Numerical Method in Fluids*, 20:421–442.

[34] LeVeque, R. J. (2007). *Finite difference methods for ordinary and partial differential equations: steady-state and time-dependent problems*. SIAM.

[35] Liles, D. R. and Reed, W. H. (1978). A semi-implicit method for two-phase fluid dynamics. *Journal of Computational Physics*, 26:390–407.

[36] Lloyd, L. J. (2014). *Development of a Spatially Selective, Nonlinear Refinement Algorithm for Thermal-Hydraulic Safety Analysis*. PhD thesis, University of Wisconsin-Madison.

[37] Mahaffy, J. H. (1982). A stability-enhancing two-step method for fluid flow calculations. *Journal of Computational Physics*, 46:329–341.

[38] Mahaffy, J. H. (1993). Numerics of codes: Stability, diffusion, and convergence. *Nuclear Engineering and Design*, 145:131–145.

[39] Mariño-Salguero, J. and Schäfer, M. (2021). A modified normalized weighting factor method for improving the efficiency of the blended high-resolution advection schemes in the context of multiphase flows. *Experimental and Computational Multiphase Flow*, 3(3):208–225.

[40] Mchugh, P. R. (1995). *An Investigation of Newton-Krylov Algorithms for Solving Incompressible and Low-Mach Number Compressible Fluid Flow and Heat Transfer Problems using Finite Volume Discretization*. PhD thesis, University of Idaho.

[41] Moorthi, A., Sharma, A. K., and Velusamy, K. (2018). A review of sub-channel thermal hydraulic codes for nuclear reactor core and future directions. *Nuclear Engineering and Design*, 332:329–344.

[42] Novak, A. J. et al. (2018). Pronghorn theory manual. Technical report, Idaho National Lab.(INL), Idaho Falls, ID (United States).

[43] NRC, US (2010). TRACE v5. 0 theory manual, field equations, solution methods, and physical models. *United States Nucl. Regul. Comm.*

[44] Patankar, S. V. (1980). *Numerical Heat Transfer and Fluid Flow*. Hempisphere Publishing Corporation.

[45] Paul F. Fischer, J. W. L. and Kerkemeier, S. G. (2008). *nek5000 Web page*. http://nek5000.mcs.anl.gov.

[46] Peaceman, D. W. and Rachford, H. (1955). The numerical solution of parabolic and elliptic differential equations. *Journal for the Society of Inustrial and Applied Mathematics*, 3:28–41.

[47] Porter, N., Mousseau, V. A., and Avramova, M. (2019). CTF-R: A novel residual-based thermal hydraulic solver. *Nuclear Engineering and Design*, 348:37–45.

[48] Porter, N. W. (2018). *Development of a Novel Residual Formulation of CTF and Application of Parameter Estimation Techniques*. PhD thesis, North Carolina State University.

[49] Ramshaw, J. D. and Trapp, J. A. (1978). Characteristics, stability, and short-wavelength phenomena in two-phase flow equation systems. *Nuclear Science and Engineering*, 66:93–102.

[50] Rowe, D. and Angle, C. W. (1967). Crossflow mixing between parallel flow channels during boiling. part ii. measurement of flow and enthalpy in two parallel channels. Technical report, Battelle-Northwest, Richland, Wash. Pacific Northwest Lab.

[51] Saad, Y. and Schultz, M. H. (1986). Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on scientific and statistical computing*, 7(3):856–869.

[52] Salko Jr, R. et al. (2020). *CTF theory manual*.

[53] Shamanskii, V. E. (1967). A modification of newton's method. *Ukranian Mathematical Journal*, 19:118–122.

[54] Shyy, W. (1985). A study of finite difference approximations to steady-state, convection-dominated flow problems. *Journal of Computational Physics*, 57(3):415–438.

[55] Siemens, P. (2017). *Simcenter STAR-CCM*.

[56] Stewart, C., Cuta, J., Koontz, A., Kelly, J., Basehore, K., George, T., and Rowe, D. (1983). Vipre-01. a thermal-hydraulic analysis code for reactor cores. volume 1. mathematical modeling.[pwr; bwr]. Technical report, Battelle Pacific Northwest Labs., Richland, WA (USA).

[57] Thurgood, M. (1983). Cobra/trac-a thermal-hydraulic code for transient analysis of nuclear reactor vessels and primary coolant systems. *Pacific Northwest Lab., Rept.*

[58] Van der Vorst, H. A. (1992). Bi-cgstab: A fast and smoothly converging variant of bi-cg for the solution of nonsymmetric linear systems. *SIAM Journal on scientific and Statistical Computing*, 13(2):631–644.

[59] Weller, H. G., Tabor, G., Jasak, H., and Fureby, C. (1998). *A tensorial approach to computational continuum mechanics using object-oriented techniques*.

[60] Yang, B.-W. et al. (2021). Subchannel analysis–current practice and development for the future. *Nuclear Engineering and Design*, 385:111477.