ABSTRACT

Nuclear engineers constantly use nuclear safety analysis codes to prevent design basis and beyond design basis accidents. In the past, nuclear engineers have used analysis codes such as TRAC and RELAP to predict and prevent fatal accident design scenarios. Now, the NRC uses its flagship thermal-hydraulic safety analysis code called TRAC/RELAP Advanced Computational Engine, or TRACE. TRACE - which performs analysis through heat transfer equations, thermal hydraulics equations, and point reactor kinetic equations – models transients in pressurized water reactors and boiling water reactors. The code performs analysis using standard finite volume method – a numerical method that discretizes the problem and solves for the physical parameters in small parts in order to develop a solution. The neutron population, found through the solution of the point kinetics equations, determines the power of the reactor. The purpose of this project is to develop and check the accuracy of an implicit implementation of the point kinetics model in TRACE. TRACE, which solves the point kinetics equations explicitly, could produce more accurate and stable solutions without the need for time steps of approximately $10^{-4}$ seconds. This project takes the modified version of the TRACE code that solves the point-kinetics equations implicitly, and compares the results of both the explicit and implicit solutions in terms of the neutronics feedback parameters, and searches for scenarios where analysis would benefit from using the implicit model. This project shows the potential benefit of using fully implicit models in nuclear reactor safety analysis, especially in accident scenarios.
# TABLE OF CONTENTS

LIST OF FIGURES ........................................................................................................ iii
LIST OF VARIABLES ..................................................................................................... iv
LIST OF SUBSCRIPTS ..................................................................................................... v
LIST OF TERMS .............................................................................................................. vi
ACRONYMS ................................................................................................................... vii
ACKNOWLEDGEMENTS ................................................................................................. vii
Chapter 1 Introduction ................................................................................................. 1
  History of Reactor Simulation ....................................................................................... 2
  Multiphysics Modeling and Literature Review .......................................................... 5
Chapter 2 Theory of Finite Volume Methods ............................................................... 12
Chapter 3 Experimentation and Results ..................................................................... 19
Chapter 4 Conclusions ................................................................................................. 25
Chapter 5 Future Work ................................................................................................ 26
Appendix A  TRACE Boiling Water Reactor Input File ................................................. 27
BIBLIOGRAPHY .............................................................................................................. 37
LIST OF FIGURES

Figure 1: Timeline of Reactor Simulation Relevant to TRACE .........................................................4
Figure 2: SNAP Opening Interface ..................................................................................................4
Figure 3: Full TRACE PWR Model in SNAP ..................................................................................4
Figure 4: Power Series Solution of PKE Equations ........................................................................7
Figure 5: Singularly Perturbed Method vs. Other PKE Calculation Methods .................................8
Figure 6: Comparison of two Analytical Expressions of PKE Equations .......................................9
Figure 7: Methods of Nonlinear Inconsistent Coupling Paradigms ...............................................10
Figure 8: Comparison of Different Coupling Paradigms ...............................................................11
Figure 9: Basic 1-D Pipe Labeled for Finite Volume Method .......................................................13
Figure 10: Simple BWR Model for TRACE in SNAP .................................................................19
Figure 11: FILL Component Parameters .......................................................................................20
Figure 12: BREAK Component Parameters ..................................................................................20
Figure 13: Total Power Comparison .............................................................................................21
Figure 14: Total Power Comparison Single Plot ............................................................................21
Figure 15: Peak Power Zoomed In ...............................................................................................22
Figure 16: The Max Average Rod Temperature; implicitly calculated (left) and explicitly calculated (right) ...........................................................................................................23
Figure 17: The Max Average Rod Temperature Same Plot ..........................................................23
Figure 18: The Liquid Velocity within the Channel; implicitly calculated (left) and explicitly calculated (right) ...........................................................................................................23
Figure 19: The Liquid Velocity Same Plot ....................................................................................24
Figure 20: CPU Time for Simulation ............................................................................................24
LIST OF VARIABLES

\( \alpha \) = Void Fraction of Gas
\( \rho \) = Reactivity
\( g \) = Acceleration Due to Gravity
\( U \) = Internal Energy
\( P \) = Pressure
\( \tau \) = Shear Stress
\( M \) = Momentum
\( \Gamma \) = Net Volumetric Vapor-Production Rate caused by Phase Change
\( S \) = Net Entrainment
\( W \) = Mass Transfer
\( q \) = Specific Heat Transfer
\( Q \) = Heat Transfer
\( \Lambda \) = Prompt Neutron Generation Time
\( n(t) \) = Number of Neutrons as a Function of Time
\( \beta \) = Total Fraction of Delayed Neutrons
\( C(t) \) = Average Density of Delayed Neutron Precursors
\( \kappa \) = Decay Constant
LIST OF SUBSCRIPTS

\( w = \text{Wall} \)
\( m = \text{Mixture} \)
\( l = \text{Liquid Phase} \)
\( v = \text{Vapor Phase} \)
\( g = \text{Gas Phase} \)
\( i = \text{Interface of Pipe} \)
\( r = \text{Relative Quantity or Variable} \)
\( s = \text{Entrainment} \)
LIST OF TERMS

\( k \) = Neutron Multiplication Factor
\( \beta(t) \) = Fraction delayed neutrons function
\( \rho \) = Reactivity
\( \Lambda \) = Neutron lifetime
\( \lambda \) = Delayed neutron decay constant
\( C(t) \) = Precursor Concentration function
\( n(t) \) = Neutron density function
LIST OF ACRONYMS

PWR = Pressurized Water Reactor
BWR = Boiling Water Reactor
LWR = Light Water Reactor
TRACE = TRAC/RELAP Advanced Computational Engine
NRC = Nuclear Regulatory Commission
RELAP = Reactor Excursion and Leak Analysis Program
SNAP = Symbolic Nuclear Analysis Package
PKE = Point Kinetics Equations
PKM = Point Kinetics Model
LOCA = Loss of Coolant Accident
TRAC = Transient Reactor Analysis Code
HEM = Homogeneous Equilibrium Model
ICE = Implicit Continuous Eulerian
INL = Idaho National Laboratory
PARCS = Purdue Advanced Core Simulator
FDM = Finite Difference Method
FVM = Finite Volume Method
PWS – Power Series Solution
SPM – Singularly Perturbed Method
PPJ – Power Prompt Jump Approximation
PrPJ – Precursor Prompt Jump Approximation
TPJ – Temperature Prompt Jump Approximation
ACKNOWLEDGEMENTS

I would like to first and foremost thank Dr. Justin Watson, who tirelessly, patiently, and persistently helped me through this project, provided this project for me, and advised me every single step of the way. I would like to thank Grant Garrett for being patient, helpful, and advising throughout the project. I would like to thank the Toshiba-Westinghouse Mechanical and Nuclear Engineering Scholar Program that provided funds for this project and inspired me to get into research. Finally, I would like to thank my mom who has always been helping me and praying for me even when I am away from home.
Chapter 1
Introduction

Nuclear energy, despite public opinion and perceived risks, boasts to be one of the safest forms of power production currently available [1]. This is because nuclear engineers carefully model and design reactors using advanced modeling and simulation computer programs. Computer simulation codes are important in the nuclear industry because of the expenses and physical difficulties to perform actual experiments using nuclear reactors. Through computer analysis, nuclear engineers are better able to understand different transient that can occur while running a reactor and prevent deadly accident scenarios. The practice of more accurate modeling and simulation allows engineers to avoid potential design accidents that other industries may overlook. As a result, nuclear engineers need to have the most precise and accurate simulation tools available to predict accident scenarios such as loss of coolant accidents and reactivity-initiated accidents. One of the most used reactor analysis tools used today is TRACE, which uses finite volume methods to solve heat transfer equations, fluid equations, and neutronics within a reactor. The neutronics portion of TRACE calculates the power of modeled reactors by explicitly solving for the point kinetics equations. The goal of this paper is to analyze an implicit implementation of these equations in an effort to make TRACE more accurate under certain transient conditions.
History of Reactor Simulation

The NRC began developing different system simulation codes through Bettis Atomic Laboratories and Los Alamos National Laboratory. TRACE, one of the main thermal-hydraulic simulation codes that the NRC uses today, was the result of the development of multiple different systems coming together into a single whole. The two driving codes in the development of TRACE were RELAP and TRAC-P/B [2]. Both of these codes endured decades of development and are still currently in use.

RELAP is one of the most powerful and state-of-the-art nuclear modeling codes currently used in the United States, and experienced numerous updates and development cycles to get to where it is today. Bettis Atomic Laboratories developed the first iteration of RELAP, called FLASH-1, at some point during the early to mid-1960’s. Due to severe computing limitations, FLASH-1 only analyzed reactors using three control volumes evaluated under a homogeneous equilibrium model. Engineers at the National Reactor Testing Station (now INL) rewrote and revised FLASH-1 into RELAP1 in 1966 to obtain a more comprehensive analysis of pressures, temperatures, flow, reactivities, and power for LOCA events in a reactor. This update rewrote the code in FORTRAN IV and followed extremely conservative 1-D estimations in accordance with the NRC’s Appendix K standards. The code enabled quick and conservative analysis. The next two iterations of RELAP – RELAP2 and RELAP3 – occurred over two year development cycles, improving the program’s stability and accuracy. INL released RELAP4 in 1973, a version of the code that eventually went through thirteen modifications before the program evolved into the first version of RELAP5 in 1979. RELAP5 changed the physics of the code from an HEM into two fluid model physics. Nuclear engineers still widely use RELAP5 today and the code became a basis for TRACE.
While INL was developing RELAP, Los Alamos National Laboratory began development of TRAC in the mid-1970’s. In 1974, Kay Lathrop and Bill Reed submitted a research proposal to develop a modeling program that used more real-world physics than RELAP at the time. The goal was to use the 3-D flow equations; however, the increased accuracy meant longer computational time – the opposite issues that were plaguing RELAP at the time. In an effort to optimize the code, engineers at INL decided to split TRAC into two separate versions – TRAC-P and TRAC-B – for PWRs and BWRs respectively. These became a major influence to TRACE.

By 1996, RELAP5, TRAC-P, and TRAC-B began to resemble each other in capabilities. RELAP was becoming more and more accurate, and TRAC had evolved to be nearly as computationally fast as RELAP. The NRC realized that many of the codes in use performed the same basic functions, and wanted to try to consolidate. By 2007, the NRC successfully integrated the three different codes along with another code called ROMONA into a single entity called TRAC/RELAP Advanced Computational Engine or TRACE. TRACE, like many of the codes used to develop it, was written in modern FORTRAN. Along with TRACE, Applied Programming Technologies developed a graphical user interface known as SNAP. Completed in 2008, SNAP facilitated the construction of input models and made the modeling process and easier and more efficient process. TRACE, with the use of SNAP, is one of the most important nuclear safety analysis codes used in the United States today. The basic historical timeline of reactor simulation can be seen in Figure 1, and Figure 2 and Figure 3 show the graphical user interface that SNAP provides to TRACE users.
Figure 1: Timeline of Reactor Simulation Relevant to TRACE

Figure 2: SNAP Opening Interface

Figure 3: Full TRACE PWR Model in SNAP
Multiphysics Modeling and Literature Review

TRACE is a modernized and up-to-date thermal hydraulic analysis code that has Multiphysics modeling and simulation capabilities. The NRC uses TRACE to analyze and simulate large and small LOCA’s and transients in a nuclear reactor. TRACE handles 1-D and 3-D models, PWR and BWR models, and multiple different physics such as neutronics, heat transfer equations, fluids, and thermal-hydraulic equations. This makes TRACE a highly capable program in analysis and nuclear reactor design, but because it was born as a combination of multiple different simulation codes, it has some minor drawbacks.

Multiphysics modeling is simply the coupled simulation of multiple physical models [3]. Initial nuclear analysis codes were typically designed to perform specialized analysis in on physical area. Then, with the development of more advanced codes, the programs began to include more and more physics models. This is done by coupling the different physical models in such a way that the results of one model can be used to calculate the parameters or results of another model coupled within the code. The physical models can be coupled either explicitly or implicitly. Explicit time step solutions are solved by reducing unknowns using known previous time step solutions, and generally require many smaller time steps to achieve the desired accuracy. Implicit time step solutions solve for all of the unknowns simultaneously for the next time step. Although more complicated, the implicit solution method is typically more stable and can achieve the desired accuracy with less and larger time steps than the explicit solution method.

In modeling of a nuclear reactor, one of the most important physical parameters to measure and simulate is the neutron populations and the time rate of change of the neutrons within the reactor core. The change of the neutron density over time is directly proportional to
the power output of the reactor. The equations that predict the neutron population are therefore exceedingly important in neutron kinetics, reactor modeling, and simulation. These equations are called the point kinetics equations, or PKE equations for short. The PKE equations with one group of delayed neutrons and without the effect of independent neutron sources are listed in the Equation 1-1 and 1-2.

\[
\frac{dn(t)}{dt} = \left(\frac{\rho - \beta}{\Lambda}\right)n(t) + \sum_{i=1}^{6} \lambda_i C_i(t)
\]

\[
\frac{dC_i}{dt} = \frac{\beta_i}{\Lambda} n(t) - \lambda_i C_i
\]

In these equations, \(n(t)\) is the neutron density, \(t\) is the time, \(\rho(t)\) is the reactivity of the reactor as a function of time, \(\beta\) is the total fraction of the delayed neutrons, \(\Lambda\) is the prompt neutron generation time, \(\lambda\) is the radioactive decay constant of the delayed neutron precursor, and \(C(t)\) is the average density of the delayed neutron precursor. The accurate determination of these equations is of prime importance in nuclear modeling and simulation analysis.

The PKE equations are a system of nonlinear ordinary differential equations. The issue in finding a solution to these equations is that they are very stiff. This means the equations are numerically unstable – most numerical solutions are subject to radically different solutions under slightly different parameters or too large of a time step is take. Herein lies the problem and the goal of this paper. As is, TRACE solves for the PKE equations explicitly. An explicit solution to the PKE equations can cause highly unstable solutions for quick and powerful transients. Not only will the solution be unstable, but solving for the code explicitly requires TRACE to take thousands of time steps that are small and impractical. At a specific point, a time step is too small in order to achieve any useful information. This will be explained in detail in Chapter 3.
An implicit solution, in theory, should be able to achieve more stable solutions of similar or greater accuracy than an explicit solution as well as using less time steps. Although this may seem like the computational time would be guaranteed to decrease, the implicit method is often more complicated and is expected to see a greater computational time using the implicit method than the explicit method. In this project, a more simple implicit implementation of the PKE equations was implemented. Other work has been done to solve the PKE equations analytically and implicitly through a variety of methods including power series solutions, rearranging of terms, and different coupling methods.

Computationally fast results of nuclear reactor dynamics that achieve a high level of accuracy are constantly being studied. One analytic method to solve for reactor dynamics is by creating a power series solution, or PWS. A PWS algorithm that used recurrence relations until the desired precision and accuracy of time and reactivity are achieved was created using Visual FORTRAN. Aboanbar (2002) implemented this algorithm and showed the accuracy of this solution method in Figure 4 [4].

![Figure 4: Power Series Solution of PKE Equations](image)
Another analytical solution method to solve the PKE equations is called the Singular Perturbed Method [5]. The research performed to develop this method found that the results of many traditional methods of calculating reactor dynamics do not obtain good or accurate results without using extremely small-time steps, which in turn increases the computational time. This shows yet again the downfalls of solving the neutronics models explicitly loses accuracy when modeling different accident scenarios. The Singularly Perturbed Method was developed to calculate the solution of the PKE equations during delayed supercritical processes with temperature feedback and small step reactivity insertions. This method was shown to be accuracy especially against other methods such as Power Prompt Jump Approximation (PPJ), Precursor Prompt Jump Approximation (PrPJ), and the Temperature Prompt Jump Approximation (TPJ). Although the SPM method was shown to be accurate for the designed scenarios for as shown in Figure 5, obtaining reliable results was determined ineffective for accident-prone scenarios [5].

![Graph: Singularly Perturbed Method vs. Other PKE Calculation Methods](image)

**Figure 5: Singularly Perturbed Method vs. Other PKE Calculation Methods**

More analytical solutions of the point kinetics equations have been developed in recent years, although these methods have a limited range of usefulness and have not been implemented in many nuclear codes. Nahla (2009) was able to develop an analytical technique of computing
reactor dynamics using the roots of the in hour equation, eigenvalues of the coefficient matrix, temperature feedback reactivity, and six groups of delayed neutrons. This method was found to be an accurate computational technique compared to other analytical expressions [6]. Another analytical solution that was similar relied more heavily on temperature feedback was then analyzed against Nahla’s solution method. The two analytical methods were compared in a simulation with an operating reactor at a critical state of 10 MW and $\beta = 0.0065$, $\Lambda = 0.0001$ s, $\lambda = 0.07741$ s$^{-1}$, $K_C = 0.05$ K/MWs, $\alpha = 5*10^{-5}$ K$^{-1}$, and $\rho = \beta/2$. The two analytical methods are accurate for delayed supercritical processes in PWRs under adiabatic conditions and are compared in Figure 6 [7].

![Figure 6: Comparison of two Analytical Expressions of PKE Equations](image)

The former paragraphs have shown that there are different solution methods to solve the point kinetics equations analytically, explicitly, and implicitly. However, it is also necessary to ensure the coupling paradigm of the different physics components in reactor analysis are also consistent and provide accurate and efficient results. Currently, most conventional operator coupling paradigms are inconsistent with the treatment of nonlinear terms across the different physical models. This is because often one physics model is solved and then the results of that model is put into the next model to solve for another desired unknown. When some of the parameters are interdependent to both models, this can create some inconsistencies and the
physics can lag behind. The conventional coupling paradigms for nuclear analysis are displayed in Figure 7 below, with the top simultaneously updating and the bottom staggering the updates of the unknowns [8].

![Figure 7: Methods of Nonlinear Inconsistent Coupling Paradigms](image)

Ragusa performed research on nonlinear consistent coupling methods that provide far more accurate results than nonlinear inconsistent traditional coupling methods. Oftentimes it is possible to modify the coupling within the codes to increase the accuracy of the coupling model and the solution. This technique involves explicit predictions in terms where lagging may occur. However, another method to consistently couple the physical models includes entirely reforming the code so that it implements an implicit stiff solver. This method can resolve discontinuities and create much more stable solutions. The results of the improved coupling methods are shown in Figure 8 below [8]. The implicit implementation method at which the PKE equations were modified in TRACE followed more closely to the conventional iterated implicit method.
The series of figures shown above demonstrate that multiple different numerical and analytical solutions to the point kinetics equations exist. These different solution methods can provide more accurate results compared to traditional solution methods. Solving implicitly in TRACE should increase the accuracy of the equations and the stability of the model as a whole, just at the price of more computational time.
Chapter 2

Theory of Finite Volume Methods

In order to simulate the basic physics of different partial differential equations, such as flow equations or heat transfer equations, the need to discretize the region of interest into a finite number of locations in space and time becomes apparent. Oftentimes, the discretization is referred to as cells, nodes, or volumes. TRACE uses what is known as the Finite Volume Method in order to calculate its results. TRACE uses Finite Volume Methods in order to model and calculate results in both 1-D and 3-D, which makes TRACE that much more of a valuable tool to the NRC. The following pages will go into detail about the finite volume method, and how the finite volume works within TRACE that makes the tool so valuable.

The Finite Volume Method begins by taking the volume in which you are observing and splitting it up into many small parts. The best way to describe this process is within the area of flow in a pipe, such as a pipe in which water flows in a nuclear reactor. As can be seen in Figure 9, you can see the visualized and illustrated. In the total region of interest, the smaller volume being observed is label cell j. This cell is the cell in which you are determining the results for the flow of the liquid. The cell edges are at j + ½ and j – ½ respectively. At these cell edges is where the liquid water will flow into and out of the volume cell of interest. The cell in which the liquid flows through before the cell of interest is labeled j – 1 and the cell after is labeled j + 1. Each cell has a length of Δx, and in this scenario, each volume of interest can be considered to have the same exact width and depth. This is important to calculate the different physical properties that will be calculated later. The water flows through the initial pipe at velocity V.
From here, the pipe is considered to be properly discretized and can be seen in the figure below [2].

![Figure 9: Basic 1-D Pipe Labeled for Finite Volume Method](image)

From here, TRACE is able to calculate all the necessary physical properties it needs to calculate. There are three main conservation equations that TRACE is programmed to obey. Those equations are solved through the finite volume approximations, and their accuracy are dependent on the approximations and the methods used to formulate the discrete equations. These equations are important to keep the nuclear community safe from all that is evil in the world.

First, TRACE follows the equations of mass. This means that TRACE ensures that the conservation of mass is withheld and analyzed in the simulations. This is shown in both the liquid and gas form. They are shown in the equations below:

**Liquid Mass Conservation Equation:**

\[
\frac{\delta}{\delta t} \left[ (1 - \alpha) \rho_l \right] + \nabla \cdot [\rho_l \nu_l] = -\Gamma
\]

This equation shows the conservation of mass while the fluid is in its liquid phase. \( \alpha \) is the void fraction, \( \rho \) is the density of the liquid, and \( \Gamma \) is the vapor generation rate. The vapor generation rate is negative because it is the rate of condensation. This is needed to balance the liquid mass equation.
Gas Mass Conservation Equation:

\[ \frac{\delta}{\delta t} \left( \alpha \rho_g \right) + \nabla \cdot \left( \alpha \rho_g \mathbf{v}_g \right) = \Gamma \]

This equation is very similar to the liquid mass gas conservation equation only corrected
the void fraction \( \alpha \) to only account for gas. The density \( \rho \) is the density of the gas. The
positive \( \Gamma \) implies that there is a net loss of gas as the gas is condensing into a liquid. This
refers to boiling in the fluid.

TRACE also uses the energy conservation equations. There are two energy
equations that TRACE uses – a total energy equation. The first term in this equation refers
to the rate of change of total energy in the system. The second term refers to the advection
of total energy in the system; the first term on the left side of the equation is the pressure
work term. The \( q \) terms refer to the gas and liquid war heat.

\[
\frac{\delta}{\delta t} \left[ \alpha \rho_g e_g + (1 - \alpha) \rho_l e_l \right] + \nabla \cdot \left[ \alpha \rho_g e_g \mathbf{v}_g + (1 - \alpha) \rho_l e_l \mathbf{v}_l \right] = P \nabla \cdot \left[ (1 - \alpha) \mathbf{v}_l + \alpha \mathbf{v}_g \right] + q_{wg} + q_{wl}
\]

This equation refers to both the liquid and gas energy in the system. The equation below
refers to just the liquid portion of energy in the system. If one desires just the gas energy in
the system then they only need to change the 1-\( \alpha \) terms into just \( \alpha \).

\[
\frac{\delta}{\delta t} \left[ (1 - \alpha) \rho_l e_l \right] + \nabla \cdot \left[ (1 - \alpha) \rho_l e_l \mathbf{v}_l \right] = \frac{P \delta \alpha}{\delta t} + P \nabla \cdot \left[ (1 - \alpha) \mathbf{v}_l \right] + qwl - \Gamma h_{st}
\]
Finally, TRACE also uses the equations of motion as primary conservation equations in order to solve for all the unknowns. It will become more apparent in the upcoming pages why this is so important and why these equations were chosen in order to calculate the unknown. Listed below is first the gas equation of motion and then the liquid equation of motion. They are once again very similar, and balance the forces in their respective phases that they represent.

Gas Equation of Motion:

\[
\frac{\delta \vec{v}_g}{\delta t} + \vec{v}_g \cdot \nabla \vec{v}_g = -\frac{1}{\rho_g} \nabla P - \frac{C_i}{\alpha \rho_g} \left( \vec{v}_g - \vec{v}_l \right) \left| \vec{v}_g - \vec{v}_l \right| - \frac{\Gamma^+}{\alpha \rho_g} \left( \vec{v}_g - \vec{v}_l \right) - \frac{C_{wg}}{\alpha \rho_g} \vec{v}_g \left| \vec{v}_g \right| + \vec{g}
\]

Then there is the liquid equation of motion. It describes the motion and velocity as it travels through the system.

Liquid Equation of Motion:

\[
\frac{\delta \vec{v}_l}{\delta t} + \vec{v}_l \cdot \nabla \vec{v}_l = -\frac{1}{\rho_l} \nabla P - \frac{C_i}{(1 - \alpha) \rho_l} \left( \vec{v}_g - \vec{v}_l \right) \left| \vec{v}_g - \vec{v}_l \right| - \frac{\Gamma^-}{(1 - \alpha) \rho_l} \left( \vec{v}_g - \vec{v}_l \right) - \frac{C_{wl}}{(1 - \alpha) \rho_l} \vec{v}_l \left| \vec{v}_l \right| + \vec{g}
\]

In both of these equations, the \( C_i \) term is the interfacial friction term. The \( \Gamma^{+/-} \) term refers to the momentum source between the different phase changes. The \( C_{wl/g} \) terms refers to the wall friction – the loss of momentum due to the phase against the wall.
Now that we have discussed the conservation equations within TRACE, we can go into more detail about the numerical methods that TRACE uses to solve. That is Finite Volume Difference methods. These methods are important to get accurate results, and we will ultimately go into further detail on how these methods can be improved upon.

An example of using the mesh development for finite volume analysis is represented below. From Figure 9, your first goal is to derive the equation of the cell that is in use. This involves integration as shown in the equation below.

\[
\frac{\delta}{\delta t} \int_{V_j} (\alpha \rho_g) j dV_j + \int_{V_j} \nabla \cdot (\alpha \rho_g V_g) j dV_j = \int_{\Gamma_j} \Gamma_j dV_j
\]

From here, you apply Gauss’ Theorem to the advection term, which is the second term on the right hand side. This changes the integral over a volume over the whole of the volume to the whole of the sides.

\[
\frac{\delta}{\delta t} \int_{V_j} (\alpha \rho_g) j dV_j + \oint_{s_j} (\alpha \rho_g V_g) j ds_j = \int_{V_j} \Gamma_j dV_j
\]

From here, the surface integral is changed to include the areas of the two boundaries so that it is purely an area integral.

\[
\frac{\delta}{\delta t} \int_{V_j} (\alpha \rho_g) j dV_j + \int_{A_{j+\frac{1}{2}}} (\alpha \rho_g V_g) j_{+\frac{1}{2}} \cdot dA_{j+\frac{1}{2}} + \int_{A_{j-\frac{1}{2}}} (\alpha \rho_g V_g) j_{-\frac{1}{2}} \cdot dA_{j-\frac{1}{2}} = \int_{V_j} \Gamma_j dV_j
\]
It is important to define both the volume average and the area average of a function in order to simplify the above equation even further. These will be defined below.

Volume Average:

\[
\bar{Y} = \frac{\int_Y Y dV}{V}
\]

Area Average:

\[
< Y > = \frac{\int_A Y dA}{A}
\]

Applying these two simplifications into the above equation yields a simpler equation overall that can be used in TRACE.

\[
\frac{\delta}{\delta t} \left( (\alpha \rho_g) \right) V_j + < \alpha \rho_g V_g >_{j+\frac{1}{2}} A_{j+\frac{1}{2}} - < \alpha \rho_g V_g >_{j-\frac{1}{2}} A_{j-\frac{1}{2}} = \bar{\Gamma}_j V_j
\]

Then, to continue simplifying, divide this equation by the volume \( V \):

\[
\frac{\delta}{\delta t} \left( (\alpha \rho_g) \right) + \frac{< \alpha \rho_j V_g >_{j+\frac{1}{2}} - < \alpha \rho_g V_g >_{j-\frac{1}{2}}}{\Delta x_j} = \bar{\Gamma}_j
\]

Now it is possible to arrange the equation in order to predict and simulate forward in time, which gets us to explicit and implicit methods in the following chapters.

\[
\frac{\alpha \rho_g^{n+1} - \alpha \rho_g^n}{\Delta t} + \frac{< \alpha \rho_j V_g >_{j+\frac{1}{2}} - < \alpha \rho_g V_g >_{j-\frac{1}{2}}}{\Delta x_j} = \bar{\Gamma}_j
\]

The point kinetics reactor neutron equations in general deal with the flux in a reactor. The shape of the flux in a reactor follows the following equation.
\[ \phi(\vec{r}, E, t) = p(t) \psi(\vec{r}, E, t) \]

\[
U = \begin{bmatrix}
\frac{\partial F}{\partial P} & \frac{\partial F}{\partial \alpha} & \frac{\partial F}{\partial T_v} & \frac{\partial F}{\partial T_l} & \frac{\partial F}{\partial T_w} \\
\frac{\partial T}{\partial P} & \frac{\partial T}{\partial \alpha} & \frac{\partial T}{\partial T_v} & \frac{\partial T}{\partial T_l} & \frac{\partial T}{\partial T_w} & \frac{\partial T}{\partial \phi_1} & \frac{\partial T}{\partial \phi_2} \\
\frac{\partial N}{\partial P} & \frac{\partial N}{\partial \alpha} & \frac{\partial N}{\partial T_v} & \frac{\partial N}{\partial T_l} & \frac{\partial N}{\partial T_w} & \frac{\partial N}{\partial \phi_1} & \frac{\partial N}{\partial \phi_2}
\end{bmatrix}
\]

Implicit Point Kinetics Model

\[
\frac{dP}{dt} = R - \beta \frac{\Lambda}{P} + \sum_{i=1}^{l} \lambda_i C_i^{n+1}
\]

\[
C_i^{n+1} = \frac{\beta_i \Delta t}{\Lambda(1 + \lambda_i \Delta t)} P^{n+1} + \frac{C_i^n}{1 + \lambda_i \Delta t}
\]

The equations above accurately describe two group neutron diffusion and the implicit point kinetics model that is in the modified version of TRACE. They describe the power, reactivity, wall temperature, liquid temperature, and the slow and fast fluxes. These are the most important equations to describe the neutronics in TRACE. Equations 2-16 and 2-17 are the modified version of the PKE equations that produce an implicit solution.
Chapter 3

Experimentation and Results

The goal of this research is to compare two different versions of TRACE- Version 5P3 that calculates the point kinetics equations explicitly/semi-implicitly and a modified version of TRACE that calculates the PKE equations implicitly. As an added bonus, the time step results will also be calculated against TRACE V5P3, which was released more recently. The experiments will be compared within two different models. The first model is a basic BWR model. In the BWR, it will first be running at a mostly steady state in order to compare the results between the explicit and implicit model. The parameters and the models will be listed below as well as the particular input decks in Appendix A and Appendix B.

I will start this section by talking about the first model - the BWR model. The BWR model is in the Figure as displayed below. This is what the model looks like in SNAP.

Figure 10: Simple BWR Model for TRACE in SNAP
Item 23 is called a Fill. A Fill component is used to set boundary conditions for any one-dimensional hydraulic component junction. A Fill component sets a coolant velocity or mass-flow boundary condition for the connecting components. This Fill component is a constant velocity component. The parameters are listed below for the Feed water Fill.

![Figure 11: FILL Component Parameters](image)

The next component that will be observed is component number 11. Component 11 is a Break component. A Break component is in many ways the opposite of a Fill component. A Break component sets a pressure boundary condition where fluids leave the system which is ideal for modeling containment systems and LOCA calculations. The parameters of the Steam Line Break are listed in the figure below.

![Figure 12: BREAK Component Parameters](image)
The final component in this model is component 5. Component 5 is referred to as a Channel. This component is used to simulate a BWR fuel assembly within the core region of a BWR pressure vessel. This component, labeled as the Average Power Fuel Bundle, is where the primary analysis of this model will be performed. For this problem, the fuel rods are split into nine separate radial nodes.

The model begins with a power transient that occurs in the first second of the simulation. The transient is a startup transient that begins with no power to a peak power of about 7 GW. After the transient finishes, the power diminishes until it reaches a steady state of about 3.5 GW. The entire simulation occurs over the course of ten seconds. The power output of the transient is displayed in Figure 13 as a side by side comparison of both the implicit and the explicit model and in Figure 14 on a single plot.

![Figure 13: Total Power Comparison](image1)

![Figure 14: Total Power Comparison Single Plot](image2)
From a power standpoint, the implicit solution and the explicit solution produce almost exactly the same results. There are a few slight differences. The peak transient occurs at for the explicit calculation occurs at 0.100884 seconds, whereas the implicit calculation occurs at 0.103598 seconds, which is a predicted 0.002714 seconds later. The peak power is also 1.4% greater in the implicit calculation, where it is calculated to be 7.18777 GW and the explicit form of TRACE calculated the peak power to be 7.08553 GW. The differences can be observed in Figure 15 where AptPlot was used to zoom in on the peak power. These are fairly minor differences in this scenario, thus the goal of achieving similar results in both the implicit and explicit calculations was achieved.

![Figure 15: Peak Power Zoomed In](image)

Figure 16 shows the change in the max average rod temperatures during the transient for the explicit and the implicit model. Both of these parameters are closely related to the neutronics and the simulation, so the accuracy of these parameters with respect to the explicit model is important. As can be seen in Figure 17, the both the explicit and implicit versions of TRACE form the exact same shape for the max fuel rod temperature; however, the explicit calculation predicts the peak temperature occurring at 0.03 seconds after the implicit calculation, and 3 K higher. This is a minor difference again, and the desired results were achieved.
Figure 16: The Max Average Rod Temperature; implicitly calculated (left) and explicitly calculated (right)

Figure 17: The Max Average Rod Temperature Same Plot

It is necessary to ensure that the changes in the neutronics portion of TRACE did not have any unintended effect on the other portions of the code such as the fluids solvers. In Figure 15, the comparison between the fluid velocities through the channel is displayed. In Figure 16, it is verified that the fluid velocities are exactly the same. This confirms that the change in the neutronics in the implicit version of TRACE did not affect the fluid equations.

Figure 18: The Liquid Velocity within the Channel; implicitly calculated (left) and explicitly calculated (right)
There was a further issue with the code. Oftentimes it took the implicit implementation much longer to compute the solution than the explicit formulation. Figure 20 shows the increased CPU time that was required using the implicit version of TRACE. In this example, the implicit version of TRACE took twice as long to reach completion. This trend of increased processing time for the implicit version was noted when using other models as well. The increased processing time could be a result of a small coding error or in the implementation of the implicit equations themselves. A suggestion for future work would be to identify the cause for the significant increase in processing time.
Chapter 4

Conclusions

Although this project is still a work in progress, some conclusions can be made based on the output of results from the BWR model. First, the results from Figures 13 through 19 show that the implicit implementation of the PKE equations in TRACE maintains the accuracy of the explicit implementation of the equations for steady state scenarios. In fact, by observing the differences between the two power outputs, the implicit model appears to output a more stable solution. This shows that the implicit code will be able to provide accurate answers in the future. By using the implicit solution of the PKE equation errors in transient conditions that would be apparent in the explicit solution would be avoided due to the allowance of larger time steps in the implicit method. This can prove to be beneficial when modeling scenarios where transients happen over the course of microseconds, and the accuracy of the power output and the temperature in the accident scenario are needed to avoid failure.

However, the code is not complete. The processing time for the implicit code takes significantly longer than the processing time for the explicit code. When running the problems, the implicit solution doubles the processing time. This may be due to a coding error, but more work will need to go in to improve the code to obtain faster and more desirable processing times.
Chapter 5

Future Work

Future work is necessary to complete this project. The first objective is to create an accident scenario in which the explicit solution of the PKE equations will not produce accurate results. These scenarios will be the most important to be aware of when modeling reactors in industry. It is recommended to look at possible LOCA scenarios and reactivity-initiated accident scenarios. In these accident scenarios, the transient and the power profile change so quickly that explicit time steps will not be small enough to achieve accuracy. In this case, the implicit solution will be able to be more accurate as the accuracy is not dependent on the time step size.

The next portion of upcoming work would need to focus on why the implicit implementation of the PKE equations in TRACE causes the code to slow down by nearly 100% in certain accident scenarios. One of the purposes of this project is to obtain a higher computational efficiency in obtaining just as accurate and even more accurate results. As of now, the implicit implementation, although achieving accuracy, is not remaining computationally efficient. This could be a simple error in the code itself. In order to do this, it would be necessary to go through the code searching for flags and errors. At worst, it would be necessary to rearrange the structuring of the code in order to properly find the solution to the PKE equations implicitly.
Appendix A

TRACE Boiling Water Reactor Input File

free format

*m: SNAP:Symbolic Nuclear Analysis Package, Version 2.6.0, November 09, 2017
*m: PLUGIN:TRACE Version 3.8.8
*m: CODE:TRACE V 5.0 Patch 3
*m: DATE:3/30/18
*
*
*************
* main data *
*************
*
* numtcr ieos inopt nmat id2o
   1  0  1  0  0
one dimensional chan model with point kinetics
*
*
*****************
* namelist data *
*****************
*
&inopts
   iambwr=.TRUE.,
icflow=2,
ikfac=1,
   ipowr=1,
   nfrc1=2,
   nfrc3=2,
   nolt1d=0,
   usesjc=3,
npower=1
&end

* ***************
* Model Flags *
* ***************

* dstep    timet
  0     0.0
* stdyst   transi  ncomp  njun  ipak
  0    1     4     2     0
* epso    epss
  1.0E-3  0.0
* oitmax   sitmax   isolut  ncontr  nccfl
  10   10     0     0     2
* ntsv    ntcb    ntcf    ntrp    ntcp
  1     0     0     1     0

* *************
* component-number data *
* ***********************

* Component input order (IORDER)
*-- type ---- num ----------- name -----------+jun1   jun2  jun3
* CHAN    *  5 s * average power fuel bundle  +  39  8
* BREAK   *  11 s * steam line break          +  8
* FILL    *  23 s * feedwater fill            +  39
* POWER   *  5003 e * average power fuel bundle +
*

*************
* ccfl data *
*************

* cbeta*  1.0  1.0e
* ccflm*   1.0   0.59e
*  
* ccflc*   2.0493902   1.0e
*  
* **************************************************
* Starting Signal Variable Section of Model  *
**************************************************
*  
*         idsv          isvn          ilcn          icn1          icn2
1             0             0             0             0
**************************************************
* Finished Signal Variable Section of Model      *
**************************************************
*  
*  
*  
*  
*  
*  
*  
*  
*  
* **************************************************
* Starting Trip Section of Model  *
***********************************
*  
* Trip Storage Count Card
*  
*         ntse          ntct          ntsf          ntdp          ntsd
0             0             0             0             0
*  
*trip        4
*         idtp          isrt          iset          itst          idsg
4             2             1             1             1
*      setp(1)          setp(2)
-1.0E20         0.0
*      dtsp(1)          dtsp(2)
1.0E20    0.0
* ifsp(1)  ifsp(2)
   0   0
*
******************************************************************************
* Finished Trips: Starting Extras*
******************************************************************************
*
******************************************************************************
* Finished Trip Section of Model *
******************************************************************************
*
*
*
*
******** type    num    userid    component name
chan     5       0 average power fuel bundle
* ncell     nodes     jun1    jun2    epsw
   5       3       39       8       0.0
* nsides
   0
* ichf     iconic     iaxcnd    liqlev    nhcom
   1       0       0       0       0
* width     th     houtl    houtv    toutl
0.536   2.03E-3    0.0       0.0       0.0
* toutv     advbwrf    quadsym    numwrods    nvfrays
   0.0
* ngrp     nchans     nodesr    nrow    ncrz
   1       764       9       8       3
* icrnk     icrlh     nmwrx    nfci    nfcil
   1       1       1       0       0
* fmon     reflood    nzmax    nzmaxw    ibeam
   0       0       9      100       0
<table>
<thead>
<tr>
<th></th>
<th>dznht</th>
<th>dznhtw</th>
<th>dtxht1</th>
<th>dtxht2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0E-3</td>
<td>1.0E-3</td>
<td>2.0</td>
<td>10.0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>hgapo</th>
<th>pdrat</th>
<th>pldr</th>
<th>fucrac</th>
<th>norad</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0E4</td>
<td>1.2982</td>
<td>0.0</td>
<td>1.0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>emcif1</th>
<th>emcif2</th>
<th>emcif3</th>
<th>noani</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.67</td>
<td>0.0</td>
<td>0.0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>emcof1</th>
<th>emcof2</th>
<th>emcof3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.67</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>dx</th>
<th>0.188</th>
<th>1.236</th>
<th>1.236</th>
<th>1.236s</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.466e</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>vol</th>
<th>1.569E-3</th>
<th>0.01248</th>
<th>0.01248</th>
<th>0.01248s</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.559E-3e</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>fa</th>
<th>3.22E-3</th>
<th>0.01001</th>
<th>0.01001</th>
<th>0.01001s</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01001</td>
<td>7.33E-3e</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>kfae</th>
<th>3.2</th>
<th>0.84</th>
<th>1.46</th>
<th>1.46s</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>7.4e</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>kfae</th>
<th>3.2</th>
<th>0.84</th>
<th>1.46</th>
<th>1.46s</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>7.4e</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>grav</th>
<th>1.0</th>
<th>1.0</th>
<th>1.0</th>
<th>1.0s</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>1.0e</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>hd</th>
<th>0.0159</th>
<th>0.0159</th>
<th>0.0159</th>
<th>0.0159s</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0159</td>
<td>0.0159e</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>icflg</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0s</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0e</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>nff</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>1s</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1e</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>lccfl</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0s</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0e</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>alp</th>
<th>0.029493859</th>
<th>0.79913329</th>
<th>0.99929469</th>
<th>1.0s</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0e</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>vl</th>
<th>1.0 0.25014905 -0.21251518 3.38656E-3</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.89581041</td>
<td>1.1654607e</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

|        | vv    | 1.0 1.5631804 0.39995942 0.75469552s |
|--------|-------|--------------------------------------|---|
* vv  * 0.92430652 1.1940024e
* tl  * 457.914  539.027  559.047  558.996s
* tl  * 558.994e
* tv  * 558.46  922.845  953.727  1029.495s
* tv  * 936.511e
* p   * 7.004192E6  7.002364E6  7.002017E6  7.001783E6s
* p   * 7.001623E6e
* pa  * 0.0  0.0  0.0  0.0s
* pa  * 0.0e
* ilev * -1  -1  -1  -1s
* ilev * -1e
* qppp * f 0.0e
* mat  * 6  6 e
* tw  * 460.655  461.168  461.34  547.781  549.443s
* tw  * 550.069  563.738  564.006  564.112  584.116s
* tw  * 579.94  578.263  579.894  576.162  574.662e
* rdx  * 64.0e
* radrd * 0.0  2.329E-3  3.293E-3  4.033E-3  4.657E-3s
* radrd * 5.283E-3  5.397E-3  5.728E-3  6.261E-3e
* matrd * 1  1  1  1s
* matrd * 1  3  2  2 e
* nfax  * 3  3  3e
* rftn * 2084.452  2036.514  1904.124  1667.626s
* rftn * 1407.797  1188.255  1111.247  1102.11s
* rftn * 1093.609  2080.174  2033.439  1905.591s
* rftn * 1682.134  1439.355  1232.733  1160.468s
* rftn * 1152.534  1147.18  1890.204  1862.68s
* rftn * 1775.835  1625.045  1450.185  1282.808s
* rftn * 1223.837  1217.486  1213.057e
* rdpwr * 1.3608914  1.3608914  1.3608914  1.3608914  1.3608914e
* rdpwr * 1.3608914  0.0  0.0  0.0e
* cpowr * 1.0e
* radpw * 1.0  1.0  1.0e
* fpuo2 *  0.0e
* ftd *   0.94e
* gmix * f  0.0e
* pgapt * f  0.0e
* burn * 0.0  0.0  0.0e
*
***** type num userid component name
break  11 0 steam line break
*  jun1 ibty isat ioff adjpress
     8 0 0 0 0
* dxin volin alpin tin pin
     6.25 130.9 1.0 559.0 7.0E6
* pain concin rbmx poff belv
    0.0 0.0 0.0 0.0 0.0
*
***** type num userid component name
fill  23 0 feedwater fill
*  jun1 ifty ioff
    39 1 0
* twtold rfmx concin felv
    0.0 1.0E6 0.0 0.0
* dxin volin alpin vlin tlin
    1.0 1.0 0.25 1.0 454.4
* pin pain flowin vvin tvin
   7.2E6 0.0 0.0 1.0 560.891
*
*
*************************************************************************************************
* Starting Power Components *
*************************************************************************************************
*
****** type num userid component name
power 5003 0 average power fuel bundle
* numpwr chanpow
  1 1
* htnum* 5 e
* irpwty ndgx ndhx nrts nhist
  14 6 69 1000 0
* q235 q239 q238 qavg r239pf
  200.0 200.0 200.0 200.0 0.8
* fisphi rans fp235 fp238
  2.3 1.0 1.0 0.0
* irpwtr irpws v nrpwtb nrpws v nrpwr
  4 1 2 0 0
* izpwtr izpws v nzpwtb nzpws v nzpwr
  0 1 1 0 0
* ipwrad ipwdep promheat decaheat wt bypass
  0 0 0.06 0.06 0.0
* nzpwz nzpwi nfbpwt nrpwr nrpwi
  4 -1 0 1 0
* react tneut rpwoff rrpwm x rpw scl
  0.0 1.625E-5 0.0 1.0E20 1.0
* rpowri zpwin zpwoff rzpwm x
  3.293E9 0.0 0.0 1.0E20
* extsou pldr pdrat fucrac
  0.0 0.0 0.0 0.0
* ircjtb 1.0 ircjtb 2.0 ircjtb 3.0 ircjtb 4.0 ibu 0
  1 1 2 1 0
* ircjtb 1.1 ircjtb 2.1 ircjtb 3.1 ircjtb 4.1 ibu 1
  1 20 1 1 0
* ircjtb 1.2 ircjtb 2.2 ircjtb 3.2 ircjtb 4.2 ibu 2
  1 1 20 1 0
* ircjtb 1.3 ircjtb 2.3 ircjtb 3.3 ircjtb 4.3 ibu 3
  1 2 1 1 -2
<table>
<thead>
<tr>
<th></th>
<th>ircjfm1</th>
<th>ircjfm2</th>
<th>ircjfm3</th>
<th>ircjfm4</th>
<th>isnotb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-2147483648</td>
</tr>
<tr>
<td></td>
<td>powexp</td>
<td>bpp0</td>
<td>bpp1</td>
<td>bpp2</td>
<td>bpp3</td>
</tr>
<tr>
<td>Value</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>zpwzt</td>
<td>0.0</td>
<td>1.236</td>
<td>2.472</td>
<td>3.708e</td>
</tr>
<tr>
<td></td>
<td>zpwtb1</td>
<td>0.0s</td>
<td>1.0779975</td>
<td>1.0690977</td>
<td>0.85290482</td>
</tr>
<tr>
<td></td>
<td>rpwtbr</td>
<td>0.0</td>
<td>0.01s</td>
<td>1000.0</td>
<td>0.02e</td>
</tr>
<tr>
<td></td>
<td>rctf</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td></td>
<td>rctc</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>-8.44E-4</td>
</tr>
<tr>
<td></td>
<td>rctc</td>
<td>0.0</td>
<td>300.0</td>
<td>318.158</td>
<td>336.316</td>
</tr>
<tr>
<td></td>
<td>rctc</td>
<td>372.632</td>
<td>390.789</td>
<td>408.947</td>
<td>427.105</td>
</tr>
<tr>
<td></td>
<td>rctc</td>
<td>463.421</td>
<td>481.579</td>
<td>499.737</td>
<td>517.895</td>
</tr>
<tr>
<td></td>
<td>rctc</td>
<td>554.211</td>
<td>572.368</td>
<td>590.526</td>
<td>608.684</td>
</tr>
<tr>
<td></td>
<td>rctc</td>
<td>645.0</td>
<td>0.0</td>
<td>0.0</td>
<td>-6.40805E-5</td>
</tr>
<tr>
<td></td>
<td>rctc</td>
<td>-9.47594E-5</td>
<td>-1.01697E-4</td>
<td>-1.09274E-4</td>
<td>-1.1749E-4</td>
</tr>
<tr>
<td></td>
<td>rctc</td>
<td>-1.35838E-4</td>
<td>-1.4597E-4</td>
<td>-1.56741E-4</td>
<td>-1.68151E-4</td>
</tr>
<tr>
<td></td>
<td>rctc</td>
<td>-1.92887E-4</td>
<td>-2.06213E-4</td>
<td>-2.20179E-4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>rcal</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.052631579</td>
</tr>
<tr>
<td></td>
<td>rcal</td>
<td>0.15789474</td>
<td>0.21052632</td>
<td>0.26315789</td>
<td>0.31578947</td>
</tr>
<tr>
<td></td>
<td>rcal</td>
<td>0.42105263</td>
<td>0.47368421</td>
<td>0.52631579</td>
<td>0.57894737</td>
</tr>
<tr>
<td></td>
<td>rcal</td>
<td>0.68421053</td>
<td>0.73684211</td>
<td>0.78947368</td>
<td>0.84210526</td>
</tr>
<tr>
<td></td>
<td>rcal</td>
<td>0.94736842</td>
<td>1.0</td>
<td>0.0</td>
<td>-0.0478</td>
</tr>
<tr>
<td></td>
<td>rcal</td>
<td>-0.07884377</td>
<td>-0.09595374</td>
<td>-0.11412244</td>
<td>-0.13334986</td>
</tr>
<tr>
<td></td>
<td>rcal</td>
<td>-0.17498089</td>
<td>-0.19738449</td>
<td>-0.22084681</td>
<td>-0.24536787</td>
</tr>
<tr>
<td></td>
<td>rcal</td>
<td>-0.29758615</td>
<td>-0.32528338</td>
<td>-0.35403934</td>
<td>-0.38385402</td>
</tr>
<tr>
<td></td>
<td>rcal</td>
<td>-0.44665956</td>
<td>-0.47965042</td>
<td>-0.5137e</td>
<td></td>
</tr>
<tr>
<td></td>
<td>rcbm</td>
<td>0.0</td>
<td>299.817</td>
<td>644.261</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>rcbm</td>
<td>-0.02225948</td>
<td>-9.74917E-3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>beta</td>
<td>1.69E-4</td>
<td>8.32E-4</td>
<td>2.64E-3</td>
<td>1.22E-3</td>
</tr>
<tr>
<td></td>
<td>beta</td>
<td>2.47E-4</td>
<td>8.32E-4</td>
<td>2.64E-3</td>
<td>1.22E-3</td>
</tr>
<tr>
<td></td>
<td>lamda</td>
<td>3.87</td>
<td>1.4</td>
<td>0.311</td>
<td>0.115</td>
</tr>
<tr>
<td>------</td>
<td>-------</td>
<td>------</td>
<td>-----</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>0.0127e</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cdgn</td>
<td>6.00683E9</td>
<td>5.900442E11</td>
<td>7.733332E12</td>
<td>5.316526E12</td>
<td>1.209826E13s</td>
</tr>
<tr>
<td>cdgn</td>
<td>4.38167E12e</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cdhn</td>
<td>f 0.0 e</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*****************************************************

* Finished Power Components *

*****************************************************

end

*******************************

*Timestep Data *

*******************************

<table>
<thead>
<tr>
<th></th>
<th>dtmin</th>
<th>dtmax</th>
<th>tend</th>
<th>rtwfp</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0E-5</td>
<td>0.1</td>
<td>10.0</td>
<td>1000.0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>edint</th>
<th>gfint</th>
<th>dmpint</th>
<th>sedint</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0E-5</td>
<td>0.1</td>
<td>5.0</td>
<td>100.0</td>
<td></td>
</tr>
</tbody>
</table>

endflag
BIBLIOGRAPHY


ACADEMIC VITA

Austin James Malencia
ajm6279@psu.edu

Education:
Bachelor of Science in Nuclear Engineering, May 2018
Minors in Military Studies and Mathematics Applications
Schreyer Honors College
The Pennsylvania State University, University Park, PA

Work Experience:
Naval Training
Pearl Harbor, HI (Summer 2017)
- Basic submarine and officer training aboard the USS Columbia (SSN 771)
Pearl Harbor, HI (Summer 2016)
- Basic submarine training aboard the USS TEXAS (SSN 775)
Norfolk, VA/King’s Bay, GA (Summer 2015)
- Basic aviation, surface ship, submarine, and marine training through Norfolk Naval Base

Activities:
Naval ROTC Penn State University
Assistant Public Affairs Officer (Fall 2015)
- Photographed and recorded all battalion functions
Assistant Operations Officer (Fall 2016)
- Managed money and workers for a concession stand at Beaver Stadium
- A part of ROTC tri-service meetings and decisions
Operations Officer (Spring 2018)
- Managed all battalion events for 180 midshipmen, including conferences, fundraisers, and other extra-curricular activities

2017 ANS Student Conference Poster Presentation
Toshiba-Westinghouse Scholars Program

Awards:
President’s Freshman Award, Penn State University
Evan Pugh Scholar Award, Penn State University
2018 Vice President’s Award in ROTC
2018 McCombie Award in ROTC

Project Experience:
Capstone Design Project
Sponsored by Nuclear Regulatory Commission (January 2018-Present)
- Analyzed RIA scenarios based on the CABRI experiment using FRAPTRAN, TRACE, and FAST
- Collected data to perform sensitivity analyses for the different nuclear analysis codes

Implicit Implementation of PKE in TRACE
Sponsored by Toshiba-Westinghouse (Fall 2016-Present)
- Compared results of implicit and explicit version of TRACE for RIA scenarios and BWR operations