A Newton-Krylov Solution to the Coupled Neutronics-Porous Medium Equations

by

Andrew M. Ward

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Nuclear Engineering and Radiological Sciences)
in The University of Michigan
2012

but these are written so that you may believe that Jesus is the Christ, the Son of God, and that by believing you may have life in his name.

—John 20:31

for Laura, to Kahlan

Your patience and unwavering support gave me strength and diligence.

Acknowledgments

I would like to thank my advisor Dr. Thomas Downar for his direction and guidance throughout my graduate studies. I would also like to thank Dr. Volkan Seker for this patience with all my question and guidance in learning about gas reactors and the porous medium equations. Dr. Yunlin Xu was also instrumental to my understanding of Newton's Method, iterative solves, and all things coding. Without their direction, this would not have been possible. Thank you.

My committee was an excellent source of direction and feedback. I would like to thank Dr. John Lee, Dr. James Holloway, and Dr. Divakar Viswanath. Their suggestions and criticisms were very helpful, and several discussions helped move this research forward.

I would like to thank Dr. Ben Collins for his encouragement, observations, assistance, and of course company. His help was vital to the success of this project. I would also like to thank Danial Jabaay and Timothy Drzewiecki for their help in brainstorming and also their contribution to our office environment; makes coming to work all that more enjoyable.

Finally, I would like to thank my wife Laura and my daughter Kahlan. Their sacrifice and patience were so important. I would also like to thank my Mom and Dad for their raising me and their support throughout my education. Lastly, all thanks, praise, glory, and honor is due Jesus Christ, the son of God and savior of the world. Through his blood, there is salvation for all.

Table of Contents

List of Tables

List of Figures

List of Appendices

Abstract

The solution of the coupled field equations for nuclear reactor analysis has typically been performed by solving separately the individual field equations and transferring information between fields. This has generally been referred to as operating splitting and has been applied to a wide range of reactor steady-state and transient problems. Although this approach has generally been successful, it has been computationally inefficient and imposed some limitations on the range of problems considered. The research here investigated fully implicit methods which do not split the coupled field operators and the solution of the coupled equations using Neutron-Krylov methods. The focus of the work here was on the solution of the coupled neutron and temperature/fluid field equations for the specific application to the high temperature gas reactor. The solution of the neutron field equations was restricted to the steady-state multi-group neutron diffusion equations and the temperature fluid solution for the gas reactor involved the solution of the solid energy, fluid energy, and the single phase mass-momentum equations.

In the research performed here, several Newton-Krylov solution approaches have been employed to improve the behavior and performance of the coupled neutronics / porous medium equations as implemented in the PARCS/AGREE code system. The Exact and Inexact Newton's method were employed first, using an analytical Jacobian, followed by a finite difference based Jacobian, and lastly a Jacobian-Free method was employed for the thermal-fluids. Results in the thermal fluids indicate that the Exact Newton's method outperformed the other methods, including the current operator split solution. Finite difference Jacobian and Jacobian-Free were slighty slower than the current solution, though fewer outer iterations were required. In the coupled solution, the exact Newton method performed the best. The finite difference Jacobian with optimized perturbation integrated into the GMRES solve also performed very well, which represented the best iterative solution to the coupled problem. Future analysis will consider the transient problem.

Chapter 1
Introduction

1.1 Motivation

The solution of the coupled field equations for nuclear reactor analysis has typically been performed by solving separately the individual field equations and transferring information between fields. This has generally been referred to as operating splitting and has been applied to a wide range of reactor steady-state and transient problems [1] [2] [3] [4] [5] [6]. Although this approach has generally been successful, it can be computationally inefficient and has imposed some limitations on the range of problems considered. The research here investigated fully implicit methods which do not split the coupled field operators and the solution of the coupled equations was performed using Neutron-Krylov methods. The focus of the work here was on the solution of the coupled neutron and temperature/fluid field equations for the specific application to the high temperature gas reactor. The solution of the neutron field equations was restricted to the steady-state multi-group neutron diffusion equations and the temperature fluid solution for the gas reactor which involved the solution of the solid energy, fluid energy, and the single phase mass-momentum equations.

1.2 Current Coupled Code Solution Methods

The focus of most previous research in coupled code solution methods has been for the Light Water Reactor (LWR) [7] [8] [9]. It has long been recognized that the steady-state and transient analysis of a LWR core is a complex multi-physics problem, involving the simulation of neutron production and transport, heat transfer throughout the structures, and the description of the coolant flow field throughout the system [10] [11] [12] [13]. Such analysis has involved the coupling of separated simulation codes, with each one devoted to the solving of one of the coupled physics problems

[14]. Because of the complexity of the individual field solutions, most of the existing coupled code systems have applied an Operator Splitting (OS) coupling technique, where one code is iterated first to provide boundary conditions to the second code and so on until the last code of the simulation system completes one overall temporal step [15] [14]. The accuracy of such coupling is generally driven by the one code that uses the least accurate numerical scheme and therefore traditional OS coupling methods have been limited to 1st order accuracy. The computational efficiency has further been limited because the non-implicit nature of this step-by-step decomposition required the use of small times steps to ensure the stability of the solution [16] [17]. Some methods have been developed to improve the efficiency of these methods, but the fundamental inefficiency and limitations of operated splitting methods remain. The use of an iterative approach to solve a loosely coupled non-linear set of field equations has generally resulted in long computation times and imposed some limitations on the range of problems that could be solved [18] [19] [20].

The most commonly used thermal-fluid codes in the industry today, such as TRACE [21], RELAP5 [22], or RETRAN [23] and originally TRAC [24], generally apply a mixture of explicit and implicit equation formulations. The numerical approach is generally chosen to minimize stability issues inherent with the first order formulations [25]. The solution of the linear reactor physics problem is generally more straightforward, particularly for the steady-state condition which is the focus of the work here [26] [27]. The methods implemented in the codes SIMULATE-3K [28] and PARCS [29] are representative of the current generation of reactor physics methods in the industry today [30].

The most common technique to couple the thermal-fluid and neutronics equations is depicted in Figure 1.1 in which one field is solved while holding constant the unsolved equations variables in the other field. The simplest example for the temperature/fluid and neutronics equations is a staggered update between three field equation subsets in which the energy equation steps forward, and the n + 1 time step solution is used to solve the fluid pressure equation. The solution is passed back to time n, where the neutronics is solved and the time step is advanced to n+1. Experience has shown this method to be relatively stable for most practical reactor applications.

This method has been used for several years to couple reactor systems codes such as RELAP5 and TRAC to reactor neutronics codes such as PARCS. The coupling of these codes is shown in Figure 1.2 which shows there is a slight difference in the sequence in which the fields are updated. In the RELAP5/PARCS coupling the heat conduction precedes the fluids solution whereas in the TRACE/PARCS coupling the

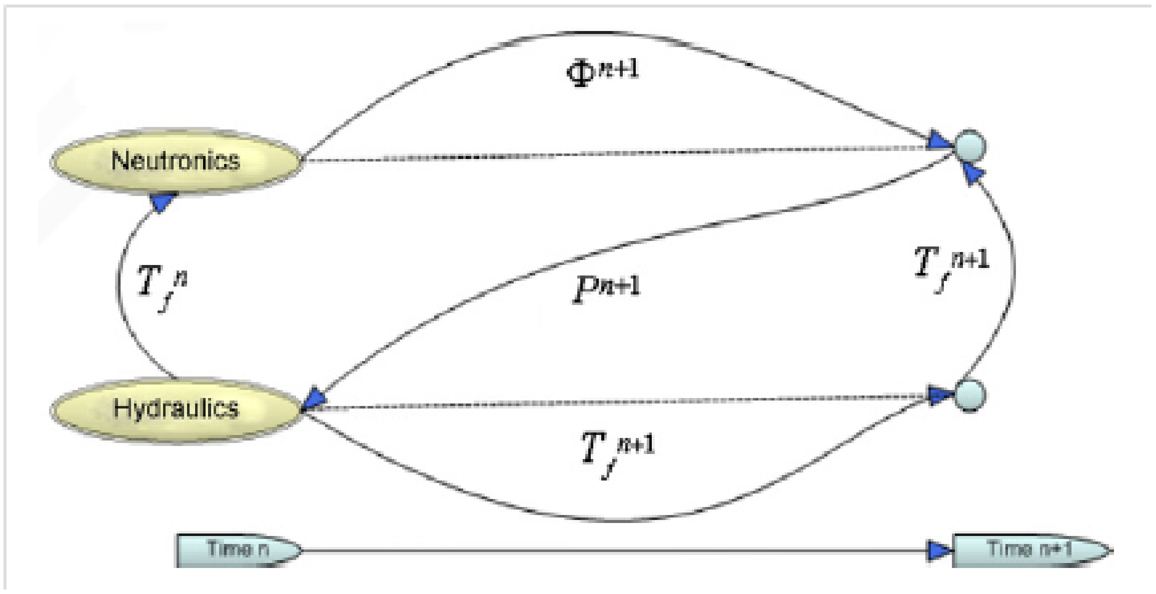

Figure 1.1: Coupling Technique - Staggered Update

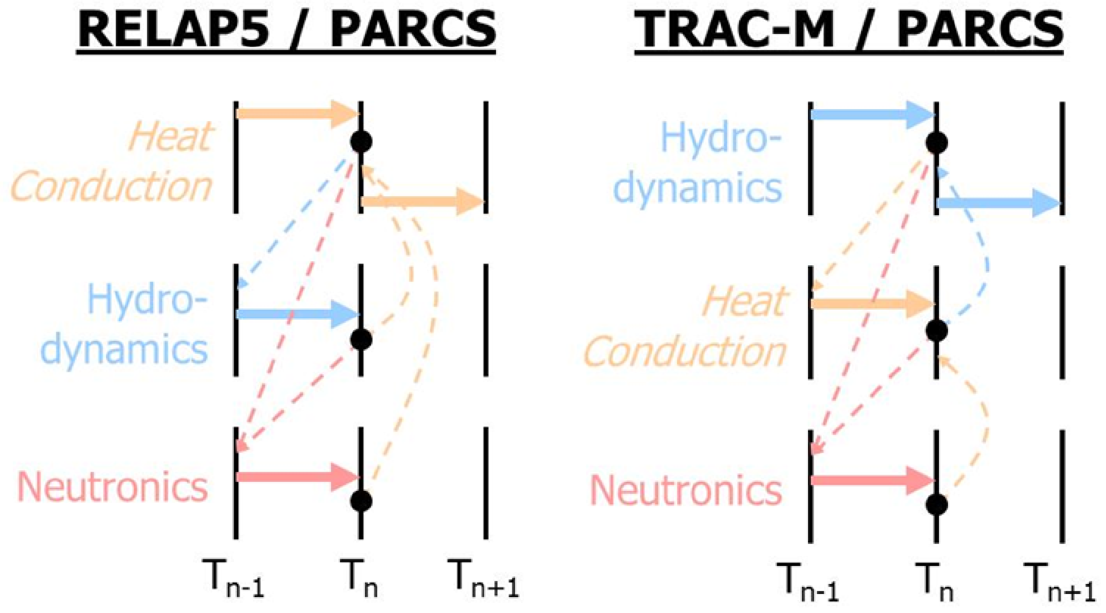

Figure 1.2: Coupling Techniques for RELAP5 / TRAC-M(TRACE)

sequence is reversed. These two schemes represent different physics considerations as well. The conduction driven scheme in RELAP5 repsents the assumption that power will generally change and in term drive the other fields, while the momemtum driven scheme in TRACE, assumes a pressure driven transient, more likely in a BWR.

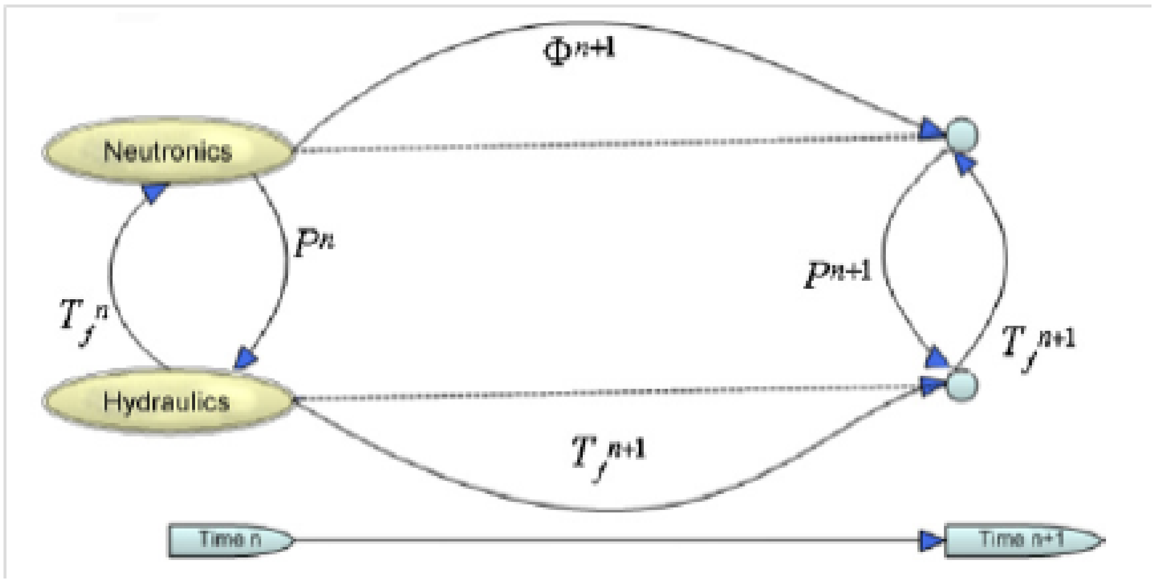

Figure 1.3: Coupling Technique - Simultaneous Update

Variations on the staggered operator splitting method have been proposed over the years and generally involve slight variations such as a simultaneous update of the fields as shown in Figure 1.3. In this method, the neutronics and thermal-fluids time advance together, but between time steps, several data transfers and recalculations are performed to converge the fields. This is the method currently used in the coupling of the U.S. NRC HTR fluids code AGREE to the neutronics PARCS which provided the framework for the research here. Ideally, this simultaneous update method allows for larger time steps, but at the cost of additional computational time.

There has been considerable analysis of the current practice of coupled codes for reactor applications. One of the most comprehensive studies was performed by the OECD [31]. Because there have been only minor advances for practical reactor applications during the last several years, the OECD report still provides a reasonable assessment of the current state-of-the-art. A detailed description of the basic issues of Neutron-Kinetics/Thermal-Hyrdaulic (N-K/T-H) coupling is given in the report, followed by a description of some of the coupling issues which include:

- Coupling approach - integration algorithm or parallel processing
- Ways of Coupling - internal or external
- Spatial mesh overlays
- Coupled time step algorithms
- Coupling numerics - explicit, semi-implicit or implicit schemes
- Coupled convergence schemes

The two first items refer to the different methods than can be used to couple two existing solvers, either integrating one code into the other one (thus resulting into one code), or establishing a dynamic data exchange routine (PVM or MPI based) between the two codes, thus corresponding to a black-box interfacing where only limited modifications to the two solvers are needed. The third item, which corresponds to the problem of exchanging coupling fields computed on different meshing schemes was not an issue for the HTR application here. The review provided of the last three items did provide some useful background, for example the development of the SIMTRAN 3D core dynamics code [32] where staggered alternate time step advancement and extrapolation strategies were used between the two physics (N-K and T-H). This provided the ability to transfer the T-H feedback variables in a nearly implicit manner for the core power calculation. The work summarized in the OECD report addressed primarily the coupling problem for the Light Water Reactor which was not the principal focus of the research here. However, this work did provide a comprehensive perspective on previous coupled code approaches provides and a basis for the investigation of methods to improve the convergence of the existing coupling techniques (OS based).

The presentation of the work performed in this research is organized as follows. The next chapter will summarize the field equations which are currently used in the U.S. NRC codes AGREE and PARCS for High Temperature Gas Reactor analysis and which provided the basis for the research here. Chapter 3 will provide an overview of the implicit methods that were used in the work here, and Chapter 4 will describe the development of these methods for the coupled temperature/fluid and neutronics HTR problem as solved in AGREE/PARCS. Chapter 5 will introduce the test problems used for the thermal-fluids, neutronics, and coupled analysis. Chapter 6 will present the results of applying these methods to practical HTR problems and summary and conclusions are provided in Chapter 7. The principal original and significant contribution of this research is the development of a fully implicit, Newton-Krylov method for the solution of the coupled temperature/fluid and neutronics equations and the application of these methods within the framework of the U.S. NRC code system AGREE/PARCS for the regulatory level analysis of the High Temperature Gas Reactor.

Chapter 2

Current AGREE/PARCS Formulation

The solution of the temperature/fluid equations for the gas reactor problem used in the work here was based on the methods implemented in the U.S. NRC code AGREE which solves three separate field equations: a mass-momentum equation, a fluid energy equation, and a solid energy equation. These are solved separately and explicitly, and several iterations are typically required to converge the TH solution. The converged temperature/fluid solution was then coupled explicity to the PARCS code which solves the multigroup neutron diffusion equation. The equations are given in the following sections.

The application here is to the pebble bed design of the High Temperature Gas Reactor which is described using a cylindrical coordinate system as shown in Figure 2.1. The finite difference method is used to discretize both the neutronics and temperature/fluid equations which provides for coupling to six neighbors as shown in the Figure. The spatial coupling convention of N, S, E, W, T, B for the neighbors will be used throughout the work here.

2.1 Thermal Fluids

The solid energy equation used in AGREE for Pebble Bed applications is the conventional porous medium conduction equation. Because of the pebble contact and mesh size, the conductivity definition is expanded to include other heat transfer mechanisms such as radiative heat transfer. The fluid energy solution also utilizes the porous medium approach and the calculation of the fluid velocity and convective terms requires specialized definitions which will be summarized in the following sections. Finally, the fluid pressure is calculated by combining the mass and momentum equations which considerably simplifies the final form of the equations.

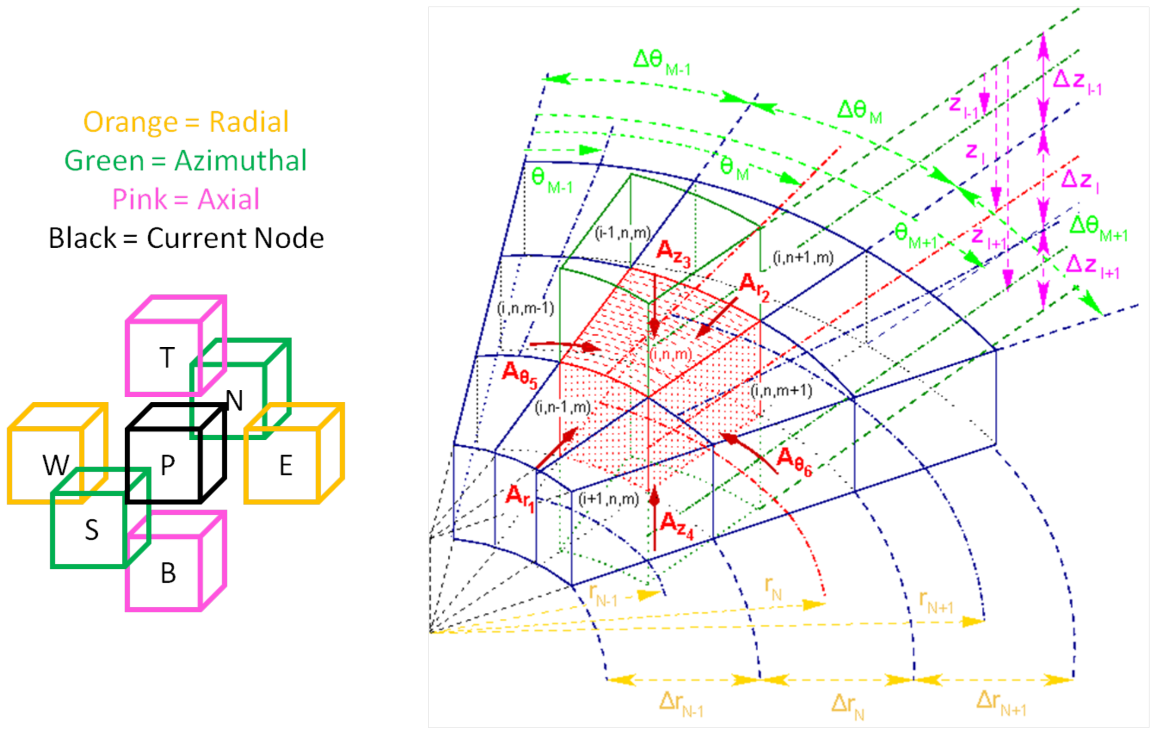

Figure 2.1: AGREE / PARCS Nodalization

The formulation of the three thermal fluid equations results in three nine-stripe matrices. The typical three dimensional finite difference seven stripes which include the node of interest and six neighbors, has two extra stipes to model the periodic boundary condition of the azimuthal direction. After the coefficients of the matrix are determined, the coefficient matrix is constructed for each equation, and then matrices are solved in the following order: pressure, fluid energy, and solid energy. An iterative loop is repeated several times until convergence. Mass flow rate and velocity are calculated after each outer iteration. The velocity is used to calculation Relynolds number, which is used to calculate several coefficients in the system.

2.1.1 Solid Energy Equation

The fuel in the pebble bed reactor is in the form of a pebble and therefore the field equation for the solid energy begins with the spherical conduction equation as shown in equation (2.1), which utilizes the conventional definition of the porosity, epsilon, as the ratio of the fluid volume to the total volume. The LHS represents the time dependent change in the energy stored in the solid. The first three terms on the

RHS describe the radial conduction, azimuthal conduction, and axial conduction, respectively. The heat transfer to the fluid is then given, which depends on the solid and fluid temperatures. Lastly, Q is the heat generation, which in the coupled system is the kappa-fission reaction rate total.

$$\begin{split}\frac{\partial}{\partial t}\left(\left(1-\epsilon\right)\rho_s c_{p,s} T_s\right) \\ = \frac{1}{r}\frac{\partial}{\partial r}\left(\left(1-\epsilon\right)k_s r\frac{\partial T_s}{\partial r}\right) + \frac{1}{r}\frac{\partial}{\partial \theta}\left(\left(1-\epsilon\right)\frac{k_s}{r}\frac{\partial T_s}{\partial \theta}\right) + \frac{\partial}{\partial z}\left(\left(1-\epsilon\right)k_s\frac{\partial T_s}{\partial z}\right) \\ -\alpha\left(T_s - T_f\right) + Q\end{split} \tag{2.1}$$

The finite-volume method is used to solve this problem, and therefore the equation is integrated in space over the discretized geometric grid. The coefficients for the radial conduction terms are given below in which the derivative terms are a function of the underlying grid geometry.

$$D_e = \left(1-\epsilon\right)k_{s,e}\frac{r_e}{\Delta r_e}\Delta\theta\Delta z \tag{2.2a}$$

$$D_w = \left(1-\epsilon\right)k_{s,w}\frac{r_w}{\Delta r_w}\Delta\theta\Delta z \tag{2.2b}$$

$$D_n = \left(1-\epsilon\right)k_{s,n}\frac{1}{r_n}\frac{r_n}{\Delta\theta_n}\Delta r\Delta z \tag{2.2c}$$

$$D_s = \left(1-\epsilon\right)k_{s,s}\frac{1}{r_s}\frac{r_s}{\Delta\theta_s}\Delta r\Delta z \tag{2.2d}$$

$$D_b = \left(1-\epsilon\right)k_{s,b}\frac{1}{\Delta z_b}\frac{r_e^2-r_w^2}{2}\Delta\theta \tag{2.2e}$$

$$D_t = \left(1-\epsilon\right)k_{s,t}\frac{1}{\Delta z_e}\frac{r_e^2-r_w^2}{2}\Delta\theta \tag{2.2f}$$

The conductivity is a strong function of the solid temperature, and therefore must be updated during the outer iterations in the finite difference scheme. Using these variables, and integrating in time, the final form of the equation can be written as equation (2.3)

$$
\begin{aligned}
&(1-\epsilon)\,\rho_s c_{p,s}\frac{1}{\Delta t}\left(T_{s,P}-T_{s,P}^{n-1}\right)\Delta V \\
&\qquad\qquad = D_e\left(T_{s,E}-T_{s,P}\right)+D_w\left(T_{s,W}-T_{s,P}\right) \\
&\qquad\qquad + D_n\left(T_{s,N}-T_{s,P}\right)+D_s\left(T_{s,S}-T_{s,P}\right) \\
&\qquad\qquad + D_b\left(T_{s,B}-T_{s,P}\right)+D_t\left(T_{s,T}-T_{s,P}\right) \\
&\qquad\qquad\qquad + Q\Delta V-\alpha\left(T_{s,P}-T_{f,P}\right)\Delta V
\end{aligned}
\tag{2.3}
$$

Equation (2.3) is the basis for solution of the time dependent heat conduction problem in AGREE. The primary variable is the solid temperature, including the primary node and its neighbors. The solution also depends on the fluid temperature in the primary node. The coefficients depend on the geometry, the node "P" solid temperature, fluid temperature, and fluid pressure.

2.1.2 Fluid Energy Equation

This field equation begins as the conduction / convection for a single phase fluid in cylindrical coordinates and is provided in equation (2.4).

$$
\begin{aligned}
&\frac{\partial}{\partial t}\left(\epsilon\rho_f c_{p,f}T_f\right) \\
&\qquad = -\frac{1}{r}\frac{\partial}{\partial r}\left(r\dot{m}_r c_{p,f}T_f\right)-\frac{1}{r}\frac{\partial}{\partial \theta}\left(\dot{m}_\theta c_{p,f}T_f\right)-\frac{\partial}{\partial z}\left(\dot{m}_z c_{p,f}T_f\right) \\
&\qquad + \frac{1}{r}\frac{\partial}{\partial r}\left(\epsilon k_f r\frac{\partial T_f}{\partial r}\right)+\frac{1}{r}\frac{\partial}{\partial \theta}\left(\epsilon\frac{k_f}{r}\frac{\partial T_f}{\partial \theta}\right)+\frac{\partial}{\partial z}\left(\epsilon k_f\frac{\partial T_f}{\partial z}\right) \\
&\qquad\qquad -\alpha\left(T_f-T_s\right)
\end{aligned}
\tag{2.4}
$$

The LHS term is the time dependent energy stored in the fluid. The terms on the RHS are the radial, azimuthal, and axial convection, or movement of heat by the fluid. The next set of terms are the radial, azimuthal, and axial conduction of energy through the fluid. The final term is the heat exchange between the fluid and solid which couples the fluid and solid energy equations.

The conventional finite-volume method is used to solve this problem, therefore the equation must be integrated in space to be used on the discretized geometric grid. The coefficients for the conduction terms are given in equation (2.5a) and equation (2.6a).

$$D_e = \epsilon k_{f,e} \frac{r_e}{\Delta r_e} \Delta\theta \Delta z \tag{2.5a}$$

$$D_w = \epsilon k_{f,w} \frac{r_w}{\Delta r_w} \Delta\theta \Delta z \tag{2.5b}$$

$$D_n = \epsilon k_{f,n} \frac{1}{r_n} \frac{r_n}{\Delta\theta_n} \Delta r \Delta z \tag{2.5c}$$

$$D_s = \epsilon k_{f,s} \frac{1}{r_s} \frac{r_s}{\Delta\theta_s} \Delta r \Delta z \tag{2.5d}$$

$$D_b = \epsilon k_{f,b} \frac{1}{\Delta z_b} \frac{r_e^2 - r_w^2}{2} \Delta\theta \tag{2.5e}$$

$$D_t = \epsilon k_{f,t} \frac{1}{\Delta z_e} \frac{r_e^2 - r_w^2}{2} \Delta\theta \tag{2.5f}$$

$$F_e = \dot{m}_e c_{p,f} \Delta\theta \Delta z \tag{2.6a}$$

$$F_w = \dot{m}_w c_{p,f} \Delta\theta \Delta z \tag{2.6b}$$

$$F_n = \dot{m}_n c_{p,f} \Delta r \Delta z \tag{2.6c}$$

$$F_s = \dot{m}_s c_{p,f} \Delta r \Delta z \tag{2.6d}$$

$$F_b = \dot{m}_b c_{p,f} \frac{r_e^2 - r_w^2}{2} \Delta\theta \tag{2.6e}$$

$$F_t = \dot{m}_t c_{p,f} \frac{r_e^2 - r_w^2}{2} \Delta\theta \tag{2.6f}$$

In order to treat the convective terms, the velocity must be calculated during the outer iteration. The differencing scheme is a user option in AGREE which provides for upwind, central, and hybrid differencing. The general form is given in equation (2.7a). For purposes of testing and comparison of the solvers developed in the research here, the upwind differencing scheme was used exclusively. This means $A(\mid P \mid)$ is equal to one in all cases.

$$A_E = D_E A\left(\mid P_E \mid\right) + [\| - F_E, 0\|] \tag{2.7a}$$
$$A_W = D_W A\left(\mid P_W \mid\right) + [\| F_W, 0\|] \tag{2.7b}$$
$$A_N = D_N A\left(\mid P_N \mid\right) + [\| - F_N, 0\|] \tag{2.7c}$$
$$A_S = D_S A\left(\mid P_S \mid\right) + [\| F_S, 0\|] \tag{2.7d}$$
$$A_T = D_T A\left(\mid P_T \mid\right) + [\| - F_T, 0\|] \tag{2.7e}$$
$$A_B = D_B A\left(\mid P_B \mid\right) + [\| F_B, 0\|] \tag{2.7f}$$

The final equation is then given in equation (2.8) which is used to solve the time dependent conduction / convection equations in AGREE. The coefficients depend on the solid temperature, fluid temperature, and the fluid pressure. During the transient, it also depends on the time step and the previous fluid temperature.

$$\begin{aligned}\epsilon\rho_f c_{p,f}\frac{1}{\Delta t}\left(T_{f,P} - T_{f,P}^{n-1}\right)\Delta V \\ = A_e\left(T_{f,E} - T_{f,P}\right) + A_w\left(T_{f,W} - T_{f,P}\right) \\ + A_n\left(T_{f,N} - T_{f,P}\right) + A_s\left(T_{f,S} - T_{f,P}\right) \\ + A_b\left(T_{f,B} - T_{f,P}\right) + A_t\left(T_{f,T} - T_{f,P}\right) \\ - \alpha\left(T_{f,P} - T_{s,P}\right)\Delta V\end{aligned} \tag{2.8}$$

2.1.3 Fluid Momentum Equation

This field equation is a combination of the mass conservation and momentum conservation equations. This is achieved by inserting the continuity equation into the momentum equation which is given in equation (2.9).

$$\frac{\partial\left(\epsilon\langle\rho_f\rangle^f\langle v_f\rangle^f\right)}{\partial t} = -\bigtriangledown\langle p\rangle + \epsilon\langle\rho_f\rangle^f\vec{g} - W\epsilon\langle\rho_f\rangle^f\langle v_f\rangle^f \tag{2.9}$$

The time dependent momentum of the fluid is given by the LHS term of equation (2.9) and the pressure gradient is the first term on the RHS. The next term on the RHS is the change in momentum resulting from gravity, and the final term is pressure loss due to the motion of the fluid through the porous medium which utilizes the resistivity as shown in equation (2.10). For the pebble bed HTR application shear and convection are usually small and can be neglected.

$$W = \left(\frac{320}{\frac{Re}{1-\epsilon}} + \frac{6}{\frac{Re}{1-\epsilon}^{0.1}} \right) \frac{1-\epsilon}{\epsilon^3} \frac{1}{d_p} \frac{\mid \langle\rho_f\rangle^f \langle v_f\rangle^f \mid}{2\langle\rho_f\rangle^f} \tag{2.10}$$

The flow rate definition from the continuity equation can be used to replace the time dependent terms in the momentum equation as shown in equation (2.11).

$$\frac{\partial \left(\epsilon\langle\rho_f\rangle^f \langle v_f\rangle^f \right)}{\partial t} = \frac{1}{A} \frac{\partial \dot{m}}{\partial t} \tag{2.11}$$

The final form of the continuity/momentum field equation is given by equation (2.12), which can then be applied to the finite difference formulation used as in the other field equations.

$$\frac{1}{A} \frac{\partial \dot{m}}{\partial t} = -\frac{W}{A} \dot{m} - \left(\frac{\partial p_r}{\partial r} + \frac{1}{r} \frac{\partial p_\theta}{\partial \theta} + \frac{\partial p_z}{\partial z} \right) + \epsilon \langle\rho_f\rangle^f \vec{g} \tag{2.12}$$

The momentum contribution through each face under the discretization scheme above is given by equation (2.13a). A similar spatial integration is used as in the other field equations.

$$\frac{r_P - r_E}{A_E} \frac{\dot{m}_E^n - \dot{m}_E^{n-1}}{\Delta t} = \frac{r_P - r_E}{A_E} W_E^n \dot{m}_E^n - \triangle p_E^n \tag{2.13a}$$

$$\frac{r_P - r_W}{A_W} \frac{\dot{m}_W^n - \dot{m}_W^{n-1}}{\Delta t} = \frac{r_P - r_W}{A_W} W_W^n \dot{m}_W^n - \triangle p_W^n \tag{2.13b}$$

$$\frac{\theta_P - \theta_N}{A_N} \frac{\dot{m}_N^n - \dot{m}_N^{n-1}}{\Delta t} = \frac{r_P - r_N}{A_E} W_N^n \dot{m}_N^n - \frac{1}{r_N} \triangle p_N^n \tag{2.13c}$$

$$\frac{\theta_P - \theta_S}{A_S} \frac{\dot{m}_S^n - \dot{m}_S^{n-1}}{\Delta t} = \frac{r_P - r_S}{A_E} W_S^n \dot{m}_S^n - \frac{1}{r_S} \triangle p_S^n \tag{2.13d}$$

$$\frac{z_P - z_B}{A_B} \frac{\dot{m}_B^n - \dot{m}_B^{n-1}}{\Delta t} = \frac{r_P - r_B}{A_B} W_B^n \dot{m}_B^n - \triangle p_B^n + \vec{g} \triangle z_B / rho_B^n \tag{2.13e}$$

$$\frac{z_P - z_T}{A_T} \frac{\dot{m}_T^n - \dot{m}_T^{n-1}}{\Delta t} = \frac{r_P - r_T}{A_T} W_T^n \dot{m}_T^n - \triangle p_T^n - \vec{g} \triangle z_T / rho_T^n \tag{2.13f}$$

Each equation is solved for the m^n terms. These terms are then summed, and the above definition of the continuity equation is used to solve for the time-dependent mass flow in terms of the change in the density. This can then be expanded as the partial derivative with time for the temperature and pressure to give the final form of the equation below which is solved for the change in pressure for each neighbor and the node of interest.

$$V\frac{\Delta\rho}{\Delta t} = \dot{m}_E^n + \dot{m}_W^n + \dot{m}_N^n + \dot{m}_S^n + \dot{m}_B^n + \dot{m}_T^n \tag{2.14}$$

$$\begin{aligned} &V\left(\frac{\partial\rho}{\partial t}\frac{1}{\Delta t}\left(T_f^n - T_f^{n-1}\right) + \frac{\partial\rho}{\partial p}\frac{1}{\Delta t}\left(p_f^n - p_f^{n-1}\right)\right) = \\ &\frac{\frac{\dot{m}_E^{n-1}}{\Delta t} - G_E \triangle p_E^n}{\left(\frac{1}{\Delta t} + W_E^{n-1}\right)} + \frac{\frac{\dot{m}_W^{n-1}}{\Delta t} - G_W \triangle p_W^n}{\left(\frac{1}{\Delta t} + W_W^{n-1}\right)} + \frac{\frac{\dot{m}_N^{n-1}}{\Delta t} - G_N \triangle p_N^n}{\left(\frac{1}{\Delta t} + W_N^{n-1}\right)} + \frac{\frac{\dot{m}_S^{n-1}}{\Delta t} - G_S \triangle p_S^n}{\left(\frac{1}{\Delta t} + W_S^{n-1}\right)} \\ &+ \frac{\frac{\dot{m}_B^{n-1}}{\Delta t} - G_B \triangle p_B^n + G_B\vec{g} \triangle z_B \rho_B^{n-1}}{\left(\frac{1}{\Delta t} + W_B^{n-1}\right)} + \frac{\frac{\dot{m}_T^{n-1}}{\Delta t} - G_T \triangle p_T^n - G_T\vec{g} \triangle z_T \rho_T^{n-1}}{\left(\frac{1}{\Delta t} + W_T^{n-1}\right)} \end{aligned} \tag{2.15}$$

2.1.4 Fluid Mass Flow Rate and Velocity

The mass flow is calculated on the face of each node, and is a function of the resistivity and pressure. The equation is given in the axial direction is shown in equation (2.16).

$$\dot{m}_z = \frac{G_z}{W_z}\left((p_b - p_p) - \vec{g}\Delta z \rho_f\right) \tag{2.16}$$

where G is a geometric constant, W is the resistivity, $\vec{g}$ is the gravitational constant, and ρ_f is the fluid density. In the solution approach used here, this is calculated after the pressure equation is solved. The mass flow rates are then used to calculate the velocity, which is stored at the cell center. For the axial direction, the velocity is given in equation (2.17).

$$v_z = \frac{1}{2}\frac{G_z}{\rho_f}\left(\dot{m}_T + \dot{m}_B\right) \tag{2.17}$$

The velocity can be expressed directly in terms of pressure and is the quantity which is actually used in the correlations and empirical relationships. The mass flow rate is generally only used for benchmarking and comparison to external data.

2.1.5 Matrix Structure of Theoretical Problem

In order to understand the matrix structure, a small model was used as shown in Figure 2.2. Because the application here is to the Pebble Bed Reactor, the problem is

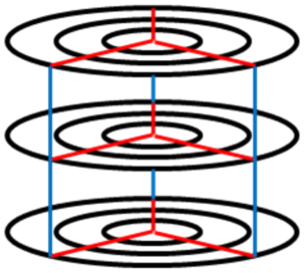
Figure 2.2: Theoretical 3 x 3 x 3 Problem

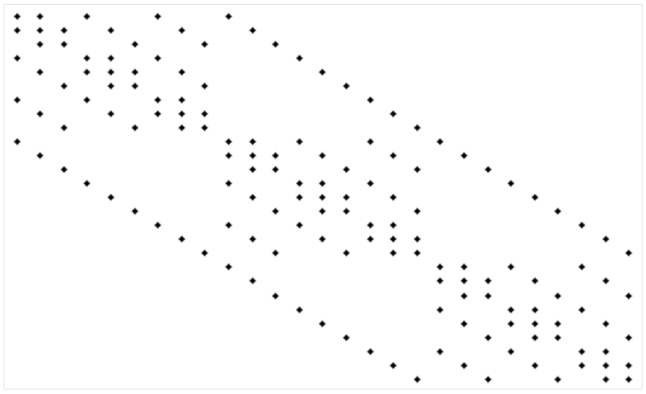
Figure 2.3: Theoretical 3 x 3 x 3 Matrix

modeled in cylindrical geometry and as shown is a 3 x 3 x 3 problem.

As indicated, there are three nodes in each direction and with a numbering scheme of radial, inner to outer, azimuthal, counter-clockwise, and axial, bottom to top, the matrix structure is as shown in Figure 2.3 for a single field equation.

The current solution method in AGREE solves the equations in the following order: Pressure (p_f), Fluid Temperature (T_f), and Solid Temperature (T_s). Because each equation is solved separately, there is no coupling between equations as shown in Figure 2.4.

This can be improved by coupling the available fields together. For example, the heat transfer coefficient is common to the solid energy and fluid energy equations. This represents the off diagonal elements in each equation. Also, the convection terms in the fluid energy equation can be expressed in terms of pressure and the density in the pressure equation can be expressed in terms of fluid temperature. This is shown in Figure 2.5. This represents the tightest coupling possible for the conventional operator split approach currently used in AGREE. In the Newton method which will be described in the following sections, the dependence of the coefficients will be

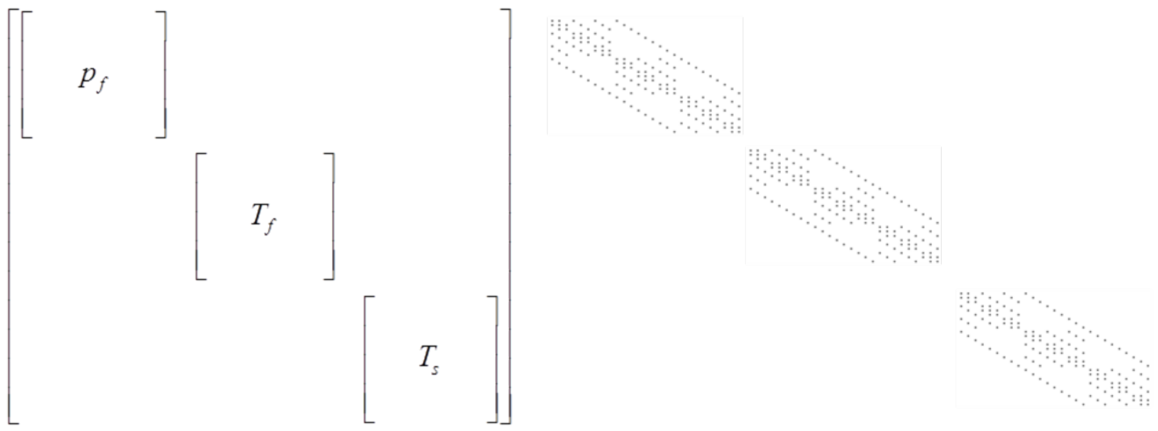

Figure 2.4: Actual Structure of Current AGREE Linear System

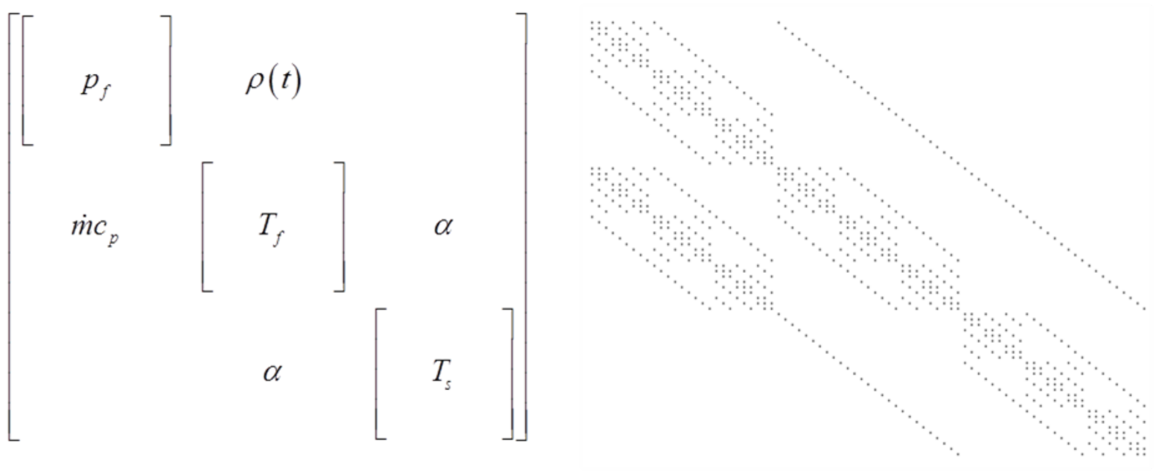

Figure 2.5: Theoretical Structure of Coupled AGREE Linear System

expressed directly which will fill in the off-diagonal elements and provide for tighter coupling of the equations.

2.2 Neutronics

The multigroup neutronics equations for PARCS are a finite difference formulation of the standard diffusion equation in cylindrical geometry. Energy groups are solved independently from each other using a group sweep, and therefore coupled only through the source terms of the other groups.

2.2.1 Multigroup Neutron Diffusion

The diffusion approximation to the transport equation in cylindrical geometry is integrated in space and angle. As in the usual diffusion approximation depicted in equation (2.18) the surface currents are approximated using Fick's law to relate the currents with the fluxes and the diffusion coefficient has the usual relation with the transport cross section defined by the P1 equations [33]. The conventional multigroup approach is used in which the cross sections and group fluxes are defined over a suitable energy range with isotropic fission and scattering sources.

$$\frac{1}{v_g}\frac{d\phi_g}{dt} = \frac{\chi_{p,g}}{k_{eff}}\sum_{g=1}^{G}\nu_{p,g}\Sigma_{f,g}\phi_g + \chi_{d,g}\sum_{k=1}^{K}\lambda_k C_k + \sum_{g'=1(\neq g)}^{G}\Sigma_{s,g'\to g}\phi_{g'} - \Sigma_{t,g}\phi_g + \bigtriangledown_0 D_g \bigtriangledown \phi_g \tag{2.18}$$

The LHS of equation (2.18) indicates the change in neutron flux with respect to time and the RHS contains the source, loss, and migration terms. First is the fission source contribution, with k_{eff} to scale the source and adjust the balance equation. The second term is the delayed neutron source, the third term is the scattering into group g from all other groups. The next term represents the loss of neutrons to absorption and scattering into other groups. Finally, the movement of neutrons into or out of the node of interest is given in the final term. Discretizing in space with the finite difference grid, the diffusion terms become function of the neighbor fluxes. The diffusion coefficients are given in equation (2.19a).

$$\tilde{D}_e = D_e\frac{r_e}{\Delta r_e}\Delta\theta\Delta z \tag{2.19a}$$

$$\tilde{D}_w = D_w\frac{r_w}{\Delta r_w}\Delta\theta\Delta z \tag{2.19b}$$

$$\tilde{D}_n = D_n\frac{1}{r_n}\frac{r_n}{\Delta\theta_n}\Delta r\Delta z \tag{2.19c}$$

$$\tilde{D}_s = D_s\frac{1}{r_s}\frac{r_s}{\Delta\theta_s}\Delta r\Delta z \tag{2.19d}$$

$$\tilde{D}_b = D_b\frac{1}{\Delta z_b}\frac{r_e^2 - r_w^2}{2}\Delta\theta \tag{2.19e}$$

$$\tilde{D}_t = D_t\frac{1}{\Delta z_e}\frac{r_e^2 - r_w^2}{2}\Delta\theta \tag{2.19f}$$

The LHS of equation (2.18) is also expanded in time and the final discretized form of the diffusion equation in cylindrical geometry is given in equation (2.20).

$$
\begin{aligned}
V_p \frac{1}{v_g} \frac{\phi^n_{g,P} - \phi^{n-1}_{g,P}}{\Delta t} = \chi_{g,d} \sum_{k=1}^{K} \lambda_k C_k \\
+ \tilde{D}_{g,E} \left(\phi^n_{g,E} - \phi^n_{g,P}\right) + \tilde{D}_{g,W} \left(\phi^n_{g,W} - \phi^n_{g,P}\right) \\
+ \tilde{D}_{g,N} \left(\phi^n_{g,N} - \phi^n_{g,P}\right) + \tilde{D}_{g,S} \left(\phi^n_{g,S} - \phi^n_{g,P}\right) \\
+ \tilde{D}_{g,B} \left(\phi^n_{g,B} - \phi^n_{g,P}\right) + \tilde{D}_{g,T} \left(\phi^n_{g,T} - \phi^n_{g,P}\right) \\
- \left(\Sigma_{a,g} + \sum_{g'=1(\neq g)}^{G} \Sigma_{s,g\to g'} \right) V_p \phi^n_{g,P} + V_p \left(\sum_{g'=1(\neq g)}^{G} \Sigma_{s,g'\to g} \phi^n_{g',P} + \beta_p \lambda^{(k)} \chi_{p,g} \sum_{g'=1}^{G} \nu \Sigma_{f,g'} \phi^n_{g',P} \right)
\end{aligned}
\tag{2.20}
$$

In steady state, the LHS becomes zero and all neutron production is assumed to be prompt, so the delayed neutrons precursors are also zero. Finally, the delayed neutron fraction becomes one. The value $\frac{1}{k_{eff}}$ is replaced by λ, which is a more practical value to calculate as a primary variable.

2.2.2 Eigenvalue

For the homogeneous, steady-state problem, the eigenvalue is introduced to scale the fission reaction rate over successive iterations and upon convergence provide a unique solution to the neutron diffusion equation. In the methods used here the eigenvalue is defined as in equation (2.21)

$$
\lambda^{(k)} \sum_{g=1}^{G} \nu \Sigma^{(k)}_{f,g} \phi^{(k)}_g = \lambda^0 \sum_{g=1}^{G} \nu \Sigma^0_{f,g} \phi^0_g \tag{2.21}
$$

During the iterations, the equation is actually solved only for $\lambda^{(k)}$, with the cross sections, fluxes, and λ^0 fixed from previous iterations or the flux solve. In practice, higher order methods (e.g. Chebyshev acceleration) can accelerate convergence, but the form used here is the most practical for implementation of Newton's Method.

2.3 Physics Coupling of AGREE / PARCS

When the temperature fluid solution in AGREE is coupled to PARCS, the neutronics solution is solved separately which is similar to the simultaneous updated method discussed in Chapter 1. However, in practice some implicitness can be introduced into the temperature fluid solution by employing a staggered scheme in which some of the coefficients are evaluated outside the iteration loop. For example, because the mass-momentum equation does not contain the solid or fluid temperature variables, the heat transfer coefficients can be outdated during the iteration. The dependence of the primary coefficients are the primary variables is given in Table 2.1. As can be seen, the equivalent conductivity in the solid energy equation depends only on the solid temperature. However, the heat transfer coefficient depends on both temperature fields, pressure, and all three velocity fields. The mass momentum equation is coupled through the fluid density, conductivity, and other parameters. Some coefficients or components are not listed, which include Reynolds number and Prandlt number.

Table 2.1: AGREE / PARCS Coefficient Depedence

Coefficient	Symbol	Dependence
Equivalent Conductivity	k	T_s
Heat Transfer Coefficient	α	$T_s, T_f, p_f, v_x, v_y, v_z$
Flow Resistance	W	$T_s, T_f, p_f, v_x, v_y, v_z$
Cross Section	Σ, D	T_d, T_m

The solution of the spherical conduction equation is used to determine the moderator temperature and Doppler temperature. Because this is a separate calculation requiring boundary conditions, the cross sections actually depend on all the thermal-fluid fields. When coupled to PARCS, the current one-dimensional conduction equation is used to update the cross sections, and then the next iteration uses the new fluxes in the heat generation term.

2.3.1 Matrix Structure of Theoretical Problem

Using the same model problem as used in the previous section for the thermal-fluids, the matrix structure can be represented for the coupled fields as shown in Figure 2.6 and Figure 2.7. Although the coupled problem is not solved in AGREE/PARCS exactly as shown in this representation, the coupling depicted in the figure is the tighest coupling possible with the conventional approach. The most significant drawback is the operator split coupling approach is an absence of coupling between the thermal fluids and neutronics. Specifically, the cross sections strongly depend on the conduction solution, but the cross section temperatures are calculated separately from

$$\begin{bmatrix} [\phi_1] & (\Sigma_s) & & & & & (\lambda F) \\ (\Sigma_s) & [\phi_2] & & & & & (\lambda F) \\ (\kappa\Sigma_f\phi_1) & (\kappa\Sigma_f\phi_2) & [T_s] & (\alpha) & & & \\ & & (\alpha) & [T_f] & (\dot{m}c_p) & & \\ & & & (\rho(t)) & [p_f] & & \\ (\kappa\Sigma_f\phi_1) & (\kappa\Sigma_f\phi_2) & (k,\alpha) & (\alpha) & & [1d] & \\ (\nu\Sigma_f\phi_1) & (\nu\Sigma_f\phi_2) & & & & & [\lambda] \end{bmatrix}$$

Figure 2.6: Theoretical AGREE / PARCS Symbolic Structure

the neutronics. The inclusion of a direct coupling between the fields is one of the strengths of the Newtons Method which is described in the subsequent chapters.

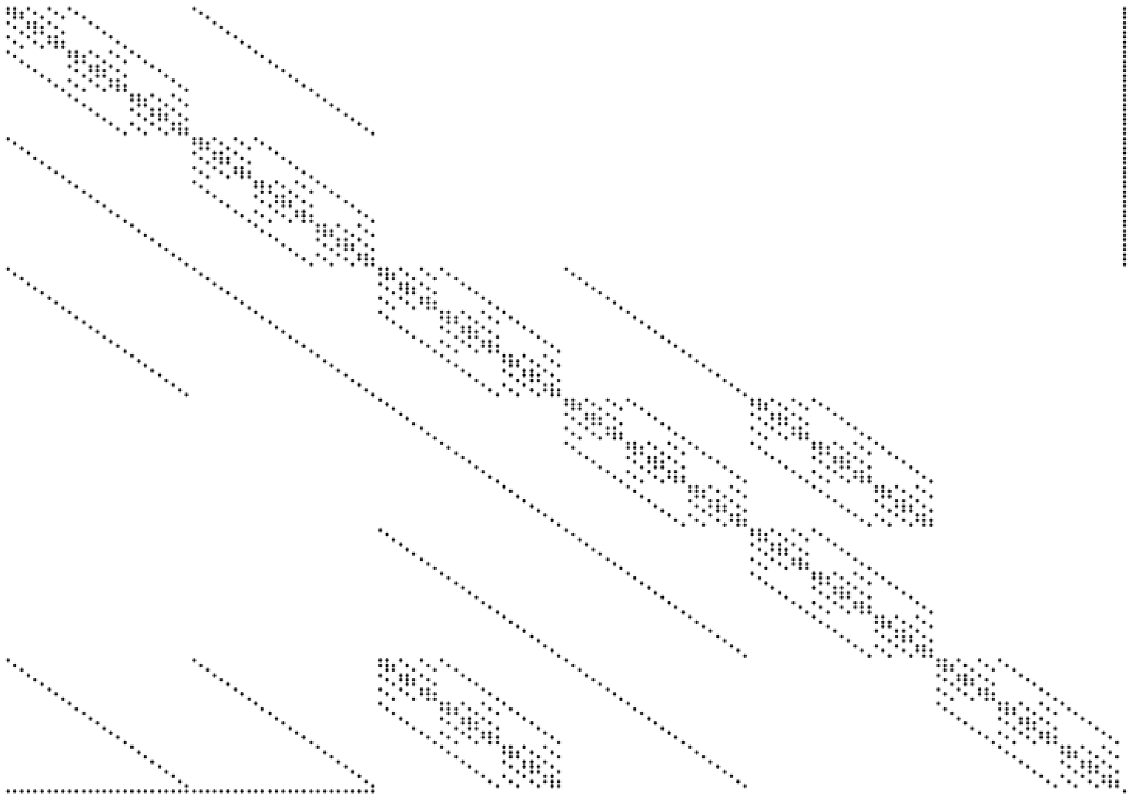

Figure 2.7: Theoretical AGREE / PARCS Matrix Structure

Chapter 3
Implicit Coupling Techniques

3.1 Introduction

Previous researchers have investigated the use implicit numerical techniques for reactor analysis applications. In general, these efforts have been restricted to a single set of field equations, such as a thermal fluids code [34], but implicit methods for coupled codes have been demonstrated using reduced order methods and test codes in a variety of fields [35]. On a larger scale, a recent effort was made to form the complete implicit formulation for PARCS / TRACE [36]. This achieved noticeable speedup for some test problems but the work is still in the development phase. Packed beds and the porous medium equation have received some interest over the years [37] [38] [39] as well. Several researches have outlined the mathematical approaches [40] [41] [42] [43] [44] [45], but as yet there has not been a successful implementation in an engineering grade reactor analysis code. Newton's method has been studied in other fields as well, with multiphysics modeling the common goals.

One of the principal reasons for the limited success of previous implicit coupling methods has been the considerable expense of forming the Jacobian for practical applications. Only recently, methods have been developed to form the analytical Jacobian [46] [47], with increased interest in automatic differentiation. Several code have been developed to use Finite Difference based Jacobians, mainly for the purposes of Uncertainty quantification and sensitivity analysis [48] [49] [50] [51]. Because of the expense of forming and storing the Jacobian, the largest area of research the last several years has been in Jacobian-Free Newton Methods [52] [53] [20] [54] [55]. In these methods, the effect of the Jacobian is approximated using a Taylor series expansion of a matrix-vector product [56]. Many fields are considering these methods, and these include thermal-fluids [57] [58] [59] [60] [61] [62] [63], radiation [64] [65] [66], neutronics [67] [68] [69] [70] [71], structural mechanics [72], fuel performance

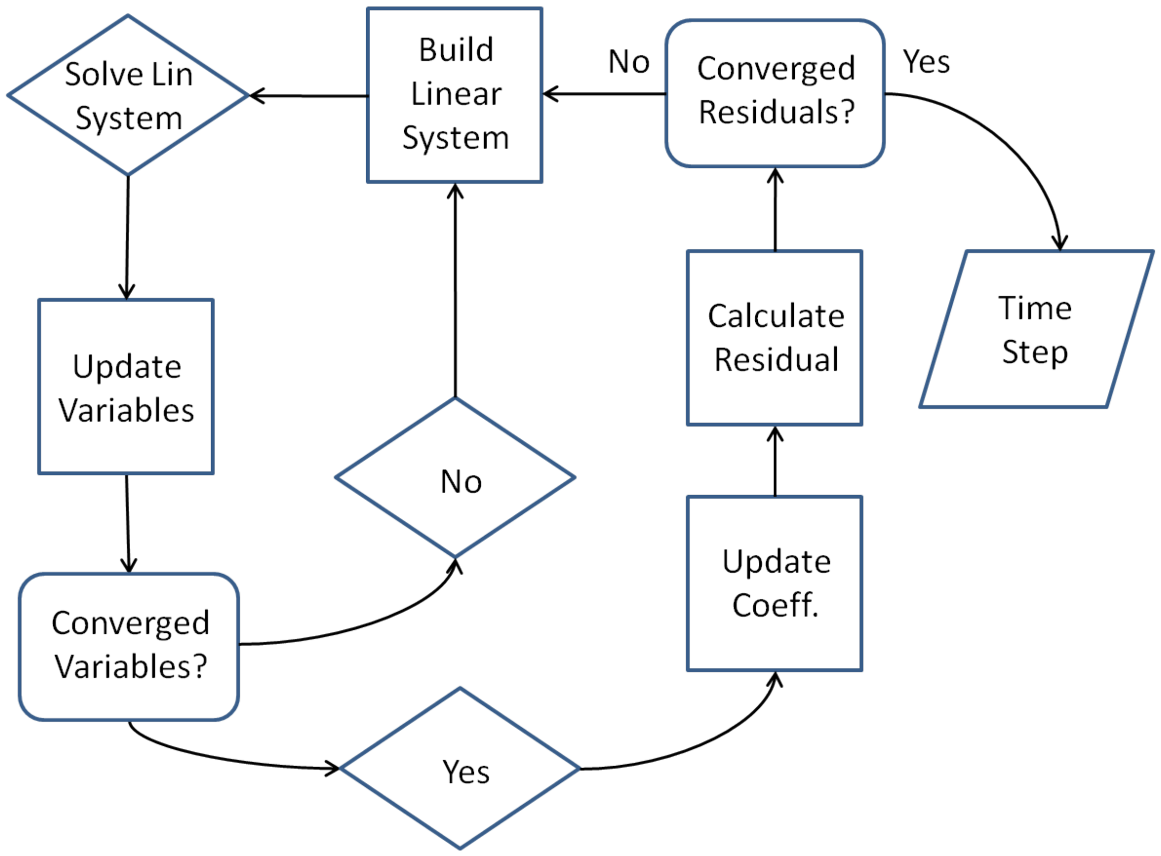

Figure 3.1: Coupling Technique - Fully Implicit

[73], and several other fields [74] [75]. Research into this method has been underway for the PARCS code as well [53]. It is important to note that convergence is not guaranteed with these methods [76] and stagnation has been a problem [44] with approximate methods using Krylov solvers [77] [78] [79]. In order to address potential non-convergence and stagnation issues, the research here will include both the exact and inexact Newton's methods.

A fully implicit solution should solve all the field equations together, and it should also converge the coefficients before time stepping. In general, this is similar to the simultaneous update described in Chapter 1, but there need not be an inner iteration between the coefficient updates and field equation solutions. This is depicted in Figure 3.1 in which the linear system is first solved with an initial guess of the coefficients and after convergence of the linear system the coefficients are updated and a new linear system is constructed. In this scheme, the time steps size can change which can be dictated by the change in coefficients with respect to the field primary variables.

The principal obstacle to the widespread use of implicit coupling of the multiphysics field equations for reactor analysis has been the difficulty in integrating existing legacy physics codes. Additionally, there have been concerns with the cost of maintaining such an integrated code which have been typically validated using an extensive set of benchmarks for the individual fields. Arguments for a more widespread use of implicit methods include a truer physics representation, improved convergence speed, simplification or elimination of code coupling, increasing demand for coupled physics analysis, reduced computational costs, and a simplified method for the quantification of uncertainties.

The majority of coupled physics solutions begin with a linearization of the system of equations, because the system is inherently nonlinear. This enables the use of reliable linear solvers and simple fixed point iteration schemes. There has been considerable research on improved non-linear solvers, but these have not been implemented in production level codes. The impact of non-linearity can be demonstrated using a term, equation (3.1a), of the momentum equation in the TRACE code. The value of the density, ρ, is evaluated at time step n-1, as is the first value of the velocity, v. The second value of v is the variable for the field equation, which created a linear field equation representation for the non-linear problem.

$$\vec{x} = \begin{bmatrix} v & p & T \end{bmatrix} \tag{3.1a}$$

$$\rho v v = a_{1,1} x_1 \tag{3.1b}$$

$$(\rho v) = a_{1,1} \quad v = x_1 \tag{3.1c}$$

$$(\rho v) v = (a_{1,1}) x_1 \tag{3.1d}$$

In this case, the mathematical system is non-linear, but can be represented by a linear system. Numerically, this can introduce stability issues and an increased computational costs. This example is representative of many cross section feedback effects, heat generation terms, heat transfer coefficients, etc. Understanding and addressing these terms is important to achieving an improvement in the convergence of the coupled field solutions in both steady state and transient simulations.

3.2 Newton Iteration

Newtons method has been well established as a method to achieve second order convergence to a non-linear problem. In contrast, operator split approaches can generally only achieve linear convergence. The so-called Newton Iteration is really a Taylor expansion of the primary field variables and coefficients, in which only the first order derivative term is used and the higher order terms ignored.

$$\vec{x} = \vec{x}_0 + \Delta\vec{x} \tag{3.2a}$$

$$f(\vec{x}_0 + \Delta\vec{x}) = f(\vec{x}_0) + \Delta\vec{x} \cdot f'(\vec{x}_0) + \frac{1}{2!}(\Delta\vec{x})^2 \cdot f"(\vec{x}_0) + \ldots \tag{3.2b}$$

$$A(\vec{x}_0 + \Delta\vec{x}) = A(\vec{x}_0) + \Delta\vec{x}A'(\vec{x}_0) \tag{3.2c}$$

This expansion for the coefficients is similar, but a second step is used as shown in equation (3.3a) in which the (k) represents the previous Newton iteration, and the δ terms are solved for in the current Newton step.

$$\vec{x}^T = \begin{bmatrix} y & z \end{bmatrix} \tag{3.3a}$$

$$\delta A \rightarrow \left.\frac{\partial A}{\partial y}\right|_{z=const.}^{(k)} \delta y + \left.\frac{\partial A}{\partial z}\right|_{y=const.}^{(k)} \delta z \tag{3.3b}$$

$$A \rightarrow A^{(k)} + \left.\frac{\partial A}{\partial y}\right|_{z=const.}^{(k)} \delta y + \left.\frac{\partial A}{\partial z}\right|_{y=const.}^{(k)} \delta z \tag{3.3c}$$

The earlier example from the momenetum equation can be expanded and as shown in equation (3.4a), with an assumed source term. Instead of solving for the velocity directly, the solution variable is the change in the velocity which provides a second order convergent approach. The actual velocity is computed by adding the solution to the previous iteration. The coupling between field equations is also much tighter since the non-velocity variables are considered through the expansion of the coefficients. This method therefore has the advantage of not only improving convergence, but also more tightly coupling the field equations.

$$\vec{x}^T = \begin{bmatrix} v & p & T \end{bmatrix} \tag{3.4a}$$

$$(\rho + \delta\rho)(v + \delta v)(v + \delta v) = (\vec{b}) \tag{3.4b}$$

$$\delta\rho vv + 2\rho v \delta v = (\vec{b}) - \rho vv \tag{3.4c}$$

$$2\rho v \delta v + vv\frac{\partial \rho}{\partial T}\delta T + vv\frac{\partial \rho}{\partial p}\delta p = (\vec{b}) - \rho vv \tag{3.4d}$$

In principle, the method is straightforward, but the there are some drawbacks which were alluded to previously. The partial derivatives must be either approximated by a perturbation / difference method, or formulated by symbolically taking the derivative of the coefficient equation. Evaluating these partial derivatives is more expensive in either case. The true impact of this increase on the computational time must be included in assessing the value of improved convergence from the Newton Method.

3.3 Implementations of Newton's Method

3.3.1 Analytical Jacobian

The most robust implementation of Newtons method is the Exact Newton's method in which the Jacobian operator is formed completely for all equations. No finite difference is used to approximate the coefficients, rather each coefficient is formed from the analytical derivatives of each component of the matrix A. Ideally, the derivatives of any empirical relationships which could potentially introduce discontinuities are smoothed to prevent oscillations in convergence. The Jacobian is then inverted, and an exact Newton step is performed and repeated until the solution is converged. A diagram of the algorithm is given in Figure 3.2.

Inexact Newton's method [80] is simply a variation on the direct solution of the linear system. Because Iterative solvers, such as GMRES or BiCGStab, provide an approximate solution, the term "inexact" is used to characterize this variation of the Newtons method. Because the size of the linear system can be large for practical reactor problems, the cost of the linear system solution can be considerable and Inexact Newton methods are more commonly used for most applications.

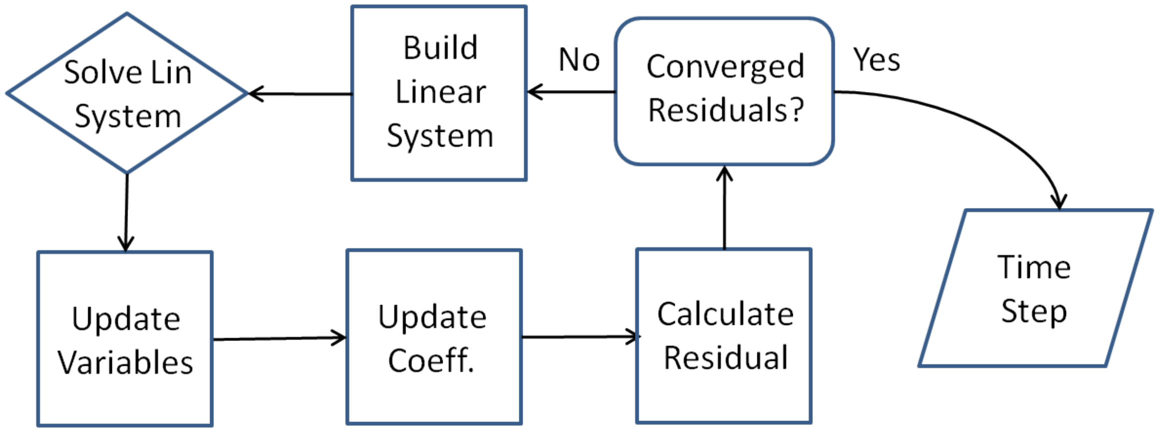

Figure 3.2: Exact Newton's Method Logic

3.3.2 Finite Difference Jacobian

A noted above, the exact Newton's method usually based on the analytical Jacobian. If this is too expensive and if Jacobian-Free methods are ineffective, a finite difference based derivative provides an alternative. In this approach, the routines for calculating a given A matrix coefficient are run twice. The primary variable is perturbed, and a simple linear approximation to the first derivative is obtained. The overall structure of the Jacobian is the same but an analytical derivative step is replaced by the finite difference calculation. Again, either a direct or Krylov solver can be used to solve the system. However, since the Jacobian is an approximation, it is more consistent to use an inexact linear solver such as a Krylov method.

3.3.3 Jacobian-Free / Approximate Block Newton

The achievement of near second order convergence without forming the Jacobian matrix is the goal of using Jacobian-Free approach. Neither the Jacobian-Free nor Approximate Block Newton [81] require the formation of the actual Jacobian, but rather use an approximation based on a first order Taylor expansion. Because the Jacobian is not needed, coding for analytical derivatives or finite difference calcuations are not necessary and in principle the existing formulation can almost be used as implemented. This method does have some drawbacks, which include susceptibility to instabilities and divergence. A diagram of the Jacobian-Free method is shown in Figure 3.3.

The JFNK algorithm is given below and detailed descriptions of the JFNK method

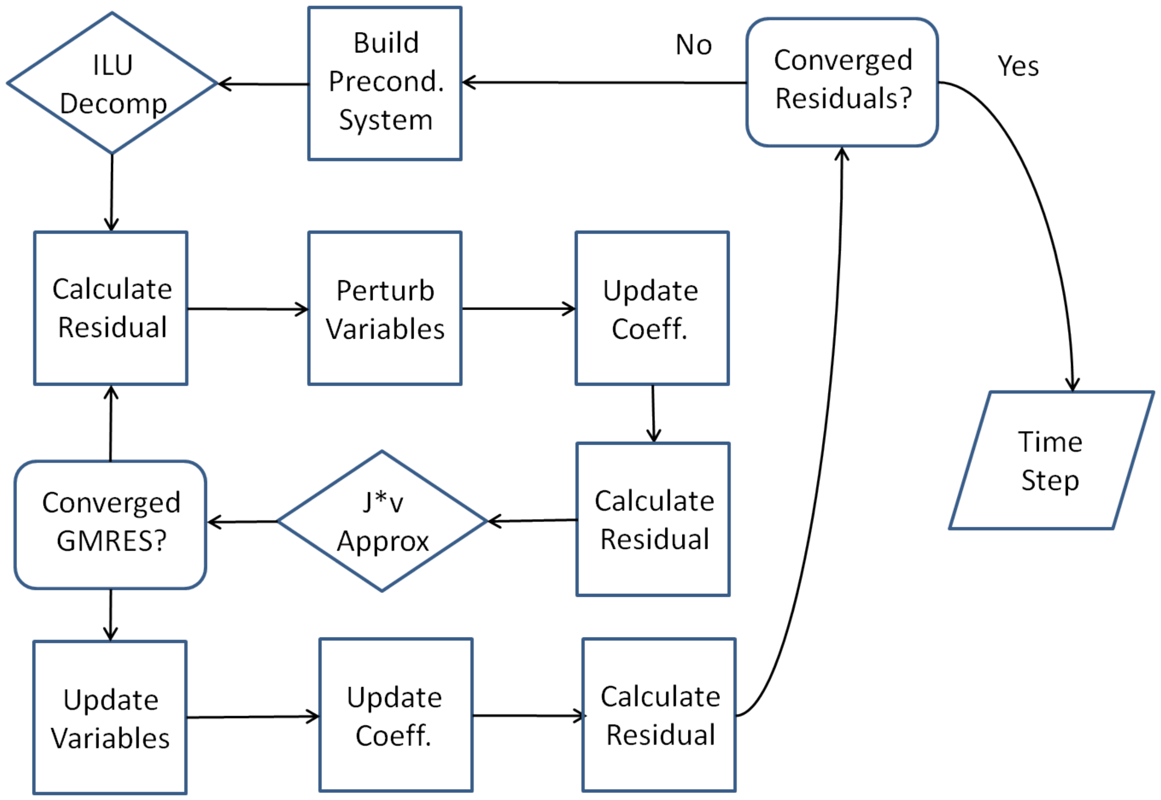

Figure 3.3: Jacobian-Free Method Logic

can be found in several publications.

```
1.  Compute $r_0 = b - Ax$,  $\beta{:} = \|r_0\|_2$  $v_1{:} = r_0/\beta$
2.  Define the $(m+1) \times m$ matrix $H_m = 0$
3.  For $j = 1, 2, ..., m$
4.      Compute $w_j{:} = Jv_j$
5.      For $i = 1, ..., j$
6.          $h_{i,j}{:} = (w_j, v_i)$
7.          $w_j{:} = w_j - h_{i,j}v_i$
8.      End
9.      $h_{j+1,j} = \|w_j\|_2$
10.     $v_{j+1} = w_j/h_{j+1,j}$
11. End
12. Compute $y_m$ the minimizer of $\|\beta e_1 - H_m y\|_2$ and $\delta x_m = \delta x_0 + V_m y_m$
```

Step 4 in the algorithm would normally require the full Jacobian matrix, but a finite difference step can approximate the action of the Jacobian on the vector. This matrix-vector operation will induce the residual in the conventional sense, so the same change must be induced. This is given in equation (3.5a)

$$\vec{r}(\vec{x}) = \vec{b}(\vec{x}) - A(\vec{x})\vec{x} \tag{3.5a}$$

$$\vec{w} = J\vec{v} \approx \frac{\vec{r}(\vec{x} + \epsilon\vec{v}) - \vec{r}(\vec{x})}{\epsilon} \tag{3.5b}$$

Therefore, evaluation of the residual is completed for each new vector w, which is required to fill the subspace. The general approach is straightforward and has been implemented without major code modifications for some applications. One of the principal requirements is to have access to the residuals of each subfield which is not always available for some of the legacy codes.

3.4 Summary

The objective of the research here was to examine both the exact and inexact Newtons method, as well as the Jacobian Free Newton Krylov method in order to assess their effectiveness for practical HTR applications. The overall metric for performance was the robustness of the algorithm and the overall computational time to achieve a converged solution. The following section will describe the methods used to implement Newtons method within the framework of the equations used in the AGREE/PARCS code system which were described in Chapter 2.

Chapter 4

Implicit Formulation for AGREE/PARCS

The most complex and computationally demanding aspect of the Newton Method is the formation of the Jacobian. This chapter will describe the form of the Jacobian for the coupled field equations and then the details of the methods used in this research to construct the Jacobian first for the thermal-fluid equations in AGREE, and then for the coupled field solution in AGREE/PARCS. The exact Jacobian is typically not constructed for practical applications, however, it has been used in some cases.

The Jacobian is formed for the complete set of field equations of the coupled field solution using a Taylor expansion for both the primary variables and coefficients of the systems equations. The matrix elements represent the partial derivatives of each equation with respect to each variable. The full Jacobian is given in Figure 4.1 for the AGREE/PARCS equations described in Chaper 2. Some terms may be zero since the coefficients in an equation may not depend on all primary variables. A red box denotes the thermal-fluid sub-matrix, which will be described first. The RHS is denoted by r_x, which is the residual of each field equation. The LHS coefficient matrix is therefore the partial derivative of each residual with respect to each variable.

4.1 Derivation of Jacobian for the Thermal-Fluids Equations

The Jacobian matrix components for the fully implicit coupled thermal-fluid equations is given in Figure 4.2. The primary variables are the multigroup neutron flux (ϕ_g), solid temperature (T_s), fluid temperature (T_f), fluid pressue (p_f),x-velocity (V_x), y-velocity (V_y), z-velocity (V_z), 1-d conduction (T_{sh} or $1d$), moderator temperature (T_m), Doppler temperature (T_d), lambda (λ). One of the major differences between the operator

$$
\begin{bmatrix}
\frac{\partial r_{\phi_1}}{\partial \phi_1} & \cdots & \frac{\partial r_{\phi_1}}{\partial \phi_g} & \frac{\partial r_{\phi_1}}{\partial T_S} & \frac{\partial r_{\phi_1}}{\partial T_F} & \frac{\partial r_{\phi_1}}{\partial p_f} & \frac{\partial r_{\phi_1}}{\partial v_x} & \frac{\partial r_{\phi_1}}{\partial v_y} & \frac{\partial r_{\phi_1}}{\partial v_z} & \frac{\partial r_{\phi_1}}{\partial 1d} & \frac{\partial r_{\phi_1}}{\partial T_m} & \frac{\partial r_{\phi_1}}{\partial T_d} & \frac{\partial r_{\phi_1}}{\partial \lambda} \\
\vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\frac{\partial r_{\phi_g}}{\partial \phi_1} & \cdots & \frac{\partial r_{\phi_g}}{\partial \phi_g} & \frac{\partial r_{\phi_g}}{\partial T_S} & \frac{\partial r_{\phi_g}}{\partial T_F} & \frac{\partial r_{\phi_g}}{\partial p_f} & \frac{\partial r_{\phi_g}}{\partial v_x} & \frac{\partial r_{\phi_g}}{\partial v_y} & \frac{\partial r_{\phi_g}}{\partial v_z} & \frac{\partial r_{\phi_g}}{\partial 1d} & \frac{\partial r_{\phi_g}}{\partial T_m} & \frac{\partial r_{\phi_g}}{\partial T_d} & \frac{\partial r_{\phi_g}}{\partial \lambda} \\
\frac{\partial r_{T_S}}{\partial \phi_1} & \cdots & \frac{\partial r_{T_S}}{\partial \phi_g} & \frac{\partial r_{T_S}}{\partial T_S} & \frac{\partial r_{T_S}}{\partial T_F} & \frac{\partial r_{T_S}}{\partial p_f} & \frac{\partial r_{T_S}}{\partial v_x} & \frac{\partial r_{T_S}}{\partial v_y} & \frac{\partial r_{T_S}}{\partial v_z} & \frac{\partial r_{T_S}}{\partial 1d} & \frac{\partial r_{T_S}}{\partial T_m} & \frac{\partial r_{T_S}}{\partial T_d} & \frac{\partial r_{T_S}}{\partial \lambda} \\
\frac{\partial r_{T_f}}{\partial \phi_1} & \cdots & \frac{\partial r_{T_f}}{\partial \phi_g} & \frac{\partial r_{T_f}}{\partial T_S} & \frac{\partial r_{T_f}}{\partial T_F} & \frac{\partial r_{T_f}}{\partial p_f} & \frac{\partial r_{T_f}}{\partial v_x} & \frac{\partial r_{T_f}}{\partial v_y} & \frac{\partial r_{T_f}}{\partial v_z} & \frac{\partial r_{T_f}}{\partial 1d} & \frac{\partial r_{T_f}}{\partial T_m} & \frac{\partial r_{T_f}}{\partial T_d} & \frac{\partial r_{T_f}}{\partial \lambda} \\
\frac{\partial r_{p_f}}{\partial \phi_1} & \cdots & \frac{\partial r_{p_f}}{\partial \phi_g} & \frac{\partial r_{p_f}}{\partial T_S} & \frac{\partial r_{p_f}}{\partial T_F} & \frac{\partial r_{p_f}}{\partial p_f} & \frac{\partial r_{p_f}}{\partial v_x} & \frac{\partial r_{p_f}}{\partial v_y} & \frac{\partial r_{p_f}}{\partial v_z} & \frac{\partial r_{p_f}}{\partial 1d} & \frac{\partial r_{p_f}}{\partial T_m} & \frac{\partial r_{p_f}}{\partial T_d} & \frac{\partial r_{p_f}}{\partial \lambda} \\
\frac{\partial r_{v_x}}{\partial \phi_1} & \cdots & \frac{\partial r_{v_x}}{\partial \phi_g} & \frac{\partial r_{v_x}}{\partial T_S} & \frac{\partial r_{v_x}}{\partial T_F} & \frac{\partial r_{v_x}}{\partial p_f} & \frac{\partial r_{v_x}}{\partial v_x} & \frac{\partial r_{v_x}}{\partial v_y} & \frac{\partial r_{v_x}}{\partial v_z} & \frac{\partial r_{v_x}}{\partial 1d} & \frac{\partial r_{v_x}}{\partial T_m} & \frac{\partial r_{v_x}}{\partial T_d} & \frac{\partial r_{v_x}}{\partial \lambda} \\
\frac{\partial r_{v_y}}{\partial \phi_1} & \cdots & \frac{\partial r_{v_y}}{\partial \phi_g} & \frac{\partial r_{v_y}}{\partial T_S} & \frac{\partial r_{v_y}}{\partial T_F} & \frac{\partial r_{v_y}}{\partial p_f} & \frac{\partial r_{v_y}}{\partial v_x} & \frac{\partial r_{v_y}}{\partial v_y} & \frac{\partial r_{v_y}}{\partial v_z} & \frac{\partial r_{v_y}}{\partial \phi_1} & \frac{\partial r_{v_y}}{\partial T_m} & \frac{\partial r_{v_y}}{\partial T_d} & \frac{\partial r_{v_y}}{\partial \lambda} \\
\frac{\partial r_{v_z}}{\partial \phi_1} & \cdots & \frac{\partial r_{v_z}}{\partial \phi_g} & \frac{\partial r_{v_z}}{\partial T_S} & \frac{\partial r_{v_z}}{\partial T_F} & \frac{\partial r_{v_z}}{\partial p_f} & \frac{\partial r_{v_z}}{\partial v_x} & \frac{\partial r_{v_z}}{\partial v_y} & \frac{\partial r_{v_z}}{\partial v_z} & \frac{\partial r_{v_z}}{\partial 1d} & \frac{\partial r_{v_z}}{\partial T_m} & \frac{\partial r_{v_z}}{\partial T_d} & \frac{\partial r_{v_z}}{\partial \lambda} \\
\frac{\partial r_{1d}}{\partial \phi_1} & \cdots & \frac{\partial r_{1d}}{\partial \phi_g} & \frac{\partial r_{1d}}{\partial T_S} & \frac{\partial r_{1d}}{\partial T_F} & \frac{\partial r_{1d}}{\partial p_f} & \frac{\partial r_{1d}}{\partial v_x} & \frac{\partial r_{1d}}{\partial v_y} & \frac{\partial r_{1d}}{\partial v_z} & \frac{\partial r_{1d}}{\partial 1d} & \frac{\partial r_{1d}}{\partial T_m} & \frac{\partial r_{1d}}{\partial T_d} & \frac{\partial r_{1d}}{\partial \lambda} \\
\frac{\partial r_{T_m}}{\partial \phi_1} & \cdots & \frac{\partial r_{T_m}}{\partial \phi_g} & \frac{\partial r_{T_m}}{\partial T_S} & \frac{\partial r_{T_m}}{\partial T_F} & \frac{\partial r_{T_m}}{\partial p_f} & \frac{\partial r_{T_m}}{\partial v_x} & \frac{\partial r_{T_m}}{\partial v_y} & \frac{\partial r_{T_m}}{\partial v_z} & \frac{\partial r_{T_m}}{\partial 1d} & \frac{\partial r_{T_m}}{\partial T_m} & \frac{\partial r_{T_m}}{\partial T_d} & \frac{\partial r_{T_m}}{\partial \lambda} \\
\frac{\partial r_{T_d}}{\partial \phi_1} & \cdots & \frac{\partial r_{T_d}}{\partial \phi_g} & \frac{\partial r_{T_d}}{\partial T_S} & \frac{\partial r_{T_d}}{\partial T_F} & \frac{\partial r_{T_d}}{\partial p_f} & \frac{\partial r_{T_d}}{\partial v_x} & \frac{\partial r_{T_d}}{\partial v_y} & \frac{\partial r_{T_d}}{\partial v_z} & \frac{\partial r_{T_d}}{\partial 1d} & \frac{\partial r_{T_d}}{\partial T_m} & \frac{\partial r_{T_d}}{\partial T_d} & \frac{\partial r_{T_d}}{\partial \lambda} \\
\frac{\partial r_{\lambda}}{\partial \phi_1} & \cdots & \frac{\partial r_{\lambda}}{\partial \phi_g} & \frac{\partial r_{\lambda}}{\partial T_S} & \frac{\partial r_{\lambda}}{\partial T_F} & \frac{\partial r_{\lambda}}{\partial p_f} & \frac{\partial r_{\lambda}}{\partial v_x} & \frac{\partial r_{\lambda}}{\partial v_y} & \frac{\partial r_{\lambda}}{\partial v_z} & \frac{\partial r_{\lambda}}{\partial 1d} & \frac{\partial r_{\lambda}}{\partial T_m} & \frac{\partial r_{\lambda}}{\partial T_d} & \frac{\partial r_{\lambda}}{\partial \lambda}
\end{bmatrix}
\begin{bmatrix}
\delta\phi_1 \\ \vdots \\ \delta\phi_g \\ \delta T_S \\ \delta T_F \\ \delta p_f \\ \delta v_x \\ \delta v_y \\ \delta v_z \\ \delta 1d \\ \delta T_m \\ \delta T_d \\ \delta\lambda
\end{bmatrix}
=
\begin{bmatrix}
r_{\phi_1} \\ \vdots \\ r_{\phi_g} \\ r_{T_S} \\ r_{T_F} \\ r_{p_f} \\ r_{v_x} \\ r_{v_y} \\ r_{v_z} \\ r_{1d} \\ r_{T_m} \\ r_{T_d} \\ r_{\lambda}
\end{bmatrix}
$$

Figure 4.1: Theoretical AGREE / PARCS Jacobian

$$\begin{bmatrix} [\delta T_s] & \alpha,\partial\alpha & \partial\alpha & \partial\alpha & \partial\alpha & \partial\alpha \\ \alpha,\partial a,\partial k,\partial\dot{m}c_p & [\delta T_f] & \partial a,\partial k,\dot{m}c_p,\partial\dot{m}c_p & \partial a,\partial k,\partial\dot{m}c_p & \partial a,\partial k,\partial\dot{m}c_p & \partial a,\partial k,\partial\dot{m}c_p \\ \partial W & \rho,\partial W & [\delta p_f] & \partial W & \partial W & \partial W \\ \partial W & \partial W & W,\partial W & [\delta v_x] & \partial W & \partial W \\ \partial W & \partial W & W,\partial W & \partial W & [\delta v_y] & \partial W \\ \partial W & \partial W & W,\partial W & \partial W & \partial W & [\delta v_z] \end{bmatrix}$$

Figure 4.2: Symbolic Representation of AGREE Jacobian

split matrix structure shown in Chapter 2 and the fully implicit Jacobian shown here is the inclusion of the velocity equations as primary variables in the fully implicit formulation. As noted previously, the heat transfer coefficients are a function of the Reynolds number which depends on velocity. Therefore in order to achieve a fully implicit solution the velocity must be included as a primary variable and analytic derivates of the velocity must be included in the Jacobian. There is no difference in the striping around the primary diagonal, but additional stripes can be seen in Figure 4.2 in the off-diagonal blocks. These stripes provide coupling of the equations through the coefficient expansions and importance of these terms will become evident when comparing solutions of the operator split and fully implicit methods.

The structure of the Jacobian matrix for the thermal-fluid equations is shown in Figure 4.3 for the same 3x3 model problem used to demonstrate the structure of the operator split matrix in Chapter 2.

4.2 Derivation of Jacobian for AGREE/PARCS

The coupled steady state equation system currently solved in AGREE/PARCS was shown previously in Chapter 2. The Jacobian for the fully coupled thermal-fluid and neutronics system is shown in Figure 4.4.

The Jacobian matrix structure for the same 3-D, 3x3x3 model problem described in Chapter 2 is shown in Figure 4.4. Because of the complexity of neutron spectrum in a graphite moderated system, the number of energy groups used in the neutronics solution is typically more than twenty. However, the structure here only shows two energy groups since the additional groups will have a structure similar to the two groups shown here. The highest and lowest energy group will have down- or up-scattering

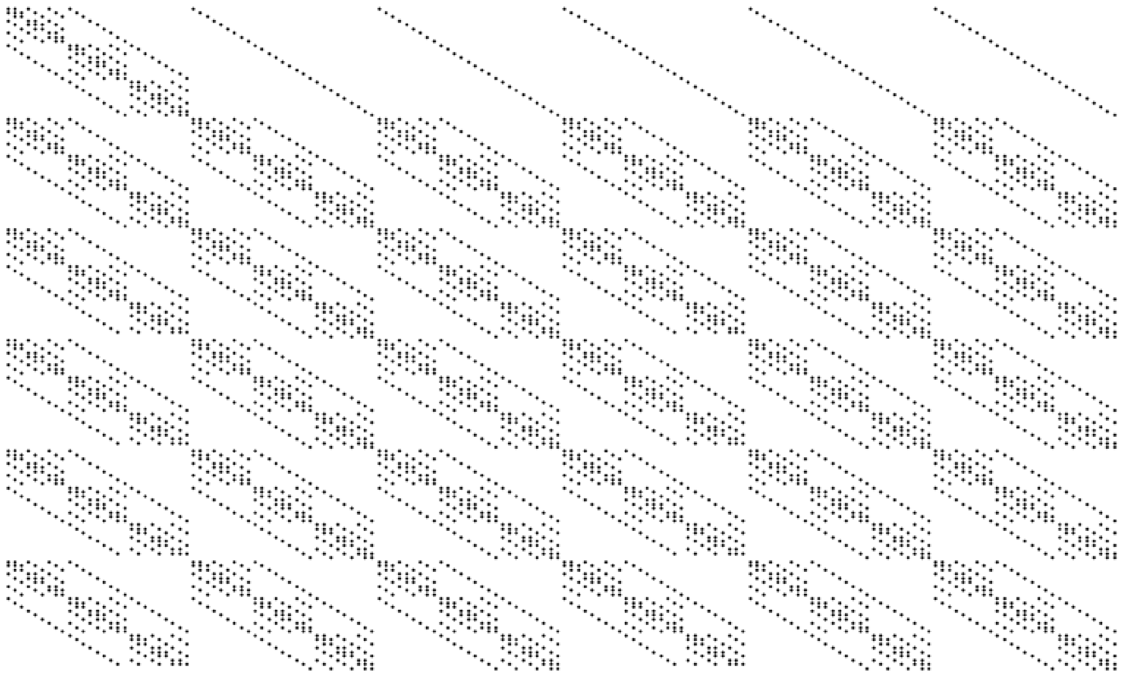

Figure 4.3: Matrix Structure of AGREE Jacobian

$$\begin{bmatrix}
[\delta\phi_1] & \cdots & \Sigma_s & & & & & & & & \partial\Sigma,\partial D & \partial\Sigma,\partial D & \nu\Sigma_f \\
\vdots & \ddots & & & & & & & & & \vdots & \vdots & \vdots \\
\Sigma_s & \cdots & [\delta\phi_g] & & & & & & & & \partial\Sigma,\partial D & \partial\Sigma,\partial D & \nu\Sigma_f \\
\kappa\Sigma_f & \cdots & \kappa\Sigma_f & [\delta T_s] & \alpha,\partial\alpha & \partial\alpha & \partial\alpha & \partial\alpha & \partial\alpha & & \partial\kappa\Sigma_f & \partial\kappa\Sigma_f & \\
 & & & \frac{\partial r_{T_f}}{\partial T_S} & [\delta T_f] & \partial a,\partial k,\dot{m}c_p,\partial\dot{m}c_p & \partial a,\partial k,\partial\dot{m}c_p & \partial a,\partial k,\partial\dot{m}c_p & \partial a,\partial k,\partial\dot{m}c_p & & & & \\
 & & & \partial W & \rho,\partial W & [p_f] & \partial W & \partial W & \partial W & & & & \\
 & & & \partial W & \partial W & \partial W & [\delta v_x] & \partial W & \partial W & & & & \\
 & & & \delta W & \partial W & \partial W & \partial W & [\delta v_y] & \partial W & & & & \\
 & & & \delta W & \delta W & \partial W & \partial W & \partial W & [\delta v_z] & & & & \\
\kappa\Sigma_f & \cdots & \kappa\Sigma_f & \alpha,k,\partial\alpha,\partial k & \alpha,\partial\alpha & \partial\alpha & \partial\alpha & \partial\alpha & \partial\alpha & [\delta 1d] & \partial\kappa\Sigma_f & \partial\kappa\Sigma_f & \\
 & & & & & & & & & 1d & [\delta T_m] & & \\
\kappa\Sigma_f & \cdots & \kappa\Sigma_f & & & & & & & 1d,\partial\kappa\Sigma_f & \partial\kappa\Sigma_f & [\delta T_d] & \\
\nu\Sigma_f & \cdots & \nu\Sigma_f & & & & & & & & \partial\nu\Sigma_f & \partial\nu\Sigma_f & [\delta\lambda]
\end{bmatrix}$$

Figure 4.4: Symbolic Representation of AGREE/PARCS Jacobian

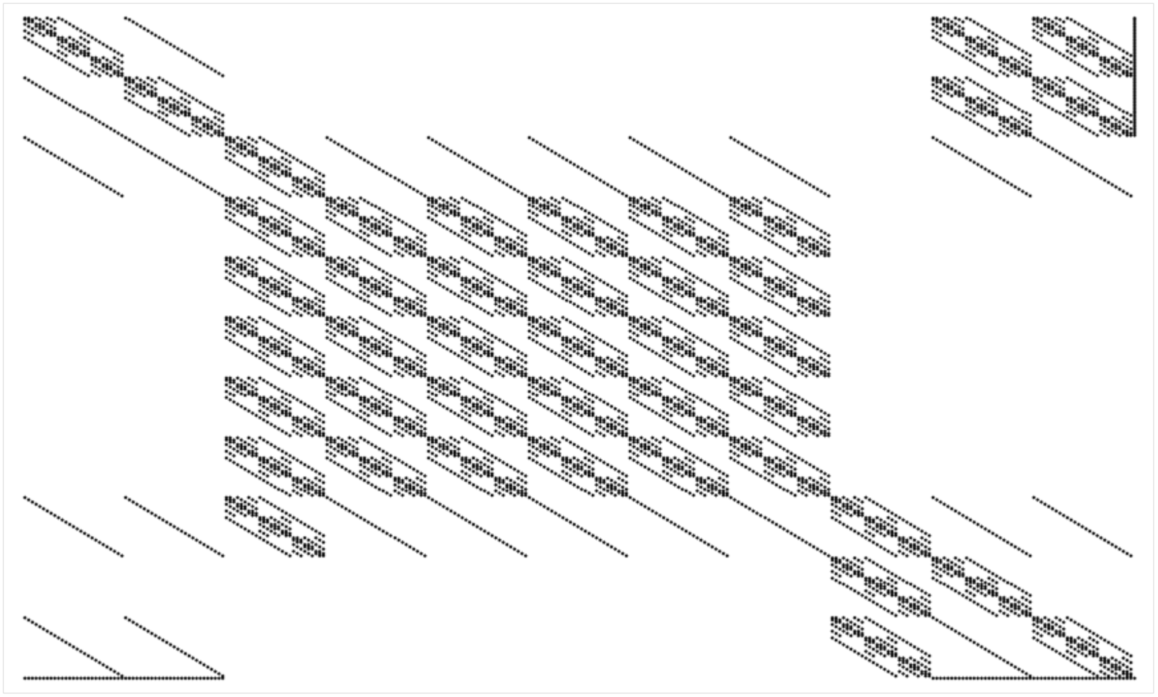

Figure 4.5: Matrix Structure of AGREE/PARCS Jacobian

only, respectively.

Construction of the Jacobian will be shown separately for the thermal-fluids and the neutronics equations. After each subsystem is solved and tested separately, the systems are then coupled to include the off-diagonal coupling terms. The steady state problem is formulated as an eigenvalue problem and the eigenvalue itself is treated as a variable in the Newton iteraction scheme. This provides for an acceleration of the convergence in a fashion similar to conventional Wielandt shift or Chebyshev acceleration methods. In total, the system includes six thermal-fluids equations (mass, energy, and momentum for gas and solid), g neutron diffusion equations, three cross section feedback equations, and one eigvenvalue equation. The following subsection will provide the form of elements for each of the field equations, but the full derivations are given in the appendices.

4.2.1 Newton's Method Neutronics Equations

The derivation for the neutronics equation begins with the diffusion equation in cylindrical geometry given in equation (2.20) and equation (2.21). As noted earlier, the basis for this approach is the Taylor expansion of both the primary variables, the flux and eigenvalue, and the coefficients, the cross sections and diffusion coefficients. Because only steady state is considered, the time dependent terms are ignored.

$$\phi_g \to \phi_g^{(k)} + \delta\phi_g \tag{4.1a}$$

$$\lambda \to \lambda^{(k)} + \delta\lambda \tag{4.1b}$$

$$\Sigma \to \Sigma^{(k)} + \delta\Sigma \tag{4.1c}$$

$$D \to D^{(k)} + \delta D \tag{4.1d}$$

Inserting these into the diffusion equations

$$\begin{aligned}
\left(\tilde{D}_{g,E}^{(k)} + \delta\tilde{D}_{g,E}\right)&\left(\phi_{g,P}^{(k)} + \delta\phi_{g,P} - \phi_{g,E}^{(k)} - \delta\phi_{g,E}\right) \\
&+ \left(\tilde{D}_{g,W}^{(k)} + \delta\tilde{D}_{g,W}\right)\left(\phi_{g,P}^{(k)} + \delta\phi_{g,P} - \phi_{g,W}^{(k)} - \delta\phi_{g,W}\right) \\
&+ \left(\tilde{D}_{g,N}^{(k)} + \delta\tilde{D}_{g,N}\right)\left(\phi_{g,P}^{(k)} + \delta\phi_{g,P} - \phi_{g,N}^{(k)} - \delta\phi_{g,N}\right) \\
&+ \left(\tilde{D}_{g,S}^{(k)} + \delta\tilde{D}_{g,S}\right)\left(\phi_{g,P}^{(k)} + \delta\phi_{g,P} - \phi_{g,W}^{(k)} - \delta\phi_{g,S}\right) \\
&+ \left(\tilde{D}_{g,B}^{(k)} + \delta\tilde{D}_{g,B}\right)\left(\phi_{g,P}^{(k)} + \delta\phi_{g,P} - \phi_{g,B}^{(k)} - \delta\phi_{g,B}\right) \\
&+ \left(\tilde{D}_{g,T}^{(k)} + \delta\tilde{D}_{g,T}\right)\left(\phi_{g,P}^{(k)} + \delta\phi_{g,P} - \phi_{g,T}^{(k)} - \delta\phi_{g,T}\right) \\
+ \left(\left(\Sigma_{a,g}^{(k)} + \delta\Sigma_{a,g}\right) + \sum_{g'=1(\neq g)}^{G}\left(\Sigma_{s,g\to g'}^{(k)} + \delta\Sigma_{s,g\to g'}\right)\right)&V_p\left(\phi_{g,P}^{(k)} + \delta\phi_{g,P}\right) \\
- V_p\sum_{g'=1(\neq g)}^{G}&\left(\Sigma_{s,g'\to g}^{(k)} + \delta\Sigma_{s,g'\to g}\right)\left(\phi_{g',P}^{(k)} + \delta\phi_{g',P}\right) \\
- V_p\lambda\chi_{p,g}\sum_{g'=1}^{G}&\left(\nu\Sigma_{f,g'}^{(k)} + \delta\nu\Sigma_{f,g'}\right)\left(\phi_{g',P}^{(k)} + \delta\phi_{g',P}\right) = 0
\end{aligned} \tag{4.2}$$

The diffusion coefficient and cross sections must be expanded as well. The $\delta\Sigma$ and δD can be expanded into a partial derivative, to accomodate the underlying dependence.

$$\delta\Sigma \to \frac{\partial\Sigma}{\partial T_m}\delta T_m + \frac{\partial\Sigma}{\partial T_d}\delta T_d \tag{4.3}$$

After some algebra, shown in the Appendicies, the final form of the diffusion equation is given as

$$
\begin{aligned}
&= S^k + V_p\chi_{gp}\sum_{g'=1}^{G}\left(\nu\Sigma_{f,g'}^{(k)}\phi_{g',P}^{(k)}\right)\delta\Lambda \\
&+ \left(\tilde{D}_{g,E}^{(k)}\delta\phi_{g,E} + \tilde{D}_{g,W}^{(k)}\delta\phi_{g,W} + \tilde{D}_{g,N}^{(k)}\delta\phi_{g,N} + \tilde{D}_{g,S}^{(k)}\delta\phi_{g,S} + \tilde{D}_{g,B}^{(k)}\delta\phi_{g,B} + \tilde{D}_{g,T}^{(k)}\delta\phi_{g,T}\right) \\
&+ \left[-\left(\tilde{D}_{g,E}^{(k)} + \tilde{D}_{g,W}^{(k)} + \tilde{D}_{g,N}^{(k)} + \tilde{D}_{g,S}^{(k)} + \tilde{D}_{g,B}^{(k)} + \tilde{D}_{g,T}^{(k)}\right) - V_p\left(\Sigma_{a,g}^{(k)} + \sum_{g'\neq g}^{G}\Sigma_{s,gg'}^{(k)}\right)\right]\delta\phi_{g,P} \\
&+ V_p\left[\sum_{g'=1(\neq g)}^{G}\Sigma_{s,g'g}^{(k)}\delta\phi_{g',P} + \chi_{gp}\Lambda^{(k)}\sum_{g'=1}^{G}\nu\Sigma_{f,g'}^{(k)}\delta\phi_{g',P}\right] \\
&+\left(\phi_{g,EWNSBT}^{(k)} - \phi_{g,P}^{(k)}\right)\left[\frac{\partial\tilde{D}_{g,EWNSBT}}{\partial T_{M,EWNSBT}}\left\{\delta T_{M,EWNSBT}\right\} + \frac{\partial\tilde{D}_{g,EWNSBT}}{\partial T_{D,EWNSBT}}\left\{\delta T_{D,EWNSBT}\right\}\right] \\
&+\left(\phi_{g,EWNSBT}^{(k)} - \phi_{g,P}^{(k)}\right)\left[\frac{\partial\tilde{D}_{g,EWNSBT}}{\partial T_{M,P}}\left\{\delta T_{M,P}\right\} + \frac{\partial\tilde{D}_{g,EWNSBT}}{\partial T_{D,P}}\left\{\delta T_{D,P}\right\}\right] \\
&- V_p\phi_{g,P}^{(k)}\left(\left(\frac{\partial\Sigma_a}{\partial T_D}\right)^{n-1} + \left(\sum_{g'=1(\neq g)}^{G}\left(\frac{\partial\Sigma_{s,gg'}}{\partial T_D}\right)^{n-1}\right)\right)\delta T_{D,P} \\
&+ V_p\left(\sum_{g'=1(\neq g)}^{G}\phi_{g',P}^{(k)}\left(\frac{\partial\Sigma_{s,g'g}}{\partial T_D}\right)^{n-1} + \chi_{gp}\Lambda^{(k)}\left(\sum_{g'=1}^{G}\phi_{g',P}^{(k)}\left(\frac{\partial\nu\Sigma_{f,g'}}{\partial T_D}\right)^{n-1}\right)\right)\delta T_{D,P} \\
&- V_p\phi_{g,P}^{(k)}\left(\left(\frac{\partial\Sigma_a}{\partial T_M}\right)^{n-1} + \left(\sum_{g'=1(\neq g)}^{G}\left(\frac{\partial\Sigma_{s,gg'}}{\partial T_M}\right)^{n-1}\right)\right)\delta T_{M,P} \\
&+ V_p\left(\sum_{g'=1(\neq g)}^{G}\phi_{g',P}^{(k)}\left(\frac{\partial\Sigma_{s,g'g}}{\partial T_M}\right)^{n-1} + \chi_{gp}\Lambda^{(k)}\left(\sum_{g'=1}^{G}\phi_{g',P}^{(k)}\left(\frac{\partial\nu\Sigma_{f,g'}}{\partial T_M}\right)^{n-1}\right)\right)\delta T_{M,P}
\end{aligned}
\tag{4.4}
$$

An important difference from the standard diffusion equation is that the eigenvalue is now a primary variable. This allows the eigenvalue to change in reponse to changes in the flux, and vice versa. Another difference is inclusion of the cross section temperature dependence. In the standard diffusion equation formulation, the cross sections dependence on temperature are not included within the neutronics solution, but are included after a thermal-fluids solve. Therefore, in the Newton's Method it is expected the nonlinear coefficients will converge more quickly.

$$
\begin{aligned}
&+\left[\left(\tilde{D}^{(k)}_{g,E}+\tilde{D}^{(k)}_{g,W}+\tilde{D}^{(k)}_{g,N}+\tilde{D}^{(k)}_{g,S}+\tilde{D}^{(k)}_{g,B}+\tilde{D}^{(k)}_{g,T}\right)+V_p\left(\Sigma^{(k)}_{a,g}+\sum_{g'\neq g}^{G}\Sigma^{(k)}_{s,gg'}\right)\right]\delta\phi_{g,P}\\
&-V_p\left(\sum_{g'=1(\neq g)}^{G}\Sigma^{(k)}_{s,g'g}\delta\phi_{g',P}+\chi_{gp}\Lambda^{(k)}\sum_{g'=1}^{G}\nu\Sigma^{(k)}_{f,g'}\delta\phi_{g',P}\right)\\
&-V_p\chi_{gp}\sum_{g'=1}^{G}\left(\nu\Sigma^{(k)}_{f,g'}\phi^{(k)}_{g',P}\right)\delta\Lambda\\
&-\left(\tilde{D}^{(k)}_{g,E}\delta\phi_{g,E}+\tilde{D}^{(k)}_{g,W}\delta\phi_{g,W}+\tilde{D}^{(k)}_{g,N}\delta\phi_{g,N}+\tilde{D}^{(k)}_{g,S}\delta\phi_{g,S}+\tilde{D}^{(k)}_{g,B}\delta\phi_{g,B}+\tilde{D}^{(k)}_{g,T}\delta\phi_{g,T}\right)\\
&=S^k
\end{aligned}
\tag{4.5}
$$

In the absense of thermal-fluids feedback, the cross sections are fixed, and the equation is much simpler. This was the form implemented for neutronics only solutions given in Section 2 of Chapter 6.

4.2.2 Newton's Method Thermal-Fluids Equations

In the conventional O.S. solve, the solution is governed by three field equations. As noted above, the dependence of Relynold's number on the velocity requires that the Jacobian include these equations. This augments the original three equations with three more velocity equations. In this sense, the thermal-fluids are better represented by the momentum and continuity equations.

The solid energy equation derivation begins with the form given above. The time dependence is ignored.

$$
\begin{aligned}
&\frac{\partial}{\partial t}\left[(1-\varepsilon)\rho_s c_{p_s} T_s\right]\Delta V\\
&\qquad= D_e T_{s,E}+D_w T_{s,W}+D_n T_{s,N}+D_s T_{s,S}+D_t T_{s,T}+D_b T_{s,B}\\
&\qquad-(D_e+D_w+D_n+D_s+D_t+D_b)T_{s,P}-\alpha(T_{s,P}-T_{f,P})\Delta V+Q\Delta V
\end{aligned}
\tag{4.6}
$$

Inserting expansion for the thermal-fluid coefficients and primary variables. Including the heat generation expansion as well.

$$
\begin{aligned}
&+ K_E\left(T_{s,P}^{(k)}-T_{s,E}^{(k)}\right)\left(\frac{\partial k_{s,E}}{\partial T_{s,E}}\right)^{n-1}\delta T_{s,E} + K_W\left(T_{s,P}^{(k)}-T_{s,W}^{(k)}\right)\left(\frac{\partial k_{s,W}}{\partial T_{s,W}}\right)^{n-1}\delta T_{s,W}\\
&+ K_N\left(T_{s,P}^{(k)}-T_{s,N}^{(k)}\right)\left(\frac{\partial k_{s,N}}{\partial T_{s,N}}\right)^{n-1}\delta T_{s,N} + K_S\left(T_{s,P}^{(k)}-T_{s,S}^{(k)}\right)\left(\frac{\partial k_{s,S}}{\partial T_{s,S}}\right)^{n-1}\delta T_{s,S}\\
&+ K_T\left(T_{s,P}^{(k)}-T_{s,T}^{(k)}\right)\left(\frac{\partial k_{s,T}}{\partial T_{s,T}}\right)^{n-1}\delta T_{s,T} + K_B\left(T_{s,P}^{(k)}-T_{s,B}^{(k)}\right)\left(\frac{\partial k_{s,B}}{\partial T_{s,B}}\right)^{n-1}\delta T_{s,B}\\
&+ \Bigg(K_E\left(T_{s,P}^{(k)}-T_{s,E}^{(k)}\right)\left(\frac{\partial k_{s,E}}{\partial T_{s,P}}\right)^{n-1} + K_W\left(T_{s,P}^{(k)}-T_{s,W}^{(k)}\right)\left(\frac{\partial k_{s,W}}{\partial T_{s,P}}\right)^{n-1}\\
&+ K_N\left(T_{s,P}^{(k)}-T_{s,N}^{(k)}\right)\left(\frac{\partial k_{s,N}}{\partial T_{s,P}}\right)^{n-1} + K_S\left(T_{s,P}^{(k)}-T_{s,S}^{(k)}\right)\left(\frac{\partial k_{s,S}}{\partial T_{s,P}}\right)^{n-1}\\
&+K_T\left(T_{s,P}^{(k)}-T_{s,T}^{(k)}\right)\left(\frac{\partial k_{s,T}}{\partial T_{s,P}}\right)^{n-1} + K_B\left(T_{s,P}^{(k)}-T_{s,B}^{(k)}\right)\left(\frac{\partial k_{s,B}}{\partial T_{s,P}}\right)^{n-1}\Bigg)\delta T_{s,P}\\
&+ \Delta V\left(T_{s,P}^{(k)}-T_{f,P}^{(k)}\right)\left(\left(\frac{\partial \alpha_P}{\partial T_{s,P}}\right)^{n-1}\delta T_{s,P} + \left(\frac{\partial \alpha_P}{\partial T_{f,P}}\right)^{n-1}\delta T_{f,P} + \left(\frac{\partial \alpha_P}{\partial P_P}\right)^{n-1}\delta P_P\right)\\
&+ \Delta V\left(T_{s,P}^{(k)}-T_{f,P}^{(k)}\right)\left(\left(\frac{\partial \alpha_P}{\partial v_x}\right)^{n-1}\delta v_x + \left(\frac{\partial \alpha_P}{\partial v_y}\right)^{n-1}\delta v_y + \left(\frac{\partial \alpha_P}{\partial v_z}\right)^{n-1}\delta v_z\right)\\
&- \Delta V\bar{P}_{n,fuel}\left(\phi_1^{(k)}\delta\kappa\Sigma_1 + \phi_2^{(k)}\delta\kappa\Sigma_2 + \phi_3^{(k)}\delta\kappa\Sigma_3\right)\\
&- \left(K_E k_{s,E}^{(k)}\delta T_{s,E} + \ldots + K_B k_{s,B}^{(k)}\delta T_{s,B}\right)\\
&+ \left(K_E k_{s,E}^{(k)} + K_W k_{s,W}^{(k)} + K_N k_{s,N}^{(k)} + K_S k_{s,S}^{(k)} + K_T k_{s,T}^{(k)} + K_B k_{s,B}^{(k)}\right)\delta T_{s,P}\\
&+ \Delta V\alpha^{(k)}\delta T_{s,P} - \Delta V\alpha^{(k)}\delta T_{f,P}\\
&- \Delta V\bar{P}_{n,fuel}\left(\kappa\Sigma_1^{(k)}\delta\phi + \kappa\Sigma_2^{(k)}\delta\phi_2 + \kappa\Sigma_3^{(k)}\delta\phi_3\right) = S_k \qquad (4.7)
\end{aligned}
$$

Where the source is the residual, given as,

$$
\begin{aligned}
S^{(k)} = &\left(K_E k_{s,E}^{(k)} T_{s,E}^{(k)} + \ldots + K_B k_{s,B}^{(k)} T_{s,B}^{(k)}\right)\\
&- \left(K_E k_{s,E}^{(k)} + K_W k_{s,W}^{(k)} + K_N k_{s,N}^{(k)} + K_S k_{s,S}^{(k)} + K_T k_{s,T}^{(k)} + K_B k_{s,B}^{(k)}\right) T_{s,P}^{(k)}\\
&- \alpha^{(k)}\Delta V T_{s,P}^{(k)} + \alpha^{(k)}\Delta V T_{f,P}^{(k)}\\
&+ \Delta V\bar{P}_{n,fuel}\left(\kappa\Sigma_1^{(k)}\phi_1^{(k)} + \kappa\Sigma_2^{(k)}\phi_2^{(k)} + \kappa\Sigma_3^{(k)}\phi_3^{(k)}\right) \qquad (4.8)
\end{aligned}
$$

Expanding the cross section terms,

$$
\begin{aligned}
&+ K_E \left(T_{s,P}^{(k)} - T_{s,E}^{(k)}\right) \left(\frac{\partial k_{s,E}}{\partial T_{s,E}}\right)^{n-1} \delta T_{s,E} + K_W \left(T_{s,P}^{(k)} - T_{s,W}^{(k)}\right) \left(\frac{\partial k_{s,W}}{\partial T_{s,W}}\right)^{n-1} \delta T_{s,W} \\
&+ K_N \left(T_{s,P}^{(k)} - T_{s,N}^{(k)}\right) \left(\frac{\partial k_{s,N}}{\partial T_{s,N}}\right)^{n-1} \delta T_{s,N} + K_S \left(T_{s,P}^{(k)} - T_{s,S}^{(k)}\right) \left(\frac{\partial k_{s,S}}{\partial T_{s,S}}\right)^{n-1} \delta T_{s,S} \\
&+ K_T \left(T_{s,P}^{(k)} - T_{s,T}^{(k)}\right) \left(\frac{\partial k_{s,T}}{\partial T_{s,T}}\right)^{n-1} \delta T_{s,T} + K_B \left(T_{s,P}^{(k)} - T_{s,B}^{(k)}\right) \left(\frac{\partial k_{s,B}}{\partial T_{s,B}}\right)^{n-1} \delta T_{s,B} \\
&+ \Bigg(K_E \left(T_{s,P}^{(k)} - T_{s,E}^{(k)}\right) \left(\frac{\partial k_{s,E}}{\partial T_{s,P}}\right)^{n-1} + K_W \left(T_{s,P}^{(k)} - T_{s,W}^{(k)}\right) \left(\frac{\partial k_{s,W}}{\partial T_{s,P}}\right)^{n-1} \\
&+ K_N \left(T_{s,P}^{(k)} - T_{s,N}^{(k)}\right) \left(\frac{\partial k_{s,N}}{\partial T_{s,P}}\right)^{n-1} + K_S \left(T_{s,P}^{(k)} - T_{s,S}^{(k)}\right) \left(\frac{\partial k_{s,S}}{\partial T_{s,P}}\right)^{n-1} \\
&+ K_T \left(T_{s,P}^{(k)} - T_{s,T}^{(k)}\right) \left(\frac{\partial k_{s,T}}{\partial T_{s,P}}\right)^{n-1} + K_B \left(T_{s,P}^{(k)} - T_{s,B}^{(k)}\right) \left(\frac{\partial k_{s,B}}{\partial T_{s,P}}\right)^{n-1} \Bigg) \delta T_{s,P} \\
&+ \Delta V \left(T_{s,P}^{(k)} - T_{f,P}^{(k)}\right) \left(\left(\frac{\partial \alpha_P}{\partial T_{s,P}}\right)^{n-1} \delta T_{s,P} + \left(\frac{\partial \alpha_P}{\partial T_{f,P}}\right)^{n-1} \delta T_{f,P} + \left(\frac{\partial \alpha_P}{\partial P_P}\right)^{n-1} \delta P_P \right) \\
&+ \Delta V \left(T_{s,P}^{(k)} - T_{f,P}^{(k)}\right) \left(\left(\frac{\partial \alpha_P}{\partial v_x}\right)^{n-1} \delta v_x + \left(\frac{\partial \alpha_P}{\partial v_y}\right)^{n-1} \delta v_y + \left(\frac{\partial \alpha_P}{\partial v_z}\right)^{n-1} \delta v_z \right) \\
&- \Delta V \bar{P}_{n,fuel} \left(\phi_1^{(k)} \left(\frac{\partial \kappa\Sigma_1}{\partial T_D}\right)^{n-1} + \phi_2^{(k)} \left(\frac{\partial \kappa\Sigma_2}{\partial T_D}\right)^{n-1} + \phi_3^{(k)} \left(\frac{\partial \kappa\Sigma_3}{\partial T_D}\right)^{n-1} \right) \delta T_{s,D} \\
&- \Delta V \bar{P}_{n,fuel} \left(\phi_1^{(k)} \left(\frac{\partial \kappa\Sigma_1}{\partial T_M}\right)^{n-1} + \phi_2^{(k)} \left(\frac{\partial \kappa\Sigma_2}{\partial T_M}\right)^{n-1} + \phi_3^{(k)} \left(\frac{\partial \kappa\Sigma_3}{\partial T_M}\right)^{n-1} \right) \delta T_{s,M} \\
&- \left(K_E k_{s,E}^{(k)} \delta T_{s,E} + \ldots + K_B k_{s,B}^{(k)} \delta T_{s,B} \right) \\
&+ \left(K_E k_{s,E}^{(k)} + K_W k_{s,W}^{(k)} + K_N k_{s,N}^{(k)} + K_S k_{s,S}^{(k)} + K_T k_{s,T}^{(k)} + K_B k_{s,B}^{(k)} \right) \delta T_{s,P} \\
&+ \Delta V \alpha^{(k)} \delta T_{s,P} - \Delta V \alpha^{(k)} \delta T_{f,P} \\
&- \Delta V \bar{P}_{n,fuel} \left(\kappa\Sigma_1^{(k)} \delta\phi + \kappa\Sigma_2^{(k)} \delta\phi_2 + \kappa\Sigma_3^{(k)} \delta\phi_3 + \kappa\Sigma_4^{(k)} \delta\phi_4 \right) = S_k \quad (4.9)
\end{aligned}
$$

The heat generation can be expanded in terms of the flux and kappa fission cross section, shown to depend on the cross section temperature

$$
\begin{gathered}
\kappa\Sigma = f\left(T_D, T_M\right) \\
\delta\kappa\Sigma = \left(\left(\tfrac{\partial \kappa\Sigma}{\partial T_D}\right)^{n-1} \delta T_S + \left(\tfrac{\partial \kappa\Sigma}{\partial T_M}\right)^{n-1} \delta T_M \right)
\end{gathered} \quad (4.10)
$$

After some algebra and collecting of terms, the final form is given, with some shorthand for brevity.

$$
\begin{aligned}
&+ K_E \left[\left(T_{s,P}^{(k)} - T_{s,E}^{(k)} \right) \left(\frac{\partial k_{s,E}}{\partial T_{s,E}} \right)^{n-1} - k_{s,E}^{(k)} \right] \delta T_{s,E} \\
&\qquad\qquad\qquad\qquad + \ldots + \\
&K_B \left[\left(T_{s,P}^{(k)} - T_{s,B}^{(k)} \right) \left(\frac{\partial k_{s,B}}{\partial T_{s,B}} \right)^{n-1} - k_{s,B}^{(k)} \right] \delta T_{s,B} \\
&+ \Bigg[K_E \left[\left(T_{s,P}^{(k)} - T_{s,E}^{(k)} \right) \left(\frac{\partial k_{s,E}}{\partial T_{s,P}} \right)^{n-1} + k_{s,E}^{(k)} \right] + K_W \left[\left(T_{s,P}^{(k)} - T_{s,W}^{(k)} \right) \left(\frac{\partial k_{s,W}}{\partial T_{s,P}} \right)^{n-1} + k_{s,W}^{(k)} \right] \\
&+ K_N \left[\left(T_{s,P}^{(k)} - T_{s,N}^{(k)} \right) \left(\frac{\partial k_{s,N}}{\partial T_{s,P}} \right)^{n-1} + k_{s,N}^{(k)} \right] + K_S \left[\left(T_{s,P}^{(k)} - T_{s,S}^{(k)} \right) \left(\frac{\partial k_{s,S}}{\partial T_{s,P}} \right)^{n-1} + k_{s,S}^{(k)} \right] \\
&+ K_T \left[\left(T_{s,P}^{(k)} - T_{s,T}^{(k)} \right) \left(\frac{\partial k_{s,T}}{\partial T_{s,P}} \right)^{n-1} + k_{s,T}^{(k)} \right] + K_B \left[\left(T_{s,P}^{(k)} - T_{s,B}^{(k)} \right) \left(\frac{\partial k_{s,B}}{\partial T_{s,P}} \right)^{n-1} + k_{s,B}^{(k)} \right] \\
&+ \Delta V \left(\left(T_{s,P}^{(k)} - T_{f,P}^{(k)} \right) \left(\frac{\partial \alpha_P}{\partial T_{s,P}} \right)^{n-1} + \alpha^{(k)} \right) \Bigg] \delta T_{s,P} \\
&+ \Delta V \left(T_{s,P}^{(k)} - T_{f,P}^{(k)} \right) \left(\left(\frac{\partial \alpha}{\partial v_x} \right)^{n-1} \delta v_x + \left(\frac{\partial \alpha}{\partial v_y} \right)^{n-1} \delta v_y + \left(\frac{\partial \alpha}{\partial v_z} \right)^{n-1} \delta v_z \right) \\
&+ \Delta V \left[\left(T_{s,P}^{(k)} - T_{f,P}^{(k)} \right) \left(\frac{\partial \alpha_P}{\partial T_{f,P}} \right)^{n-1} - \alpha^{(k)} \right] \delta T_{f,P} + \Delta V \left[\left(T_{s,P}^{(k)} - T_{f,P}^{(k)} \right) \left(\frac{\partial \alpha_P}{\partial P_P} \right)^{n-1} \right] \delta P_P \\
&- \Delta V \bar{P}_{n,fuel} \left(\left[\kappa \Sigma_1^{(k)} \right] \delta \phi_1 - \left[\kappa \Sigma_2^{(k)} \right] \delta \phi_2 - \left[\kappa \Sigma_3^{(k)} \right] \delta \phi_3 \right) \\
&- \Delta V \bar{P}_{n,fuel} \left(\phi_1^{(k)} \left(\frac{\partial \kappa \Sigma_1}{\partial T_D} \right)^{n-1} + \phi_2^{(k)} \left(\frac{\partial \kappa \Sigma_2}{\partial T_D} \right)^{n-1} + \phi_3^{(k)} \left(\frac{\partial \kappa \Sigma_3}{\partial T_D} \right)^{n-1} \right) \delta T_{s,D} \\
&- \Delta V \bar{P}_{n,fuel} \left(\phi_1^{(k)} \left(\frac{\partial \kappa \Sigma_1}{\partial T_M} \right)^{n-1} + \phi_2^{(k)} \left(\frac{\partial \kappa \Sigma_2}{\partial T_M} \right)^{n-1} + \phi_3^{(k)} \left(\frac{\partial \kappa \Sigma_3}{\partial T_M} \right)^{n-1} \right) \delta T_{s,M} = S_k
\end{aligned}
\tag{4.11}
$$

The final equation depends on the neighbor solid temperatures and fluid temperature as before. However, the heat transfer coefficient derivatives increase the coupling to the other fields. Also, the flux and kappa fission terms tightly couple the heat generation. Because there is no heat assumed to be generated in the fluid, this is the primary coupling between the flux and thermal-fluids.

The fluid energy equation is actually the more complex field equation. The presence of the conduction, convection, and heat transfer terms, increases the number of terms and complexity of the equation. As noted above, only upwind differencing was considered, but further analysis could consider higher order schemes. Beginning with the conventional derivation,

$$\begin{aligned}\frac{\partial}{\partial t}\left[\varepsilon\rho_f c_{p_f} T_f\right]\Delta V = \\ &A_E T_{f,E} + A_W T_{f,W} + A_N T_{f,N} + A_S T_{s,S} + A_T T_{f,T} + A_B T_{f,B} \\ &-\ (\mathrm{A}_E + A_W + A_N + A_S + A_T + A_B) T_{f,P} - \alpha(T_{f,P} - T_{s,P})\Delta V\end{aligned} \tag{4.12}$$

Expanding the primary variables,

$$\begin{aligned}&(A_E + A_W + A_N + A_S + A_T + A_B)\left(T_{f,P} + \delta T_{f,P}\right) \\ &\quad - A_E\left(T_{f,E} + \delta T_{f,E}\right) - A_W\left(T_{f,W} + \delta T_{f,W}\right) - A_N\left(T_{f,N} + \delta T_{f,N}\right) \\ &\quad - A_S\left(T_{s,S} + \delta T_{f,S}\right) - A_T\left(T_{f,T} + \delta T_{f,T}\right) - A_B\left(T_{f,B} + \delta T_{f,B}\right) \\ &\quad + \left(\left(T_{f,P} + \delta T_{f,P}\right) - \left(T_{s,P} + \delta T_{s,P}\right)\right)\Delta V \delta\alpha = 0\end{aligned} \tag{4.13}$$

Expansions for the coefficients,

$$\begin{aligned}&\alpha = \alpha^{(k)} + \delta\alpha \\ &F_e = G_E\left(c_{pf,E}^{(k)} + \delta c_{pf,E}\right)\left(\dot m_E^{(k)} + \delta\dot m_E\right) \qquad D_e = K_E\left(k_{f,E}^{(k)} + \delta k_{f,E}\right) \\ &F_w = G_W\left(c_{pf,W}^{(k)} + \delta c_{pf,W}\right)\left(\dot m_W^{(k)} + \delta\dot m_W\right) \qquad D_w = K_W\left(k_{f,W}^{(k)} + \delta k_{f,W}\right) \\ &F_n = G_N\left(c_{pf,N}^{(k)} + \delta c_{pf,N}\right)\left(\dot m_N^{(k)} + \delta\dot m_N\right) \qquad D_n = K_N\left(k_{f,N}^{(k)} + \delta k_{f,N}\right) \\ &F_s = G_S\left(c_{pf,S}^{(k)} + \delta c_{pf,S}\right)\left(\dot m_S^{(k)} + \delta\dot m_S\right) \qquad D_s = K_S\left(k_{f,S}^{(k)} + \delta k_{f,S}\right) \\ &F_t = G_T\left(c_{pf,T}^{(k)} + \delta c_{pf,T}\right)\left(\dot m_T^{(k)} + \delta\dot m_T\right) \qquad D_t = K_T\left(k_{f,T}^{(k)} + \delta k_{f,T}\right) \\ &F_b = G_B\left(c_{pf,B}^{(k)} + \delta c_{pf,B}\right)\left(\dot m_B^{(k)} + \delta\dot m_B\right) \qquad D_b = K_B\left(k_{f,B}^{(k)} + \delta k_{f,B}\right)\end{aligned} \tag{4.14}$$

The final form is then given as,

$$
\begin{gathered}
\left(T_{f,P}-T_{f,E}\right)K_E\left(\frac{\partial k_E}{\partial T_{f,P}}\delta T_{f,P}+\frac{\partial k_E}{\partial T_{f,E}}\delta T_{f,E}+\frac{\partial k_E}{\partial p_{f,P}}\delta p_{f,P}+\frac{\partial k_E}{\partial p_{f,E}}\delta p_{f,E}\right)\\
+\left(T_{f,P}-T_{f,E}\right)G_E\dot{m}_E\left(\frac{\partial c_{p,E}}{\partial T_{f,P}}\delta T_{f,P}+\frac{\partial c_{p,E}}{\partial T_{f,E}}\delta T_{f,E}+\frac{\partial c_{p,E}}{\partial p_{f,P}}\delta p_{f,P}+\frac{\partial c_{p,E}}{\partial p_{f,E}}\delta p_{f,E}\right)\\
+\left(T_{f,P}-T_{f,E}\right)G_E c_{p,E}\left(\frac{\partial \dot{m}_E}{\partial T_{s,P}}\delta T_{s,P}+...+\frac{\partial \dot{m}_E}{\partial v_{z,P}}\delta v_{z,P}+\frac{\partial \dot{m}_E}{\partial T_{s,E}}\delta T_{s,E}+...+\frac{\partial \dot{m}_E}{\partial v_{z,E}}\delta v_{z,E}\right)\\
+...+\\
\left(T_{f,P}-T_{f,B}\right)K_B\left(\frac{\partial k_B}{\partial T_{f,P}}\delta T_{f,P}+\frac{\partial k_B}{\partial T_{f,B}}\delta T_{f,B}+\frac{\partial k_B}{\partial p_{f,P}}\delta p_{f,P}+\frac{\partial k_B}{\partial p_{f,B}}\delta p_{f,B}\right)\\
+\left(T_{f,P}-T_{f,B}\right)G_B\dot{m}_B\left(\frac{\partial c_{p,B}}{\partial T_{f,P}}\delta T_{f,P}+\frac{\partial c_{p,B}}{\partial T_{f,B}}\delta T_{f,B}+\frac{\partial c_{p,B}}{\partial p_{f,P}}\delta p_{f,P}+\frac{\partial c_{p,B}}{\partial p_{f,B}}\delta p_{f,B}\right)\\
+\left(T_{f,P}-T_{f,B}\right)G_B c_{p,B}\left(\frac{\partial \dot{m}_B}{\partial T_{s,P}}\delta T_{s,P}+...+\frac{\partial \dot{m}_B}{\partial v_{z,P}}\delta v_{z,P}+\frac{\partial \dot{m}_B}{\partial T_{s,B}}\delta T_{s,B}+...+\frac{\partial \dot{m}_B}{\partial v_{z,B}}\delta v_{z,B}\right)\\
+\Delta V\left(T_{f,P}-T_{s,P}\right)\left(\frac{\partial \alpha}{\partial T_{s,P}}T_{s,P}+...+\frac{\partial \alpha}{\partial v_{z,P}}\delta v_{z,P}\right)\\
+A_E\left(\delta T_{f,P}-\delta T_{f,E}\right)+A_W\left(\delta T_{f,P}-\delta T_{f,W}\right)+A_N\left(\delta T_{f,P}-\delta T_{f,N}\right)\\
+A_S\left(\delta T_{f,P}-\delta T_{f,S}\right)A_T\left(\delta T_{f,P}-\delta T_{f,T}\right)+A_B\left(\delta T_{f,P}-\delta T_{f,B}\right)+\left(\delta T_{f,P}-\delta T_{s,P}\right)\Delta V\alpha=\\
-A_E\left(T_{f,P}-T_{f,E}\right)-A_W\left(T_{f,P}-T_{f,W}\right)-A_N\left(T_{f,P}-T_{f,N}\right)\\
-A_S\left(T_{f,P}-T_{f,S}\right)-A_T\left(T_{f,P}-T_{f,T}\right)-A_B\left(T_{f,P}-T_{f,B}\right)-\left(T_{f,P}-T_{s,P}\right)\Delta V\alpha
\end{gathered}
\tag{4.15}
$$

As noted earlier, the fluid energy depends on all primary variables, with all neighbors. This creates tight coupling between the fluid energy and every other equation.

The pressure equation is generally simple in form. However, the coefficients depend on every primary variable, including the velocities. The equation written in terms of pressure, is given as,

$$
\begin{aligned}
0=&\frac{-G_1}{W_{r1}^{k+1}}\Delta p_{r1}^{k+1}+\frac{-G_2}{W_{r2}^{k+1}}\Delta p_{r2}^{k+1}+\frac{-G_3}{W_{\theta3}^{k+1}}\Delta p_{\theta3}^{k+1}+\frac{-G_4}{W_{\theta4}^{k+1}}\Delta p_{\theta4}^{k+1}\\
&+\frac{-G_5}{W_{z5}^{k+1}}\left(\Delta p_{z5}^{k+1}+g\Delta z_5\rho_5^{k+1}\right)+\frac{-G_6}{W_{z6}^{k+1}}\left(\Delta p_{z6}^{k+1}+g\Delta z_6\rho_6^{k+1}\right)
\end{aligned}
\tag{4.16}
$$

where W represents the flow resisitivity in each geometric direction, respectively.

Using some coefficients for convenience,

$$\begin{array}{llll} R_{r,E} = \frac{G_1}{W_{r1}^{k+1}} & R_{r,W} = \frac{G_2}{W_{r2}^{k+1}} & R_{a,N} = \frac{G_3}{W_{\theta 3}^{k+1}} & R_{a,S} = \frac{G_4}{W_{\theta 4}^{k+1}} \\ R_{l,B} = \frac{G_5}{W_{z5}^{k+1}} & R_{l,T} = \frac{G_6}{W_{z6}^{k+1}} & b_{f,B} = g\Delta z_5 \rho_5^{k+1} & b_{f,T} = g\Delta z_6 \rho_6^{k+1} \end{array} \tag{4.17}$$

The equations are then given as,

$$\begin{aligned} & - R_{r,E}\Delta p_{r1}^{k+1} - R_{r,W}\Delta p_{r2}^{k+1} - R_{a,N}\Delta p_{\theta 3}^{k+1} - R_{r,S}\Delta p_{\theta 4}^{k+1} \\ & \qquad - R_{l,B}\left(\Delta p_{z5}^{k+1} + b_{f,B}\right) - R_{l,T}\left(\Delta p_{z6}^{k+1} + b_{f,B}\right) = 0 \end{aligned} \tag{4.18}$$

Expanding the primary variables, and coefficients,

$$\begin{aligned} & R_{r,E}\left(\delta p_{f,P} - \delta p_{f,E}\right) + R_{r,W}\left(\delta p_{f,P} - \delta p_{f,W}\right) \\ & \quad + R_{a,N}\left(\delta p_{f,P} - \delta p_{f,N}\right) + R_{r,S}\left(\delta p_{f,P} - \delta p_{f,S}\right) \\ & \quad + R_{l,B}\left(\delta p_{f,P} - \delta p_{f,B}\right) + R_{l,T}\left(\delta p_{f,P} - \delta p_{f,T}\right) \\ & \quad + \delta R_{r,E}\left(p_{f,P} - p_{f,E}\right) + \delta R_{r,W}\left(p_{f,P} - p_{f,W}\right) \\ & \quad + \delta R_{a,N}\left(p_{f,P} - p_{f,N}\right) + \delta R_{r,S}\left(p_{f,P} - p_{f,S}\right) \\ & \quad - R_{l,B}\delta b_{f,B} - R_{l,T}\delta b_{f,B} \\ & \quad + \delta R_{l,B}\left(\left(p_{f,P} - p_{f,B}\right) - b_{f,B}\right) + \delta R_{l,T}\left(\left(p_{f,P} - p_{f,T}\right) - b_{f,B}\right) \\ & \quad = -R_{r,E}\left(p_{f,P} - p_{f,E}\right) - R_{r,W}\left(p_{f,P} - p_{f,W}\right) \\ & \quad - R_{a,N}\left(p_{f,P} - p_{f,N}\right) - R_{r,S}\left(p_{f,P} - p_{f,S}\right) \\ & \quad - R_{l,B}\left(\left(p_{f,P} - p_{f,B}\right) - b_{f,B}\right) - R_{l,T}\left(\left(p_{f,P} - p_{f,T}\right) - b_{f,B}\right) \end{aligned} \tag{4.19}$$

The final form is given, with some short hand for brevity,

$$
\begin{aligned}
&- (p_{f,P} - p_{f,E}) \left[\frac{\partial R_{r,E}}{\partial T_{s,P}} \delta T_{s,p} + ... + \frac{\partial R_{r,E}}{\partial v_{z,P}} \delta v_{z,P} \right] \\
&\quad - (p_{f,P} - p_{f,E}) \left[\frac{\partial R_{r,E}}{\partial T_{s,E}} \delta T_{s,E} + ... + \frac{\partial R_{r,E}}{\partial v_{z,E}} \delta v_{z,E} \right] \\
&\quad - (p_{f,P} - p_{f,W}) \left[\frac{\partial R_{r,W}}{\partial T_{s,P}} \delta T_{s,p} + ... + \frac{\partial R_{r,W}}{\partial v_{z,P}} \delta v_{z,P} \right] \\
&\quad - (p_{f,P} - p_{f,W}) \left[\frac{\partial R_{r,W}}{\partial T_{s,W}} \delta T_{s,W} + ... + \frac{\partial R_{r,W}}{\partial v_{z,W}} \delta v_{z,W} \right] \\
&\quad + (p_{f,P} - p_{f,S}) \left[\frac{\partial R_{a,S}}{\partial T_{s,P}} \delta T_{s,P} + ... + \frac{\partial R_{a,S}}{\partial v_{z,P}} \delta v_{z,P} \right] \\
&\quad + (p_{f,P} - p_{f,S}) \left[\frac{\partial R_{a,S}}{\partial T_{s,S}} \delta T_{s,S} + ... + \frac{\partial R_{a,S}}{\partial v_{z,S}} \delta v_{z,S} \right] \\
&\quad + (p_{f,P} - p_{f,N}) \left[\frac{\partial R_{a,N}}{\partial T_{s,P}} \delta T_{s,P} + ... + \frac{\partial R_{a,N}}{\partial v_{z,P}} \delta v_{z,P} \right] \\
&\quad + (p_{f,P} - p_{f,N}) \left[\frac{\partial R_{a,N}}{\partial T_{s,N}} \delta T_{s,N} + ... + \frac{\partial R_{a,N}}{\partial v_{z,N}} \delta v_{z,N} \right] \\
&\quad + (p_{f,P} - p_{f,T} - b_{f,T}) \left[\frac{\partial R_{l,T}}{\partial T_{s,T}} \delta T_{s,T} + ... + \frac{\partial R_{l,T}}{\partial v_{z,T}} \delta v_{z,T} \right] \\
&\quad + (p_{f,P} - p_{f,T} - b_{f,T}) \left[\frac{\partial R_{l,T}}{\partial T_{s,P}} \delta T_{s,P} + ... + \frac{\partial R_{l,T}}{\partial v_{z,P}} \delta v_{z,P} \right] \\
&\quad + (p_{f,P} - p_{f,B} - b_{f,B}) \left[\frac{\partial R_{l,B}}{\partial T_{s,B}} \delta T_{s,B} + ... + \frac{\partial R_{l,B}}{\partial v_{z,B}} \delta v_{z,B} \right] \\
&\quad + (p_{f,P} - p_{f,B} - b_{f,B}) \left[\frac{\partial R_{l,B}}{\partial T_{s,P}} \delta T_{s,P} + ... + \frac{\partial R_{l,B}}{\partial v_{z,P}} \delta v_{z,P} \right] \\
&- R_{l,T} \left[\frac{\partial b_{f,T}}{\partial T_{s,T}} \delta T_{s,T} + ... + \frac{\partial b_{f,T}}{\partial v_{z,T}} \delta v_{z,T} \right] - R_{l,T} \left[\frac{\partial b_{f,T}}{\partial T_{s,P}} \delta T_{s,P} + ... + \frac{\partial b_{f,T}}{\partial v_{z,P}} \delta v_{z,P} \right] \\
&- R_{l,B} \left[\frac{\partial b_{f,B}}{\partial T_{s,B}} \delta T_{s,B} + ... + \frac{\partial b_{f,B}}{\partial v_{z,B}} \delta v_{z,B} \right] - R_{l,B} \left[\frac{\partial b_{f,B}}{\partial T_{s,P}} \delta T_{s,P} + ... + \frac{\partial b_{f,B}}{\partial v_{z,P}} \delta v_{z,P} \right] \\
&\quad R_{r,E} (\delta p_{f,P} - \delta p_{f,E}) + R_{r,W} (\delta p_{f,P} - \delta p_{f,W}) \\
&\quad + R_{a,N} (\delta p_{f,P} - \delta p_{f,N}) + R_{r,S} (\delta p_{f,P} - \delta p_{f,S}) \\
&\quad + R_{l,B} (\delta p_{f,P} - \delta p_{f,B}) + R_{l,T} (\delta p_{f,P} - \delta p_{f,T}) \\
&\quad = R_{r,E} (p_{f,P} - p_{f,E}) + R_{r,W} (p_{f,P} - p_{f,W}) \\
&\quad + R_{a,N} (p_{f,P} - p_{f,N}) + R_{r,S} (p_{f,P} - p_{f,S}) \\
&\quad + R_{l,B} ((p_{f,P} - p_{f,B}) - b_{f,B}) + R_{l,T} ((p_{f,P} - p_{f,T}) - b_{f,B})
\end{aligned}
\tag{4.20}
$$

The pressure equation now depends on all primary thermal-fluid variables. This more tightly couples the equation to the other fields, which is drastically different from the convetional implementation. The velocity equations are simple compared to the above field equations, and therefore, the derivations are provided in the appendicies only.

4.2.3 Newton's Method Cross Section Feedback Equations

The cross section feedback for the PBMR is governed by three field equatiosn. The primary equation is a special 1-D conduction solve which is appled to an average power pebble within each material mesh. Along with the 1-D conduction and heat conduction to the fluid, heat transfer through the pebble is included as well. Beginning with the 1-D conduction equation in steady state,

$$-\frac{\partial}{\partial x}\left[kA\frac{\partial T}{\partial x}\right] = Q \tag{4.21}$$

Integrating over space, the finite difference form is given as,

$$k_{in}A_{in}\left(T_{sur}-T_{in}\right)+k_{out}A_{out}\left(T_{sur}-T_{out}\right)=q_{in}^{'''}V_{in}+q_{out}^{'''}V_{out} \tag{4.22}$$

Expanding all of the coefficients, heat generation terms, and primary variables,

$$\begin{aligned}
A_{in}\left(T_{sur}-T_{in}\right)&\left(\frac{\partial k_{in}}{\partial T_{sur}}\delta T_{sur}+\frac{\partial k_{in}}{\partial T_{in}}\delta T_{in}\right)\\
&+A_{out}\left(T_{sur}-T_{out}\right)\left(\frac{\partial k_{out}}{\partial T_{sur}}\delta T_{sur}+\frac{\partial k_{out}}{\partial T_{out}}\delta T_{out}\right)\\
-\left(f_{in}V_{in}+f_{out}V_{out}\right)\bar{P}_{core}&\left(\left(\sum_{g=1}^{ngroup}\phi_g\frac{\partial\kappa\Sigma_f}{\partial T_m}\right)\delta T_m+\left(\sum_{g=1}^{ngroup}\phi_g\frac{\partial\kappa\Sigma_f}{\partial T_d}\right)\delta T_d\right)\\
&+k_{in}A_{in}\left(\delta T_{sur}-\delta T_{in}\right)+k_{out}A_{out}\left(\delta T_{sur}-\delta T_{out}\right)\\
=\left(f_{in}V_{in}+f_{out}V_{out}\right)q_{peb}^{'''}&-k_{in}A_{in}\left(T_{sur}-T_{in}\right)+k_{out}A_{out}\left(T_{sur}-T_{out}\right)
\end{aligned}\tag{4.23}$$

Keeping only the first order terms, the final form is given as

$$
\begin{aligned}
&A_{in}\left(T_{sur}-T_{in}\right)\left(\frac{\partial k_{in}}{\partial T_{sur}}\delta T_{sur}+\frac{\partial k_{in}}{\partial T_{in}}\delta T_{in}\right)\\
&+\alpha A_{peb}\left(\delta T_{sur}-\delta T_{flu}\right)+A_{peb}\left(\delta T_{sur}-\delta T_{flu}\right)\left(\frac{\partial \alpha}{\partial T_{sur}}\delta T_{sur}+\frac{\partial \alpha}{\partial T_{sol}}\delta T_{sol}+\frac{\partial \alpha}{\partial T_{flu}}\delta T_{flu}\right)\\
&+k_{eq,W}\left(\delta T_{sur}-\delta T_{sol,W}\right)+\left(T_{sur}-T_{sol,W}\right)\left(\frac{\partial k_{eq,W}}{\partial T_{sol,W}}\delta T_{sol,W}+\frac{\partial k_{eq,W}}{\partial T_{sol,P}}\delta T_{sol,P}\right)\\
&+k_{eq,E}\left(\delta T_{sur}-\delta T_{sol,E}\right)+\left(T_{sur}-T_{sol,E}\right)\left(\frac{\partial k_{eq,E}}{\partial T_{sol,E}}\delta T_{sol,E}+\frac{\partial k_{eq,E}}{\partial T_{sol,P}}\delta T_{sol,P}\right)\\
&+k_{eq,N}\left(\delta T_{sur}-\delta T_{sol,N}\right)+\left(T_{sur}-T_{sol,N}\right)\left(\frac{\partial k_{eq,N}}{\partial T_{sol,N}}\delta T_{sol,E}+\frac{\partial k_{eq,N}}{\partial T_{sol,P}}\delta T_{sol,P}\right)\\
&+k_{eq,S}\left(\delta T_{sur}-\delta T_{sol,S}\right)+\left(T_{sur}-T_{sol,S}\right)\left(\frac{\partial k_{eq,S}}{\partial T_{sol,S}}\delta T_{sol,E}+\frac{\partial k_{eq,S}}{\partial T_{sol,P}}\delta T_{sol,P}\right)\\
&+k_{eq,B}\left(\delta T_{sur}-\delta T_{sol,B}\right)+\left(T_{sur}-T_{sol,B}\right)\left(\frac{\partial k_{eq,B}}{\partial T_{sol,B}}\delta T_{sol,E}+\frac{\partial k_{eq,B}}{\partial T_{sol,P}}\delta T_{sol,P}\right)\\
&+k_{eq,T}\left(\delta T_{sur}-\delta T_{sol,T}\right)+\left(T_{sur}-T_{sol,T}\right)\left(\frac{\partial k_{eq,T}}{\partial T_{sol,T}}\delta T_{sol,T}+\frac{\partial k_{eq,T}}{\partial T_{sol,P}}\delta T_{sol,P}\right)\\
&=q_{in}^{'''}V_{in}-k_{in}A_{in}\left(T_{sur}-T_{in}\right)-\alpha A_{peb}\left(T_{sur}-T_{flu}\right)\\
&-k_{eq,W}\left(T_{sur}-T_{sol,W}\right)-k_{eq,E}\left(T_{sur}-T_{sol,E}\right)\\
&-k_{eq,N}\left(T_{sur}-T_{sol,N}\right)-k_{eq,S}\left(T_{sur}-T_{sol,S}\right)\\
&-k_{eq,B}\left(T_{sur}-T_{sol,B}\right)-k_{eq,T}\left(T_{sur}-T_{sol,T}\right)
\end{aligned}
\tag{4.24}
$$

As can be seen, the solution depends on the fluid temperature and all solid temperature neighbors. The results of this calculation are used to determine the cross sections temperatures for each node.

The next cross sections temperature equation is the Moderator temperaure. This is actually a simple volume weighting of the 1-D conduction solution given above. The inclusion of this equation results from a need to reduce the spatial complexity of the cross section dependence. The fundamental equation is given as,

$$V_{peb}T_m=\sum_{sh=1}^{nshell}V_s\frac{T_{sh}+T_{sh+1}}{2}\tag{4.25}$$

Expanding the 1-D condution temperature and Moderator temperature, the final form is given as,

$$V_{peb}\delta T_m - \sum_{sh=1}^{nshell} V_s \frac{\delta T_{sh} + \delta T_{sh+1}}{2} = \left(\sum_{sh=1}^{nshell} V_s \frac{T_{sh} + T_{sh+1}}{2} \right) - (V_{peb} T_m) \tag{4.26}$$

The final equation for cross section feedback is the Doppler temperature calculation. This equation is similar to the Moderator temperature calculation, with an added empirical heat generation term, $f - pueb$, used to approximate the peaking within the triso particles. The basic equation is given as,

$$V_{fuel} T_d = \sum_{sh=1}^{nshell(fuel)} \left(V_s \frac{T_{sh} + T_{sh+1}}{2} + V_s f_{pueb} Q_{peb} \right) \tag{4.27}$$

Expanding the temperature and heat generation term, the expanded form is given as,

$$\begin{aligned} V_{fuel}\delta T_d - \sum_{sh=1}^{nshell(fuel)} V_s \frac{\delta T_{sh} + \delta T_{sh+1}}{2} - \sum_{sh=1}^{nshell(fuel)} V_s f_{pueb} \delta Q_{peb} \\ = \left(\sum_{sh=1}^{nshell(fuel)} \left(V_s \frac{T_{sh} + T_{sh+1}}{2} + V_s f_{pueb} Q_{peb} \right) \right) - (V_{fuel} T_d) \end{aligned} \tag{4.28}$$

Including the cross section dependence, the final form is given as,

$$
\begin{aligned}
V_{fuel}\delta T_d - \sum_{sh=1}^{nshell(fuel)} V_s \frac{\delta T_{sh} + \delta T_{sh+1}}{2} \\
- \left(\sum_{sh=1}^{nshell(fuel)} V_s f_{pueb}\right) \sum_{1}^{ngrp} \frac{\partial Q_{peb}}{\partial T_m} \delta T_m \\
- \left(\sum_{sh=1}^{nshell(fuel)} V_s f_{pueb}\right) \sum_{1}^{ngrp} \frac{\partial Q_{peb}}{\partial T_D} \delta T_D \\
- \left(\sum_{sh=1}^{nshell(fuel)} V_s f_{pueb}\right) + V_{fuel} \overline{P}_{n,fuel} \sum_{1}^{ngrp} \kappa \Sigma_g \delta\phi_g \\
= \left(\sum_{sh=1}^{nshell(fuel)} \left(V_s \frac{T_{sh} + T_{sh+1}}{2} + V_s f_{pueb} Q_{peb}\right)\right) - (V_{fuel} T_d)
\end{aligned}
\tag{4.29}
$$

The Doppler temperature depends on the 1-D condution solve, the neutron flux, Moderator temperature, and the Doppler temperature.

4.3 Solution of Newton's Methods

As noted in Chapter 3, the Jacobian was solved in this research using both exact and inexact methods. The exact solution provided a reference solution whereas practical solutions of the Jacobian are typically performed using linear solver libraries similar to the GMRES [82] [83] [84] which was used here from the INTEL-MKL library. A routine was developed to convert the different parts of the Jacobian into the required format for the MKL library but could easily be extended to include the formats needed for other libraries.

4.3.1 Solution Methods

The direct solver used in the exact Newton method was PARDISO which is a state-of-the-art direct solver used throughout the scientific community. In PARDISO the standard Gaussian elimination is replaced by an LU decomposition produced using pivoting and iterative refinement. In this way, PARDISO attempts to minimize the truncation and roundoff error produced during a standard matrix inversion. The resulting solution is a very reliable and accurate exact solution. Several of the advanced features will be disabled to allow a clearer comparison to the iterative method.

The iterative solver used in the inexact Newton method was the GMRES solver from the INTEL-MKL library. The solver is composed of three main routines. The first generates the orthogonal basis for the given vector. The second routine is used to perform the Jv matrix-vector operation, and lastly, a third routine performs the least squares problem to obtain the solution for the given Newton step. The orthogonal basis is constructed using the Arnoldi process with Householder transformations. An incomplete LU factorization is also available to precondition the GMRES iterations. In the work here the restarted GMRES solution was also tested since the memory requirements can become considerable for the large coupled problems. This was found to be somewhat successful for the thermal fluids problem, but not for the coupled problem. Along with a fixed convergence strategy, a dynamic strategy was implemented as well allow the inner iterations to converge relative to the Newton residual calculated in the previous iteration. In this way, the GMRES inners would not converge tightly to an inaccurate solution early in the nonlinear iterations.

4.3.2 Convergence Testing

There are several possible methods to evaluate the convergence of the Newton iterations. The first method typically used is to evaluate the magnitude of the shift of each solution variable in successive iterations. This can evaluate convergence within the Newton iteration itself, but does not necessarily indicate the convergence of the overall non-linear system. For a system with constant coefficients this is typically sufficient, but a higher order estimate of the error is required for systems with non-constant coefficients which is the more practical case.

The second approach typically used to evaluate convergence is to compare the magnitude of the nonlinear residuals after coefficient updates. This provides a more accurate estimate of the error for the case here in which the coefficients are not constant. A variation on this approach is to evaluate the change in the coefficients together with a test of the primary system variables. This can include either or both the standard coefficients or the partial derivatives. The goal is to therefore drive the residuals to zero, or a small as possible with machine percision.

Chapter 5
Models Used in Analysis

This chapter describes the problems used to test and evaluate the Newton's method developed in this work. The problems are based on the OECD PBMR-400 Benchmark problem which was developed to assess the performance of coupled codes for the steady-state and transient analysis of the Pebble Bed High Temperature Gas Reactor. The following section will introduce the problem specifications and the subsequent sections will provide the detailed coupled code solution. All results presented in this section are the validation basis for the Newton's Method calculations, and were taken from Seker [85].

5.1 Problem Specifications

The development of the PBMR concept began in South Africa in 1993. The Pebble Bed Modular Reactor (Pty) Ltd Company was established in 1999 to complete the design study and carry out the construction of the first PBMR module. The PBMR is a pebble bed type high temperature gas reactor with a direct cycle gas turbine power conversion system which achieves a thermodynamic efficiency of about 42 %. The reactor operates at a thermal power of 400MWt with inlet and outlet temperatures of 500 °C and 900 °C, respectively. The major design and operational characteristics are summarized in Table 5.1.

The reactor has an annular core with an outer diameter of 3.7 meters and an effective core height of 11 meters. The solid graphite reflector with a diameter of 2 meters is located in the center of the core and a graphite side reflector with a thickness of 90 cm surrounds the fuel region. The layout of the reactor unit is shown in Figure 5.1.

The helium gas enters the reactor unit from the inlet plenum, flows upwards in the helium flow channels which are located in the side reflector and downwards through

Table 5.1: PBMR Characteristics

PBMR Characteristic	Value
Thermal Power	400 MW
Electric Power	165 MW
Capacity Factor	$\geq$ 95 %
Core Configuration	Vertical with Inner Reflector
Fuel	TRISO Coated Graphite Spehers
Primary Coolant	Helios
Reactor Pressure	9 MPa
Moderator	Graphite
Core Outlet Temp	900 °C
Core Inlet Temp	500 °C
Cycle Type	Direct
Cycle Efficiency	$\sim$ 42 %

the pebble bed before leaving the reactor from the outlet plenum. The reactor control system consists of 24 control rod positions in the side reflector. 12 control rods operate at the upper part of the core serving as control rods while the other 12 rods operate at the lower part of the core as shut down rods.

The fuel is loaded online into the core from the three positions located above the core and removed from the three defueling chutes located below the core and positioned equidistant to the centre of the fuel annulus. The core contains approximately 452000 fuel pebbles with a packing fraction of 0.61. The fuel pebble has a diameter of 6 cm. The inner 5 cm of the fuel contains about 15 million UO_2 TRISO particles embedded in a graphite matrix and is surrounded by an outer graphite shell with a thickness of 0.5 cm. The form of the fuel pebble is represented Figure 5.2. Each fuel pebble contains 9 grams of Uranium with a U-235 enrichment of 9.6 wt %.

The PBMR design is primarily based on the German AVR and THTR reactors. The tools and methods to perform the design and safety analyses of the reactor lagged behind the state of the art compared to other reactor technologies. This has motivated the testing of existing methods for pebble bed reactor concept. The first attempt for this verification and validation effort was the benchmark problem defined for the PBMR 268MW reactor design. A test case for the PBMR 400MW design was also defined and accepted as an international benchmark problem by the OECD/NEA/NSC. The main objective of the benchmark is to establish a well defined problem based on a common set of cross-sections to compare the methods and tools in the core simulation and thermal hydraulic analysis.

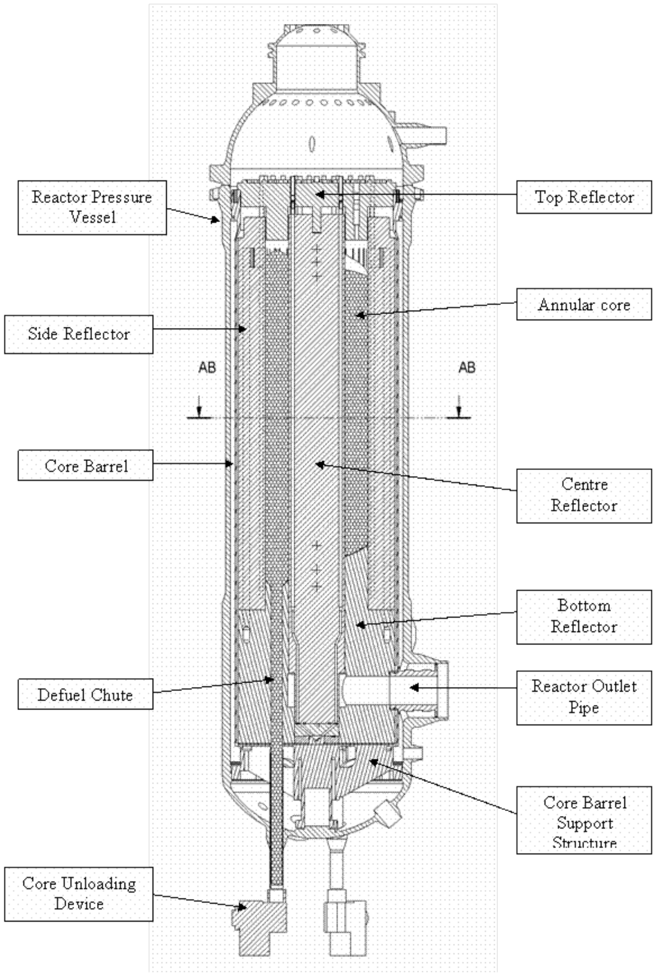

Figure 5.1: PBMR 400 Reactor

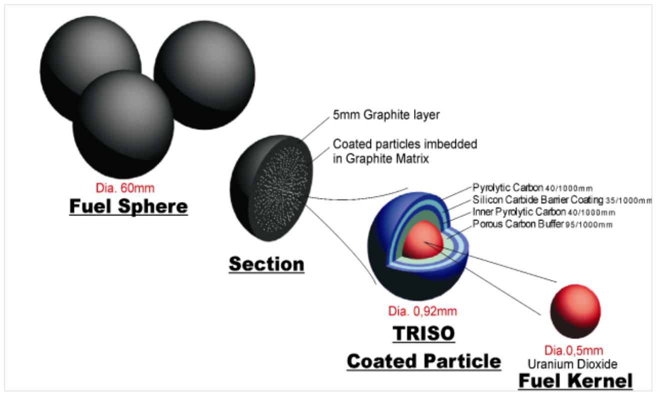
Dia. 60mm
Fuel Sphere
5mm Graphite layer
Coated particles imbedded in Graphite Matrix
Section
Pyrolytic Carbon 40/1000mm
Silicon Carbide Barrier Coating 35/1000mm
Inner Pyrolytic Carbon 40/1000mm
Porous Carbon Buffer 95/1000mm
Dia. 0,92mm
TRISO Coated Particle
Dia.0,5mm
Uranium Dioxide
Fuel Kernel

Figure 5.2: TRISO Fuel Pebble

The reference design for this benchmark is derived from the 400MW PBMR shown in Figure 5.1. The reactor core is modeled in two-dimensions (r,z) with some simplifications such as flattening the pebble beds upper surface and removal of the bottom cones and the de-fuel channels which results in a flat bottom reflector, which simplified the input model. The effect of this simplification was not quantified, but would be neccesary in futher validation efforts. The pebble flow is simplified to be in parallel channels and at equal speed. The control rods in the side reflector are modeled with a given B-10 concentration as a cylindrical skirt. In order to analyze some specific cases such as a single rod or a segment of rods ejections, a three-dimensional (r,θ,z) model is also employed. In the thermal-fluids design, the stagnant helium and air is specified between the core barrel and RPV and RPV and heat sink, respectively. The details of the benchmark problem such as the core geometry, material properties, fuel and structural material specifications and etc. are given in the official benchmark description document [86].

The core layout of the PBMR-400 is shown in Figure 5.3. A set of cross-sections in two energy groups and five state parameters dependent is provided as a part of benchmark specification. The cross sections are dependent on fuel temperature, moderator temperature, xenon concentration, fast and thermal bucklings. The 5-D interpolation method implemented in PARCS was used to update the cross section data.

5.2 Validation Basis

This section will summarize the AGREE-PARCS results of the steady-state cases and compare the results with those of other benchmark participants. These results were previously completed, and are not Newton-Krylov methods results. There are three steady-state cases that were performed which include

- Case S-1: Neutronics Solution with Fixed Cross Sections
- Case S-2: Thermal Fluid solution with given power / heat sources
- Case S-3: Combined neutronics thermal fluids calculation starting condition for the transients

The first three cases were performed to verify the standalone neutronics, standalone thermal-fluids and coupled neutronics/thermal-fluids. The comparison of the eigenvalue, maximum power density and axial/radial thermal flux profiles for the Case S-1 are shown in Figure 5.4 through Figure 5.7.

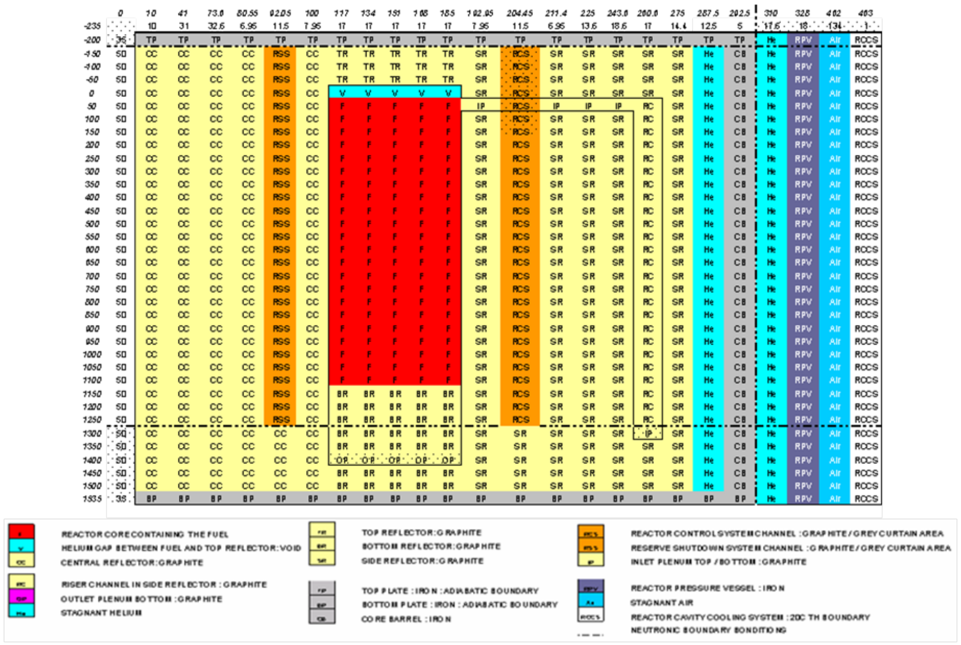
REACTOR CORE CONTAINING THE FUEL
HELIUM GAP BETWEEN FUEL AND TOP REFLECTOR: VOID
CENTRAL REFLECTOR: GRAPHITE
RISER CHANNEL IN SIDE REFLECTOR : GRAPHITE
OUTLET PLENUM BOTTOM : GRAPHITE
STAGNANT HELIUM
TOP REFLECTOR: GRAPHITE
BOTTOM REFLECTOR: GRAPHITE
SIDE REFLECTOR: GRAPHITE
TOP PLATE : IRON : ADIABATIC BOUNDARY
BOTTOM PLATE : IRON : ADIABATIC BOUNDARY
CORE BARREL : IRON
REACTOR CONTROL SYSTEM CHANNEL : GRAPHITE / GREY CURTAIN AREA
RESERVE SHUTDOWN SYSTEM CHANNEL : GRAPHITE / GREY CURTAIN AREA
INLET PLENUM TOP / BOTTOM : GRAPHITE
REACTOR PRESSURE VESSEL : IRON
STAGNANT AIR
REACTOR CAVITY COOLING SYSTEM : 20C TH BOUNDARY
NEUTRONIC BOUNDARY BONDITIONG

Figure 5.3: Model of Core Layout

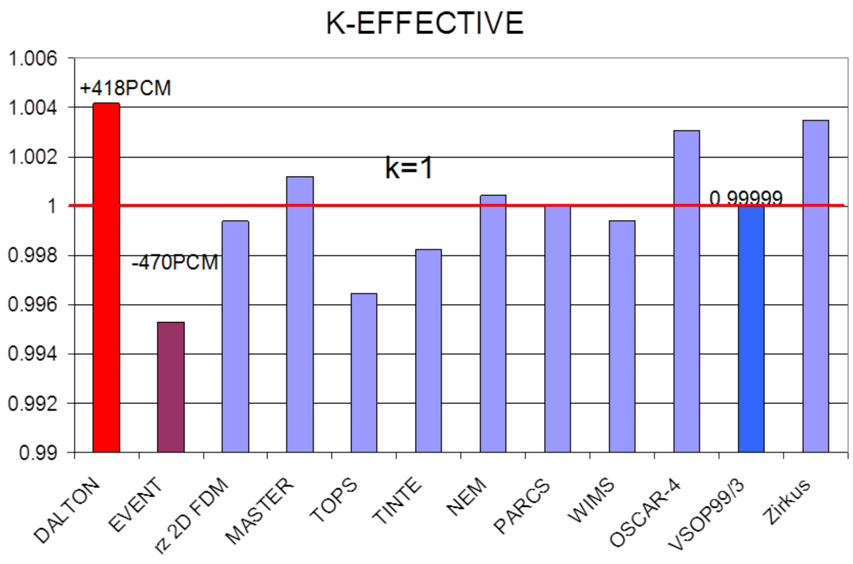

Figure 5.4: K-Effective

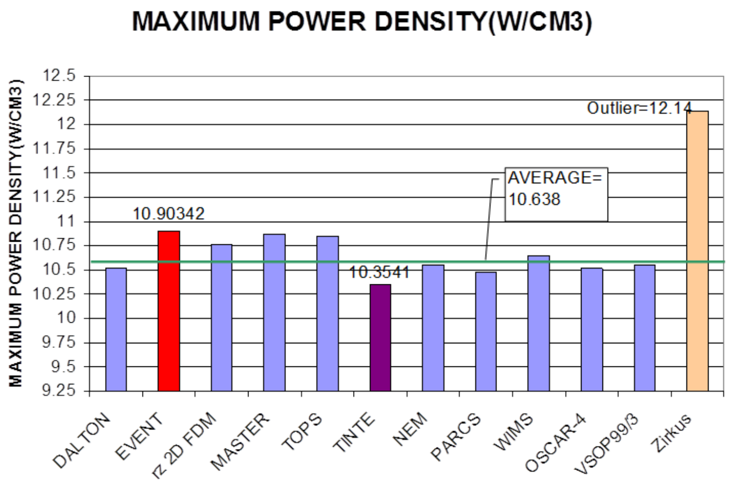

Figure 5.5: Maximum Power Density

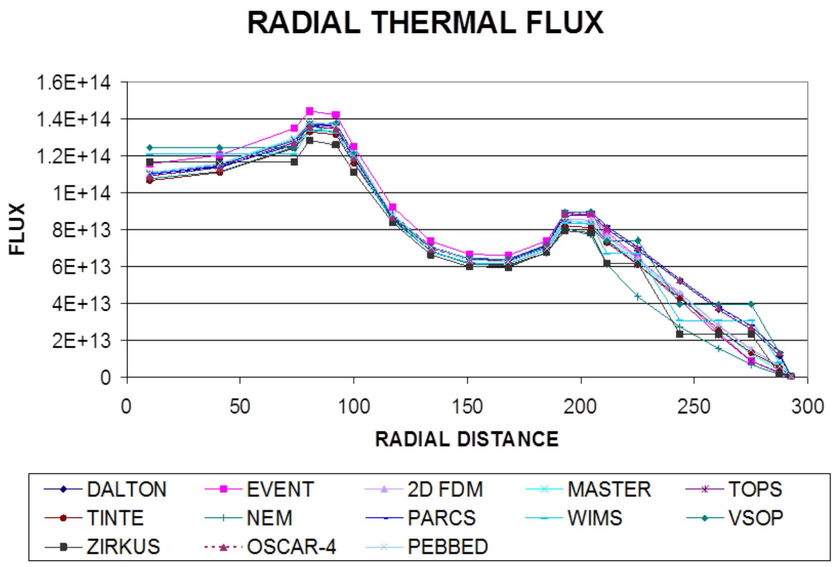

Figure 5.6: Radial Thermal Flux

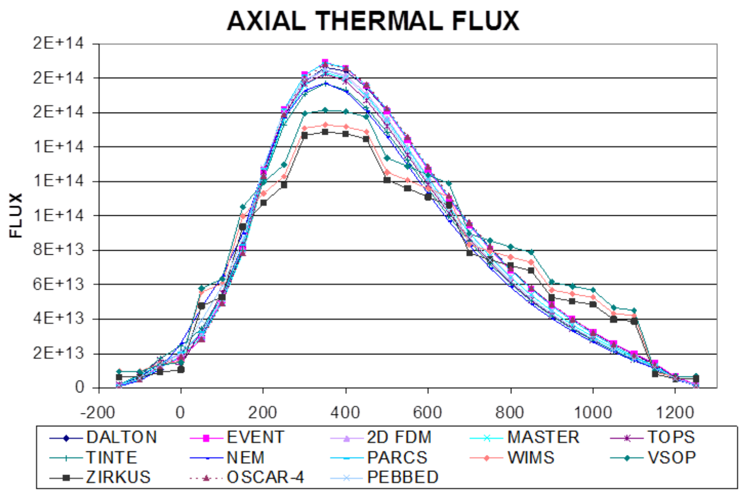

Figure 5.7: Axial Thermal Flux

In case S-2 the power profile obtained from the equilibrium calculation is set in the standalone thermal fluids calculation. The comparison of various average values and profiles are given in Figure 5.8 through Figure 5.11. The coupled neutronics/thermal-fluids problem Case-3 was solved and the results were compared with those obtained by the CAPP/MARS coupled code system.

These three steady-state problems were used to evaluate the performance of the Newtons method described in the previous chapters. The following chapter will show the comparisons of the convergence history of the Newtons methods with the existing solver for each of the three steady state cases.

Table 5.2: Case S-3 Eigenvalue Comparison

Code	k_{eff} (HFP)	k_{eff} (HZP)
CAPP/MARS	0.99270	-
PARCS	0.99283	1.04099
PARCS$*$	0.99282	1.04090

5.3 The Jacobian Matrix

The general guideline is that larger matricies are more costly to solve, however, the sparsity of the matrix is also very important. Table 5.3 gives the dimensions and

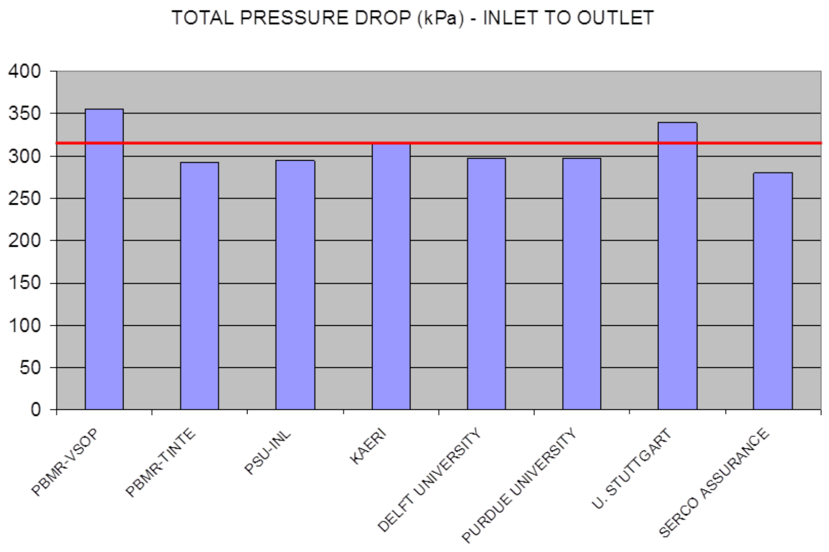

Figure 5.8: Pressure Drop

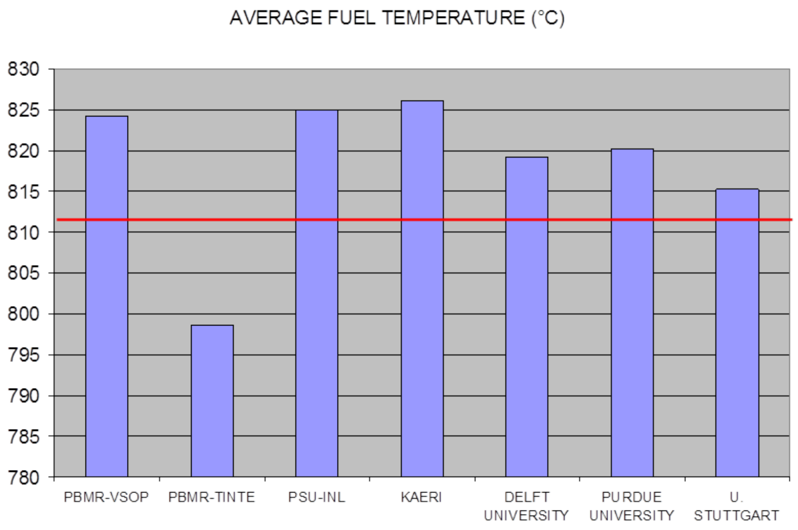

Figure 5.9: Fuel Temperature

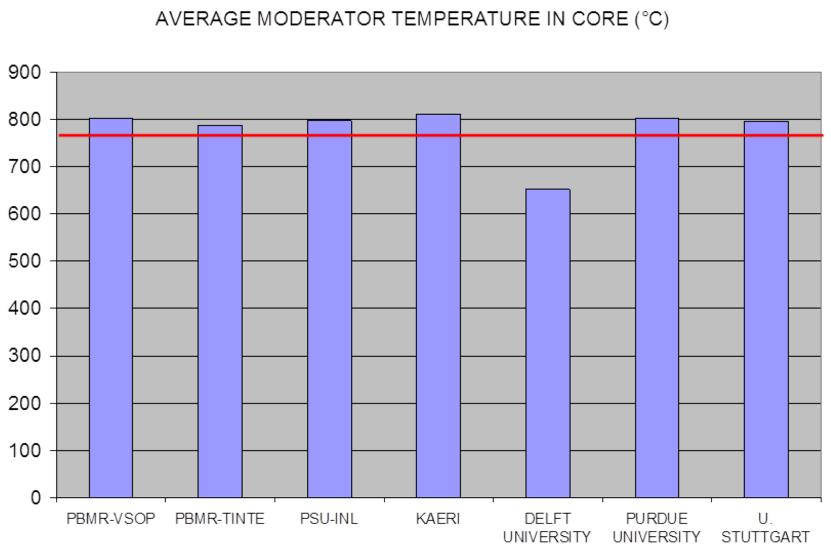

Figure 5.10: Moderator Temperature

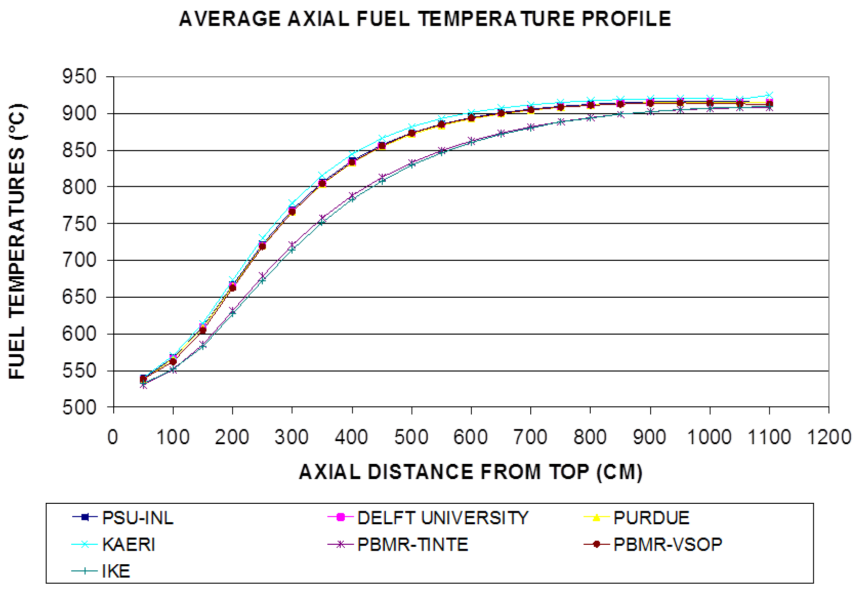

Figure 5.11: Axial Fuel Temperature

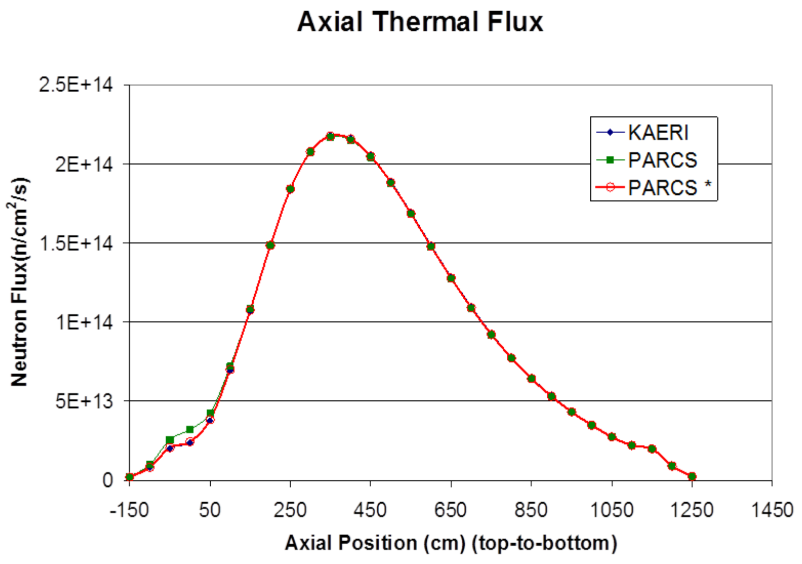

Figure 5.12: Axial Thermal Flux

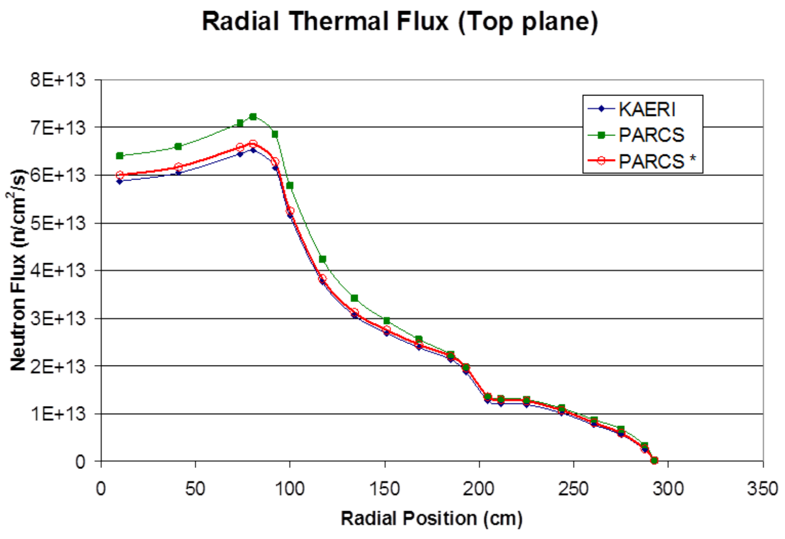

Figure 5.13: Radial Thermal Flux

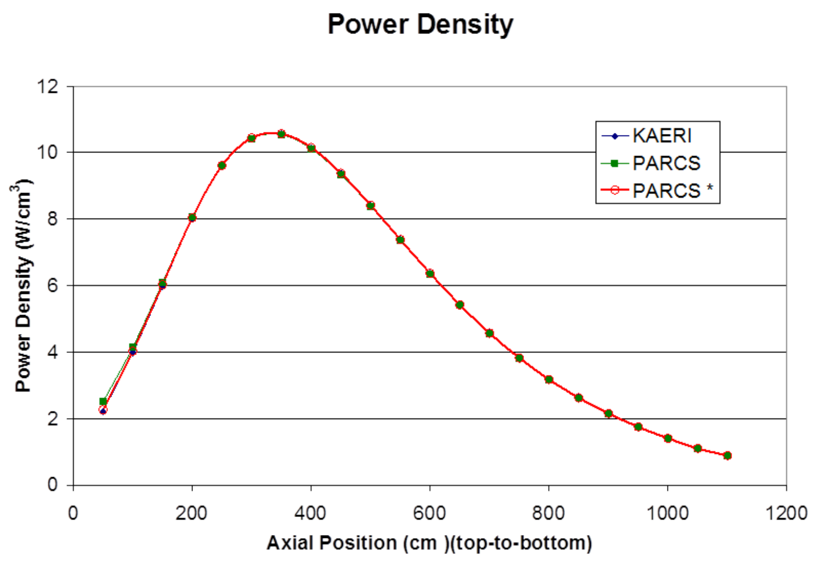

Figure 5.14: Power Density

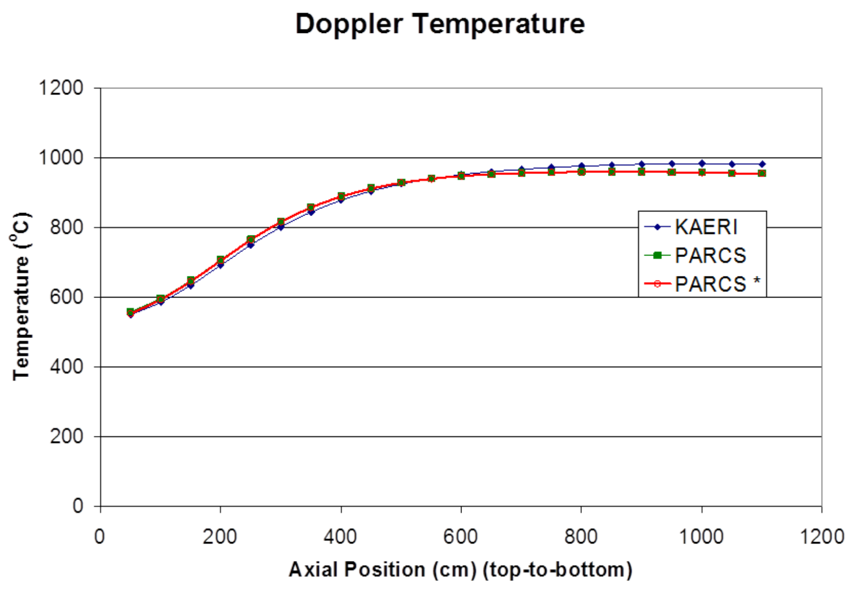

Figure 5.15: Axial Doppler Temperature

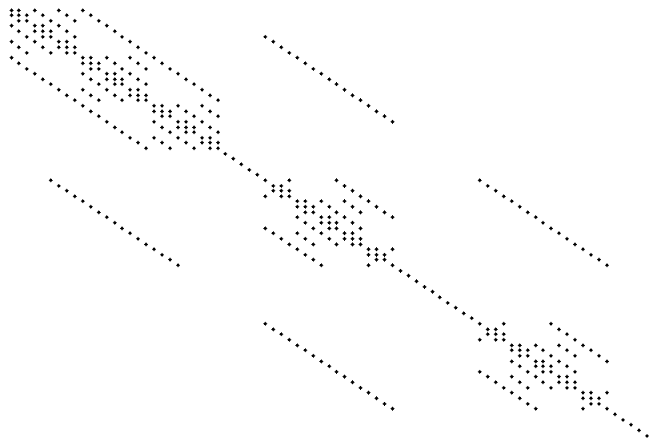

Figure 5.16: Partial Filling of Practical Matrix

sparsity of the three matrices used in the test problems. The large increase in the size of the coupled system results from the large number of 1-D conduction problems that must be solved for the XS feedback. In all cases, the percent of non-zero elements is about 0.05%.

Table 5.3: Resulting Jacobian Linear Systems

System	Neutronics	Therm-Flu	Coupled
Dimension	1161	8772	61825
Non-Zeros	6591	36559	207747

Because of the computational cost of building the Jacobian, the performance of the Newton's Method will depend largely on the size of the system analyzed. An estimation of the required operation counts was useful also performed. Because these systems are very sparse, as shown above, the commonly held assumptions regarding performance of direct vs iterative system may not be valid. Another important consideration is the storage convention used in the coding. Figure 5.3 shows that not all nodes contain a calculation for all variables. The boundary nodes have only a conduction solve and do not generate heat and therefore, the Jacobian matrix will contain identity submatrices, as illustrated in Figure 5.16.

The second and third submatrices are smaller than the first submatrix. This means that both the direct solve and iterative solve will have some zero elements in the RHS.

This is not a significant issue for the direct solver but represents a potential problem for the iterative solver. Each time a subspace vector is generated, operations are wasted on these zero residuals, which then results in poor performance for GMRES. An estimate of the required operations is shown in Table 5.4. The direct solve statistics were taken from PARDISO, and the iterative estimate was adapted from Sosonkina [87]. N is the matrix dimension, N_{NZ} is the number of non-zero elements in the Jacboian, K is the number of stored Krylov vectors, and I is the number of inner iterations per Newton step.

$$W_{GMRES} \approx 4I\left(N_{NZ} + \left(N_{NZ} - N\right)\right)) + \frac{1}{2}NKI + \mathcal{O}(I) \tag{5.1}$$

Table 5.4: Estimate of Solve Operations

Operations Per Newton	Neutronics	Therm-Flu	Coupled
Direct Flops	1.308e6	1.000e7	6.972e8
GMRES Inners	22	29	114
GMERS Flops	1.407e6	1.009e7	6.973e8

The number of inners iterations was calculated to obtain similar total operations per outer iterations. The results here indicate that if GMRES requires more than 22, 29, and 114 iterations for the neutronics, thermal-fluids, and coupled field solution, respectively, then the direct solver can be expected to perform better than the iterative method.

These esimates depend on the number of stored vectors as well, which varies between the neutronics, thermal-fluids, and coupled calculations. The current implementation could be improved in several ways, which include storing the residual more efficiently, reducing N, storing less subspace vectors, reducing K, or improving the preconditioner, reducing I.

Chapter 6
Analysis of Implicit Formulations

This chapter provides the results and analysis of applying the Newton Method to the HTR problem described in the preceding chapter. First the application of Newtons method to the individual field solutions will be presented with the temperature-fluid solution presented in Section 6.1 and the neutronics in section 6.2. The analysis of the individual field solution was important to verify the accuracy of the Jacobian since errors in the Jacobian itself would not always prevent convergence. The solution of the coupled field solution are then presented in section 6.3. In order to obtain a reasonable initial guess for the Newton iteration, the conventional solvers were run for a few iterations. The primary variables were then used to form the Jacobian and initiate the Newton iteration until convergence.

The fundamental expansion of the primary variables and coefficients serves as the foundation for the Jacobian matrix. In order to test each derivative and determine the "importance" of individual coefficients, the Jacobian could be modified to include or exclude any coefficient expansion. However, the Jacobian with all analytic derivatives was the basis for the Exact Newton's method.

6.1 Newton-Krylov Thermal-Fluid Analysis

6.1.1 Exact Newton

The problem described in the previous chapter was used with a fixed power in order to analyze the solution of the thermal-fluid equations. This problem required the solution of all six field equations, which allowed testing of the coupling terms among the field equations. In the fully implicit method all field equations are solved together and therefore the convergence is generally different from the conventional solution which uses a nested iteration scheme in which the heat transfer coefficient is updated in an outer iteration.

The first Newton method studied was the Exact Newton's Method in which the Jacobian was formed with analytical derivatives as described in Chapter 3. A comparison of the convergence of the energy and pressure equations using the operator split and Exact Newtons Method is shown in Figure 6.1 and Figure 6.2, for the 2-norm and infinite norm, respectively. The superlinear convergence and the considerable decrease in the number of outer iterations observed with the Exact Newtons Method compared to the operator split method is indicative of the performance expected of the Exact Newton's Method. The normalized change in each variable is a practical measure of convergence, but the convergence of the residuals is also shown in Figure 6.3 and Figure 6.4.

The pressure appears to converge much quicker than either energy equation. This results from the decoupling of the pressure from energy equations, as well as the limited effect of fluid temperature on fluid density. The flow resistivity, the primary coefficient in the pressure equation, is a strong function of pressure, as it depends on several empirical, and unit less constitutive relationships, which drives the convergence of the pressure equation. The convergence of the energy equations is very similar. This is due to the heat transfer coefficient, alpha, but also the definition of the wall temperature, uses to evaluate several coefficients in the energy equations. Coupling the energy equation considerably improves convergence.

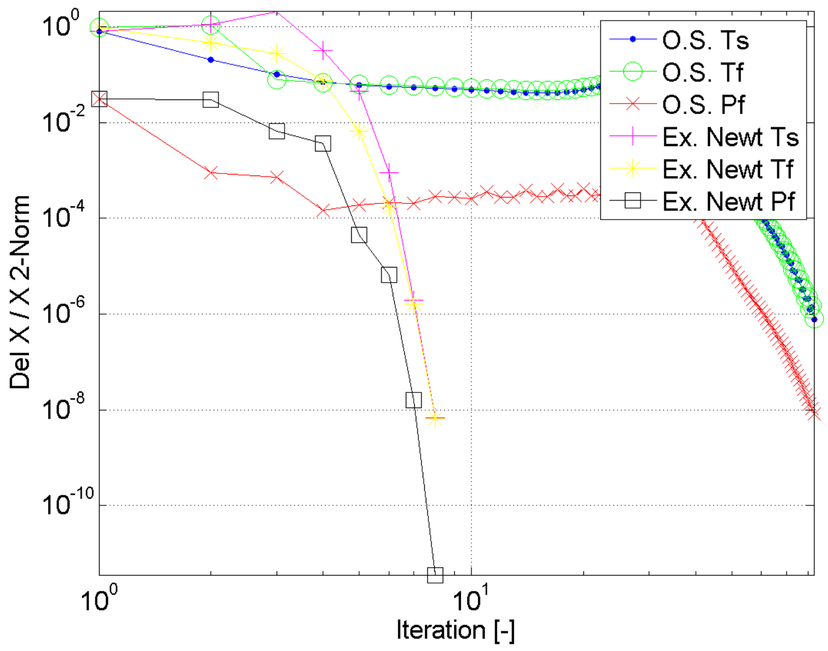

Figure 6.1: Thermal Fluids Exact Newton Del-X / X 2-Norm

The physical derivatives were also compared. Each residual has different units, and therefore are at different levels. However, each residual should converge to the same level between the conventional solution and the Newton's Method.

The results in Figure 6.1 through Figure 6.4 indicate that the Exact Newton's method converges more quickly than the conventional operator split method. The expected "2nd Order" convergence is apparent with each of the norms. However, also as expected, because of the overhead to form the Jacobian, the decrease in the computational time for the Exact Newtons method is not as significant as the decrease in the number of iterations. The number of iterations and the wall time are given in Table 6.1 for the O.S. and Exact Newtons method. As indicated there is an order of magnitude reduction in the number of outer iterations but only a factor of two reduction in the computational time. The increased computational expense of calculating the derivatives and solving the large matrix equations offsets much of the potential increase in performance.

A comparison between the convergence of individual solution variables obtained from the Exact Newton and Operator Split method is shown in Table 6.2. As indicated there is very little difference between the two solutions which provides confidence in the accuracy of the Newtons method solution.

A closer analysis of the individual solution variables in the Exact Newton's Method was also performed as shown in Figure 6.5 and Figure 6.6. As indicated a similar rate of convergence is observed for each of the variables with the exception of the fluid pressure which converges slightly faster than the temperatures and velocities.

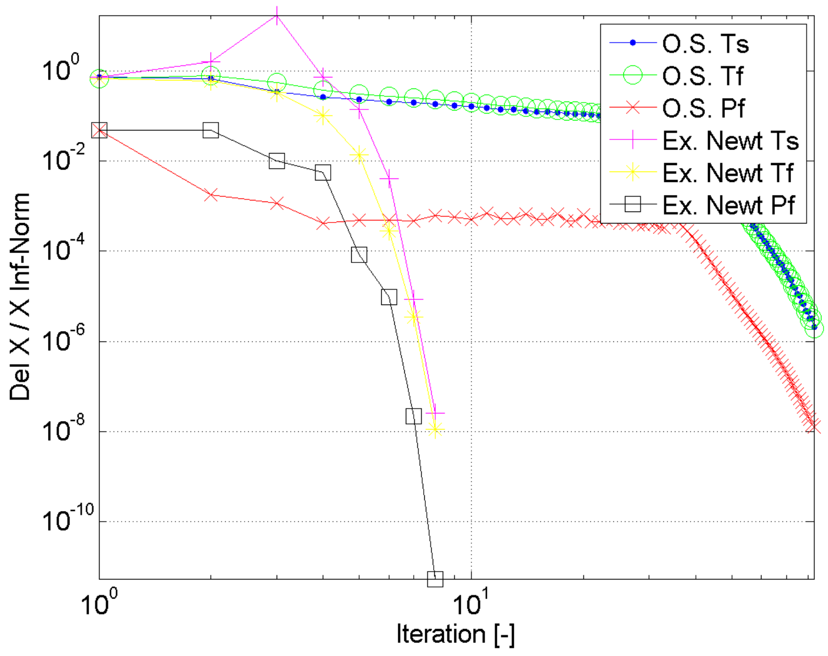

Figure 6.2: Thermal Fluids Exact Newton Del-X / X Inf-Norm

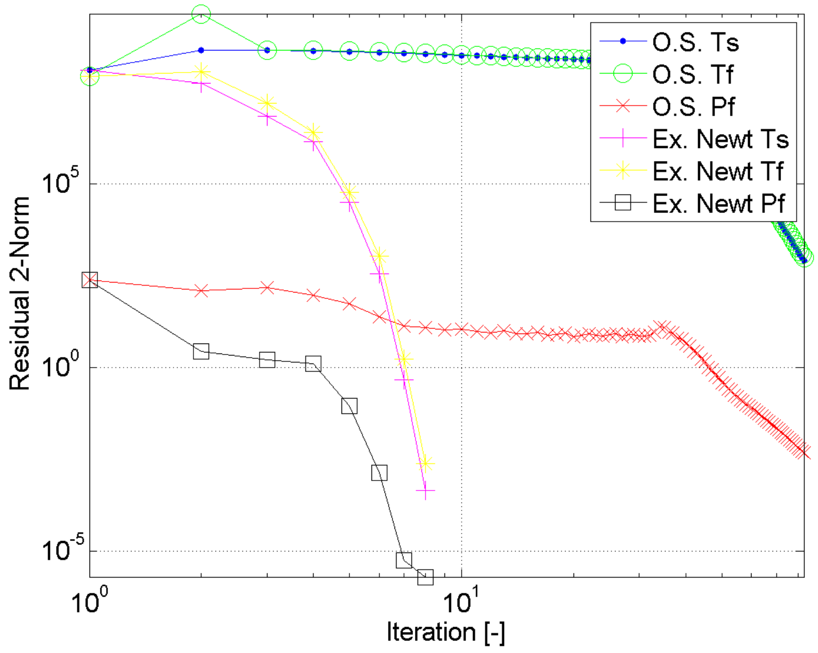

Figure 6.3: Thermal Fluids Exact Newton Residual 2-Norm

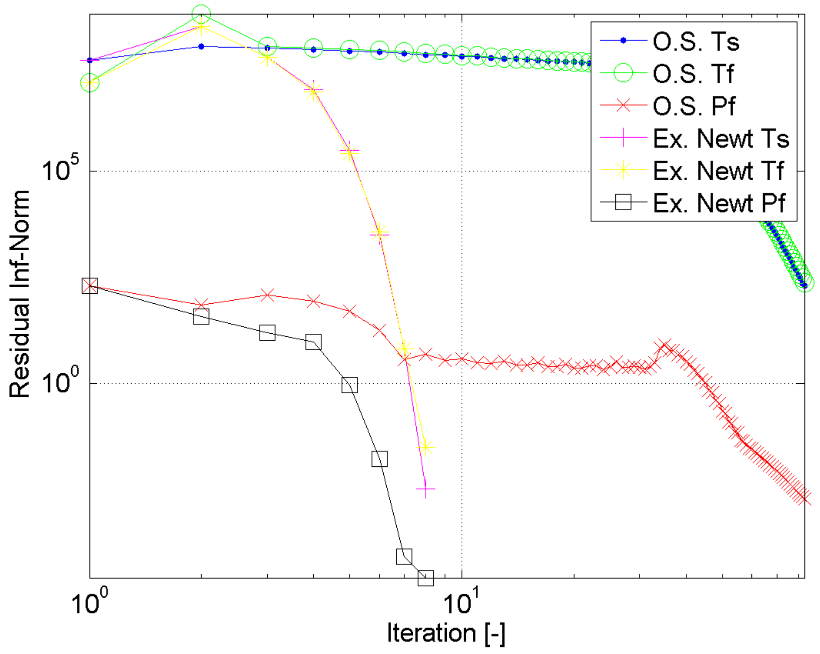

Figure 6.4: Thermal Fluids Exact Newton Residual Inf-Norm

Table 6.1: Thermal-Fluids Performance Comparison

Method	Current/O.S.	Exact Newton
Time [s]	1.901	1.074
Outer Itr	83	7

Table 6.2: Thermal-Fluids Results Comparison

Variable	Ts	Tf	Pf	Vx	Vz
% Err Max	6.98E-04	6.71E-04	3.96E-06	7.95E-06	1.35E+01
% Err RMS	1.03E-05	1.84E-05	2.48E-07	6.81E-07	3.95E-01
Abs Err Max	6.44E-03	6.31E-03	3.65E-04	1.00E-06	6.00E-05
Abs Err RMS	9.01E-05	1.70E-04	2.28E-05	8.57E-08	1.82E-06

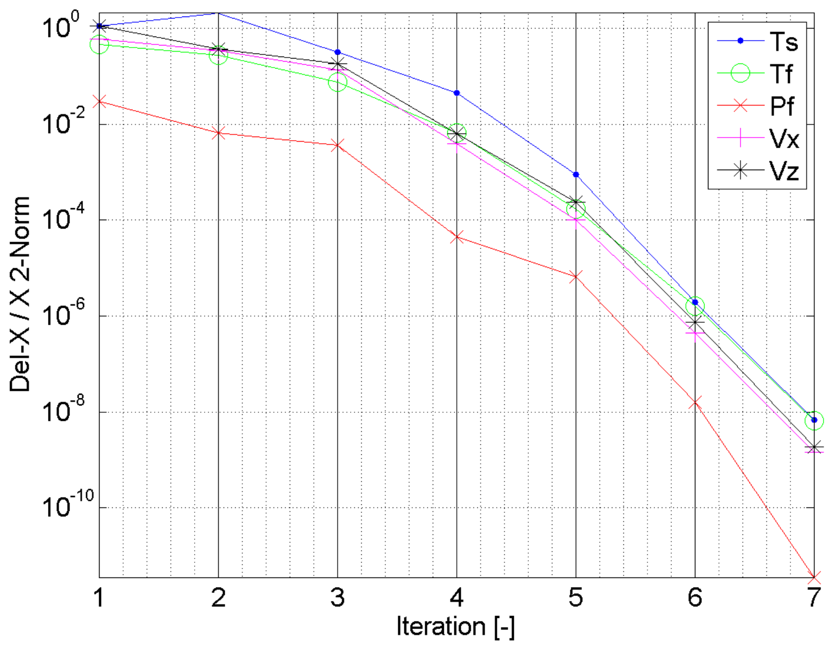

Figure 6.5: Thermal Fluids Exact Newton Del-X / X Comparison

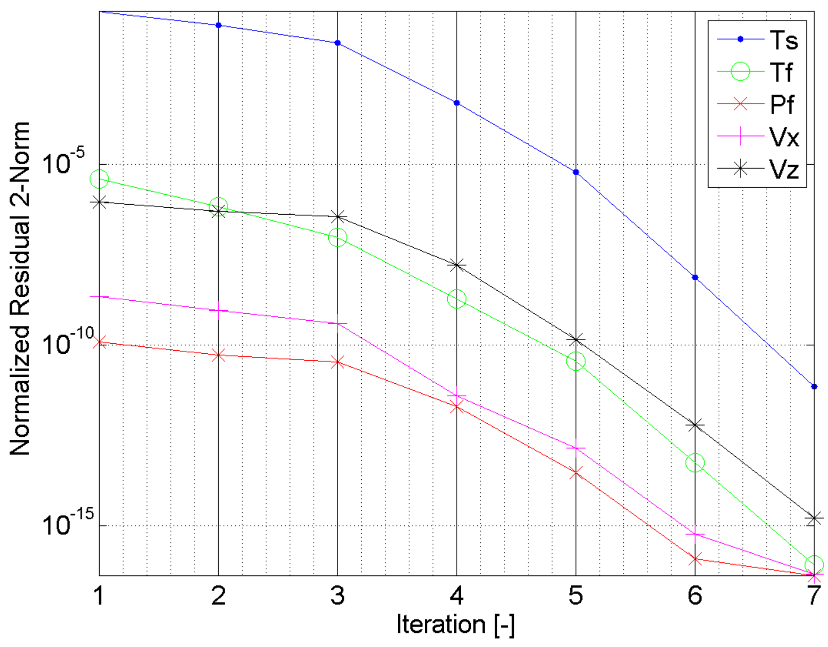

Figure 6.6: Thermal Fluids Exact Newton Residuals Comparison

6.1.2 Inexact Newton

The inexact Newton method was then analyzed with the inner iteration performed using an ILU preconditioned GMRES solver as discussed in Chapter 4. Two methods were used to determine the convergence of the inner iteration. First, the same fixed tolerance of 10^{-8} was used for all inner iterations and then a more practical algorithm was used in which the convergence tolerance of the inner iteration was determined by the residual of the outer iteration.

The first method using a tight convergence of each inner iteration is useful to evaluate the numerical performance of the Exact and Inexact Newtons method. The convergence of the solid temperature and fluid velocity is shown in Figure 6.7 and Figure 6.8, respectively, and as indicated the rate of convergence of the 2-norm and infinite norm are very similar. The residuals are shown in Figure 6.9 and Figure 6.10.

In the second inexact method the tolerance of the inner iteration was determined by the residual of the outer iteration. During the initial outer iterations the inner iteration convergence was relaxed and then increased as the outer residual reduced during the final iterations. As shown in Figure 6.11, this considerably reduced the overall number of GMRES iteration compared to the first method of using a fixed tolerance throughout the Newton iteration.

The impact of the dynamic inner iteration convergence on the convergence of the Inexact Newton method is shown in Figure 6.12 and Figure 6.13. As expected there is some decrease in the rate of convergence, but as shown in Table 6.3, there is only a slight increase in the number of outer iterations. However, as also shown in the Figure 6.11 even the slight increase in the number of outer iterations leads to only a slight reduction in the overall computational time.

Fixed Convergence: rel. tol = (10e-6 v0 = v1), abs tol = 0.0
Dynamic Convergence: rel tol = (10e-5 v0 = v1), abs tol = 10e-4*r0

One of the most important observations from Table 6.3 is that even with the dynamic inner convergence method the computational time of the inexact Newton method is considerably higher than the exact Newton method.One of the primary reasons for this was discussed in Chapter 5 which estimated the breakeven performance of the direct and iterative solvers. As shown in Figure 6.11, even with dynamic inner iteration tolerance, the number of GMRES iterations is above the breakeven threshold, and Exact Newton with the direct solver should be expected to perform better.

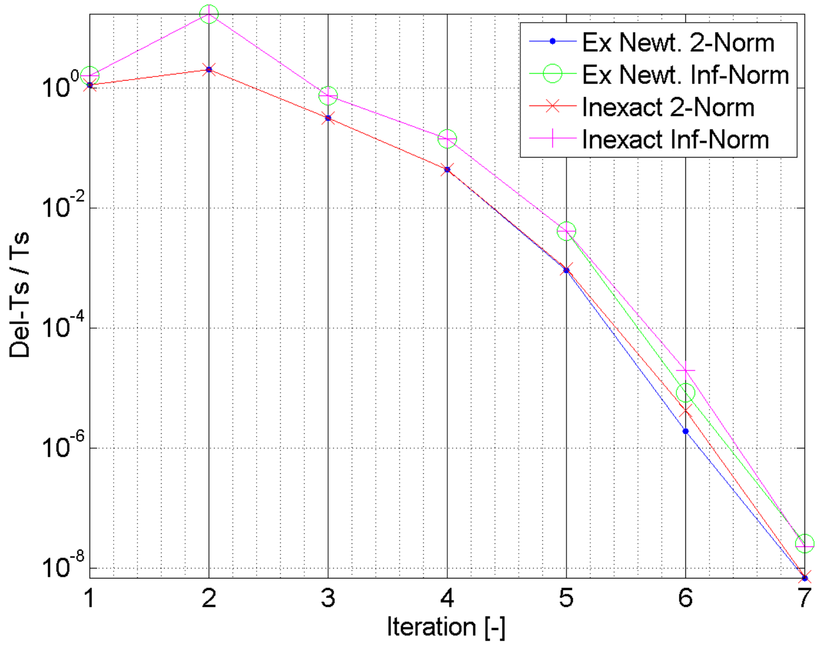

Figure 6.7: Thermal Fluids Inexact Newton Del-Ts / Ts

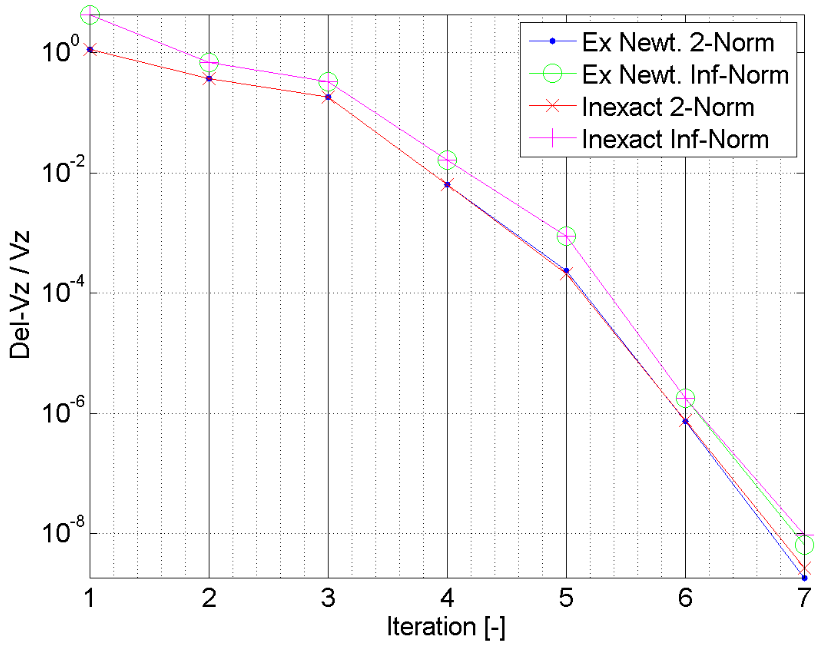

Figure 6.8: Thermal Fluids Inexact Newton Del-Vz / Vz

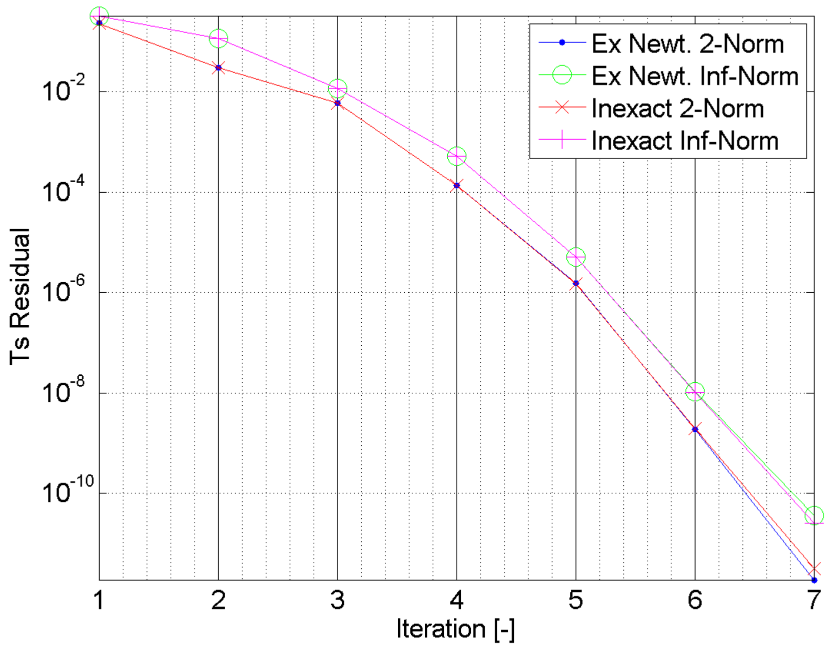

Figure 6.9: Thermal Fluids Inexact Newton Normalized Ts Residual

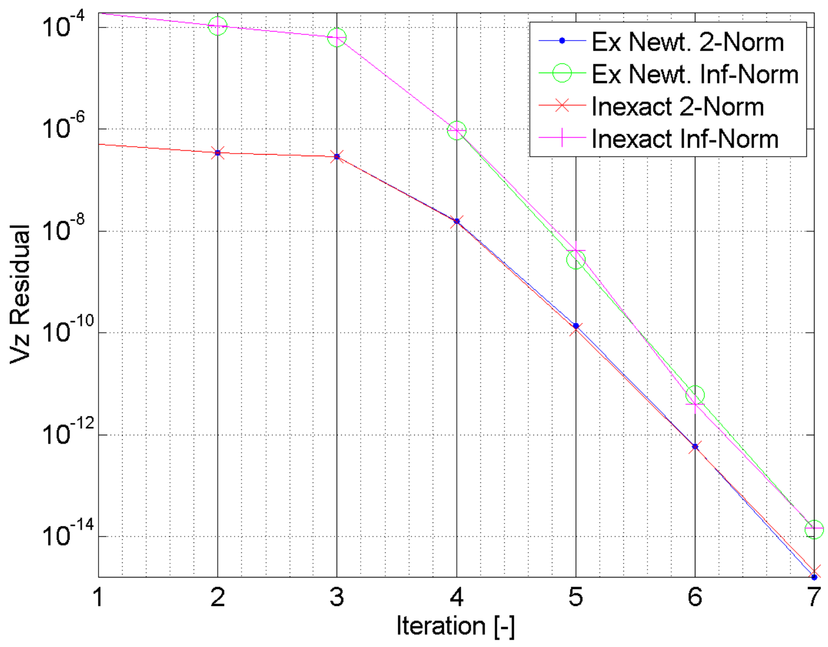

Figure 6.10: Thermal Fluids Inexact Newton Normalized Vz Residual

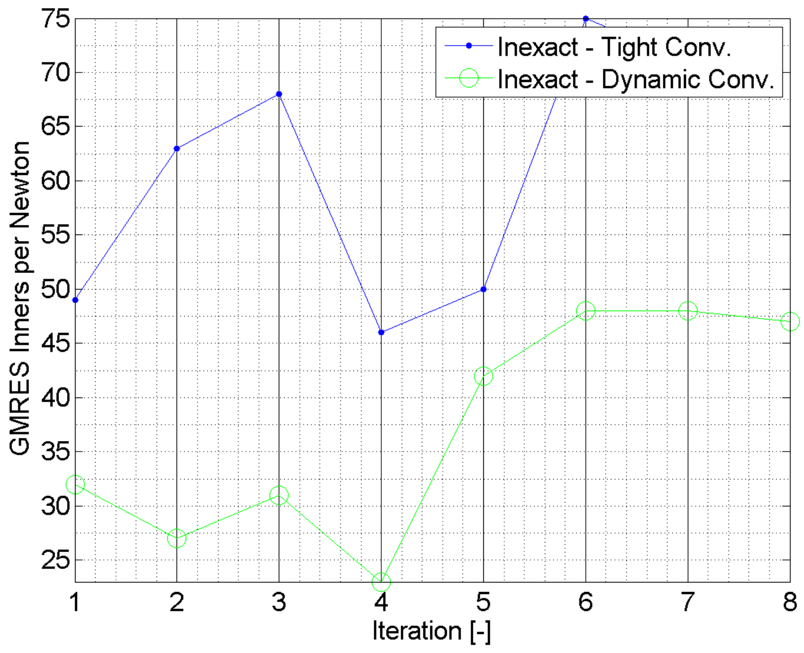

Figure 6.11: Thermal Fluids Inexact GMRES Conv.

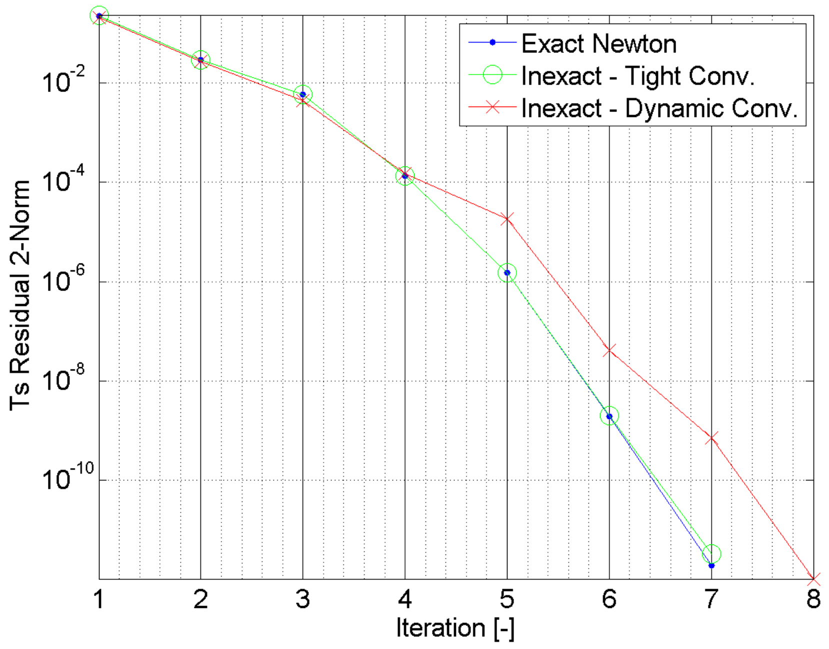

Figure 6.12: Thermal Fluids Inexact Newton Normalized Ts Residual

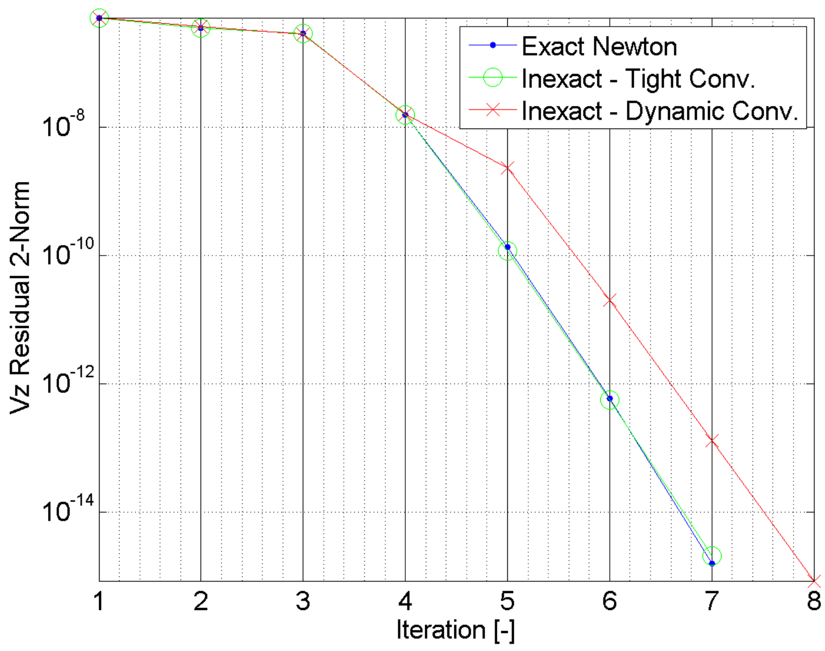

Figure 6.13: Thermal Fluids Inexact Newton Normalized Vz Residual

Table 6.3: Thermal-Fluids Inexact Newton

Method	Exact Newton	Inexact Newton Tight Conv.	Inexact Newton Dynam. Conv.
Time [s]	1.104	1.167	1.177
Newton Itr	7	7	8
Flops / Nwt.	1.00e7	1.009e7	1.309e7

6.1.3 Finite Difference Jacobian

The next phase of the research was to investigate a finite difference method for calculating the elements of the Jacobian. Two methods were used to form the perturbation size for the finite difference evaluation. The first method was to simply use a fixed perturbation size and the second method was to use an algorithm to determine the perturbation size which depended on the matrix properties. Both methods were evaluated using both the inexact and exact Newtons method.

The first method of fixed perturbation size was first evaluated using the Exact Newtons method and as expected the rate of convergence was reduced. Figure 6.14 and Figure 6.15 compares the convergence of the Exact Newton method with analytic derivates and with finite derivatives and the overall performance is summarized in Table 6.4. As indicated in the Table, the required number of outer iterations was slightly higher than the Exact Newtons method with analytic derivates. The increased computational cost reflects both the increased number of outer iterations but also the slightly higher cost to calculate the finite difference derivatives than in the analytical approach. Also shown in the Table is the result of applying the first method of fixed perturbation size to the Inexact Newtons method. As indicated both the number of outer iteration and the overall computational time are higher than for the Inexact Newtons method with analytic derivates.

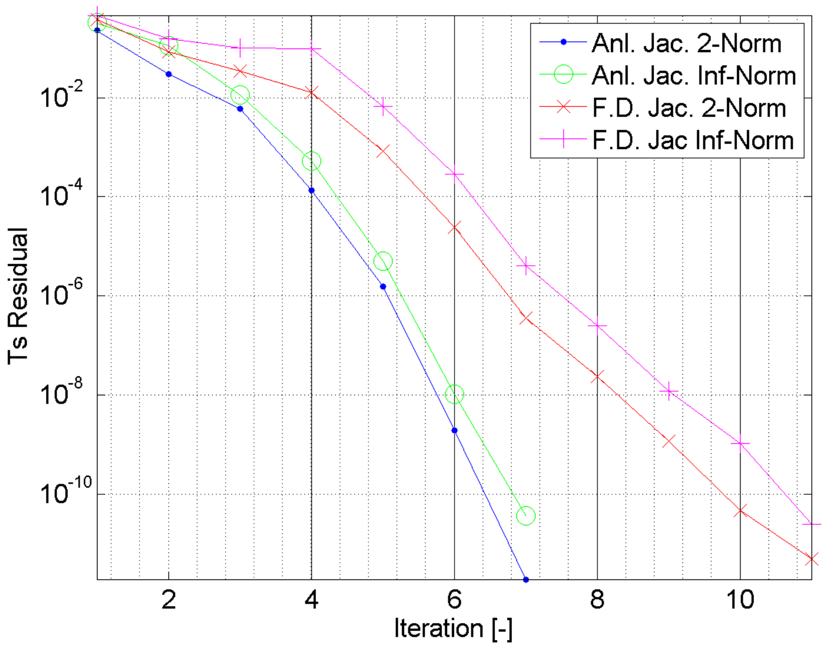

Figure 6.14: Thermal Fluids F.D. Jac Ts Residual

The second method evaluated was to allow the perturbation size to vary based upon an optimized pertubation. Because the finite difference is an approximation, GMRES was used as the basis for the perturbation. This approach was suggested by Xu [55] and depends on the norms of the Jacobian which are estimated using methods described in [55].

$$\gamma \geq \tilde{\gamma} = \frac{2\left\|F(x^{k+1})\right\|}{\left\|s^k\right\|^2} \quad \eta \approx \varepsilon \approx 2^{-52} \approx 2.22^{-16} \tag{6.1}$$

$$\sigma_{opt} = \frac{1}{\|v\|}\sqrt{\varepsilon\frac{\|F'(x)\|}{\gamma}\|x\| + \frac{2\eta\|F(x)\|}{\gamma}} \tag{6.2}$$

The rate of convergence of the Finite Difference method with the optimized method for evaluating the perturbation size shown in Figure 6.16 and Figure 6.17, and summarized in Table 6.5. As indicated, the optimized method does not perform as well as the fixed perturbation size. However, the fixed perturbation size used in this work was chosen after several sensitivity studies, and it is therefore highly susceptible to errors. Although the optimized perturbation method does not perform as efficiently as the fixed perturbation used here, it may provide a more robust method for a wider range of problems.

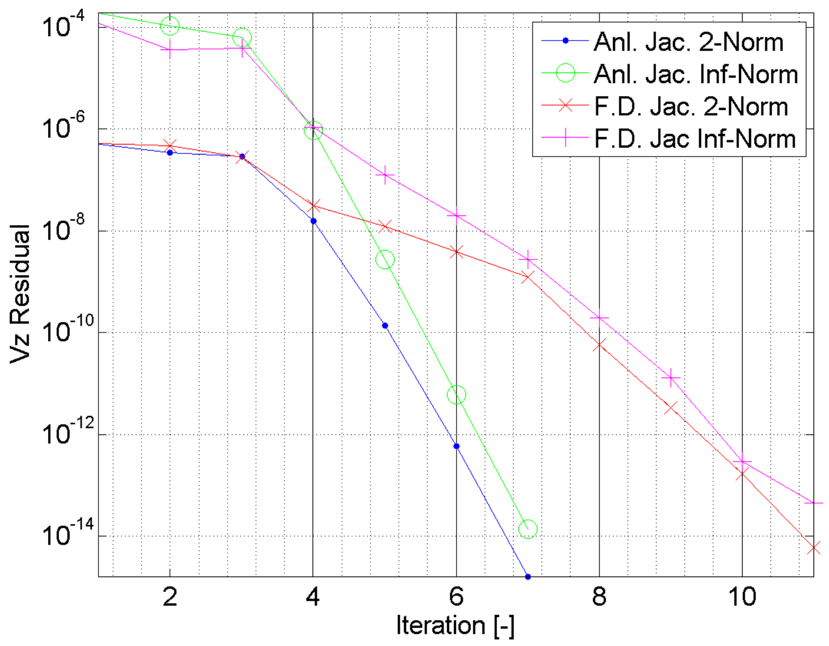

Figure 6.15: Thermal Fluids F.D. Jac Vz Residual

Table 6.4: Thermal-Fluids F.D. Jac Direct Solve

Method	Anl. Jacobian Direct Solve	F.D. Jacobian Direct Solve
Time [s]	1.014	1.528
Newton Itr	7	11

Table 6.5: Thermal-Fluids F.D. Jac with Optimized Pert.

Method	Anl Jacobian	F.D. Jacobian Fixed Pert.	F.D. Jacobian Opt. Pert.
Time [s]	1.177	1.914	2.004
Newton Itr	8	13	15
Flops / Nwt.	1.309e7	1.783e7	1.515e7

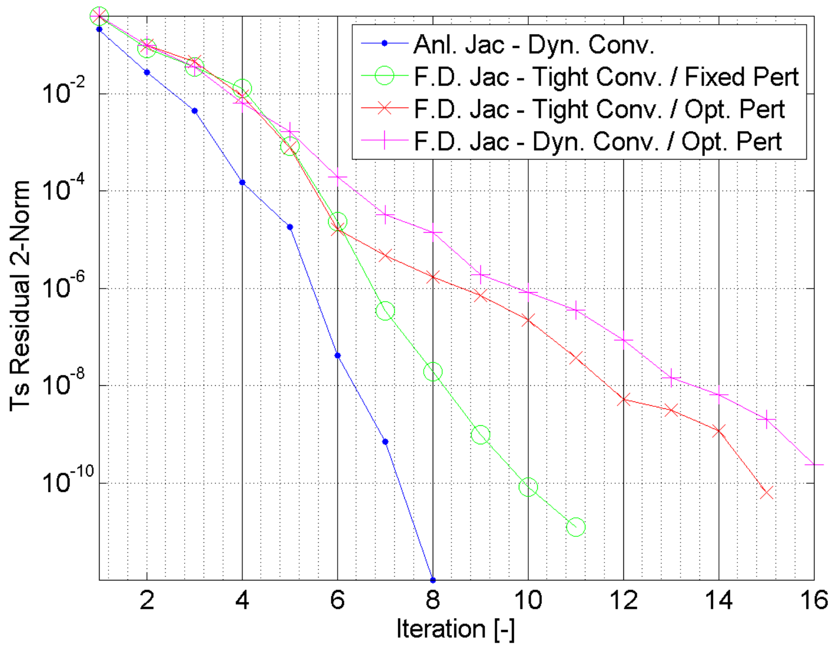

Figure 6.16: Thermal Fluids F.D. Jac Normalized Ts Residual

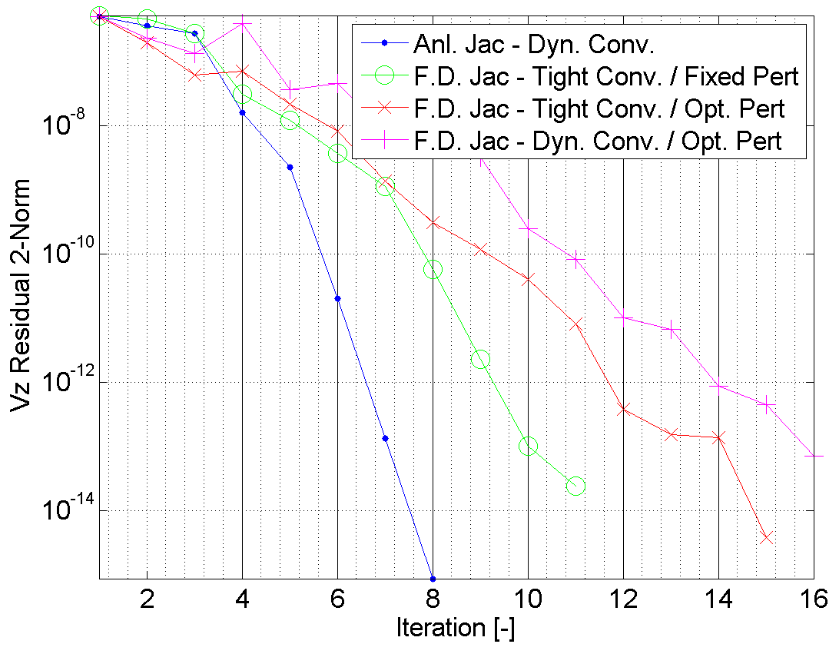

Figure 6.17: Thermal Fluids F.D. Jac Normalized Vz Residual

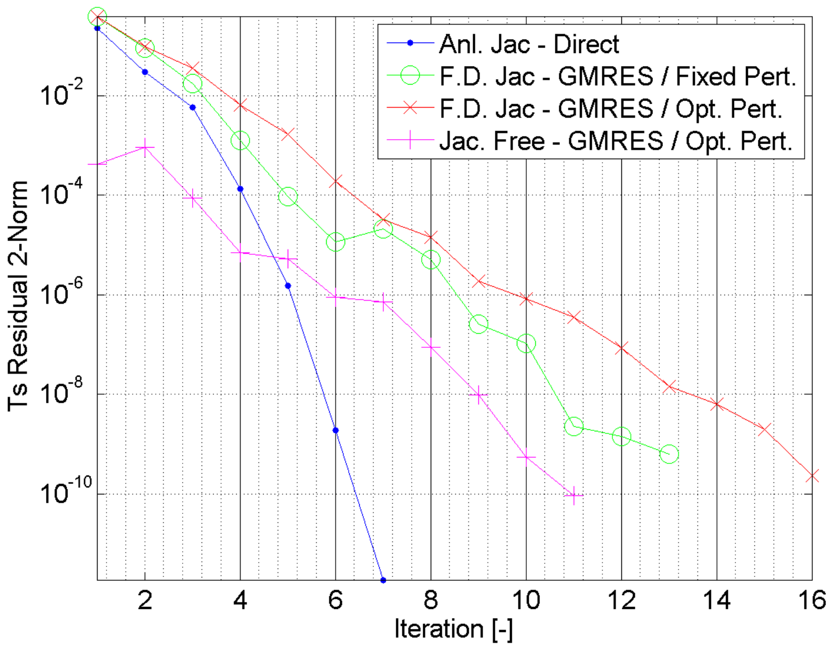

Figure 6.18: Thermal Fluids Jac-Free Normalized Ts Residual

6.1.4 Jacobian-Free

The final method evaluated for the thermal fluids was the Jacobian Free method which has been studied extensively in recent years as discussed in Chapter 4. The GMRES solver was used as the basis for the Jacobian-Free method and the formation of the residual for the six thermal fluids equations was performed similarly to the methods used in the previous sections. As will be evident in evaluating the performance of the method, the cost of computing the residual was significant and one of the areas for future research will be to investigate more efficient methods to compute the residuals.

The residuals are given in Figure 6.18 and Figure 6.19 below, which compares the Exact Newton, F. D. Jacobian, and Jacobian-Free methods. As indicated in the Figure, the Jacobian-Free converges in fewer Newton steps than the F.D. Jacobian, but all methods show superlinear convergence performance.

The computation time is shown in Table 6.6 and as indicated, the number of Newton iterations for the Jacobian-Free method is comparable to the other methods, however the overall computational time is much higher. As noted above, this is primarily because of the high cost of forming the residual which may be reduced with a more compact residual representation. However, it is not expected that any reduction would be larger than 50

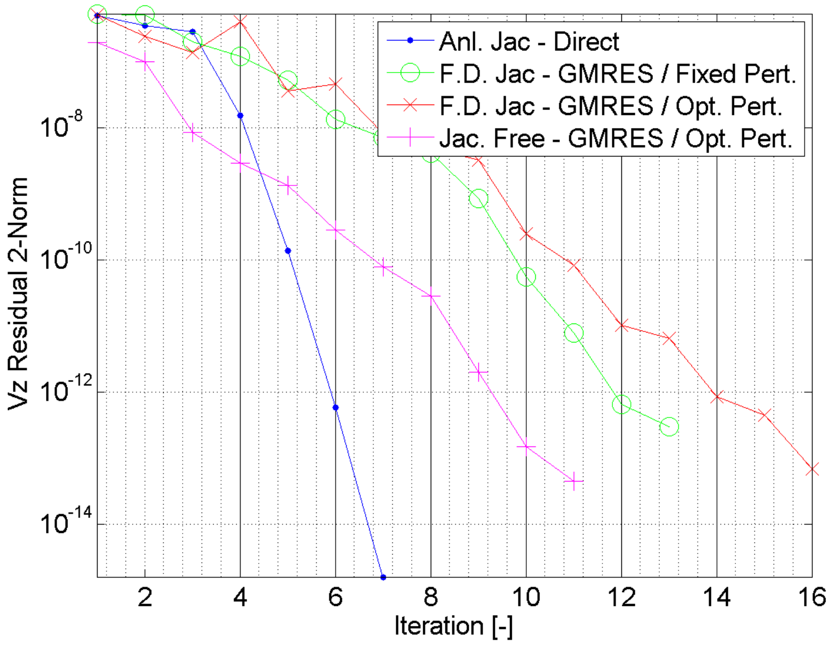

Figure 6.19: Thermal Fluids Jac-Free Normalized Vz Residual

Table 6.6: Thermal-Fluids Jacobian-Free

Method	Anl. Jacobian Direct	Anl. Jacobian Dynam. Conv.	F.D. Jacobian Dyn. / Opt.	Jac.-Free Dyn. / Opt.
Time [s]	1.074	1.177	2.004	5.883
Newton Itr	7	8	15	11
Flops / Nwt.	1.000e7	1.309e7	1.515e7	1.057e7

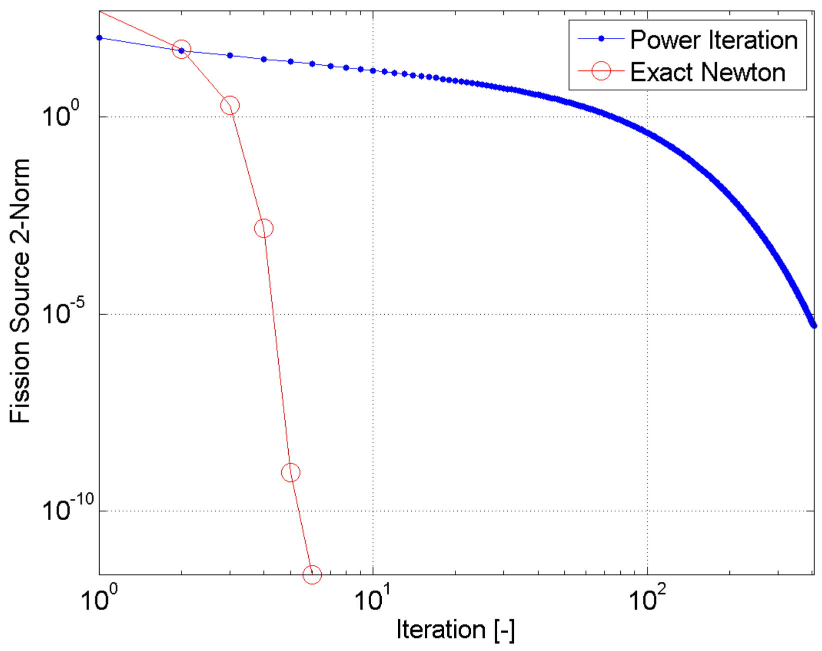

Figure 6.20: Neutronics Exact Newton Fission Source

6.2 Newton-Krylov Neutronics Analysis

6.2.1 Exact Newton's Method

The neutronics solution was analyzed using the test problem described in Chapter 5 with a fixed temperature-fluid field solution and therefore no cross section feedback. The Jacobean of the multigroup neutronics equations was first computed using analytic derivates as described in Chapter 4 and solved exactly using the same direct solver as described in the previous section. The Exact Newton solution was then compared to the solution using the standard fission source and power iterations. The fluxes and eigenvalue were essentially identical but the convergence performance was very different as shown in Figure 6.20. As indicated, the Newton's method reduces significantly the required number of iterations compared to the standard power iteration. The primary reason for this significant difference is that the eigenvalue is solved directly as a primary variable in the Newtons method whereas the standard power iteration involves an inner and outer iteration. The computational time is summarized in Table 6.7 which also shows the small difference in the fluxes.

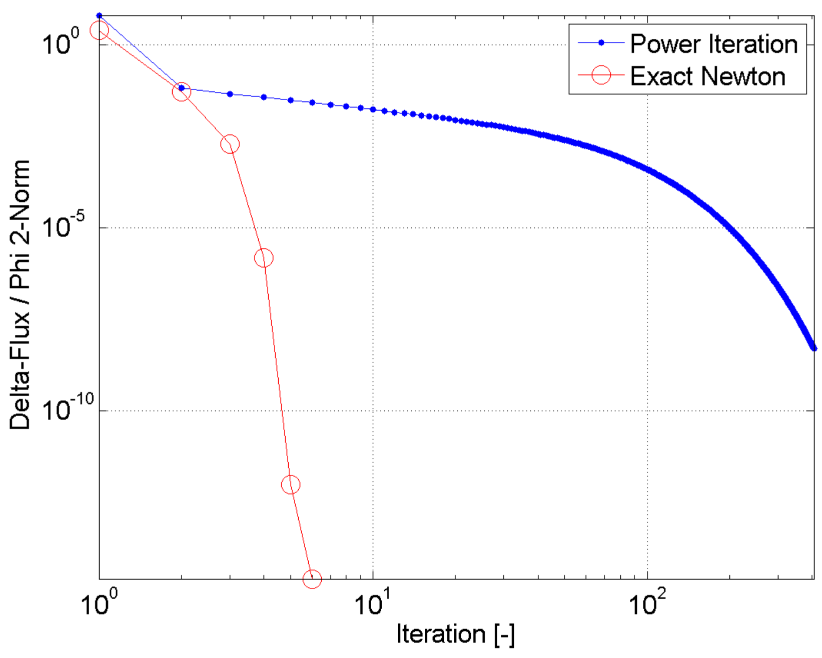

Figure 6.21: Neutronics Exact Newton Del-Flux / Flux

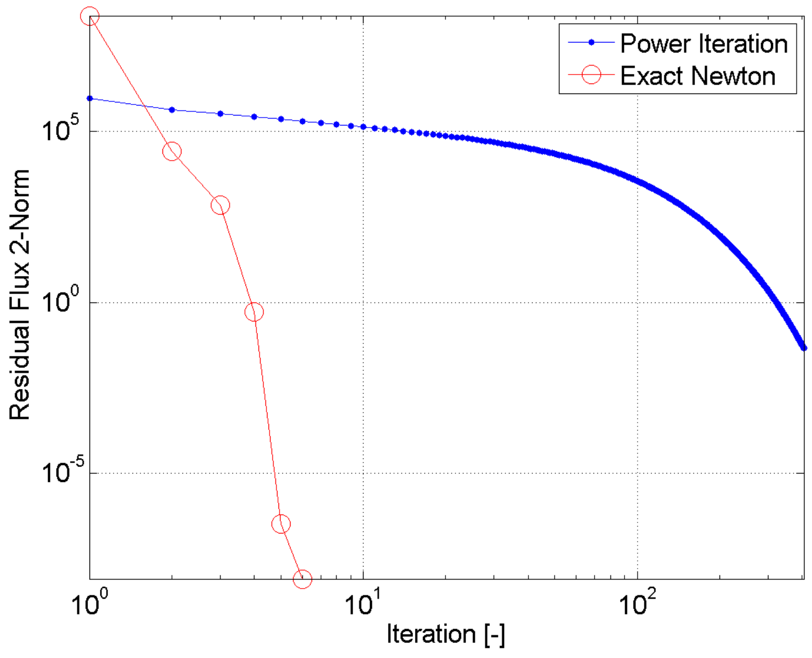

Figure 6.22: Neutronics Exact Newton Flux Residual

Table 6.7: Neutronics Performance Comparison

Method	Current Power Itr.	Exact Newton
Solver	Direct	Direct
Time [s]	2.324	0.265
Outer Itr	406	6

6.2.2 Inexact Newton's Method

The neutronics problem was then solved with the same GMRES method and inexact Newtons method used in the previous section for the Thermal-Fluids equations. A fixed tight convergence 10^{-8} was used to converge GMRES. As shown in Figure 6.23, a similar convergence behavior was observed in the convergence of the flux between the inexact and exact Newtons method. However, some stagnation of the GMRES solution was observed in the residual as shown in Figure 6.24. A comparison of the computation time is shown in Table 6.8. As indicated the same number of iterations are used in the Exact and Inexact Newtons method, but similar to the trend observed in the thermal-fluids solution there is an overall increase in the computational time required for the inexact Newton method, primarily because of the increased cost of the GMRES iterations compared to the direct solver, as noted in Chapter 5.

Table 6.8: Neutronics Results Comparison

Method	Current Power Itr.	Exact Newton	Inexact Newton
Solver	Direct	Direct	GMRES
Time [s]	2.324	0.265	0.405
Outer Itr	406	6	6
% Err RMS	-	3.09e-6	2.86e-6
% Err Max	-	9.46e-4	2.25e-3

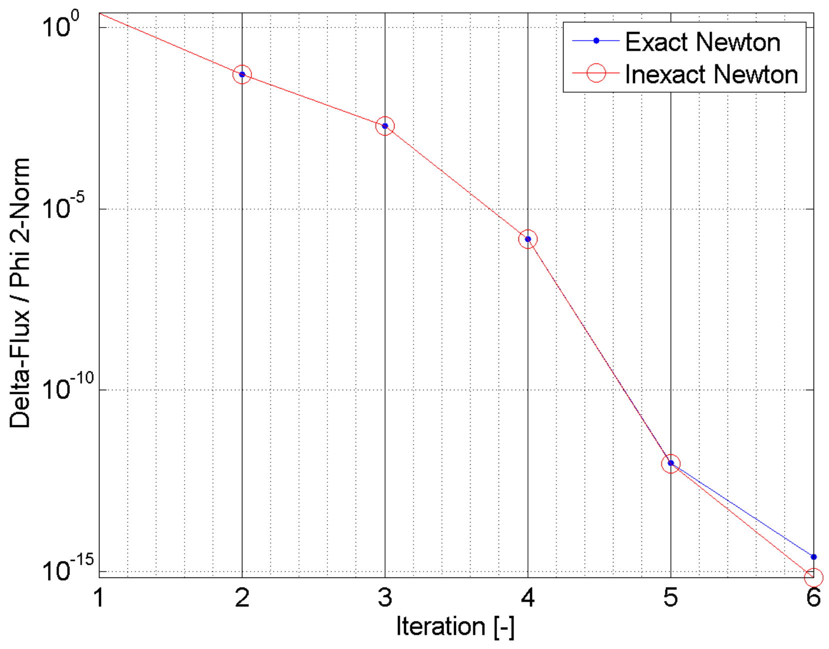

Figure 6.23: Neutronics Inexact Newton Del-Flux / Flux

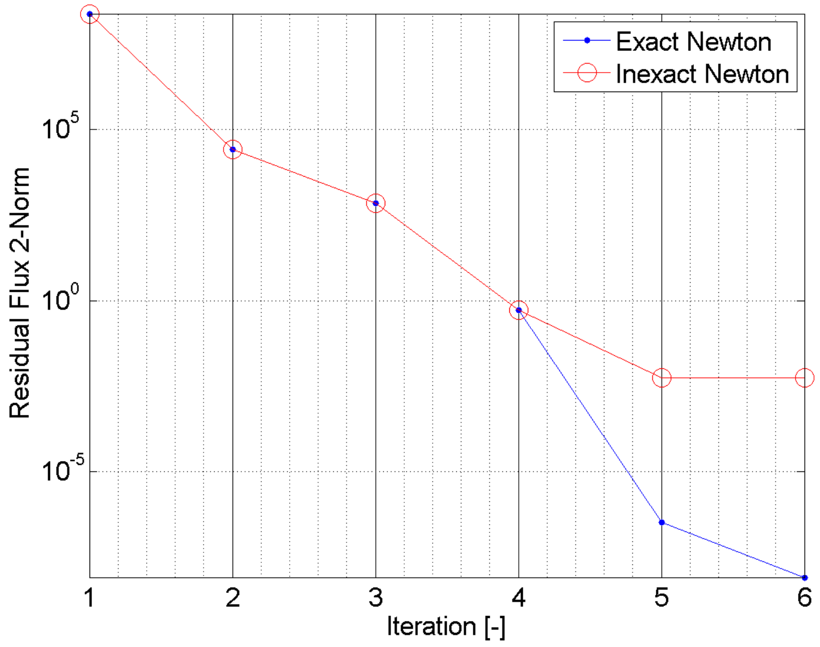

Figure 6.24: Neutronics Inexact Newton Flux Residual

6.3 Newton-Krylov Coupled Analysis

The final phase of the research was to apply the Newton's method to the coupled thermal-fluids and neutronics solution. The same coupled steady state problem was used as described in Chapter 5 and applied to the individual field solutions in the previous sections. As discussed in Chapter 2, the conventional operator split solution first achieves a converged thermal-fluid solution by iterating between the separate solution of each thermal fluids equation and then updates the neutron cross section and solves the neutronics equations. Upon convergence of the neutronics equations, the temperature fluid coefficients are updated and the thermal fluid equations are solved again. The iteration strategy is repeated until convergence.

The operator split method was compared with the same three variations of the Newton solution applied to the thermal fluid equations in Section 6.1. One of the essential differences for the coupled field is that the analytical Jacobian was expanded to include the derivatives of the cross section to the temperature fluid variables. This significantly increased the size of the problem since several submesh are required to calculate the temperature distribution in the pebble and it was necessary to represent this explicitly in the Jacobian.

In the following subsections, results will be presented sequentially for the exact Newton method, the inexact Newton method, and then for the inexact Newton method with a finite difference Jacobian.

6.3.1 Exact Newton

One of the important advantages of the Exact Newton's method is the rapid convergence of the flux source as shown in Figure 6.25. The more rapidly the flux is converged the sooner a converged power and heat source distribution can be obtained for the thermal-fluids solution. Therefore, the accelerated fission source convergence considerably improves the convergence of the coupled system. The two-norm and infinite-norm of the solid temperature and pressure are shown in Figure 6.26 and Figure 6.27.

The residuals of the coupled field solution are given below. After about eight iterations, the Newton's method begins to converge superlinearly as expected. This is similar to the convergence observed in the thermal-fluids. Also, because of the tighter coupling between the fields the residuals are generally much smaller in the Exact Newton's method than in the conventional solution method.

A comparison of the performance of the exact Newton and the operator split method is given in Table 6.9. The considerable reduction in the total number of iterations is similar to the thermal-fluids case. However, the increased size of the system results in a higher computational burden to form the Jacobian which offsets the savings in the reduced number of iterations.

Table 6.9: Coupled Performance Comparison

Method	Current/O.S.	Exact Newton
Time [s]	23.275	14.508
Outer Itr	495	17

The convergence of the 2-norm of the solution variables and the residuals is shown in Table 6.31 and Table 6.32. All of the residuals are normalized to converge to below double precision error, but execution terminates as the stopping criteria is met. Some oscillations are introduced from the discontinuous derivatives, but the solution stabilizes after about five iterations. The overall convergence behavior is similar to what was observed in the thermal fluids.

A detailed comparison of the converged coupled field solution variables is shown in Table 6.10 and Table 6.11. As indicated there is very little difference in the solution variables.

Table 6.10: Coupled Results Comparison Neutronics

Variable	Phi-1	Phi-2	Tm	Td
% Err Max	1.25E-05	1.19E-05	4.48E-06	8.82E-06
% Err RMS	9.93E-04	9.98E-04	2.86E-04	2.70E-04
Abs Err Max	4.30E+06	6.33E+06	3.84E-05	8.01E-05
Abs Err RMS	1.00E+09	1.00E+09	2.66E-03	2.52E-03

* Extremely Small Values Increase Percent Error

Table 6.11: Coupled Results Comparison Thermal Fluids

Variable	Ts	Tf	Pf	Vx	Vz*
% Err Max	4.48E-06	9.89E-06	1.23E-07	5.39E-04	1.84E+00
% Err RMS	2.86E-04	2.88E-04	2.07E-06	8.34E-03	5.69E+01
Abs Err Max	3.84E-05	8.73E-05	1.14E-05	9.54E-07	2.27E-04
Abs Err RMS	2.66E-03	2.68E-03	1.92E-04	1.30E-05	6.90E-03

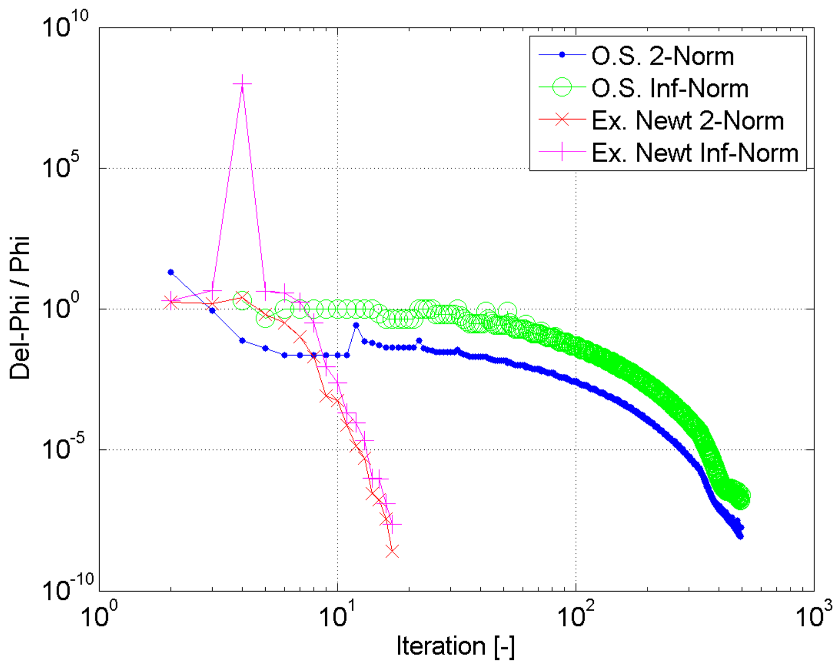

Figure 6.25: Coupled Exact Newton Del-Flux / Flux

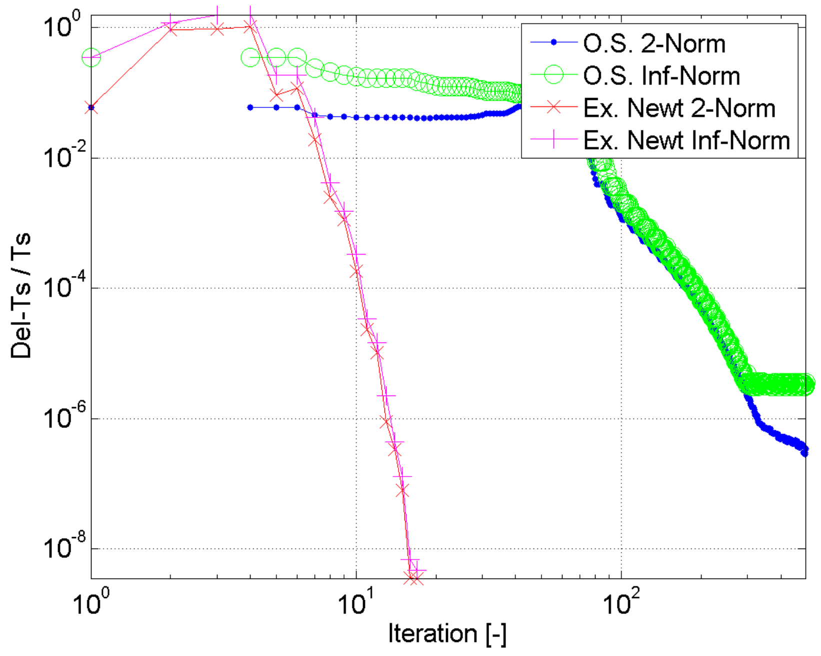

Figure 6.26: Coupled Exact Newton Del-Ts / Ts

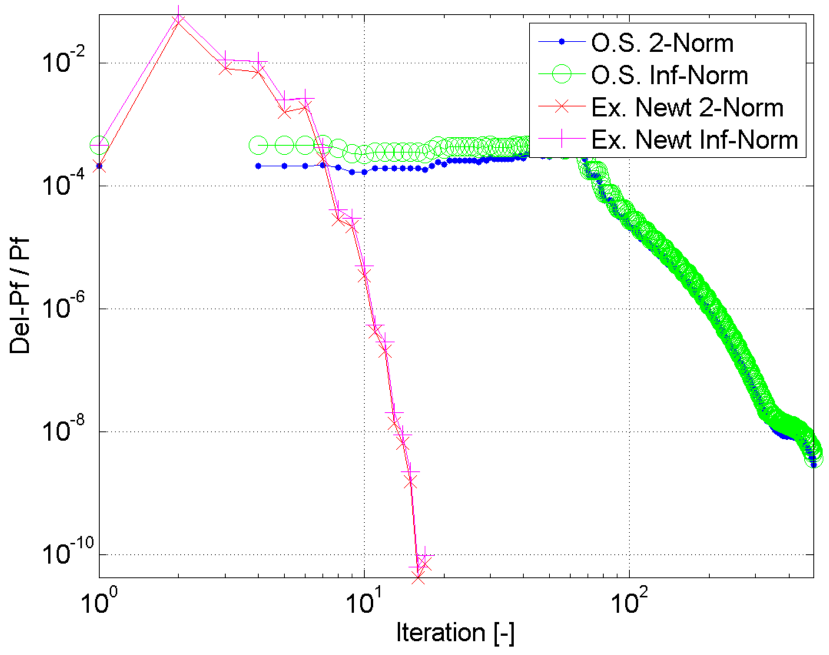

Figure 6.27: Coupled Exact Newton Del-Pf / Pf

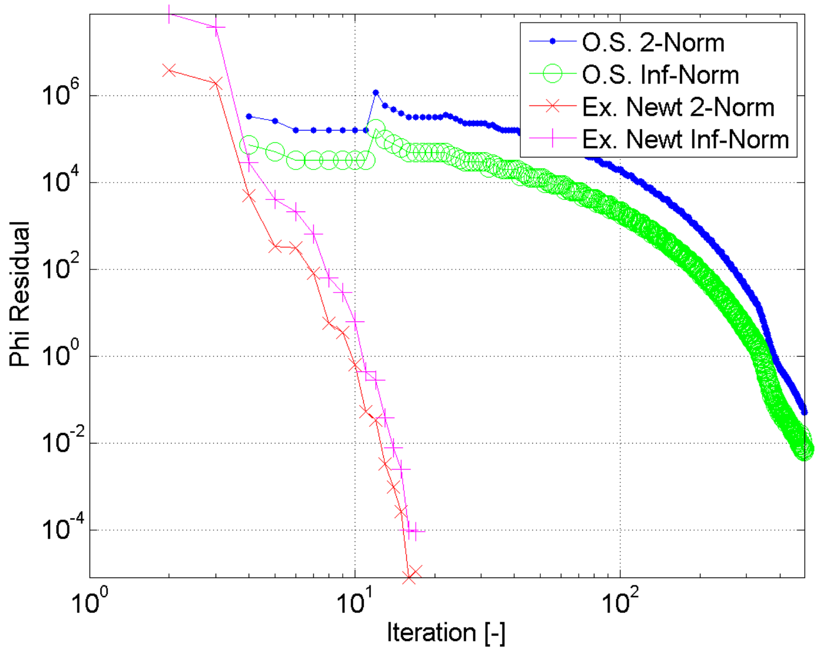

Figure 6.28: Coupled Exact Newton Residual Flux

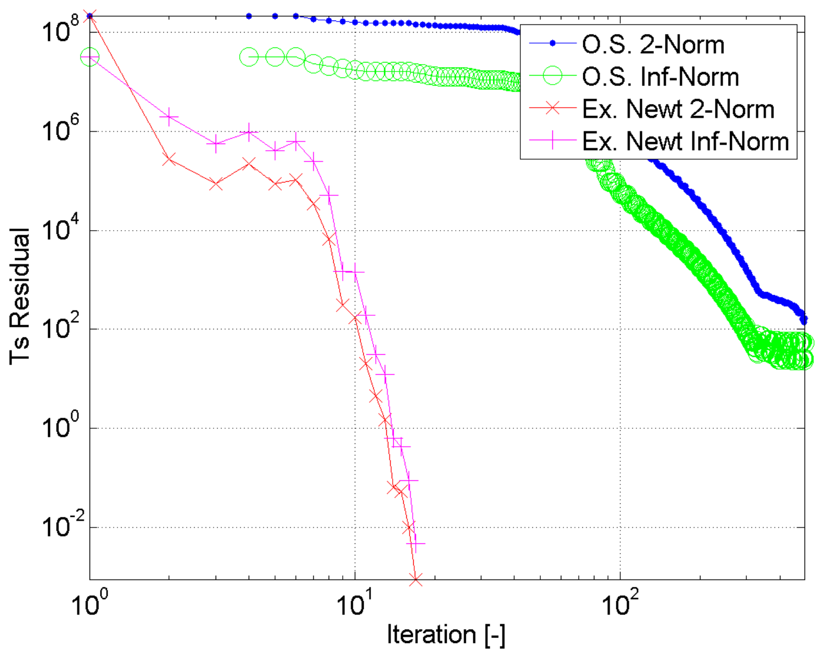

Figure 6.29: Coupled Exact Newton Residual Ts

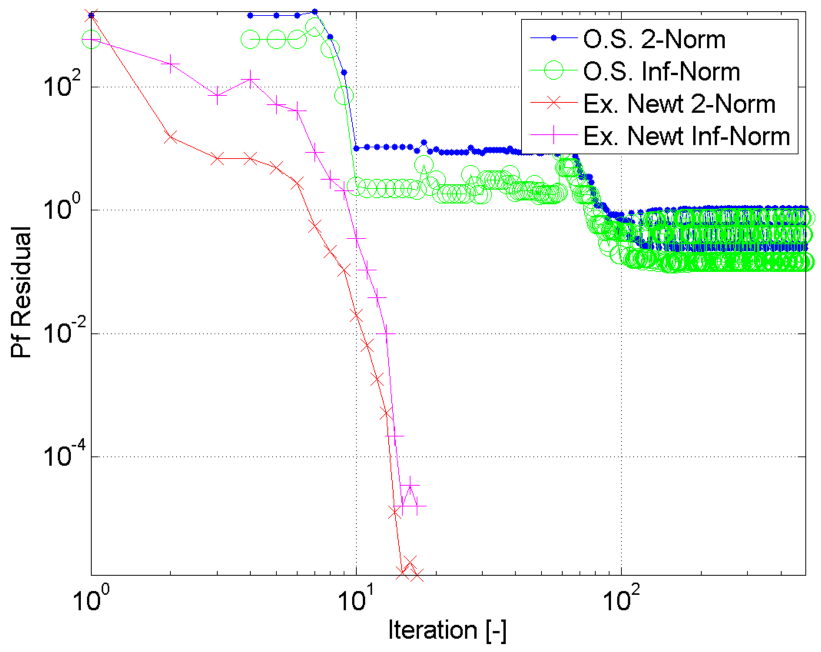

Figure 6.30: Coupled Exact Newton Residual Pf

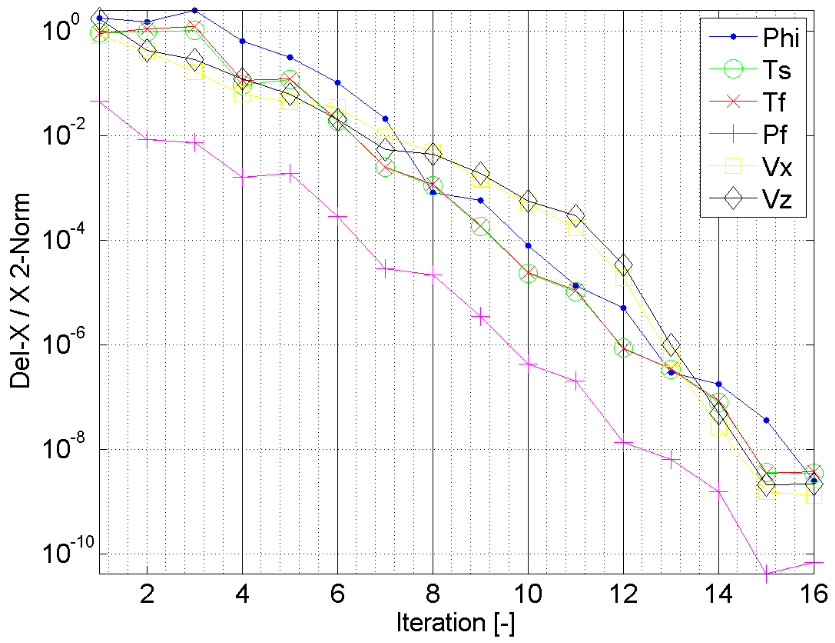

Figure 6.31: Coupled Exact Newton Del-X / X

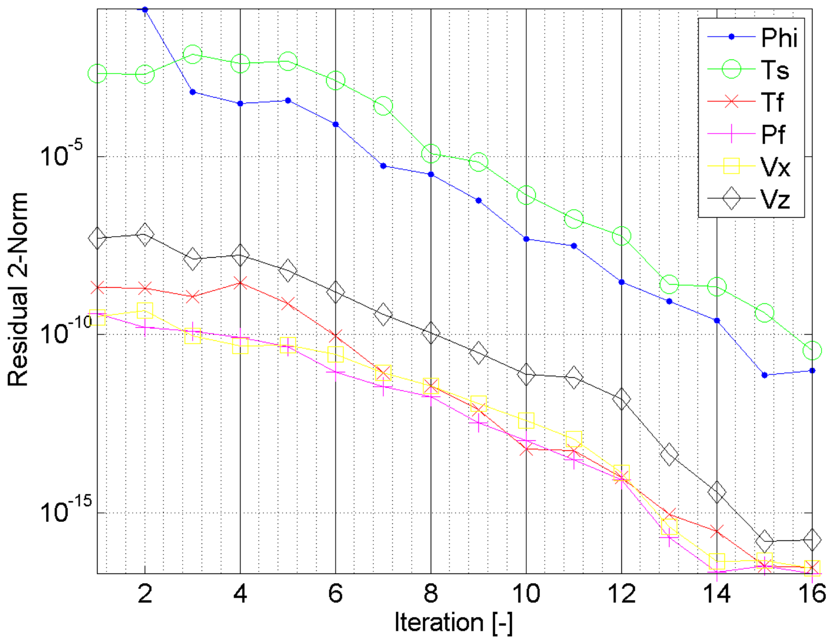

Figure 6.32: Coupled Exact Newton Residuals

6.3.2 Inexact Newton

The inexact Newtons method was then applied to the coupled field. As determined in the thermal-fluids analysis, the most efficient convergence strategy with GMRES was to use a dynamic tolerance for the inner iterations and this was used in the analysis of the coupled system. A comparison of the number of GMRES iterations for each Newton iteration for the fixed and dynamic tolerance is shown in Figure 6.33.

As dicussed in Section 5.3, the number of inner iterations for the coupled calculation considerably exceeds the breakeven to which the iterative solver would be competative with the direct solver. Therefore, the inexact Newton's method cannot be expected to perform better than the Exact Newton's method.

The convergence of the coupled field residuals are shown in the following Figures for the exact Newton method and inexact Newton with a fixed and dynamic tolerance. As indicated the fixed tolerance performance is similar to the Exact Newton method but the dynamic tolerance converges more slowly. However, as shown in Table 6.12, because of the reduced number of inner iterations the overall computational time of the dynamic tolerance is less than the fixed tolerance inexact Newton method.

Table 6.12: Coupled Inexact Newton Performance

Method	Exact Newton	Inexact Newton Tight Conv.	Inexact Newton Dynamic Conv.
Time [s]	14.508	48.030	49.829
Newton Itr	17	17	25
Flops / Nwt.	6.972e8	1.391e9	8.180e8

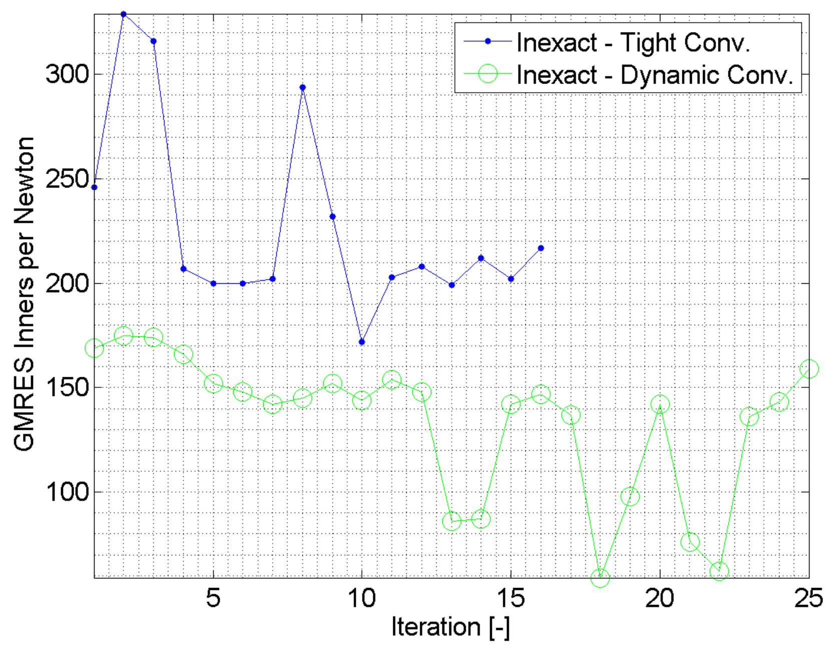

Figure 6.33: Coupled Inexact GMRES Conv.

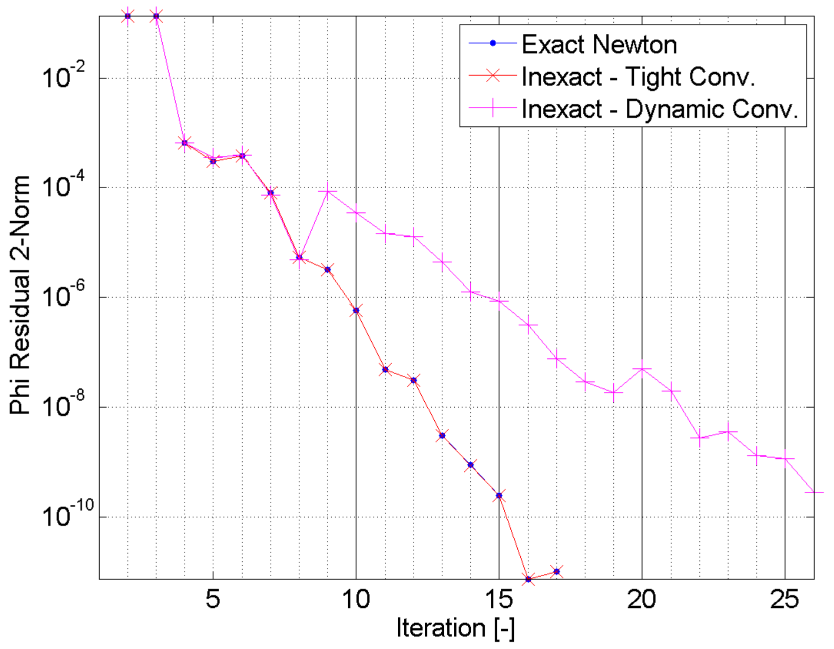

Figure 6.34: Coupled Inexact Newton Normalized Flux Residual

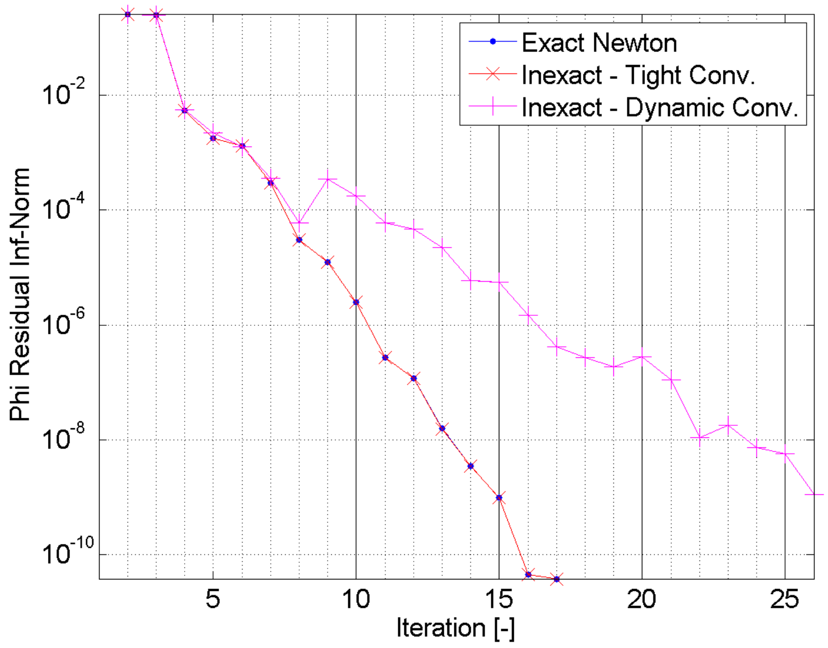

Figure 6.35: Coupled Inexact Newton Normalized Ts Residual

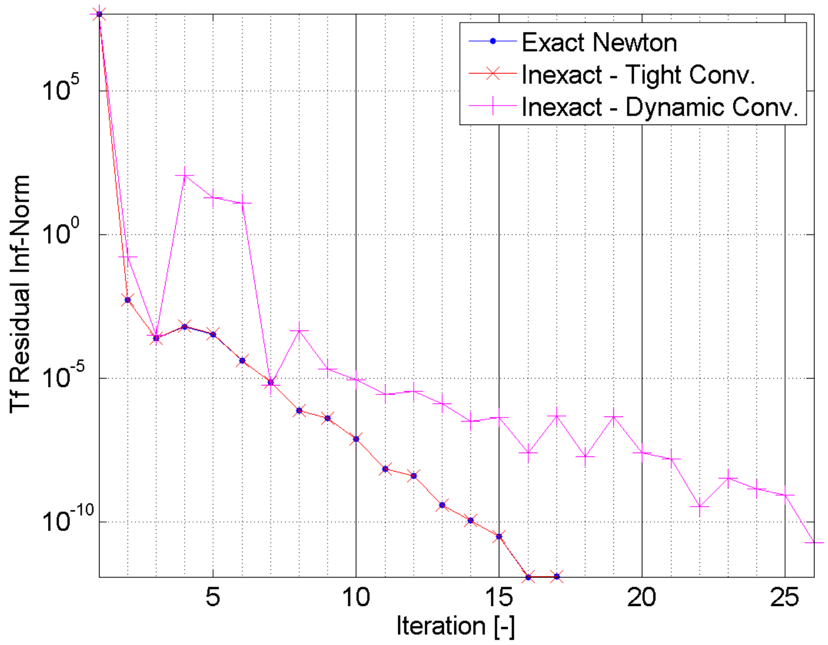

Figure 6.36: Coupled Inexact Newton Normalized Vz Residual

6.3.3 Finite Difference Jacobian

The final phase of the research was to investigate the behavior of the Finite Difference Jacobian, which can be particularly attractive because of the expense of forming analytic derivates when code methods change. The finite difference Jacobian was analyzed using both exact and inexact Newtons method. The residuals for the finite difference based Jacobian are given in Figure 6.37 through Figure 6.39. As can be observed, Newtons method with the analytical Jacobian converges more quickly than the finite difference Jacobian.

The finite difference Jacobian was then compared to the analytic Jacobian using the inexact Newton method. The results are shown in Figure 6.40 through Figure 6.42. As indicated the rate of convergence achieved with finite difference derivates is slightly less than the analytic derivates and in both cases the convergence is slower than the finite difference derivatives with the direct solver.

The computational performance is summarized in Table 6.13. As indicated the number of iterations of the finite difference Jacobian is higher than the Jacobian with analytic derivates for both the exact and inexact Newton method.

Table 6.13: Coupled F.D. Jac with Fixed Pert.

Method	Anl. Jacobian Direct Solve	Anl. Jacobian Dynam. Conv.	F.D. Jacobian Dynam. Conv.
Time [s]	14.508	49.829	46.101
Newton Itr	17	25	29
Flops / Nwt	6.972e8	8.180e8	7.484e8

The optimized perturbation size that was analyzed for the thermal-fluid solution was also tested for the coupled neutronics / thermal fluid solution. The results are shown in Table 6.14 for the Inexact Newton case and the optimized perturbation finite difference method was found to converge in fewer iterations than the finite difference with a fixed perturbation size. If the analytic Jacobian is not available, then the results here indicate that finite difference Jacobian with the optimized perturbation method would be an attractive option.

Table 6.14: Coupled F.D. Jac with Dynamic Pert.

Method	Anl. Jacobian	F.D. Jacobian Fixed Pert.	F.D. Jacobian Opt. Pert
Time [s]	49.829	46.101	39.794
Newton Itr	25	29	26
Flops / Nwt.	8.180e9	7.484e8	7.139e8

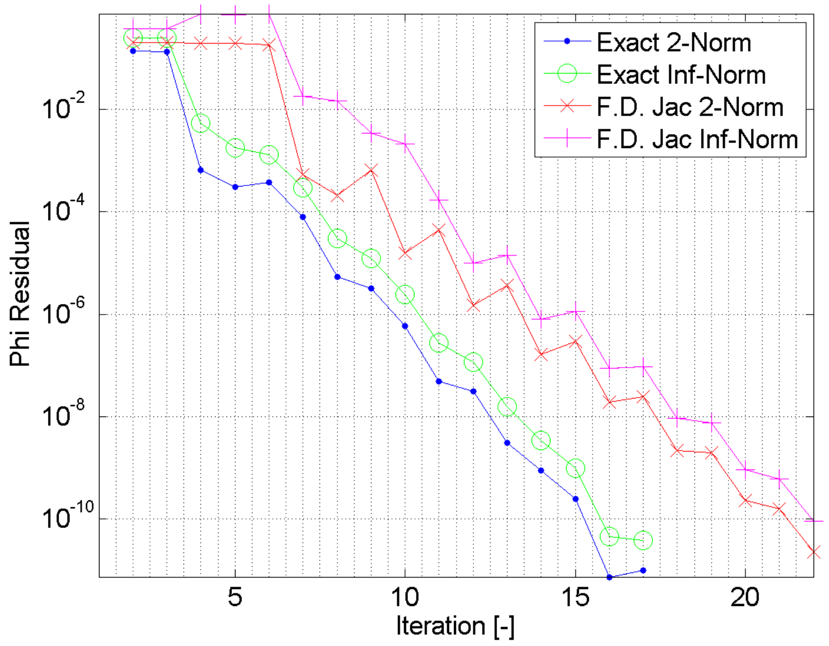

Figure 6.37: Coupled F.D. Jac Flux Residual

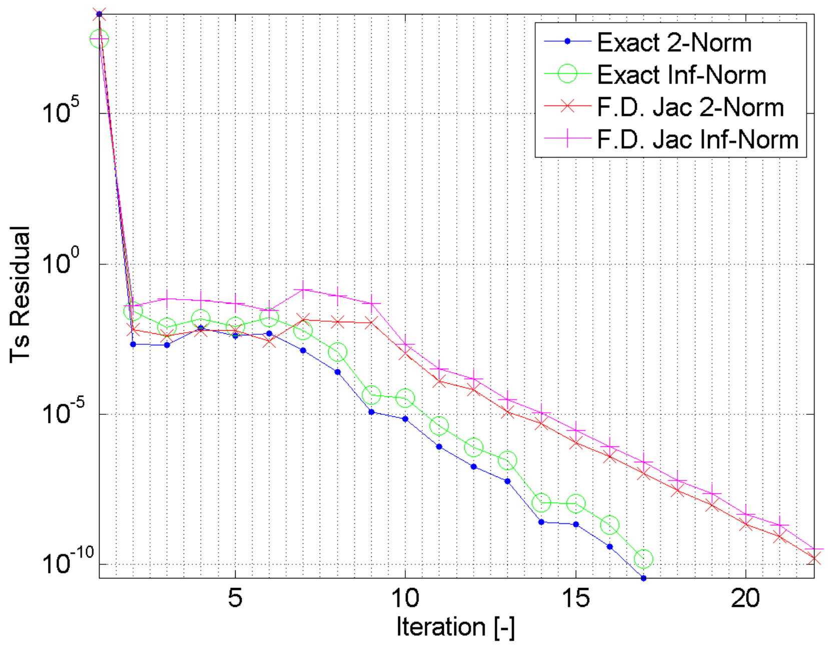

Figure 6.38: Coupled F.D. Jac Ts Residual

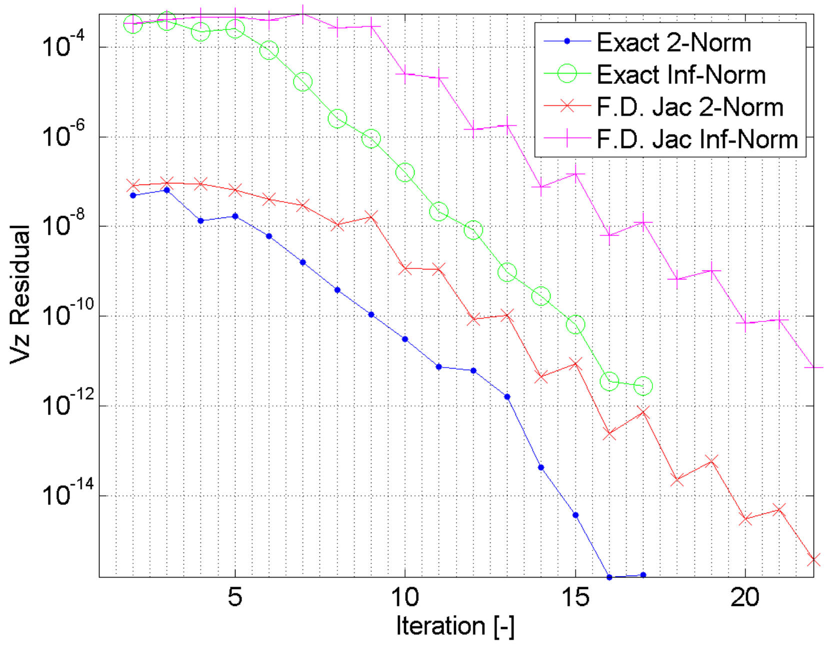

Figure 6.39: Coupeld F.D. Jac Vz Residual

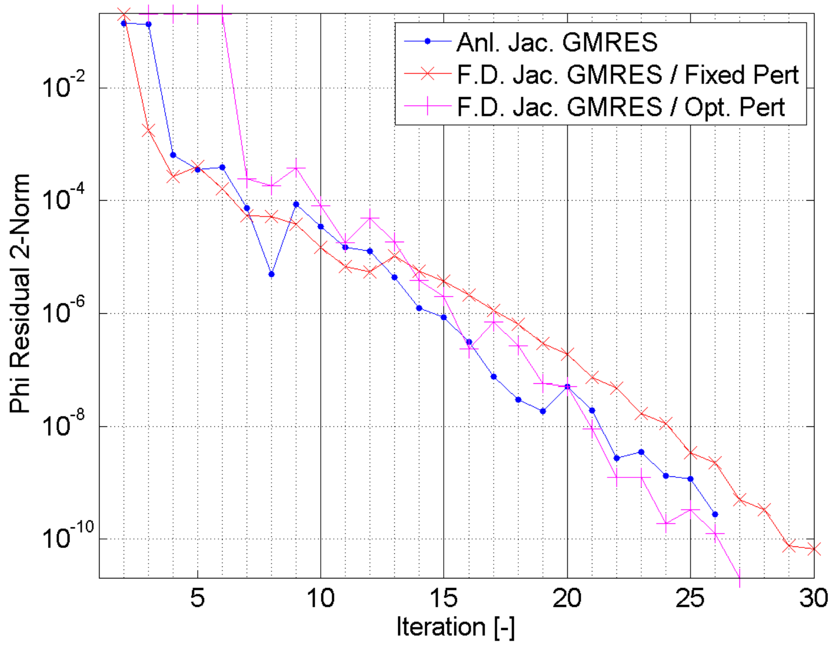

Figure 6.40: Coupled F.D. Jac Normalized Flux Residual

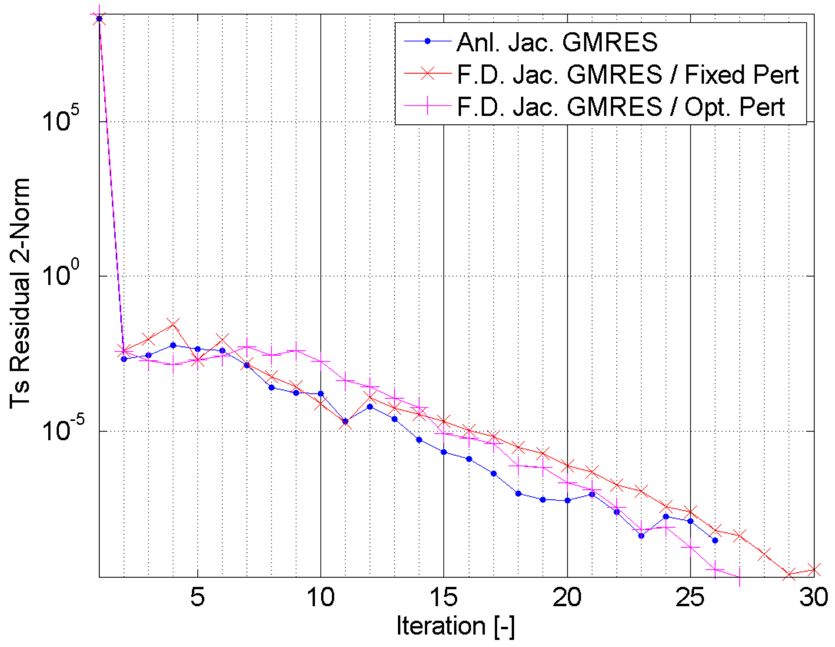

Figure 6.41: Coupled F.D. Jac Normalized Ts Residual

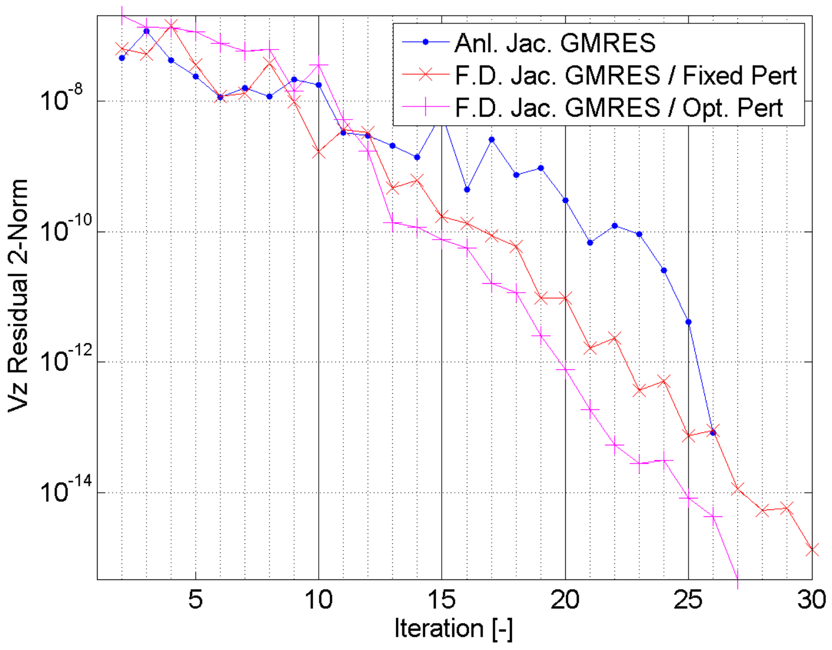

Figure 6.42: Coupled F.D. Jac Normalized Vz Residual

Chapter 7
Summary and Conclusion

7.1 Summary of Work

The solution of the coupled field equations for nuclear reactor analysis has typically been performed by solving separately the individual field equations and transferring information between fields. This has generally been referred to as operating splitting and has been successfully applied to a wide range of reactor steady-state and transient problems. Although this approach has generally been successful, it has been computationally inefficient and imposed some limitations on the range of problems considered. The research here investigated fully implicit methods which do not split the coupled field operators and which solves the coupled equations using Newton-Krylov methods.

The focus of the work here was on the solution of the coupled neutron and temperature/fluid field equations for the specific application to the high temperature gas reactor. However, the reserach here also investigated the application of the Newton's Method to the individual field equations. The solution of the neutron field equations was restricted to the steady-state multi-group neutron diffusion equations and the temperature fluid solution for the gas reactor involved only the single phase fluid which was adequate for the gas reactor. The results here indicate that steady state convergence of the coupled field equations can significantly improve both the stand alone neutronics and thermal fluid using Newton methods.

A large part of the improvement in the convergence in the neutronics was the inclusion of the eigenvalue as a primary variable in the Jacobian. This provided an acceleration in the eigenvalue search similar to the well know Wielandt shift method. Because the reference solution does not use any acceleration techniques, the comparison is not truly representative of the improved solution.

The thermal fluids convergence also improved considerably using a fully implicit Newtons method. The primary acceleration was due to the coupling of the solid and liquid energy equations. Because of the strong heat transfer coefficient coupling and the dependence of the heat transfer coefficient on the primary variables, the energy equation coupling improved both the rate and the stability of convergence. The implicit treatment of the pressure equation also contributed to the improvement in the convergence, but in general the pressure equation converged more rapidly than the energy equations.

Improvement in the coupled field solution was similar to the improvement observed in the convergence of the individual neutronics and thermal-fluids problems. Over an order of magnitude reduction was observed in the number of iterations required to achieve convergence. However, the overall computational time reduction for the Exact Newtons method was only about 50 %.

Another important conclusion from the research here was that the Inexact Newton's method did not outperform the Exact Newton's Method. Performance of the iterative method in the thermal fluids was nearly as fast as the direct solution, but the iterative method was much slower in the coupled solution. A detailed analytis of the floating point operation count showed that in all cases, the number of inner iterations required of the iterative method exceeded the breakeven for which an iterative method could be expected to outperform the direct solver.

7.2 Future Work

Based on the results and analysis in the work here, it is recommended that the next phase of the research should focus on approximate solutions to the exact Jacobian which do not require the formation of all the elements of the Jacobian. One of the most promising approaches would be to investigate the Approximate Block Newton (ABN) [88] method which appears to achieve a suitable balance between the expense of forming the Jacobian and an improvement in the rate of convergence. The ABN method also has been shown to provide improved performance without considerable code changes which is an important consideration codes undergo improvements in methods. Further research should include transient problems which can have the most significant impact on the computational time for practical HTR safety analysis.

Finally, another potential area of research is the application of these methods to uncertainty quantification for both the steady state and transient problems. Research suggests that the analytical Jacobian may not be neccesary for accurate determination of uncertainty and sensitivities. The availiability of both the analytical and finite-difference based Jacobian permit analysis the error introduced by approximating the Jacobian in both the thermal-fluids and coupled models.

Appendices

Appendix A
Additional Analysis of Krylov Solvers

Much of the focus of this research has been on GMRES as the Krylov iterative solver. Research has shown that the performance of Krylov methods can vary widely depending on the type of problem. For this reason, the behavior of the Bi - Conjugate Gradient Stabilized method was analyzed. BiCGStab is used throughout reactor analysis and have been shown to outperform GMRES is some cases.

The thermal fluids analytical Jacobian was chosen as the test case. Although this case is smaller than the coupled case, the comparison should provide some insight into the coupled performance as well. Because the two approaches are very different, and were implemented differently, the comparison must be a mixture of quantitative and qualitative analysis. The GMRES solver used above is an INTEL/MKL implementation, and is well optimized. The BiCGStab solver is a mix of user coding and mkl routines to peform the matrix-vector multiplication and application of the preconditioner.

The first comparison is of the the inner iteration per Newton step. In order to see a substantial improvement in performance, BiCGStab must solve the problem with significantly less inners. Figure A.1 illustrates the differences. As can be seen, the two solvers require similar numbers of inners iterations. With the same number of Newton steps, the performances should be similar

Table A.1: Krylov Solver Performance

Krylov Method	GMRES	BiCGStab
Time [s]	1.167	1.554
Newton Itr	7	7
Inner Itr	451	398
Inners / Newton	64	57

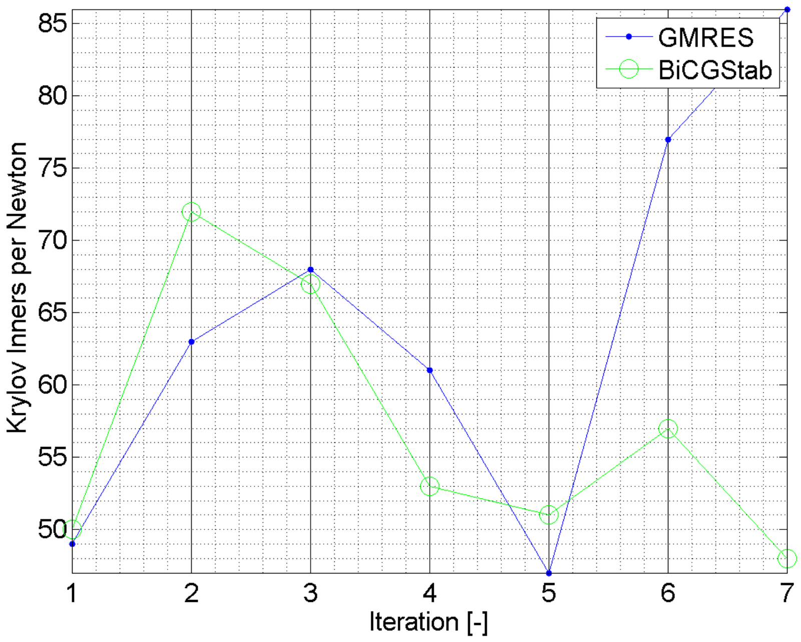

Figure A.1: Krylov Solvers Inners per Newton Step

As noted above, the similar number of inner iteration suggests the overal performance should be similar. Because the GMRES solver is likely more efficient, it may well outperform the BiCGStab solver. The performance is given in Table A.1, which indicates that GMRES is slightly faster than BiCGStab. Again, with bigger problems, one solver might perform much better than the other, and that is outside the scope of the research performed here. Analysis of the work performed by GMRES seems to suggest that the Krylov methods favor dense residual vectors. Any increase in the dimension, N, or stored subspace, K, dramatically increases the orthgonalization work of the iterative method. This puts it at a disadvantage compared to the direct solution methods. Futher development will work to improve storage of the coefficient matrix and residual vector. This should improve the performance of GMRES dramatically.

Appendix B
Additional Analysis of Preconditioners

As noted above, the incomplete LU factorization, without fill-in (ILU0), was used to precondition the GMRES solver. This approach is widely used and has been implemented in Neutronics solvers and thermal fluids codes throughout the nuclear industry. Research indicates that the preconditioner can have a substantial impact on the convergence of the Krylov solver. The state-of-practice in neutronics is to perform the ILU decomposition on a matrix similar to A. The decomposed matrix, M, is generally a block-wise version of A, with the coupling terms between subfields eliminated.

The decomposition was actually performed on the full Jacobian matrix. In order to gain more understanding of the preconditioner impact, some other approaches were analyzed. Following the conventional approach, the decomposed matrix was reduced from the Jacobian to a block-wise version of the Jacobian. The decomposed matrix was further reduced to the original components of A. This was approach used in the Jacobian-Free solver, as the partial derivatives that compose part of J were assumed to not exist.

The analytical Jacobian from the thermal-fluids was used as the test case. A good measure of the impact of the preconditioner is the number of inner iteration required by GMRES. This is given in Figure B.1. The number of inners and Newton steps decreased when the preconditioned block-A matrix is augmented with some partial derivatives. The improvement in performance is much smaller when the Block-wise Jacobian is expanded to be the full Jacobian.

The timings are given in Table B.1. The block-wise Jacobian is nearly as effective a preconditioner as the complete Jacobian. The larger matrix used in the coupled solution may benefit from a more diagonal preconditioner like the block-wise Jacobian.

Another preconditioner was the incomplete LU factorization, with threshold fill-in

Table B.1: ILU0 Decomposed Matrix Performance

Matrix	Jacobian	Block-wise Jac.	Block-wise A
Time [s]	1.167	1.526	1.663
Newton Itr	7	8	8
GMRES Inners	451	1076	1333
Inners / Newton	64	135	167

(ILUT). This was tested, and was found to underperform when compared to the ILU0 preconditioning. The behavior is not shown, but the ILUT has two problems. The first is difficulty in reducing the residual for the first few iterations. It appears it is more challenging for the ILUT to enter the converence region, where Newton's method performs the best. Secondly, the solution appears to stagnate just above pratical convergence of the primary variables, $\Delta X/X \approx 10^{-6}$.

These results suggest that the ILU0 preconditioner is adequate for the research here, but further research and development are needed to improve the performance of the Krylov methods. Problem specific preconditioning has received significant attention, and these may have value for the Newton's method class of approaches in reactor analysis.

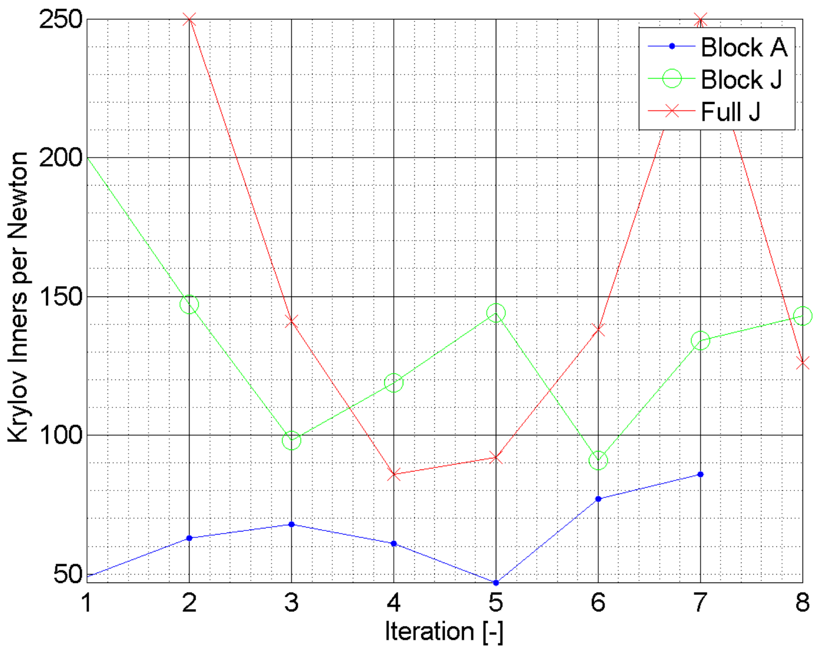

Figure B.1: ILU0 Inner Iterations per Newton Step

Appendix C

Derivation: Solid Energy Equation

Solid Energy Equation Derivation for Implicit/Newton Iteration

$$\begin{aligned}\frac{\partial}{\partial t}\left[(1-\varepsilon)\rho_s c_{p_s} T_s\right]\Delta V & \\ &= D_e T_{s,E} + D_w T_{s,W} + D_n T_{s,N} + D_s T_{s,S} + D_t T_{s,T} + D_b T_{s,B} \\ &- (D_e + D_w + D_n + D_s + D_t + D_b) T_{s,P} - \alpha (T_{s,P} - T_{f,P})\Delta V + Q\Delta V \end{aligned} \tag{C.1}$$

Coefficients

$$\begin{aligned} D_e &= (1-\varepsilon_e) k_{s_e} \tfrac{r_e}{\partial r_e} \Delta\theta\Delta z & D_w &= (1-\varepsilon_w) k_{s_w} \tfrac{r_w}{\partial r_w} \Delta\theta\Delta z \\ D_n &= (1-\varepsilon_n) k_{s_n} \tfrac{1}{r_n} \tfrac{1}{\partial\theta_n} \Delta r\Delta z & D_s &= (1-\varepsilon_s) k_{s_s} \tfrac{1}{r_s} \tfrac{1}{\partial\theta_s} \Delta r\Delta z \\ D_t &= (1-\varepsilon_t) k_{s_t} \tfrac{1}{\partial z_t} \tfrac{r_e^2 - r_w^2}{2} \Delta\theta & D_b &= (1-\varepsilon_b) k_{s_b} \tfrac{1}{\partial z_b} \tfrac{r_e^2 - r_w^2}{2} \Delta\theta \end{aligned} \tag{C.2}$$

Using a simplifying coefficient

$$\begin{aligned} K_e &= (1-\varepsilon_e) \tfrac{r_e}{\partial r_e} \Delta\theta\Delta z & K_w &= (1-\varepsilon_w) \tfrac{r_w}{\partial r_w} \Delta\theta\Delta z \\ K_n &= (1-\varepsilon_n) \tfrac{1}{r_n} \tfrac{1}{\partial\theta_n} \Delta r\Delta z & K_s &= (1-\varepsilon_s) \tfrac{1}{r_s} \tfrac{1}{\partial\theta_s} \Delta r\Delta z \\ K_t &= (1-\varepsilon_t) \tfrac{1}{\partial z_t} \tfrac{r_e^2 - r_w^2}{2} \Delta\theta & K_b &= (1-\varepsilon_b) \tfrac{1}{\partial z_b} \tfrac{r_e^2 - r_w^2}{2} \Delta\theta \end{aligned} \tag{C.3}$$

The revised coefficients are then given as,

$$\begin{aligned} D_E &= K_E\left(k_{s,E}^{(k)} + \delta k_{s,E}\right) & D_W &= K_W\left(k_{s,W}^{(k)} + \delta k_{s,W}\right) \\ D_N &= K_N\left(k_{s,N}^{(k)} + \delta k_{s,N}\right) & D_S &= K_S\left(k_{s,S}^{(k)} + \delta k_{s,S}\right) \\ D_T &= K_T\left(k_{s,T}^{(k)} + \delta k_{s,T}\right) & D_B &= K_B\left(k_{s,B}^{(k)} + \delta k_{s,B}\right) \\ \alpha &= \alpha^{(k)} + \delta\alpha \end{aligned} \tag{C.4}$$

The primary variables are then given as

$$\begin{aligned} T_{s,E} &= T_{s,E}^{(k)} + \delta T_{s,E} \quad & T_{s,W} &= T_{s,W}^{(k)} + \delta T_{s,W} \\ T_{s,N} &= T_{s,N}^{(k)} + \delta T_{s,N} \quad & T_{s,S} &= T_{s,S}^{(k)} + \delta T_{s,S} \\ T_{s,T} &= T_{s,T}^{(k)} + \delta T_{s,T} \quad & T_{s,B} &= T_{s,B}^{(k)} + \delta T_{s,B} \\ T_{s,P} &= T_{s,P}^{(k)} + \delta T_{s,P} \quad & T_{f,P} &= T_{f,P}^{(k)} + \delta T_{f,P} \end{aligned} \tag{C.5}$$

The new equation is then given as,

$$\begin{aligned} K_E \left(k_{s,E}^{(k)} + \delta k_{s,E}\right)\left(T_{s,E}^{(k)} + \delta T_{s,E}\right) + K_W \left(k_{s,W}^{(k)} + \delta k_{s,W}\right)\left(T_{s,W}^{(k)} + \delta T_{s,W}\right) \\ + K_N \left(k_{s,N}^{(k)} + \delta k_{s,N}\right)\left(T_{s,N}^{(k)} + \delta T_{s,N}\right) + K_S \left(k_{s,S}^{(k)} + \delta k_{s,S}\right)\left(T_{s,S}^{(k)} + \delta T_{s,S}\right) \\ + K_T \left(k_{s,T}^{(k)} + \delta k_{s,T}\right)\left(T_{s,T}^{(k)} + \delta T_{s,T}\right) + K_B \left(k_{s,B}^{(k)} + \delta k_{s,B}\right)\left(T_{s,B}^{(k)} + \delta T_{s,B}\right) \\ - K_E \left(k_{s,E}^{(k)} + \delta k_{s,E}\right)\left(T_{s,P}^{(k)} + \delta T_{s,P}\right) - K_W \left(k_{s,W}^{(k)} + \delta k_{s,W}\right)\left(T_{s,P}^{(k)} + \delta T_{s,P}\right) \\ - K_N \left(k_{s,N}^{(k)} + \delta k_{s,N}\right)\left(T_{s,P}^{(k)} + \delta T_{s,P}\right) - K_S \left(k_{s,S}^{(k)} + \delta k_{s,S}\right)\left(T_{s,P}^{(k)} + \delta T_{s,P}\right) \\ - K_T \left(k_{s,T}^{(k)} + \delta k_{s,T}\right)\left(T_{s,P}^{(k)} + \delta T_{s,P}\right) - K_B \left(k_{s,B}^{(k)} + \delta k_{s,B}\right)\left(T_{s,P}^{(k)} + \delta T_{s,P}\right) \\ - \left(\alpha^{(k)} + \delta\alpha\right)\left[\left(T_{s,P}^{(k)} + \delta T_{s,P}\right) - \left(T_{f,P}^{(k)} + \delta T_{f,P}\right)\right]\Delta V + Q\Delta V = 0 \end{aligned} \tag{C.6}$$

Getting rid of the higher order terms and simplifying

$$\begin{aligned} K_E \left(k_{s,E}^{(k)} T_{s,E}^{(k)} + \delta k_{s,E} T_{s,E}^{(k)} + k_{s,E}^{(k)} \delta T_{s,E}\right) + K_W \left(k_{s,W}^{(k)} T_{s,W}^{(k)} + \delta k_{s,W} T_{s,W}^{(k)} + k_{s,W}^{(k)} \delta T_{s,W}\right) \\ + K_N \left(k_{s,N}^{(k)} T_{s,N}^{(k)} + \delta k_{s,N} T_{s,N}^{(k)} + k_{s,N}^{(k)} \delta T_{s,N}\right) + K_S \left(k_{s,S}^{(k)} T_{s,S}^{(k)} + \delta k_{s,S} T_{s,S}^{(k)} + k_{s,S}^{(k)} \delta T_{s,S}\right) \\ + K_T \left(k_{s,T}^{(k)} T_{s,T}^{(k)} + \delta k_{s,T} T_{s,T}^{(k)} + k_{s,T}^{(k)} \delta T_{s,T}\right) + K_B \left(k_{s,B}^{(k)} T_{s,B}^{(k)} + \delta k_{s,B} T_{s,B}^{(k)} + k_{s,B}^{(k)} \delta T_{s,B}\right) \\ - K_E \left(k_{s,E}^{(k)} T_{s,P}^{(k)} + \delta k_{s,E} T_{s,P}^{(k)} + k_{s,E}^{(k)} \delta T_{s,P}\right) - K_W \left(k_{s,W}^{(k)} T_{s,P}^{(k)} + \delta k_{s,W} T_{s,P}^{(k)} + k_{s,W}^{(k)} \delta T_{s,P}\right) \\ - K_N \left(k_{s,N}^{(k)} T_{s,P}^{(k)} + \delta k_{s,N} T_{s,P}^{(k)} + k_{s,N}^{(k)} \delta T_{s,P}\right) - K_S \left(k_{s,S}^{(k)} T_{s,P}^{(k)} + \delta k_{s,S} T_{s,P}^{(k)} + k_{s,S}^{(k)} \delta T_{s,P}\right) \\ - K_T \left(k_{s,T}^{(k)} T_{s,P}^{(k)} + \delta k_{s,T} T_{s,P}^{(k)} + k_{s,T}^{(k)} \delta T_{s,P}\right) - K_B \left(k_{s,B}^{(k)} T_{s,P}^{(k)} + \delta k_{s,B} T_{s,P}^{(k)} + k_{s,B}^{(k)} \delta T_{s,P}\right) \\ - \left(\alpha^{(k)} T_{s,P}^{(k)} + \delta\alpha T_{s,P}^{(k)} + \alpha^{(k)} \delta T_{s,P}\right)\Delta V + \left(\alpha^{(k)} T_{f,P}^{(k)} + \delta\alpha T_{f,P}^{(k)} + \alpha^{(k)} \delta T_{f,P}\right)\Delta V \\ + Q\Delta V = 0 \end{aligned} \tag{C.7}$$

Further simplifications,

$$\begin{aligned}
&\left(K_E\delta k_{s,E}T_{s,E}^{(k)}+\ldots+K_B\delta k_{s,B}T_{s,B}^{(k)}\right)\\
&\quad-\left(K_E\delta k_{s,E}+K_W\delta k_{s,W}+K_N\delta k_{s,N}+K_S\delta k_{s,S}+K_T\delta k_{s,T}+K_B\delta k_{s,B}\right)T_{s,P}^{(k)}\\
&\quad-\Delta V\delta\alpha T_{s,P}^{(k)}+\Delta V\delta\alpha T_{f,P}^{(k)}\\
&\quad+\left(K_Ek_{s,E}^{(k)}\delta T_{s,E}+\ldots+K_Bk_{s,B}^{(k)}\delta T_{s,B}\right)\\
&\quad-\left(K_Ek_{s,E}^{(k)}+K_Wk_{s,W}^{(k)}+K_Nk_{s,N}^{(k)}+K_Sk_{s,S}^{(k)}+K_Tk_{s,T}^{(k)}+K_Bk_{s,B}^{(k)}\right)\delta T_{s,P}\\
&\quad-\Delta V\alpha^{(k)}\delta T_{s,P}+\Delta V\alpha^{(k)}\delta T_{f,P}\\
&\quad+\left(K_Ek_{s,E}^{(k)}T_{s,E}^{(k)}+\ldots+K_Bk_{s,B}^{(k)}T_{s,B}^{(k)}\right)\\
&\quad-\left(K_Ek_{s,E}^{(k)}+K_Wk_{s,W}^{(k)}+K_Nk_{s,N}^{(k)}+K_Sk_{s,S}^{(k)}+K_Tk_{s,T}^{(k)}+K_Bk_{s,B}^{(k)}\right)T_{s,P}^{(k)}\\
&\quad-\Delta V\alpha^{(k)}T_{s,P}^{(k)}+\Delta V\alpha^{(k)}T_{f,P}^{(k)}=0
\end{aligned}\tag{C.8}$$

Addition of the flux source term,

$$\begin{aligned}
Q\Delta V&=\bar{P}_{n,fuel}\left(\kappa\Sigma_1\phi_1+\kappa\Sigma_2\phi_2+\kappa\Sigma_3\phi_3\right)\Delta V\\
\kappa\Sigma\phi&=\left(\kappa\Sigma^{(k)}+\delta\kappa\Sigma\right)\left(\phi^{(k)}+\delta\phi\right)=\left(\kappa\Sigma^{(k)}\phi^{(k)}+\delta\kappa\Sigma\phi^{(k)}+\kappa\Sigma^{(k)}\delta\phi\right)
\end{aligned}\tag{C.9}$$

Inserting into the equation

$$
\begin{aligned}
&- \Delta V \bar{P}_{n,fuel} \left(\kappa \Sigma_1^{(k)} \phi_1^{(k)} + \delta\kappa \Sigma_1 \phi_1^{(k)} + \kappa \Sigma_1^{(k)} \delta\phi_1 \right) \\
&\qquad - \Delta V \bar{P}_{n,fuel} \left(\kappa \Sigma_2^{(k)} \phi_2^{(k)} + \delta\kappa \Sigma_2 \phi_2^{(k)} + \kappa \Sigma_2^{(k)} \delta\phi_2 \right) \\
&\qquad - \Delta V \bar{P}_{n,fuel} \left(\kappa \Sigma_3^{(k)} \phi_3^{(k)} + \delta\kappa \Sigma_3 \phi_3^{(k)} + \kappa \Sigma_3^{(k)} \delta\phi_3 \right) \\
&\qquad = \left(K_E \delta k_{s,E} T_{s,E}^{(k)} + \ldots + K_B \delta k_{s,B} T_{s,B}^{(k)} \right) \\
&- \left(K_E \delta k_{s,E} + K_W \delta k_{s,W} + K_N \delta k_{s,N} + K_S \delta k_{s,S} + K_T \delta k_{s,T} + K_B \delta k_{s,B} \right) T_{s,P}^{(k)} \\
&\qquad - \Delta V \delta\alpha T_{s,P}^{(k)} + \Delta V \delta\alpha T_{f,P}^{(k)} \\
&\qquad + \left(K_E k_{s,E}^{(k)} \delta T_{s,E} + \ldots + K_B k_{s,B}^{(k)} \delta T_{s,B} \right) \\
&- \left(K_E k_{s,E}^{(k)} + K_W k_{s,W}^{(k)} + K_N k_{s,N}^{(k)} + K_S k_{s,S}^{(k)} + K_T k_{s,T}^{(k)} + K_B k_{s,B}^{(k)} \right) \delta T_{s,P} \\
&\qquad - \Delta V \alpha^{(k)} \delta T_{s,P} + \Delta V \alpha^{(k)} \delta T_{f,P} \\
&\qquad + \left(K_E k_{s,E}^{(k)} T_{s,E}^{(k)} + \ldots + K_B k_{s,B}^{(k)} T_{s,B}^{(k)} \right) \\
&- \left(K_E k_{s,E}^{(k)} + K_W k_{s,W}^{(k)} + K_N k_{s,N}^{(k)} + K_S k_{s,S}^{(k)} + K_T k_{s,T}^{(k)} + K_B k_{s,B}^{(k)} \right) T_{s,P}^{(k)} \\
&\qquad - \Delta V \alpha^{(k)} T_{s,P}^{(k)} + \Delta V \alpha^{(k)} T_{f,P}^{(k)}
\end{aligned}
\tag{C.10}
$$

Collecting like terms for organization,

$$
\begin{aligned}
&\left(K_E\delta k_{s,E}T_{s,E}^{(k)}+\ldots+K_B\delta k_{s,B}T_{s,B}^{(k)}\right)\\
&\quad-\left(K_E\delta k_{s,E}+K_W\delta k_{s,W}+K_N\delta k_{s,N}+K_S\delta k_{s,S}+K_T\delta k_{s,T}+K_B\delta k_{s,B}\right)T_{s,P}^{(k)}\\
&\qquad-\Delta V\delta\alpha T_{s,P}^{(k)}+\Delta V\delta\alpha T_{f,P}^{(k)}-\\
&\qquad+\Delta V\bar{P}_{n,fuel}\left(\delta\kappa\Sigma_1\phi_1^{(k)}+\delta\kappa\Sigma_2\phi_2^{(k)}+\delta\kappa\Sigma_3\phi_3^{(k)}\right)\\
&\qquad+\left(K_Ek_{s,E}^{(k)}\delta T_{s,E}+\ldots+K_Bk_{s,B}^{(k)}\delta T_{s,B}\right)\\
&\quad-\left(K_Ek_{s,E}^{(k)}+K_Wk_{s,W}^{(k)}+K_Nk_{s,N}^{(k)}+K_Sk_{s,S}^{(k)}+K_Tk_{s,T}^{(k)}+K_Bk_{s,B}^{(k)}\right)\delta T_{s,P}\\
&\qquad-\Delta V\alpha^{(k)}\delta T_{s,P}+\Delta V\alpha^{(k)}\delta T_{f,P}\\
&\qquad+\Delta V\bar{P}_{n,fuel}\left(\kappa\Sigma_1^{(k)}\delta\phi+\kappa\Sigma_2^{(k)}\delta\phi_2+\kappa\Sigma_3^{(k)}\delta\phi_3\right)\\
&\qquad+\left(K_Ek_{s,E}^{(k)}T_{s,E}^{(k)}+\ldots+K_Bk_{s,B}^{(k)}T_{s,B}^{(k)}\right)\\
&\quad-\left(K_Ek_{s,E}^{(k)}+K_Wk_{s,W}^{(k)}+K_Nk_{s,N}^{(k)}+K_Sk_{s,S}^{(k)}+K_Tk_{s,T}^{(k)}+K_Bk_{s,B}^{(k)}\right)T_{s,P}^{(k)}\\
&\qquad-\alpha^{(k)}T_{s,P}^{(k)}\Delta V+\alpha^{(k)}T_{f,P}^{(k)}\Delta V\\
&\qquad+\Delta V\bar{P}_{n,fuel}\left(\kappa\Sigma_1^{(k)}\phi_1^{(k)}+\kappa\Sigma_2^{(k)}\phi_2^{(k)}+\kappa\Sigma_3^{(k)}\phi_3^{(k)}\right)=0
\end{aligned}
\tag{C.11}
$$

To simplify, the following constants,

$$
\begin{aligned}
S^{(k)}=&\left(K_Ek_{s,E}^{(k)}T_{s,E}^{(k)}+\ldots+K_Bk_{s,B}^{(k)}T_{s,B}^{(k)}\right)\\
&-\left(K_Ek_{s,E}^{(k)}+K_Wk_{s,W}^{(k)}+K_Nk_{s,N}^{(k)}+K_Sk_{s,S}^{(k)}+K_Tk_{s,T}^{(k)}+K_Bk_{s,B}^{(k)}\right)T_{s,P}^{(k)}\\
&\qquad-\alpha^{(k)}\Delta VT_{s,P}^{(k)}+\alpha^{(k)}\Delta VT_{f,P}^{(k)}\\
&\qquad+\Delta V\bar{P}_{n,fuel}\left(\kappa\Sigma_1^{(k)}\phi_1^{(k)}+\kappa\Sigma_2^{(k)}\phi_2^{(k)}+\kappa\Sigma_3^{(k)}\phi_3^{(k)}\right)
\end{aligned}
\tag{C.12}
$$

Updated Equation, moving the non-source to the LHS

$$
\begin{aligned}
&- \left(K_E \delta k_{s,E} T_{s,E}^{(k)} + \ldots + K_B \delta k_{s,B} T_{s,B}^{(k)} \right) \\
&\quad + \left(K_E \delta k_{s,E} + K_W \delta k_{s,W} + K_N \delta k_{s,N} + K_S \delta k_{s,S} + K_T \delta k_{s,T} + K_B \delta k_{s,B} \right) T_{s,P}^{(k)} \\
&\quad + \Delta V \delta\alpha T_{s,P}^{(k)} - \Delta V \delta\alpha T_{f,P}^{(k)} \\
&\quad - \Delta V \bar{P}_{n,fuel} \left(\delta\kappa \Sigma_1 \phi_1^{(k)} + \delta\kappa \Sigma_2 \phi_2^{(k)} + \delta\kappa \Sigma_3 \phi_3^{(k)} \right) \\
&\quad - \left(K_E k_{s,E}^{(k)} \delta T_{s,E} + \ldots + K_B k_{s,B}^{(k)} \delta T_{s,B} \right) \\
&\quad + \left(K_E k_{s,E}^{(k)} + K_W k_{s,W}^{(k)} + K_N k_{s,N}^{(k)} + K_S k_{s,S}^{(k)} + K_T k_{s,T}^{(k)} + K_B k_{s,B}^{(k)} \right) \delta T_{s,P} \\
&\quad + \Delta V \alpha^{(k)} \delta T_{s,P} - \Delta V \alpha^{(k)} \delta T_{f,P} \\
&\quad - \Delta V \bar{P}_{n,fuel} \left(\kappa \Sigma_1^{(k)} \delta\phi + \kappa \Sigma_2^{(k)} \delta\phi_2 + \kappa \Sigma_3^{(k)} \delta\phi_3 \right) = S^{(k)}
\end{aligned} \tag{C.13}
$$

The next step is to the delta-coefficients. The fuel coefficients are relatively simple.

$$
\begin{aligned}
&\alpha = f\left(T_s, T_f, P, v_x, v_y, v_z\right) \\
&\qquad \delta\alpha = \left(\frac{\partial \alpha}{\partial T_s}\right)^{n-1} \delta T_s + \ldots + \left(\frac{\partial \alpha}{\partial v_z}\right)^{n-1} \delta v_z \\
&\qquad k_s = f\left(T_s\right) \quad \delta k_s = \left(\tfrac{\partial k_s}{\partial T_s}\right)^{n-1} \delta T_s
\end{aligned} \tag{C.14}
$$

Reorganizing again

$$
\begin{aligned}
&K_E\left(T_{s,P}^{(k)}-T_{s,E}^{(k)}\right)\delta k_{s,E}+K_W\left(T_{s,P}^{(k)}-T_{s,W}^{(k)}\right)\delta k_{s,W}\\
&\quad+K_N\left(T_{s,P}^{(k)}-T_{s,N}^{(k)}\right)\delta k_{s,N}+K_S\left(T_{s,P}^{(k)}-T_{s,S}^{(k)}\right)\delta k_{s,S}\\
&\quad+K_T\left(T_{s,P}^{(k)}-T_{s,T}^{(k)}\right)\delta k_{s,T}+K_B\left(T_{s,P}^{(k)}-T_{s,B}^{(k)}\right)\delta k_{s,B}\\
&\quad+\Delta V\left(T_{s,P}^{(k)}-T_{f,P}^{(k)}\right)\delta\alpha\\
&\quad-\Delta V\left(\phi_1^{(k)}\delta\kappa\Sigma_1+\phi_2^{(k)}\delta\kappa\Sigma_2+\phi_3^{(k)}\delta\kappa\Sigma_3\right)\\
&\quad-\left(K_E k_{s,E}^{(k)}\delta T_{s,E}+\ldots+K_B k_{s,B}^{(k)}\delta T_{s,B}\right)\\
&\quad+\left(K_E k_{s,E}^{(k)}+K_W k_{s,W}^{(k)}+K_N k_{s,N}^{(k)}+K_S k_{s,S}^{(k)}+K_T k_{s,T}^{(k)}+K_B k_{s,B}^{(k)}\right)\delta T_{s,P}\\
&\quad+\Delta V\alpha^{(k)}\delta T_{s,P}-\Delta V\alpha^{(k)}\delta T_{f,P}\\
&\quad-\Delta V\bar{P}_{n,fuel}\left(\kappa\Sigma_1^{(k)}\delta\phi+\kappa\Sigma_2^{(k)}\delta\phi_2+\kappa\Sigma_3^{(k)}\delta\phi_3\right)=S_k \qquad \text{(C.15)}
\end{aligned}
$$

Inserting definitions

$$
\begin{aligned}
&+ K_E\left(T_{s,P}^{(k)}-T_{s,E}^{(k)}\right)\left(\frac{\partial k_{s,E}}{\partial T_{s,E}}\right)^{n-1}\delta T_{s,E} + K_W\left(T_{s,P}^{(k)}-T_{s,W}^{(k)}\right)\left(\frac{\partial k_{s,W}}{\partial T_{s,W}}\right)^{n-1}\delta T_{s,W}\\
&+ K_N\left(T_{s,P}^{(k)}-T_{s,N}^{(k)}\right)\left(\frac{\partial k_{s,N}}{\partial T_{s,N}}\right)^{n-1}\delta T_{s,N} + K_S\left(T_{s,P}^{(k)}-T_{s,S}^{(k)}\right)\left(\frac{\partial k_{s,S}}{\partial T_{s,S}}\right)^{n-1}\delta T_{s,S}\\
&+ K_T\left(T_{s,P}^{(k)}-T_{s,T}^{(k)}\right)\left(\frac{\partial k_{s,T}}{\partial T_{s,T}}\right)^{n-1}\delta T_{s,T} + K_B\left(T_{s,P}^{(k)}-T_{s,B}^{(k)}\right)\left(\frac{\partial k_{s,B}}{\partial T_{s,B}}\right)^{n-1}\delta T_{s,B}\\
&+ \Bigg(K_E\left(T_{s,P}^{(k)}-T_{s,E}^{(k)}\right)\left(\frac{\partial k_{s,E}}{\partial T_{s,P}}\right)^{n-1} + K_W\left(T_{s,P}^{(k)}-T_{s,W}^{(k)}\right)\left(\frac{\partial k_{s,W}}{\partial T_{s,P}}\right)^{n-1}\\
&+ K_N\left(T_{s,P}^{(k)}-T_{s,N}^{(k)}\right)\left(\frac{\partial k_{s,N}}{\partial T_{s,P}}\right)^{n-1} + K_S\left(T_{s,P}^{(k)}-T_{s,S}^{(k)}\right)\left(\frac{\partial k_{s,S}}{\partial T_{s,P}}\right)^{n-1}\\
&+K_T\left(T_{s,P}^{(k)}-T_{s,T}^{(k)}\right)\left(\frac{\partial k_{s,T}}{\partial T_{s,P}}\right)^{n-1} + K_B\left(T_{s,P}^{(k)}-T_{s,B}^{(k)}\right)\left(\frac{\partial k_{s,B}}{\partial T_{s,P}}\right)^{n-1}\Bigg)\delta T_{s,P}\\
&+ \Delta V\left(T_{s,P}^{(k)}-T_{f,P}^{(k)}\right)\left(\left(\frac{\partial \alpha_P}{\partial T_{s,P}}\right)^{n-1}\delta T_{s,P} + \left(\frac{\partial \alpha_P}{\partial T_{f,P}}\right)^{n-1}\delta T_{f,P} + \left(\frac{\partial \alpha_P}{\partial P_P}\right)^{n-1}\delta P_P\right)\\
&+ \Delta V\left(T_{s,P}^{(k)}-T_{f,P}^{(k)}\right)\left(\left(\frac{\partial \alpha_P}{\partial v_x}\right)^{n-1}\delta v_x + \left(\frac{\partial \alpha_P}{\partial v_y}\right)^{n-1}\delta v_y + \left(\frac{\partial \alpha_P}{\partial v_z}\right)^{n-1}\delta v_z\right)\\
&- \Delta V\bar{P}_{n,fuel}\left(\phi_1^{(k)}\delta\kappa\Sigma_1 + \phi_2^{(k)}\delta\kappa\Sigma_2 + \phi_3^{(k)}\delta\kappa\Sigma_3\right)\\
&- \left(K_E k_{s,E}^{(k)}\delta T_{s,E} + \ldots + K_B k_{s,B}^{(k)}\delta T_{s,B}\right)\\
&+ \left(K_E k_{s,E}^{(k)} + K_W k_{s,W}^{(k)} + K_N k_{s,N}^{(k)} + K_S k_{s,S}^{(k)} + K_T k_{s,T}^{(k)} + K_B k_{s,B}^{(k)}\right)\delta T_{s,P}\\
&+ \Delta V\alpha^{(k)}\delta T_{s,P} - \Delta V\alpha^{(k)}\delta T_{f,P}\\
&- \Delta V\bar{P}_{n,fuel}\left(\kappa\Sigma_1^{(k)}\delta\phi + \kappa\Sigma_2^{(k)}\delta\phi_2 + \kappa\Sigma_3^{(k)}\delta\phi_3\right) = S_k \qquad \text{(C.16)}
\end{aligned}
$$

The cross section dependencies,

$$
\begin{aligned}
\kappa\Sigma &= f\left(T_D, T_M\right)\\
\delta\kappa\Sigma &= \left(\left(\tfrac{\partial\kappa\Sigma}{\partial T_D}\right)^{n-1}\delta T_S + \left(\tfrac{\partial\kappa\Sigma}{\partial T_M}\right)^{n-1}\delta T_M\right)
\end{aligned}
\qquad \text{(C.17)}
$$

Inserting the cross section terms, collecting XS temperature terms

$$
\begin{aligned}
&+ K_E\left(T_{s,P}^{(k)}-T_{s,E}^{(k)}\right)\left(\frac{\partial k_{s,E}}{\partial T_{s,E}}\right)^{n-1}\delta T_{s,E} + K_W\left(T_{s,P}^{(k)}-T_{s,W}^{(k)}\right)\left(\frac{\partial k_{s,W}}{\partial T_{s,W}}\right)^{n-1}\delta T_{s,W}\\
&+ K_N\left(T_{s,P}^{(k)}-T_{s,N}^{(k)}\right)\left(\frac{\partial k_{s,N}}{\partial T_{s,N}}\right)^{n-1}\delta T_{s,N} + K_S\left(T_{s,P}^{(k)}-T_{s,S}^{(k)}\right)\left(\frac{\partial k_{s,S}}{\partial T_{s,S}}\right)^{n-1}\delta T_{s,S}\\
&+ K_T\left(T_{s,P}^{(k)}-T_{s,T}^{(k)}\right)\left(\frac{\partial k_{s,T}}{\partial T_{s,T}}\right)^{n-1}\delta T_{s,T} + K_B\left(T_{s,P}^{(k)}-T_{s,B}^{(k)}\right)\left(\frac{\partial k_{s,B}}{\partial T_{s,B}}\right)^{n-1}\delta T_{s,B}\\
&+ \Bigg(K_E\left(T_{s,P}^{(k)}-T_{s,E}^{(k)}\right)\left(\frac{\partial k_{s,E}}{\partial T_{s,P}}\right)^{n-1} + K_W\left(T_{s,P}^{(k)}-T_{s,W}^{(k)}\right)\left(\frac{\partial k_{s,W}}{\partial T_{s,P}}\right)^{n-1}\\
&+ K_N\left(T_{s,P}^{(k)}-T_{s,N}^{(k)}\right)\left(\frac{\partial k_{s,N}}{\partial T_{s,P}}\right)^{n-1} + K_S\left(T_{s,P}^{(k)}-T_{s,S}^{(k)}\right)\left(\frac{\partial k_{s,S}}{\partial T_{s,P}}\right)^{n-1}\\
&+K_T\left(T_{s,P}^{(k)}-T_{s,T}^{(k)}\right)\left(\frac{\partial k_{s,T}}{\partial T_{s,P}}\right)^{n-1} + K_B\left(T_{s,P}^{(k)}-T_{s,B}^{(k)}\right)\left(\frac{\partial k_{s,B}}{\partial T_{s,P}}\right)^{n-1}\Bigg)\delta T_{s,P}\\
&+ \Delta V\left(T_{s,P}^{(k)}-T_{f,P}^{(k)}\right)\left(\left(\frac{\partial \alpha_P}{\partial T_{s,P}}\right)^{n-1}\delta T_{s,P} + \left(\frac{\partial \alpha_P}{\partial T_{f,P}}\right)^{n-1}\delta T_{f,P} + \left(\frac{\partial \alpha_P}{\partial P_P}\right)^{n-1}\delta P_P\right)\\
&+ \Delta V\left(T_{s,P}^{(k)}-T_{f,P}^{(k)}\right)\left(\left(\frac{\partial \alpha_P}{\partial v_x}\right)^{n-1}\delta v_x + \left(\frac{\partial \alpha_P}{\partial v_y}\right)^{n-1}\delta v_y + \left(\frac{\partial \alpha_P}{\partial v_z}\right)^{n-1}\delta v_z\right)\\
&- \Delta V\bar{P}_{n,fuel}\left(\phi_1^{(k)}\left(\frac{\partial \kappa\Sigma_1}{\partial T_D}\right)^{n-1} + \phi_2^{(k)}\left(\frac{\partial \kappa\Sigma_2}{\partial T_D}\right)^{n-1} + \phi_3^{(k)}\left(\frac{\partial \kappa\Sigma_3}{\partial T_D}\right)^{n-1}\right)\delta T_{s,D}\\
&- \Delta V\bar{P}_{n,fuel}\left(\phi_1^{(k)}\left(\frac{\partial \kappa\Sigma_1}{\partial T_M}\right)^{n-1} + \phi_2^{(k)}\left(\frac{\partial \kappa\Sigma_2}{\partial T_M}\right)^{n-1} + \phi_3^{(k)}\left(\frac{\partial \kappa\Sigma_3}{\partial T_M}\right)^{n-1}\right)\delta T_{s,M}\\
&- \left(K_E k_{s,E}^{(k)}\delta T_{s,E} + \ldots + K_B k_{s,B}^{(k)}\delta T_{s,B}\right)\\
&+ \left(K_E k_{s,E}^{(k)} + K_W k_{s,W}^{(k)} + K_N k_{s,N}^{(k)} + K_S k_{s,S}^{(k)} + K_T k_{s,T}^{(k)} + K_B k_{s,B}^{(k)}\right)\delta T_{s,P}\\
&+ \Delta V\alpha^{(k)}\delta T_{s,P} - \Delta V\alpha^{(k)}\delta T_{f,P}\\
&- \Delta V\bar{P}_{n,fuel}\left(\kappa\Sigma_1^{(k)}\delta\phi + \kappa\Sigma_2^{(k)}\delta\phi_2 + \kappa\Sigma_3^{(k)}\delta\phi_3 + \kappa\Sigma_4^{(k)}\delta\phi_4\right) = S_k \quad \text{(C.18)}
\end{aligned}
$$

All terms are gathered, giving the final form

$$
\begin{aligned}
&+ K_E \left[\left(T_{s,P}^{(k)} - T_{s,E}^{(k)} \right) \left(\frac{\partial k_{s,E}}{\partial T_{s,E}} \right)^{n-1} - k_{s,E}^{(k)} \right] \delta T_{s,E} \\
&\qquad\qquad\qquad\qquad + \ldots + \\
&K_B \left[\left(T_{s,P}^{(k)} - T_{s,B}^{(k)} \right) \left(\frac{\partial k_{s,B}}{\partial T_{s,B}} \right)^{n-1} - k_{s,B}^{(k)} \right] \delta T_{s,B} \\
&+ \Bigg[K_E \left[\left(T_{s,P}^{(k)} - T_{s,E}^{(k)} \right) \left(\frac{\partial k_{s,E}}{\partial T_{s,P}} \right)^{n-1} + k_{s,E}^{(k)} \right] + K_W \left[\left(T_{s,P}^{(k)} - T_{s,W}^{(k)} \right) \left(\frac{\partial k_{s,W}}{\partial T_{s,P}} \right)^{n-1} + k_{s,W}^{(k)} \right] \\
&+ K_N \left[\left(T_{s,P}^{(k)} - T_{s,N}^{(k)} \right) \left(\frac{\partial k_{s,N}}{\partial T_{s,P}} \right)^{n-1} + k_{s,N}^{(k)} \right] + K_S \left[\left(T_{s,P}^{(k)} - T_{s,S}^{(k)} \right) \left(\frac{\partial k_{s,S}}{\partial T_{s,P}} \right)^{n-1} + k_{s,S}^{(k)} \right] \\
&+ K_T \left[\left(T_{s,P}^{(k)} - T_{s,T}^{(k)} \right) \left(\frac{\partial k_{s,T}}{\partial T_{s,P}} \right)^{n-1} + k_{s,T}^{(k)} \right] + K_B \left[\left(T_{s,P}^{(k)} - T_{s,B}^{(k)} \right) \left(\frac{\partial k_{s,B}}{\partial T_{s,P}} \right)^{n-1} + k_{s,B}^{(k)} \right] \\
&+ \Delta V \left(\left(T_{s,P}^{(k)} - T_{f,P}^{(k)} \right) \left(\frac{\partial \alpha_P}{\partial T_{s,P}} \right)^{n-1} + \alpha^{(k)} \right) \Bigg] \delta T_{s,P} \\
&+ \Delta V \left(T_{s,P}^{(k)} - T_{f,P}^{(k)} \right) \left(\left(\frac{\partial \alpha}{\partial v_x} \right)^{n-1} \delta v_x + \left(\frac{\partial \alpha}{\partial v_y} \right)^{n-1} \delta v_y + \left(\frac{\partial \alpha}{\partial v_z} \right)^{n-1} \delta v_z \right) \\
&+ \Delta V \left[\left(T_{s,P}^{(k)} - T_{f,P}^{(k)} \right) \left(\frac{\partial \alpha_P}{\partial T_{f,P}} \right)^{n-1} - \alpha^{(k)} \right] \delta T_{f,P} + \Delta V \left[\left(T_{s,P}^{(k)} - T_{f,P}^{(k)} \right) \left(\frac{\partial \alpha_P}{\partial P_P} \right)^{n-1} \right] \delta P_P \\
&- \Delta V \bar{P}_{n,fuel} \left(\left[\kappa \Sigma_1^{(k)} \right] \delta \phi_1 - \left[\kappa \Sigma_2^{(k)} \right] \delta \phi_2 - \left[\kappa \Sigma_3^{(k)} \right] \delta \phi_3 \right) \\
&- \Delta V \bar{P}_{n,fuel} \left(\phi_1^{(k)} \left(\frac{\partial \kappa \Sigma_1}{\partial T_D} \right)^{n-1} + \phi_2^{(k)} \left(\frac{\partial \kappa \Sigma_2}{\partial T_D} \right)^{n-1} + \phi_3^{(k)} \left(\frac{\partial \kappa \Sigma_3}{\partial T_D} \right)^{n-1} \right) \delta T_{s,D} \\
&- \Delta V \bar{P}_{n,fuel} \left(\phi_1^{(k)} \left(\frac{\partial \kappa \Sigma_1}{\partial T_M} \right)^{n-1} + \phi_2^{(k)} \left(\frac{\partial \kappa \Sigma_2}{\partial T_M} \right)^{n-1} + \phi_3^{(k)} \left(\frac{\partial \kappa \Sigma_3}{\partial T_M} \right)^{n-1} \right) \delta T_{s,M} = S_k
\end{aligned}
\tag{C.19}
$$

Appendix D

Derivation: Fluid Energy Equation

Fluid Energy Equation Derivation for Implicit/Newton Iteration

$$\begin{aligned}\frac{\partial}{\partial t}\left[\varepsilon\rho_f c_{p_f} T_f\right]\Delta V = \\ &A_E T_{f,E} + A_W T_{f,W} + A_N T_{f,N} + A_S T_{s,S} + A_T T_{f,T} + A_B T_{f,B} \\ &-\ (\mathrm{A}_E + A_W + A_N + A_S + A_T + A_B)T_{f,P} - \alpha(T_{f,P} - T_{s,P})\Delta V \end{aligned} \quad \text{(D.1)}$$

Coefficients

$$\begin{aligned} A_E &= D_E A(|P_E|) + [\|-F_E, 0\|] & A_W &= D_w A(|P_w|) + [\|F_W, 0\|] \\ A_S &= D_S A(|P_S|) + [\|-F_S, 0\|] & A_N &= D_n A(|P_n|) + [\|F_N, 0\|] \\ A_T &= D_T A(|P_T|) + [\|-F_T, 0\|] & A_B &= D_b A(|P_b|) + [\|F_B, 0\|] \end{aligned} \quad \text{(D.2)}$$

Subcoefficients

$$\begin{aligned} F_e &= (\dot{m}c_p)_e r_e \Delta\theta\Delta z & D_e &= \varepsilon_e k_{fe}\tfrac{r_e}{\partial r_e}\Delta\theta\Delta z \\ F_w &= (\dot{m}c_p)_w r_w \Delta\theta\Delta z & D_w &= \varepsilon_w k_{fw}\tfrac{r_w}{\partial r_w}\Delta\theta\Delta z \\ F_n &= (\dot{m}c_p)_n \Delta r\Delta z & D_n &= \varepsilon_n k_{fn}\tfrac{1}{r_n}\tfrac{1}{\partial\theta_n}\Delta r\Delta z \\ F_s &= (\dot{m}c_p)_s \Delta r\Delta z & D_s &= \varepsilon_s k_{fs}\tfrac{1}{r_s}\tfrac{1}{\partial\theta_s}\Delta r\Delta z \\ F_t &= (\dot{m}c_p)_t \tfrac{r_e^2 - r_w^2}{2}\Delta\theta & D_t &= \varepsilon_t k_{ft}\tfrac{1}{\partial z_t}\tfrac{r_e^2 - r_w^2}{2}\Delta\theta \\ F_b &= (\dot{m}c_p)_b \tfrac{r_e^2 - r_w^2}{2}\Delta\theta & D_b &= \varepsilon_b k_{fb}\tfrac{1}{\partial z_b}\tfrac{r_e^2 - r_w^2}{2}\Delta\theta \end{aligned} \quad \text{(D.3)}$$

Expanding the primary variables first,

$$\begin{aligned} T_{f,E} &= T_{f,E}^{(k)} + \delta T_{f,E} & T_{f,W} &= T_{f,W}^{(k)} + \delta T_{f,W} \\ T_{f,N} &= T_{f,N}^{(k)} + \delta T_{f,N} & T_{f,S} &= T_{f,S}^{(k)} + \delta T_{f,S} \\ T_{f,T} &= T_{f,T}^{(k)} + \delta T_{f,T} & T_{f,B} &= T_{f,B}^{(k)} + \delta T_{f,B} \\ T_{s,P} &= T_{s,P}^{(k)} + \delta T_{s,P} & T_{f,P} &= T_{f,P}^{(k)} + \delta T_{f,P} \end{aligned} \quad \text{(D.4)}$$

Inserting into the equation

$$
\begin{aligned}
&(A_E + A_W + A_N + A_S + A_T + A_B)\left(T_{f,P} + \delta T_{f,P}\right) \\
&\qquad - A_E\left(T_{f,E} + \delta T_{f,E}\right) - A_W\left(T_{f,W} + \delta T_{f,W}\right) - A_N\left(T_{f,N} + \delta T_{f,N}\right) \\
&\qquad\quad - A_S\left(T_{s,S} + \delta T_{f,S}\right) - A_T\left(T_{f,T} + \delta T_{f,T}\right) - A_B\left(T_{f,B} + \delta T_{f,B}\right) \\
&\qquad\qquad + \left(\left(T_{f,P} + \delta T_{f,P}\right) - \left(T_{s,P} + \delta T_{s,P}\right)\right)\Delta V \delta\alpha = 0 \quad \text{(D.5)}
\end{aligned}
$$

The coefficients are expanded as well. This assumes only upwind differencing. The derivation will assume all "upwind" statements are true, and implementation will allow only one direction for flow per Newton iteration. Flow can change direction, changing the upwind direction.

$$
\begin{aligned}
&\alpha = \alpha^{(k)} + \delta\alpha \\
&\begin{aligned}
F_e &= G_E\left(c_{pf,E}^{(k)} + \delta c_{pf,E}\right)\left(\dot m_E^{(k)} + \delta \dot m_E\right) & D_e &= K_E\left(k_{f,E}^{(k)} + \delta k_{f,E}\right) \\
F_w &= G_W\left(c_{pf,W}^{(k)} + \delta c_{pf,W}\right)\left(\dot m_W^{(k)} + \delta \dot m_W\right) & D_w &= K_W\left(k_{f,W}^{(k)} + \delta k_{f,W}\right) \\
F_n &= G_N\left(c_{pf,N}^{(k)} + \delta c_{pf,N}\right)\left(\dot m_N^{(k)} + \delta \dot m_N\right) & D_n &= K_N\left(k_{f,N}^{(k)} + \delta k_{f,N}\right) \\
F_s &= G_S\left(c_{pf,S}^{(k)} + \delta c_{pf,S}\right)\left(\dot m_S^{(k)} + \delta \dot m_S\right) & D_s &= K_S\left(k_{f,S}^{(k)} + \delta k_{f,S}\right) \\
F_t &= G_T\left(c_{pf,T}^{(k)} + \delta c_{pf,T}\right)\left(\dot m_T^{(k)} + \delta \dot m_T\right) & D_t &= K_T\left(k_{f,T}^{(k)} + \delta k_{f,T}\right) \\
F_b &= G_B\left(c_{pf,B}^{(k)} + \delta c_{pf,B}\right)\left(\dot m_B^{(k)} + \delta \dot m_B\right) & D_b &= K_B\left(k_{f,B}^{(k)} + \delta k_{f,B}\right)
\end{aligned} \quad \text{(D.6)}
\end{aligned}
$$

Developing the coefficients for simplicity, losing higher order terms

$$
\begin{aligned}
&\left(T_{f,P} - T_{f,E}\right)\delta A_E + \left(T_{f,P} - T_{f,W}\right)\delta A_W + A_E\left(\delta T_{f,P} - \delta T_{f,E}\right) + A_W\left(\delta T_{f,P} - \delta T_{f,W}\right) \\
&\left(T_{f,P} - T_{f,N}\right)\delta A_N + \left(T_{f,P} - T_{f,S}\right)\delta A_S + A_N\left(\delta T_{f,P} - \delta T_{f,N}\right) + A_S\left(\delta T_{f,P} - \delta T_{f,S}\right) \\
&\left(T_{f,P} - T_{f,T}\right)\delta A_T + \left(T_{f,P} - T_{f,B}\right)\delta A_B + A_T\left(\delta T_{f,P} - \delta T_{f,T}\right) + A_B\left(\delta T_{f,P} - \delta T_{f,B}\right) \\
&\qquad + \left(T_{f,P} - T_{s,P}\right)\Delta V \delta\alpha + \left(\delta T_{f,P} - \delta T_{s,P}\right)\Delta V \alpha = \\
&\qquad - A_E\left(T_{f,P} - T_{f,E}\right) - A_W\left(T_{f,P} - T_{f,W}\right) \\
&\qquad - A_N\left(T_{f,P} - T_{f,N}\right) - A_S\left(T_{f,P} - T_{f,S}\right) \\
&\qquad - A_T\left(T_{f,P} - T_{f,T}\right) - A_B\left(T_{f,P} - T_{f,B}\right) \\
&\qquad\qquad - \left(T_{f,P} - T_{s,P}\right)\Delta V \alpha \quad \text{(D.7)}
\end{aligned}
$$

The new equation is then given as, using a shorthand for the coefficients,

$$
\begin{gathered}
\left(T_{f,P}-T_{f,E}\right)K_E\left(\frac{\partial k_E}{\partial T_{f,P}}\delta T_{f,P}+\frac{\partial k_E}{\partial T_{f,E}}\delta T_{f,E}+\frac{\partial k_E}{\partial p_{f,P}}\delta p_{f,P}+\frac{\partial k_E}{\partial p_{f,E}}\delta p_{f,E}\right)\\
+\left(T_{f,P}-T_{f,E}\right)G_E\dot{m}_E\left(\frac{\partial c_{p,E}}{\partial T_{f,P}}\delta T_{f,P}+\frac{\partial c_{p,E}}{\partial T_{f,E}}\delta T_{f,E}+\frac{\partial c_{p,E}}{\partial p_{f,P}}\delta p_{f,P}+\frac{\partial c_{p,E}}{\partial p_{f,E}}\delta p_{f,E}\right)\\
+\left(T_{f,P}-T_{f,E}\right)G_E c_{p,E}\left(\frac{\partial \dot{m}_E}{\partial T_{s,P}}\delta T_{s,P}+...+\frac{\partial \dot{m}_E}{\partial v_{z,P}}\delta v_{z,P}+\frac{\partial \dot{m}_E}{\partial T_{s,E}}\delta T_{s,E}+...+\frac{\partial \dot{m}_E}{\partial v_{z,E}}\delta v_{z,E}\right)\\
+...+\\
\left(T_{f,P}-T_{f,B}\right)K_B\left(\frac{\partial k_B}{\partial T_{f,P}}\delta T_{f,P}+\frac{\partial k_B}{\partial T_{f,B}}\delta T_{f,B}+\frac{\partial k_B}{\partial p_{f,P}}\delta p_{f,P}+\frac{\partial k_B}{\partial p_{f,B}}\delta p_{f,B}\right)\\
+\left(T_{f,P}-T_{f,B}\right)G_B\dot{m}_B\left(\frac{\partial c_{p,B}}{\partial T_{f,P}}\delta T_{f,P}+\frac{\partial c_{p,B}}{\partial T_{f,B}}\delta T_{f,B}+\frac{\partial c_{p,B}}{\partial p_{f,P}}\delta p_{f,P}+\frac{\partial c_{p,B}}{\partial p_{f,B}}\delta p_{f,B}\right)\\
+\left(T_{f,P}-T_{f,B}\right)G_B c_{p,B}\left(\frac{\partial \dot{m}_B}{\partial T_{s,P}}\delta T_{s,P}+...+\frac{\partial \dot{m}_B}{\partial v_{z,P}}\delta v_{z,P}+\frac{\partial \dot{m}_B}{\partial T_{s,B}}\delta T_{s,B}+...+\frac{\partial \dot{m}_B}{\partial v_{z,B}}\delta v_{z,B}\right)\\
+\Delta V\left(T_{f,P}-T_{s,P}\right)\left(\frac{\partial \alpha}{\partial T_{s,P}}T_{s,P}+...+\frac{\partial \alpha}{\partial v_{z,P}}\delta v_{z,P}\right)\\
+A_E\left(\delta T_{f,P}-\delta T_{f,E}\right)+A_W\left(\delta T_{f,P}-\delta T_{f,W}\right)+A_N\left(\delta T_{f,P}-\delta T_{f,N}\right)\\
+A_S\left(\delta T_{f,P}-\delta T_{f,S}\right)A_T\left(\delta T_{f,P}-\delta T_{f,T}\right)+A_B\left(\delta T_{f,P}-\delta T_{f,B}\right)+\left(\delta T_{f,P}-\delta T_{s,P}\right)\Delta V\alpha=\\
-A_E\left(T_{f,P}-T_{f,E}\right)-A_W\left(T_{f,P}-T_{f,W}\right)-A_N\left(T_{f,P}-T_{f,N}\right)\\
-A_S\left(T_{f,P}-T_{f,S}\right)-A_T\left(T_{f,P}-T_{f,T}\right)-A_B\left(T_{f,P}-T_{f,B}\right)-\left(T_{f,P}-T_{s,P}\right)\Delta V\alpha
\end{gathered}
\tag{D.8}
$$

Appendix E

Derivation: Pressure Equation

The mass flow equation for each node,

$$
\begin{aligned}
\frac{(r_N - r_{N-1})}{A_{r1}}\frac{\dot m_1^{k+1} - \dot m_1^k}{\Delta t} &= -\frac{(r_N - r_{N-1})}{A_{r1}} W_{r1}^{k+1}\dot m_1^{k+1} - \Delta p_{r1}^{k+1} \\
\frac{(r_N - r_{N+1})}{A_{r2}}\frac{\dot m_2^{k+1} - \dot m_2^k}{\Delta t} &= -\frac{(r_N - r_{N+1})}{A_{r2}} W_{r2}^{k+1}\dot m_2^{k+1} - \Delta p_{r2}^{k+1} \\
\frac{(\theta_M - \theta_{M-1})}{A_{\theta 3}}\frac{\dot m_3^{k+1} - \dot m_3^k}{\Delta t} &= -\frac{(\theta_M - \theta_{M-1})}{A_{\theta 3}} W_{\theta 3}^{k+1}\dot m_3^{k+1} - \frac{1}{r_N}\Delta p_{\theta 3}^{k+1} \\
\frac{(\theta_M - \theta_{M+1})}{A_{\theta 4}}\frac{\dot m_4^{k+1} - \dot m_4^k}{\Delta t} &= -\frac{(\theta_M - \theta_{M+1})}{A_{\theta 4}} W_{\theta 4}^{k+1}\dot m_4^{k+1} - \frac{1}{r_N}\Delta p_{\theta 4}^{k+1} \\
\frac{(z_I - z_{I-1})}{A_{z5}}\frac{\dot m_5^{k+1} - \dot m_5^k}{\Delta t} &= -\frac{(z_I - z_{I-1})}{A_{z5}} W_{z5}^{k+1}\dot m_5^{k+1} - \Delta p_{z5}^{k+1} + g\Delta z_5 \rho_5^{k+1} \\
\frac{(z_I - z_{I+1})}{A_{z6}}\frac{\dot m_6^{k+1} - \dot m_6^k}{\Delta t} &= -\frac{(z_I - z_{I+1})}{A_{z6}} W_{z6}^{k+1}\dot m_6^{k+1} - \Delta p_{z6}^{k+1} - g\Delta z_6 \rho_6^{k+1}
\end{aligned}
\tag{E.1}
$$

A few constants for convenience,

$$
\begin{array}{lll}
G_1 = \frac{A_{r1}}{r_N - r_{N-1}} & G_3 = \frac{A_{\theta 3}}{\theta_M - \theta_{M-1}} & G_5 = \frac{A_{z5}}{z_I - z_{I-1}} \\
G_2 = \frac{A_{r2}}{r_N - r_{N+1}} & G_4 = \frac{A_{\theta 4}}{\theta_M - \theta_{M+1}} & G_6 = \frac{A_{z6}}{z_I - z_{I+1}}
\end{array}
\tag{E.2}
$$

Using these in the equations,

$$
\begin{aligned}
\frac{\dot m_1^{k+1} - \dot m_1^k}{\Delta t} &= -W_{r1}^{k+1}\dot m_1^{k+1} - G_1\Delta p_{r1}^{k+1} \\
\frac{\dot m_2^{k+1} - \dot m_2^k}{\Delta t} &= -W_{r2}^{k+1}\dot m_2^{k+1} - G_2\Delta p_{r2}^{k+1} \\
\frac{\dot m_3^{k+1} - \dot m_3^k}{\Delta t} &= -W_{\theta 3}^{k+1}\dot m_3^{k+1} - \frac{1}{r_N}G_3\Delta p_{\theta 3}^{k+1} \\
\frac{\dot m_4^{k+1} - \dot m_4^k}{\Delta t} &= -W_{\theta 4}^{k+1}\dot m_4^{k+1} - \frac{1}{r_N}G_4\Delta p_{\theta 4}^{k+1} \\
\frac{\dot m_5^{k+1} - \dot m_5^k}{\Delta t} &= -W_{z5}^{k+1}\dot m_5^{k+1} - G_5\Delta p_{z5}^{k+1} + G_5 g\Delta z_5 \rho_5^{k+1} \\
\frac{\dot m_6^{k+1} - \dot m_6^k}{\Delta t} &= -W_{z6}^{k+1}\dot m_6^{k+1} - G_6\Delta p_{z6}^{k+1} - G_6 g\Delta z_6 \rho_6^{k+1}
\end{aligned}
\tag{E.3}
$$

Solving for the k+1 time terms in m-dot.

$$\begin{aligned} \dot{m}_1^{k+1} &= \frac{\frac{\dot{m}_1^k}{\Delta t} - G_1 \Delta p_{r1}^{k+1}}{\left(\frac{1}{\Delta t} + W_{r1}^{k+1}\right)} & \dot{m}_2^{k+1} &= \frac{\frac{\dot{m}_2^k}{\Delta t} - G_2 \Delta p_{r2}^{k+1}}{\left(\frac{1}{\Delta t} + W_{r2}^{k+1}\right)} \\ \dot{m}_3^{k+1} &= \frac{\frac{\dot{m}_3^k}{\Delta t} - G_3 \Delta p_{\theta 3}^{k+1}}{\left(\frac{1}{\Delta t} + W_{\theta 3}^{k+1}\right)} & \dot{m}_4^{k+1} &= \frac{\frac{\dot{m}_4^k}{\Delta t} - G_4 \Delta p_{\theta 4}^{k+1}}{\left(\frac{1}{\Delta t} + W_{\theta 4}^{k+1}\right)} \\ \dot{m}_5^{k+1} &= \frac{\frac{\dot{m}_5^k}{\Delta t} - G_5 \Delta p_{z5}^{k+1} + G_5 g \Delta z_5 \rho_5^{k+1}}{\left(\frac{1}{\Delta t} + W_{z5}^{k+1}\right)} & \dot{m}_6^{k+1} &= \frac{\frac{\dot{m}_6^k}{\Delta t} - G_6 \Delta p_{z6}^{k+1} - G_6 g \Delta z_6 \rho_6^{k+1}}{\left(\frac{1}{\Delta t} + W_{z6}^{k+1}\right)} \end{aligned} \tag{E.4}$$

Creating the equation in terms of pressure,

$$\begin{aligned} V\frac{\Delta \rho}{\Delta t} &= \dot{m}_1^{k+1} + \dot{m}_2^{k+1} + \dot{m}_3^{k+1} + \dot{m}_4^{k+1} + \dot{m}_5^{k+1} + \dot{m}_6^{k+1} \\ & V\left(\frac{\partial \rho}{\partial T}\frac{1}{\Delta t}\left(T^{k+1} - T^k\right) + \frac{\partial \rho}{\partial p}\frac{1}{\Delta t}\left(p^{k+1} - p^k\right)\right) \\ &= \frac{\frac{\dot{m}_1^k}{\Delta t} - G_1 \Delta p_{r1}^{k+1}}{\left(\frac{1}{\Delta t} + W_{r1}^{k+1}\right)} + \frac{\frac{\dot{m}_2^k}{\Delta t} - G_2 \Delta p_{r2}^{k+1}}{\left(\frac{1}{\Delta t} + W_{r2}^{k+1}\right)} + \frac{\frac{\dot{m}_3^k}{\Delta t} - G_3 \Delta p_{\theta 3}^{k+1}}{\left(\frac{1}{\Delta t} + W_{\theta 3}^{k+1}\right)} + \frac{\frac{\dot{m}_4^k}{\Delta t} - G_4 \Delta p_{\theta 4}^{k+1}}{\left(\frac{1}{\Delta t} + W_{\theta 4}^{k+1}\right)} \\ &\quad + \frac{\frac{\dot{m}_5^k}{\Delta t} - G_5 \Delta p_{z5}^{k+1} + G_5 g \Delta z_5 \rho_5^{k+1}}{\left(\frac{1}{\Delta t} + W_{z5}^{k+1}\right)} + \frac{\frac{\dot{m}_6^k}{\Delta t} - G_6 \Delta p_{z6}^{k+1} - G_6 g \Delta z_6 \rho_6^{k+1}}{\left(\frac{1}{\Delta t} + W_{z6}^{k+1}\right)} \end{aligned} \tag{E.5}$$

Removing the time dependent terms,

$$\begin{aligned} 0 = \frac{-G_1}{W_{r1}^{k+1}}\Delta p_{r1}^{k+1} + \frac{-G_2}{W_{r2}^{k+1}}\Delta p_{r2}^{k+1} + \frac{-G_3}{W_{\theta 3}^{k+1}}\Delta p_{\theta 3}^{k+1} + \frac{-G_4}{W_{\theta 4}^{k+1}}\Delta p_{\theta 4}^{k+1} \\ + \frac{-G_5}{W_{z5}^{k+1}}\left(\Delta p_{z5}^{k+1} + g\Delta z_5 \rho_5^{k+1}\right) + \frac{-G_6}{W_{z6}^{k+1}}\left(\Delta p_{z6}^{k+1} + g\Delta z_6 \rho_6^{k+1}\right) \end{aligned} \tag{E.6}$$

Some coefficients for convenience,

$$\begin{aligned} R_{r,E} &= \tfrac{G_1}{W_{r1}^{k+1}} & R_{r,W} &= \tfrac{G_2}{W_{r2}^{k+1}} & R_{a,N} &= \tfrac{G_3}{W_{\theta 3}^{k+1}} & R_{a,S} &= \tfrac{G_4}{W_{\theta 4}^{k+1}} \\ R_{l,B} &= \tfrac{G_5}{W_{z5}^{k+1}} & R_{l,T} &= \tfrac{G_6}{W_{z6}^{k+1}} & b_{f,B} &= g\Delta z_5 \rho_5^{k+1} & b_{f,T} &= g\Delta z_6 \rho_6^{k+1} \end{aligned} \tag{E.7}$$

The equations are then given as,

$$\begin{aligned} - R_{r,E}\Delta p_{r1}^{k+1} - R_{r,W}\Delta p_{r2}^{k+1} - R_{a,N}\Delta p_{\theta 3}^{k+1} - R_{r,S}\Delta p_{\theta 4}^{k+1} \\ - R_{l,B}\left(\Delta p_{z5}^{k+1} + b_{f,B}\right) - R_{l,T}\left(\Delta p_{z6}^{k+1} + b_{f,B}\right) = 0 \end{aligned} \tag{E.8}$$

Expanding the delta pressure in terms of the geometry,

$$\begin{aligned}
&R_{r,E}\left(p_{f,P}-p_{f,E}\right)+R_{r,W}\left(p_{f,P}-p_{f,W}\right) \\
&\qquad +R_{a,N}\left(p_{f,P}-p_{f,N}\right)+R_{r,S}\left(p_{f,P}-p_{f,S}\right) \\
&\qquad +R_{l,B}\left(\left(p_{f,P}-p_{f,B}\right)-b_{f,B}\right)+R_{l,T}\left(\left(p_{f,P}-p_{f,T}\right)-b_{f,B}\right)=0
\end{aligned} \tag{E.9}$$

Expanding the primary variables, and coefficients,

$$\begin{aligned}
&R_{r,E}\left(\delta p_{f,P}-\delta p_{f,E}\right)+R_{r,W}\left(\delta p_{f,P}-\delta p_{f,W}\right) \\
&\qquad +R_{a,N}\left(\delta p_{f,P}-\delta p_{f,N}\right)+R_{r,S}\left(\delta p_{f,P}-\delta p_{f,S}\right) \\
&\qquad +R_{l,B}\left(\delta p_{f,P}-\delta p_{f,B}\right)+R_{l,T}\left(\delta p_{f,P}-\delta p_{f,T}\right) \\
&\qquad +\delta R_{r,E}\left(p_{f,P}-p_{f,E}\right)+\delta R_{r,W}\left(p_{f,P}-p_{f,W}\right) \\
&\qquad +\delta R_{a,N}\left(p_{f,P}-p_{f,N}\right)+\delta R_{r,S}\left(p_{f,P}-p_{f,S}\right) \\
&\qquad -R_{l,B}\delta b_{f,B}-R_{l,T}\delta b_{f,B} \\
&\qquad +\delta R_{l,B}\left(\left(p_{f,P}-p_{f,B}\right)-b_{f,B}\right)+\delta R_{l,T}\left(\left(p_{f,P}-p_{f,T}\right)-b_{f,B}\right) \\
&\qquad =-R_{r,E}\left(p_{f,P}-p_{f,E}\right)-R_{r,W}\left(p_{f,P}-p_{f,W}\right) \\
&\qquad -R_{a,N}\left(p_{f,P}-p_{f,N}\right)-R_{r,S}\left(p_{f,P}-p_{f,S}\right) \\
&\qquad -R_{l,B}\left(\left(p_{f,P}-p_{f,B}\right)-b_{f,B}\right)-R_{l,T}\left(\left(p_{f,P}-p_{f,T}\right)-b_{f,B}\right)
\end{aligned} \tag{E.10}$$

The expanded coefficients, all terms included,

$$
\begin{aligned}
&-\left(p_{f,P}-p_{f,E}\right)\left[\frac{\partial R_{r,E}}{\partial T_{s,P}}\delta T_{s,p}+...+\frac{\partial R_{r,E}}{\partial v_{z,P}}\delta v_{z,P}\right]\\
&\quad-\left(p_{f,P}-p_{f,E}\right)\left[\frac{\partial R_{r,E}}{\partial T_{s,E}}\delta T_{s,E}+...+\frac{\partial R_{r,E}}{\partial v_{z,E}}\delta v_{z,E}\right]\\
&\quad-\left(p_{f,P}-p_{f,W}\right)\left[\frac{\partial R_{r,W}}{\partial T_{s,P}}\delta T_{s,p}+...+\frac{\partial R_{r,W}}{\partial v_{z,P}}\delta v_{z,P}\right]\\
&\quad-\left(p_{f,P}-p_{f,W}\right)\left[\frac{\partial R_{r,W}}{\partial T_{s,W}}\delta T_{s,W}+...+\frac{\partial R_{r,W}}{\partial v_{z,W}}\delta v_{z,W}\right]\\
&\quad+\left(p_{f,P}-p_{f,S}\right)\left[\frac{\partial R_{a,S}}{\partial T_{s,P}}\delta T_{s,P}+...+\frac{\partial R_{a,S}}{\partial v_{z,P}}\delta v_{z,P}\right]\\
&\quad+\left(p_{f,P}-p_{f,S}\right)\left[\frac{\partial R_{a,S}}{\partial T_{s,S}}\delta T_{s,S}+...+\frac{\partial R_{a,S}}{\partial v_{z,S}}\delta v_{z,S}\right]\\
&\quad+\left(p_{f,P}-p_{f,N}\right)\left[\frac{\partial R_{a,N}}{\partial T_{s,P}}\delta T_{s,P}+...+\frac{\partial R_{a,N}}{\partial v_{z,P}}\delta v_{z,P}\right]\\
&\quad+\left(p_{f,P}-p_{f,N}\right)\left[\frac{\partial R_{a,N}}{\partial T_{s,N}}\delta T_{s,N}+...+\frac{\partial R_{a,N}}{\partial v_{z,N}}\delta v_{z,N}\right]\\
&\quad+\left(p_{f,P}-p_{f,T}-b_{f,T}\right)\left[\frac{\partial R_{l,T}}{\partial T_{s,T}}\delta T_{s,T}+...+\frac{\partial R_{l,T}}{\partial v_{z,T}}\delta v_{z,T}\right]\\
&\quad+\left(p_{f,P}-p_{f,T}-b_{f,T}\right)\left[\frac{\partial R_{l,T}}{\partial T_{s,P}}\delta T_{s,P}+...+\frac{\partial R_{l,T}}{\partial v_{z,P}}\delta v_{z,P}\right]\\
&\quad+\left(p_{f,P}-p_{f,B}-b_{f,B}\right)\left[\frac{\partial R_{l,B}}{\partial T_{s,B}}\delta T_{s,B}+...+\frac{\partial R_{l,B}}{\partial v_{z,B}}\delta v_{z,B}\right]\\
&\quad+\left(p_{f,P}-p_{f,B}-b_{f,B}\right)\left[\frac{\partial R_{l,B}}{\partial T_{s,P}}\delta T_{s,P}+...+\frac{\partial R_{l,B}}{\partial v_{z,P}}\delta v_{z,P}\right]\\
&-R_{l,T}\left[\frac{\partial b_{f,T}}{\partial T_{s,T}}\delta T_{s,T}+...+\frac{\partial b_{f,T}}{\partial v_{z,T}}\delta v_{z,T}\right]-R_{l,T}\left[\frac{\partial b_{f,T}}{\partial T_{s,P}}\delta T_{s,P}+...+\frac{\partial b_{f,T}}{\partial v_{z,P}}\delta v_{z,P}\right]\\
&-R_{l,B}\left[\frac{\partial b_{f,B}}{\partial T_{s,B}}\delta T_{s,B}+...+\frac{\partial b_{f,B}}{\partial v_{z,B}}\delta v_{z,B}\right]-R_{l,B}\left[\frac{\partial b_{f,B}}{\partial T_{s,P}}\delta T_{s,P}+...+\frac{\partial b_{f,B}}{\partial v_{z,P}}\delta v_{z,P}\right]\\
&\quad R_{r,E}\left(\delta p_{f,P}-\delta p_{f,E}\right)+R_{r,W}\left(\delta p_{f,P}-\delta p_{f,W}\right)\\
&\quad+R_{a,N}\left(\delta p_{f,P}-\delta p_{f,N}\right)+R_{r,S}\left(\delta p_{f,P}-\delta p_{f,S}\right)\\
&\quad+R_{l,B}\left(\delta p_{f,P}-\delta p_{f,B}\right)+R_{l,T}\left(\delta p_{f,P}-\delta p_{f,T}\right)\\
&\quad=R_{r,E}\left(p_{f,P}-p_{f,E}\right)+R_{r,W}\left(p_{f,P}-p_{f,W}\right)\\
&\quad+R_{a,N}\left(p_{f,P}-p_{f,N}\right)+R_{r,S}\left(p_{f,P}-p_{f,S}\right)\\
&\quad+R_{l,B}\left(\left(p_{f,P}-p_{f,B}\right)-b_{f,B}\right)+R_{l,T}\left(\left(p_{f,P}-p_{f,T}\right)-b_{f,B}\right)
\end{aligned}
\tag{E.11}
$$

Appendix F

Derivation: Velocity X Equation

The two mass flows in the x direction for the west and east faces are,

$$\begin{aligned} \dot{m}_{x,E} &= R_{r,E}\left(p_{f,P} - p_{f,E}\right) \\ \dot{m}_{x,W} &= R_{r,W}\left(p_{f,W} - p_{f,P}\right) \end{aligned} \tag{F.1}$$

R is the radial flow resistance, with respect to east and west. The velocity definition,

$$v_{x,P} = \frac{m_{x,E} + m_{x,W}}{2} \tag{F.2}$$

Rewriting the equation with definitions,

$$2v_{x,P} = R_{r,E}\left(p_{f,P} - p_{f,E}\right) + R_{r,W}\left(p_{f,W} - p_{f,P}\right) \tag{F.3}$$

Expanding the primary variables

$$\begin{aligned} &2\left(v_{x,P} + \delta v_{x,P}\right) \\ &\qquad = R_{r,E}\left(\left(p_{f,P} + \delta p_{f,P}\right) - \left(p_{f,E} + \delta p_{f,E}\right)\right) \\ &\qquad + R_{r,W}\left(\left(p_{f,W} + \delta p_{f,W}\right) - \left(p_{f,P} + \delta p_{f,P}\right)\right) \end{aligned} \tag{F.4}$$

Moving to the LHS

$$
\begin{aligned}
2\delta v_{x,P} + 2v_{x,P}& \\
&- R_{r,E}\left(\delta p_{f,P} - \delta p_{f,E}\right) - R_{r,W}\left(\delta p_{f,W} - \delta p_{f,P}\right) \\
&- R_{r,E}\left(p_{f,P} - p_{f,E}\right) - R_{r,W}\left(p_{f,W} - p_{f,P}\right) \\
&= 0
\end{aligned}
\tag{F.5}
$$

Expanding the coefficients, ignoring higher order terms

$$
\begin{aligned}
2\delta v_{x,P} - R_{r,E}\left(\delta p_{f,P} - \delta p_{f,E}\right) - R_{r,W}\left(\delta p_{f,W} - \delta p_{f,P}\right)& \\
- \delta R_{r,E}\left(p_{f,P} - p_{f,E}\right) - \delta R_{r,W}\left(p_{f,W} - p_{f,P}\right)& \\
= -2v_{x,P} + R_{r,E}\left(p_{f,P} - p_{f,E}\right) + R_{r,W}\left(p_{f,W} - p_{f,P}\right)&
\end{aligned}
\tag{F.6}
$$

Expanding the coefficients completely, some more algebra

$$
\begin{aligned}
2\delta v_{x,P} - R_{r,E}\left(\delta p_{f,P} - \delta p_{f,E}\right) - R_{r,W}\left(\delta p_{f,W} - \delta p_{f,p}\right)& \\
- \left(p_{f,P} - p_{f,E}\right)\left[\frac{\partial R_{r,E}}{\partial T_{s,P}}\delta T_{s,p} + ... + \frac{\partial R_{r,E}}{\partial v_{z,P}}\delta v_{z,P}\right]& \\
- \left(p_{f,P} - p_{f,E}\right)\left[\frac{\partial R_{r,E}}{\partial T_{s,E}}\delta T_{s,E} + ... + \frac{\partial R_{r,E}}{\partial v_{z,E}}\delta v_{z,E}\right]& \\
- \left(p_{f,W} - p_{f,P}\right)\left[\frac{\partial R_{r,W}}{\partial T_{s,P}}\delta T_{s,p} + ... + \frac{\partial R_{r,W}}{\partial v_{z,P}}\delta v_{z,P}\right]& \\
- \left(p_{f,W} - p_{f,P}\right)\left[\frac{\partial R_{r,W}}{\partial T_{s,W}}\delta T_{s,W} + ... + \frac{\partial R_{r,W}}{\partial v_{z,W}}\delta v_{z,W}\right]& \\
= -2v_{x,P} + R_{r,E}\left(p_{f,P} - p_{f,E}\right) + R_{r,W}\left(p_{f,W} - p_{f,P}\right)&
\end{aligned}
\tag{F.7}
$$

Appendix G
Derivation: Velocity Y Equation

The two mass flows in the y direction for the north and south faces are,

$$\begin{aligned} \dot{m}_{y,S} &= R_{a,S}\left(p_{f,P} - p_{f,S}\right) \\ \dot{m}_{y,N} &= R_{a,N}\left(p_{f,N} - p_{f,P}\right) \end{aligned} \tag{G.1}$$

R is the azimuthal flow resistance, with respect to north and south. The velocity definition,

$$v_{y,P} = \frac{m_{y,N} + m_{y,S}}{2} \tag{G.2}$$

Rewriting the equation with definitions,

$$2v_{y,P} = R_{a,S}\left(p_{f,P} - p_{f,S}\right) + R_{a,N}\left(p_{f,N} - p_{f,P}\right) \tag{G.3}$$

Expanding the primary variables

$$\begin{aligned} &2\left(v_{y,P} + \delta v_{y,P}\right) \\ &\qquad = R_{a,S}\left(\left(p_{f,P} + \delta p_{f,P}\right) - \left(p_{f,S} + \delta p_{f,S}\right)\right) \\ &\qquad + R_{a,N}\left(\left(p_{f,N} + \delta p_{f,N}\right) - \left(p_{f,P} + \delta p_{f,P}\right)\right) \end{aligned} \tag{G.4}$$

Moving to the LHS

$$
\begin{aligned}
2\delta v_{y,P} + 2v_{y,P}& \\
&- R_{a,S}\left(\delta p_{f,P} - \delta p_{f,S}\right) - R_{a,N}\left(\delta p_{f,N} - \delta p_{f,P}\right) \\
&- R_{a,S}\left(p_{f,P} - p_{f,S}\right) - R_{a,N}\left(p_{f,N} - p_{f,P}\right) = 0
\end{aligned}
\tag{G.5}
$$

Expanding the coefficients, ignoring higher order terms

$$
\begin{aligned}
2\delta v_{y,P} - R_{a,S}\left(\delta p_{f,P} - \delta p_{f,S}\right) - R_{a,N}\left(\delta p_{f,N} - \delta p_{f,P}\right)& \\
- \delta R_{a,S}\left(p_{f,P} - p_{f,S}\right) - \delta R_{a,N}\left(p_{f,N} - p_{f,P}\right)& \\
= -2v_{y,P} + R_{a,S}\left(p_{f,P} - p_{f,S}\right) + R_{a,N}\left(p_{f,N} - p_{f,P}\right)&
\end{aligned}
\tag{G.6}
$$

Some more algebra, expanding coefficients

$$
\begin{aligned}
2\delta v_{y,p} - R_{a,S}\left(\delta p_{f,P} - \delta p_{f,S}\right) - R_{a,N}\left(\delta p_{f,N} - \delta p_{f,P}\right)& \\
- \left(p_{f,P} - p_{f,S}\right)\left[\frac{\partial R_{a,S}}{\partial T_{s,P}}\delta T_{s,P} + \ldots + \frac{\partial R_{a,S}}{\partial v_{z,P}}\delta v_{z,P}\right]& \\
- \left(p_{f,P} - p_{f,S}\right)\left[\frac{\partial R_{a,S}}{\partial T_{s,S}}\delta T_{s,S} + \ldots + \frac{\partial R_{a,S}}{\partial v_{z,S}}\delta v_{z,S}\right]& \\
- \left(p_{f,N} - p_{f,P}\right)\left[\frac{\partial R_{a,N}}{\partial T_{s,P}}\delta T_{s,P} + \ldots + \frac{\partial R_{a,N}}{\partial v_{z,P}}\delta v_{z,P}\right]& \\
- \left(p_{f,N} - p_{f,P}\right)\left[\frac{\partial R_{a,N}}{\partial T_{s,N}}\delta T_{s,N} + \ldots + \frac{\partial R_{a,N}}{\partial v_{z,N}}\delta v_{z,N}\right]& \\
= -2v_{y,P} + R_{a,S}\left(p_{f,R} - p_{f,E}\right) + R_{a,N}\left(p_{f,N} - p_{f,P}\right)&
\end{aligned}
\tag{G.7}
$$

Appendix H

Derivation: Velocity Z Equation

The two mass flows in the x direction for the west and east faces are,

$$\begin{aligned} \dot{m}_{z,T} &= R_{l,T}\left(p_{f,P} - p_{f,T} - b_{f,T}\right) \\ \dot{m}_{z,B} &= R_{l,B}\left(p_{f,B} - p_{f,P} - b_{f,P}\right) \end{aligned} \tag{H.1}$$

The velocity definition,

$$v_{z,P} = \frac{m_{z,B} + m_{z,T}}{2} \tag{H.2}$$

Rewriting the equation with definitions,

$$2v_{z,P} = R_{l,T}\left(p_{f,P} - p_{f,T} - b_{f,T}\right) + R_{l,B}\left(p_{f,B} - p_{f,P} - b_{f,B}\right) \tag{H.3}$$

Expanding the primary variables and coefficients.,

$$\begin{aligned} 2\left(v_{z,P} + \delta v_{z,P}\right) = & \\ & + R_{l,T}\left(\left(p_{f,P} + \delta p_{f,P}\right) - \left(p_{f,T} + \delta p_{f,T}\right) - \left(b_{f,T} + \delta b_{f,T}\right)\right) \\ & + R_{l,B}\left(\left(p_{f,B} + \delta p_{f,B}\right) - \left(p_{f,P} + \delta p_{f,P}\right) - \left(b_{f,B} + \delta b_{f,B}\right)\right) \\ & + \delta R_{l,T}\left(p_{f,P} - p_{f,T} - b_{f,T}\right) + \delta R_{l,B}\left(p_{f,B} - p_{f,P} - b_{f,B}\right) \end{aligned} \tag{H.4}$$

Moving to the RHS and LHS

$$\begin{aligned}2\delta v_{z,P} &- R_{l,T}\left(\delta p_{f,P}-\delta p_{f,T}-\delta b_{f,T}\right)-R_{l,B}\left(\delta p_{f,B}-\delta p_{f,P}-\delta b_{f,B}\right)\\ &-\delta R_{l,T}\left(p_{f,P}-p_{f,T}-b_{f,T}\right)-\delta R_{l,B}\left(p_{f,B}-p_{f,P}-b_{f,B}\right)\\ &=-2v_{z,P}+R_{l,T}\left(p_{f,P}-p_{f,T}-b_{f,T}\right)+R_{l,B}\left(p_{f,B}-p_{f,P}-b_{f,B}\right)\end{aligned} \tag{H.5}$$

Separating the coefficients,

$$\begin{aligned}2\delta v_{z,P} &- R_{l,T}\left(\delta p_{f,P}-\delta p_{f,T}\right)-R_{l,B}\left(\delta p_{f,B}-\delta p_{f,P}\right)\\ &+R_{l,T}\delta b_{f,T}+R_{l,B}\delta b_{f,B}\\ &-\left(p_{f,P}-p_{f,T}-b_{f,T}\right)\delta R_{l,T}\\ &-\left(p_{f,B}-p_{f,P}-b_{f,B}\right)\delta R_{l,B}\\ &=-2v_{z,P}+R_{l,T}\left(p_{f,P}-p_{f,T}-b_{f,T}\right)+R_{l,B}\left(p_{f,B}-p_{f,P}-b_{f,B}\right)\end{aligned} \tag{H.6}$$

Expanding each coefficient,

$$\begin{aligned}2\delta v_{z,P} &- R_{l,T}\left(\delta p_{f,P}-\delta p_{f,T}\right)-R_{l,B}\left(\delta p_{f,B}-\delta p_{f,P}\right)\\ &+R_{l,T}\left[\frac{\partial b_{f,T}}{\partial T_{s,T}}\delta T_{s,T}+...+\frac{\partial b_{f,T}}{\partial v_{z,T}}\delta v_{z,T}\right]+R_{l,T}\left[\frac{\partial b_{f,T}}{\partial T_{s,P}}\delta T_{s,P}+...+\frac{\partial b_{f,T}}{\partial v_{z,P}}\delta v_{z,P}\right]\\ &+R_{l,B}\left[\frac{\partial b_{f,B}}{\partial T_{s,B}}\delta T_{s,B}+...+\frac{\partial b_{f,B}}{\partial v_{z,B}}\delta v_{z,B}\right]+R_{l,B}\left[\frac{\partial b_{f,B}}{\partial T_{s,P}}\delta T_{s,P}+...+\frac{\partial b_{f,B}}{\partial v_{z,P}}\delta v_{z,P}\right]\\ &-\left(p_{f,P}-p_{f,T}-b_{f,T}\right)\left[\frac{\partial R_{l,T}}{\partial T_{s,T}}\delta T_{s,T}+...+\frac{\partial R_{l,T}}{\partial v_{z,T}}\delta v_{z,T}\right]\\ &-\left(p_{f,P}-p_{f,T}-b_{f,T}\right)\left[\frac{\partial R_{l,T}}{\partial T_{s,P}}\delta T_{s,P}+...+\frac{\partial R_{l,T}}{\partial v_{z,P}}\delta v_{z,P}\right]\\ &-\left(p_{f,B}-p_{f,P}-b_{f,B}\right)\left[\frac{\partial R_{l,B}}{\partial T_{s,B}}\delta T_{s,B}+...+\frac{\partial R_{l,B}}{\partial v_{z,B}}\delta v_{z,B}\right]\\ &-\left(p_{f,B}-p_{f,P}-b_{f,B}\right)\left[\frac{\partial R_{l,B}}{\partial T_{s,P}}\delta T_{s,P}+...+\frac{\partial R_{l,B}}{\partial v_{z,P}}\delta v_{z,P}\right]\\ &=-2v_{z,P}+R_{l,T}\left(p_{f,P}-p_{f,T}-b_{f,T}\right)+R_{l,B}\left(p_{f,B}-p_{f,P}-b_{f,B}\right)\end{aligned} \tag{H.7}$$

Appendix I
Derivation: Neutronics

Beginning with the cylindrical time-dependent flux equation,

$$\begin{aligned}
V_p \frac{1}{v_g} \frac{\phi_{g,P}^{n} - \phi_{g,P}^{n-1}}{\Delta t} = \chi_{gd} \sum_{q=1}^{Q} \lambda_q C_q \\
+ \tilde{D}_{g,E}\phi_{g,E}^{n} + \tilde{D}_{g,W}\phi_{g,W}^{n} + \tilde{D}_{g,N}\phi_{g,N}^{n} \\
+ \tilde{D}_{g,S}\phi_{g,S}^{n} + \tilde{D}_{g,B}\phi_{g,B}^{n} + \tilde{D}_{g,T}\phi_{g,T}^{n} \\
- \left(\tilde{D}_{g,E} + \tilde{D}_{g,W} + \tilde{D}_{g,N} + \tilde{D}_{g,S} + \tilde{D}_{g,B} + \tilde{D}_{g,T}\right)\phi_{g,P}^{n} \\
- \left(\Sigma_{a,g} + \sum_{g'\neq g}^{G} \Sigma_{s,gg'}\right) V_p \phi_{g,P}^{n} \\
+ \left(\sum_{g'=1(\neq g)}^{G} \Sigma_{s,g'g} + \beta_p \frac{\chi_{gp}}{k_{eff}} \sum_{g'=1}^{G} \nu\Sigma_{f,g'}\right) V_p \phi_{g',P}^{n}
\end{aligned} \tag{I.1}$$

The diffusion coefficients,

$$\begin{array}{ll}
\tilde{D}_{g,E} = D_e \frac{r_e}{\partial r_e} \Delta\theta\Delta z & \tilde{D}_{g,W} = D_w \frac{r_w}{\partial r_w} \Delta\theta\Delta z \\
\tilde{D}_{g,N} = D_n \frac{1}{r_n} \frac{1}{\partial\theta_n} \Delta r\Delta z & \tilde{D}_{g,S} = D_s \frac{1}{r_s} \frac{1}{\partial\theta_s} \Delta r\Delta z \\
\tilde{D}_{g,B} = D_e \frac{1}{\partial z_b} \frac{r_e^2 - r_w^2}{2} \Delta\theta & \tilde{D}_{g,T} = D_e \frac{1}{\partial z_t} \frac{r_e^2 - r_w^2}{2} \Delta\theta
\end{array} \tag{I.2}$$

The diffusion coefficient, cross section, and eigenvalue need to be addressed. That derivation will follow. The cross section and diffusion coefficient dependencies

$$\Sigma_s, \Sigma_a, \nu\Sigma_f, D, \frac{1}{v}, \beta_p = f\left(T_m, T_d\right) \tag{I.3}$$

The definitions for the cross sections and fluxes

$$\begin{aligned}\phi^{k+1} &= \phi^{(k)} + \delta\phi & \Lambda^{k+1} &= \Lambda^{(k)} + \delta\Lambda \\ \Sigma_a^{k+1} &= \Sigma_a^{(k)} + \delta\Sigma_a & \Sigma_s^{k+1} &= \Sigma_s^{(k)} + \delta\Sigma_s \\ \nu\Sigma_f^{k+1} &= \nu\Sigma_f^{(k)} + \delta\nu\Sigma_f & D^{k+1} &= D^{(k)} + \delta D\end{aligned} \tag{I.4}$$

Inserting these into the equation,

$$\begin{aligned}
&- \left(\tilde{D}_{g,E}^{(k)} + \delta\tilde{D}_{g,E}\right)\left(\phi_{g,E}^{(k)} + \delta\phi_{g,E}\right) - \left(\tilde{D}_{g,W}^{(k)} + \delta\tilde{D}_{g,W}\right)\left(\phi_{g,W}^{(k)} + \delta\phi_{g,W}\right) \\
&\quad - \left(\tilde{D}_{g,N}^{(k)} + \delta\tilde{D}_{g,N}\right)\left(\phi_{g,N}^{(k)} + \delta\phi_{g,N}\right) - \left(\tilde{D}_{g,S}^{(k)} + \delta\tilde{D}_{g,S}\right)\left(\phi_{g,S}^{(k)} + \delta\phi_{g,S}\right) \\
&\quad - \left(\tilde{D}_{g,T}^{(k)} + \delta\tilde{D}_{g,T}\right)\left(\phi_{g,B}^{(k)} + \delta\phi_{g,B}\right) - \left(\tilde{D}_{g,T}^{(k)} + \delta\tilde{D}_{g,T}\right)\left(\phi_{g,T}^{(k)} + \delta\phi_{g,T}\right) \\
&\quad + \left(\tilde{D}_{g,E}^{(k)} + \delta\tilde{D}_{g,E}\right)\left(\phi_{g,P}^{(k)} + \delta\phi_{g,P}\right) + \left(\tilde{D}_{g,W}^{(k)} + \delta\tilde{D}_{g,W}\right)\left(\phi_{g,P}^{(k)} + \delta\phi_{g,P}\right) \\
&\quad + \left(\tilde{D}_{g,N}^{(k)} + \delta\tilde{D}_{g,N}\right)\left(\phi_{g,P}^{(k)} + \delta\phi_{g,P}\right) + \left(\tilde{D}_{g,S}^{(k)} + \delta\tilde{D}_{g,S}\right)\left(\phi_{g,P}^{(k)} + \delta\phi_{g,P}\right) \\
&\quad + \left(\tilde{D}_{g,T}^{(k)} + \delta\tilde{D}_{g,T}\right)\left(\phi_{g,P}^{(k)} + \delta\phi_{g,P}\right) + \left(\tilde{D}_{g,T}^{(k)} + \delta\tilde{D}_{g,T}\right)\left(\phi_{g,P}^{(k)} + \delta\phi_{g,P}\right) \\
&\quad + V_p\left(\Sigma_{a,g}^{(k)} + \delta\Sigma_{a,g}\right)\left(\phi_{g,P}^{(k)} + \delta\phi_{g,P}\right) \\
&\quad + V_p\left[\sum_{g'\neq g}^{G}\left(\Sigma_{s,gg'}^{(k)} + \delta\Sigma_{s,gg'}\right)\right]\left(\phi_{g,P}^{(k)} + \delta\phi_{g,P}\right) \\
&= V_p \sum_{g'=1(\neq g)}^{G}\left[\left(\Sigma_{s,g'g}^{(k)} + \delta\Sigma_{s,g'g}\right)\left(\phi_{g',P}^{(k)} + \delta\phi_{g',P}\right)\right] \\
&\quad + V_p\chi_{gp}\left(\Lambda^{(k)} + \delta\Lambda\right)\sum_{g'=1}^{G}\left[\left(\nu\Sigma_{f,g'}^{(k)} + \nu\delta\Sigma_{f,g'}\right)\left(\phi_{g',P}^{(k)} + \delta\phi_{g',P}\right)\right]
\end{aligned} \tag{I.5}$$

Multiplying out,

$$
\begin{aligned}
&-\left(\tilde{D}_{g,E}^{(k)}\phi_{g,E}^{(k)}+\delta\tilde{D}_{g,E}\phi_{g,E}^{(k)}+\tilde{D}_{g,E}^{(k)}\delta\phi_{g,E}\right)-\left(\tilde{D}_{g,W}^{(k)}\phi_{g,W}^{(k)}+\delta\tilde{D}_{g,W}\phi_{g,W}^{(k)}+\tilde{D}_{g,W}^{(k)}\delta\phi_{g,W}\right)\\
&-\left(\tilde{D}_{g,N}^{(k)}\phi_{g,N}^{(k)}+\delta\tilde{D}_{g,N}\phi_{g,N}^{(k)}+\tilde{D}_{g,N}^{(k)}\delta\phi_{g,N}\right)-\left(\tilde{D}_{g,S}^{(k)}\phi_{g,S}^{(k)}+\delta\tilde{D}_{g,S}\phi_{g,S}^{(k)}+\tilde{D}_{g,S}^{(k)}\delta\phi_{g,S}\right)\\
&-\left(\tilde{D}_{g,B}^{(k)}\phi_{g,B}^{(k)}+\delta\tilde{D}_{g,B}\phi_{g,B}^{(k)}+\tilde{D}_{g,B}^{(k)}\delta\phi_{g,B}\right)-\left(\tilde{D}_{g,T}^{(k)}\phi_{g,T}^{(k)}+\delta\tilde{D}_{g,T}\phi_{g,T}^{(k)}+\tilde{D}_{g,T}^{(k)}\delta\phi_{g,T}\right)\\
&+\left(\tilde{D}_{g,E}^{(k)}\phi_{g,P}^{(k)}+\delta\tilde{D}_{g,E}\phi_{g,P}^{(k)}+\tilde{D}_{g,E}^{(k)}\delta\phi_{g,P}\right)+\left(\tilde{D}_{g,W}^{(k)}\phi_{g,P}^{(k)}+\delta\tilde{D}_{g,W}\phi_{g,P}^{(k)}+\tilde{D}_{g,W}^{(k)}\delta\phi_{g,P}\right)\\
&+\left(\tilde{D}_{g,N}^{(k)}\phi_{g,P}^{(k)}+\delta\tilde{D}_{g,N}\phi_{g,P}^{(k)}+\tilde{D}_{g,N}^{(k)}\delta\phi_{g,P}\right)+\left(\tilde{D}_{g,S}^{(k)}\phi_{g,P}^{(k)}+\delta\tilde{D}_{g,S}\phi_{g,P}^{(k)}+\tilde{D}_{g,S}^{(k)}\delta\phi_{g,P}\right)\\
&+\left(\tilde{D}_{g,B}^{(k)}\phi_{g,P}^{(k)}+\delta\tilde{D}_{g,B}\phi_{g,P}^{(k)}+\tilde{D}_{g,B}^{(k)}\delta\phi_{g,P}\right)+\left(\tilde{D}_{g,T}^{(k)}\phi_{g,P}^{(k)}+\delta\tilde{D}_{g,T}\phi_{g,P}^{(k)}+\tilde{D}_{g,T}^{(k)}\delta\phi_{g,P}\right)\\
&+V_p\left(\Sigma_{a,g}^{(k)}\phi_{g,P}^{(k)}+\delta\Sigma_{a,g}\phi_{g,P}^{(k)}+\Sigma_{a,g}^{(k)}\delta\phi_{g,P}\right)\\
&+V_p\left(\left(\sum_{g'\neq g}^{G}\Sigma_{s,gg'}^{(k)}\right)\phi_{g,P}^{(k)}+\left(\sum_{g'\neq g}^{G}\delta\Sigma_{s,gg'}\right)\phi_{g,P}^{(k)}+\left(\sum_{g'\neq g}^{G}\Sigma_{s,gg'}^{(k)}\right)\delta\phi_{g,P}\right)\\
&=V_p\left[\sum_{g'=1(\neq g)}^{G}\left(\Sigma_{s,g'g}^{(k)}\phi_{g',P}^{(k)}\right)+\sum_{g'=1(\neq g)}^{G}\left(\delta\Sigma_{s,g'g}\phi_{g',P}^{(k)}\right)+\sum_{g'=1(\neq g)}^{G}\left(\Sigma_{s,g'g}^{(k)}\delta\phi_{g',P}\right)\right]\\
&+V_p\chi_{gp}\left(\Lambda^{(k)}+\delta\Lambda\right)\left[\sum_{g'=1}^{G}\left(\nu\Sigma_{f,g'}^{(k)}\phi_{g',P}^{(k)}\right)+\sum_{g'=1}^{G}\left(\nu\delta\Sigma_{f,g'}\phi_{g',P}^{(k)}\right)+\sum_{g'=1}^{G}\left(\nu\Sigma_{f,g'}^{(k)}\delta\phi_{g',P}\right)\right]
\end{aligned}
\tag{I.6}
$$

Condensing Terms,

$$
\begin{aligned}
0 = &\left(\tilde{D}_{g,E}^{(k)}\phi_{g,E}^{(k)} + \tilde{D}_{g,W}^{(k)}\phi_{g,W}^{(k)} + \tilde{D}_{g,N}^{(k)}\phi_{g,N}^{(k)} + \tilde{D}_{g,S}^{(k)}\phi_{g,S}^{(k)} + \tilde{D}_{g,B}^{(k)}\phi_{g,B}^{(k)} + \tilde{D}_{g,T}^{(k)}\phi_{g,T}^{(k)}\right) \\
&+ \left(\delta\tilde{D}_{g,E}\phi_{g,E}^{(k)} + \delta\tilde{D}_{g,W}\phi_{g,W}^{(k)} + \delta\tilde{D}_{g,N}\phi_{g,N}^{(k)} + \delta\tilde{D}_{g,S}\phi_{g,S}^{(k)} + \delta\tilde{D}_{g,B}\phi_{g,B}^{(k)} + \delta\tilde{D}_{g,T}\phi_{g,T}^{(k)}\right) \\
&+ \left(\tilde{D}_{g,E}^{(k)}\delta\phi_{g,E} + \tilde{D}_{g,W}^{(k)}\delta\phi_{g,W} + \tilde{D}_{g,N}^{(k)}\delta\phi_{g,N} + \tilde{D}_{g,S}^{(k)}\delta\phi_{g,S} + \tilde{D}_{g,B}^{(k)}\delta\phi_{g,B} + \tilde{D}_{g,T}^{(k)}\delta\phi_{g,T}\right) \\
&+ \left[-\left(\tilde{D}_{g,E}^{(k)} + \tilde{D}_{g,W}^{(k)} + \tilde{D}_{g,N}^{(k)} + \tilde{D}_{g,S}^{(k)} + \tilde{D}_{g,B}^{(k)} + \tilde{D}_{g,T}^{(k)}\right) - V_p\left(\Sigma_{a,g}^{(k)} + \sum_{g'\neq g}^{G}\Sigma_{s,gg'}^{(k)}\right)\right]\phi_{g,P}^{(k)} \\
&+ \left[-\left(\delta\tilde{D}_{g,E} + \delta\tilde{D}_{g,W} + \delta\tilde{D}_{g,N} + \delta\tilde{D}_{g,S} + \delta\tilde{D}_{g,B} + \delta\tilde{D}_{g,T}\right) - V_p\left(\delta\Sigma_{a,g} + \sum_{g'\neq g}^{G}\delta\Sigma_{s,gg'}\right)\right]\phi_{g,P}^{(k)} \\
&+ \left[-\left(\tilde{D}_{g,E}^{(k)} + \tilde{D}_{g,W}^{(k)} + \tilde{D}_{g,N}^{(k)} + \tilde{D}_{g,S}^{(k)} + \tilde{D}_{g,B}^{(k)} + \tilde{D}_{g,T}^{(k)}\right) - V_p\left(\Sigma_{a,g}^{(k)} + \sum_{g'\neq g}^{G}\Sigma_{s,gg'}^{(k)}\right)\right]\delta\phi_{g,P} \\
&+ V_p\left[\sum_{g'=1(\neq g)}^{G}\left(\Sigma_{s,g'g}^{(k)}\phi_{g',P}^{(k)}\right) + \sum_{g'=1(\neq g)}^{G}\left(\delta\Sigma_{s,g'g}\phi_{g',P}^{(k)}\right) + \sum_{g'=1(\neq g)}^{G}\left(\Sigma_{s,g'g}^{(k)}\delta\phi_{g',P}\right)\right] \\
&+ V_p\left(\beta_{pr}^{(k)} + \delta\beta_{pr}\right)\left(\Lambda^{(k)} + \delta\Lambda\right)\left[\sum_{g'=1}^{G}\left(\nu\Sigma_{f,g'}^{(k)}\phi_{g',P}^{(k)}\right) + \sum_{g'=1}^{G}\left(\nu\delta\Sigma_{f,g'}\phi_{g',P}^{(k)}\right) + \sum_{g'=1}^{G}\left(\nu\Sigma_{f,g'}^{(k)}\delta\phi_{g',P}\right)\right]
\end{aligned}
\tag{I.7}
$$

Moving the non-derivative terms to the source term,

$$
\begin{aligned}
S^{(k)} = &\\
&+ \left(\tilde{D}_{g,E}^{(k)}\phi_{g,E}^{(k)} + \tilde{D}_{g,W}^{(k)}\phi_{g,W}^{(k)} + \tilde{D}_{g,N}^{(k)}\phi_{g,N}^{(k)} + \tilde{D}_{g,S}^{(k)}\phi_{g,S}^{(k)} + \tilde{D}_{g,B}^{(k)}\phi_{g,B}^{(k)} + \tilde{D}_{g,T}^{(k)}\phi_{g,T}^{(k)}\right) \\
&+ \left[-\left(\tilde{D}_{g,E}^{(k)} + \tilde{D}_{g,W}^{(k)} + \tilde{D}_{g,N}^{(k)} + \tilde{D}_{g,S}^{(k)} + \tilde{D}_{g,B}^{(k)} + \tilde{D}_{g,T}^{(k)}\right) - V_p\left(\Sigma_{a,g}^{(k)} + \sum_{g'\neq g}^{G}\Sigma_{s,gg'}^{(k)}\right)\right]\phi_{g,P}^{(k)} \\
&+ V_p\left[\sum_{g'=1(\neq g)}^{G}\left(\Sigma_{s,g'g}^{(k)}\phi_{g',P}^{(k)}\right)\right] + V_p\chi_{gp}\Lambda^{(k)}\left[\sum_{g'=1}^{G}\left(\nu\Sigma_{f,g'}^{(k)}\phi_{g',P}^{(k)}\right)\right]
\end{aligned}
\tag{I.8}
$$

Simplifying and collecting terms,

$$
\begin{aligned}
0 &= S^{(k)} \\
&+ \left(\delta\tilde{D}_{g,E}\phi_{g,E}^{(k)} + \delta\tilde{D}_{g,W}\phi_{g,W}^{(k)} + \delta\tilde{D}_{g,N}\phi_{g,N}^{(k)} + \delta\tilde{D}_{g,S}\phi_{g,S}^{(k)} + \delta\tilde{D}_{g,B}\phi_{g,B}^{(k)} + \delta\tilde{D}_{g,T}\phi_{g,T}^{(k)}\right) \\
&+ \left[-\left(\delta\tilde{D}_{g,E} + \delta\tilde{D}_{g,W} + \delta\tilde{D}_{g,N} + \delta\tilde{D}_{g,S} + \delta\tilde{D}_{g,B} + \delta\tilde{D}_{g,T}\right) - V_p\left(\delta\Sigma_{a,g} + \sum_{g'\neq g}^{G}\delta\Sigma_{s,gg'}\right)\right]\phi_{g,P}^{(k)} \\
&+ V_p \sum_{g'=1(\neq g)}^{G}\left(\delta\Sigma_{s,g'g}\phi_{g',P}^{(k)}\right) + V_p\chi_{gp}\Lambda^{(k)}\sum_{g'=1}^{G}\left(\nu\delta\Sigma_{f,g'}\phi_{g',P}^{(k)}\right) \\
&+ \left(\tilde{D}_{g,E}^{(k)}\delta\phi_{g,E} + \tilde{D}_{g,W}^{(k)}\delta\phi_{g,W} + \tilde{D}_{g,N}^{(k)}\delta\phi_{g,N} + \tilde{D}_{g,S}^{(k)}\delta\phi_{g,S} + \tilde{D}_{g,B}^{(k)}\delta\phi_{g,B} + \tilde{D}_{g,T}^{(k)}\delta\phi_{g,T}\right) \\
&+ \left[-\left(\tilde{D}_{g,E}^{(k)} + \tilde{D}_{g,W}^{(k)} + \tilde{D}_{g,N}^{(k)} + \tilde{D}_{g,S}^{(k)} + \tilde{D}_{g,B}^{(k)} + \tilde{D}_{g,T}^{(k)}\right) - V_p\left(\Sigma_{a,g}^{(k)} + \sum_{g'\neq g}^{G}\Sigma_{s,gg'}^{(k)}\right)\right]\delta\phi_{g,P} \\
&+ V_p \sum_{g'=1(\neq g)}^{G}\left(\Sigma_{s,g'g}^{(k)}\delta\phi_{g',P}\right) + V_p\chi_{gp}\Lambda^{(k)}\sum_{g'=1}^{G}\left(\nu\Sigma_{f,g'}^{(k)}\delta\phi_{g',P}\right) \\
&+ V_p\chi_{gp}\delta\Lambda\sum_{g'=1}^{G}\left(\nu\Sigma_{f,g'}^{(k)}\phi_{g',P}^{(k)}\right)
\end{aligned} \tag{I.9}
$$

Further simplifications,

$$
\begin{aligned}
0 = {} & S^{(k)} \\
& + \left(\tilde{D}^{(k)}_{g,E}\delta\phi_{g,E} + \tilde{D}^{(k)}_{g,W}\delta\phi_{g,W} + \tilde{D}^{(k)}_{g,N}\delta\phi_{g,N} + \tilde{D}^{(k)}_{g,S}\delta\phi_{g,S} + \tilde{D}^{(k)}_{g,B}\delta\phi_{g,B} + \tilde{D}^{(k)}_{g,T}\delta\phi_{g,T}\right) \\
& + \left[-\left(\tilde{D}^{(k)}_{g,E} + \tilde{D}^{(k)}_{g,W} + \tilde{D}^{(k)}_{g,N} + \tilde{D}^{(k)}_{g,S} + \tilde{D}^{(k)}_{g,B} + \tilde{D}^{(k)}_{g,T}\right) - V_p\left(\Sigma^{(k)}_{a,g} + \sum_{g'\neq g}^{G}\Sigma^{(k)}_{s,gg'}\right)\right]\delta\phi_{g,P} \\
& + V_p\left[\sum_{g'=1(\neq g)}^{G}\Sigma^{(k)}_{s,g'g} + \chi_{gp}\Lambda^{(k)}\sum_{g'=1}^{G}\nu\Sigma^{(k)}_{f,g'}\right]\delta\phi_{g',P} \\
& + \left(\phi^{(k)}_{g,E} - \phi^{(k)}_{g,P}\right)\delta\tilde{D}_{g,E} + \left(\phi^{(k)}_{g,W} - \phi^{(k)}_{g,P}\right)\delta\tilde{D}_{g,W} \\
& + \left(\phi^{(k)}_{g,N} - \phi^{(k)}_{g,P}\right)\delta\tilde{D}_{g,S} + \left(\phi^{(k)}_{g,S} - \phi^{(k)}_{g,P}\right)\delta\tilde{D}_{g,S} \\
& + \left(\phi^{(k)}_{g,B} - \phi^{(k)}_{g,P}\right)\delta\tilde{D}_{g,B} + \left(\phi^{(k)}_{g,T} - \phi^{(k)}_{g,P}\right)\delta\tilde{D}_{g,T} \\
& - V_p\phi^{(k)}_{g,P}\delta\Sigma_{a,g} - V_p\phi^{(k)}_{g,P}\sum_{g'=1(\neq g)}^{G}\delta\Sigma_{s,gg'} \\
& + V_p\sum_{g'=1(\neq g)}^{G}\phi^{(k)}_{g',P}\delta\Sigma_{s,g'g} + V_p\chi_{gp}\sum_{g'=1}^{G}\phi^{(k)}_{g',P}\nu\delta\Sigma_{f,g'} \\
& + V_p\chi_{gp}\sum_{g'=1}^{G}\left(\nu\Sigma^{(k)}_{f,g'}\phi^{(k)}_{g',P}\right)\delta\Lambda
\end{aligned}
\tag{I.10}
$$

Expanding each of the terms by the cross section by TH dependence,

$$\Sigma, D = f\left(T_D, T_M\right) \tag{I.11}$$

Inputting the expansions into the equation,

$$
\begin{aligned}
&+ \left(\phi_{g,E}^{(k)} - \phi_{g,P}^{(k)}\right) \left[\left(\frac{\partial \tilde{D}_{g,E}}{\partial T_{D,P}}\right)^{n-1} \delta T_{D,P} + \left(\frac{\partial \tilde{D}_{g,E}}{\partial T_{D,E}}\right)^{n-1} \delta T_{D,E} \right] \\
&\quad + \left(\phi_{g,E}^{(k)} - \phi_{g,P}^{(k)}\right) \left[\left(\frac{\partial \tilde{D}_{g,E}}{\partial T_{M,P}}\right)^{n-1} \delta T_{M,P} + \left(\frac{\partial \tilde{D}_{g,E}}{\partial T_{M,E}}\right)^{n-1} \delta T_{M,E} \right] \\
&\quad \ldots W, N, S, B, \ldots \\
&\quad + \left(\phi_{g,T}^{(k)} - \phi_{g,P}^{(k)}\right) \left[\left(\frac{\partial \tilde{D}_{g,T}}{\partial T_{D,P}}\right)^{n-1} \delta T_{D,P} + \left(\frac{\partial \tilde{D}_{g,T}}{\partial T_{D,T}}\right)^{n-1} \delta T_{D,T} \right] \\
&\quad + \left(\phi_{g,T}^{(k)} - \phi_{g,P}^{(k)}\right) \left[\left(\frac{\partial \tilde{D}_{g,T}}{\partial T_{M,P}}\right)^{n-1} \delta T_{M,P} + \left(\frac{\partial \tilde{D}_{g,T}}{\partial T_{M,T}}\right)^{n-1} \delta T_{M,T} \right] \\
&\quad - V_p \phi_{g,P}^{(k)} \left(\left(\frac{\partial \Sigma_a}{\partial T_D}\right)^{n-1} \delta T_{D,P} + \left(\frac{\partial \Sigma_a}{\partial T_M}\right)^{n-1} \delta T_{M,P} \right) \\
&\quad - V_p \phi_{g,P}^{(k)} \sum_{g'=1(\neq g)}^{G} \left(\left(\frac{\partial \Sigma_{s,gg'}}{\partial T_D}\right)^{n-1} \delta T_{D,P} + \left(\frac{\partial \Sigma_{s,gg'}}{\partial T_M}\right)^{n-1} \delta T_{M,P} \right) \\
&\quad + V_p \sum_{g'=1(\neq g)}^{G} \phi_{g',P}^{(k)} \left(\left(\frac{\partial \Sigma_{s,g'g}}{\partial T_D}\right)^{n-1} \delta T_{D,P} + \left(\frac{\partial \Sigma_{s,g'g}}{\partial T_M}\right)^{n-1} \delta T_{M,P} \right) \\
&\quad + V_p \chi_{gp} \Lambda^{(k)} \sum_{g'=1}^{G} \phi_{g',P}^{(k)} \left(\left(\frac{\partial \nu\Sigma_{f,g'}}{\partial T_D}\right)^{n-1} \delta T_{D,P} + \left(\frac{\partial \nu\Sigma_{f,g'}}{\partial T_M}\right)^{n-1} \delta T_{M,P} \right)
\end{aligned}
\tag{I.12}
$$

Regrouped, using some short hand,

$$
\begin{aligned}
&= S^k + V_p \chi_{gp} \sum_{g'=1}^{G} \left(\nu\Sigma_{f,g'}^{(k)} \phi_{g',P}^{(k)} \right) \delta\Lambda \\
&+ \left(\tilde{D}_{g,E}^{(k)} \delta\phi_{g,E} + \tilde{D}_{g,W}^{(k)} \delta\phi_{g,W} + \tilde{D}_{g,N}^{(k)} \delta\phi_{g,N} + \tilde{D}_{g,S}^{(k)} \delta\phi_{g,S} + \tilde{D}_{g,B}^{(k)} \delta\phi_{g,B} + \tilde{D}_{g,T}^{(k)} \delta\phi_{g,T} \right) \\
&+ \left[- \left(\tilde{D}_{g,E}^{(k)} + \tilde{D}_{g,W}^{(k)} + \tilde{D}_{g,N}^{(k)} + \tilde{D}_{g,S}^{(k)} + \tilde{D}_{g,B}^{(k)} + \tilde{D}_{g,T}^{(k)} \right) - V_p \left(\Sigma_{a,g}^{(k)} + \sum_{g' \neq g}^{G} \Sigma_{s,gg'}^{(k)} \right) \right] \delta\phi_{g,P} \\
&\qquad + V_p \left[\sum_{g'=1(\neq g)}^{G} \Sigma_{s,g'g}^{(k)} + \chi_{gp} \Lambda^{(k)} \sum_{g'=1}^{G} \nu\Sigma_{f,g'}^{(k)} \right] \delta\phi_{g',P} \\
&+ \left(\phi_{g,EWNSBT}^{(k)} - \phi_{g,P}^{(k)} \right) \left[\frac{\partial \tilde{D}_{g,EWNSBT}}{\partial T_{M,EWNSBT}} \left\{ \delta T_{M,EWNSBT} \right\} + \frac{\partial \tilde{D}_{g,EWNSBT}}{\partial T_{D,EWNSBT}} \left\{ \delta T_{D,EWNSBT} \right\} \right] \\
&+ \left(\phi_{g,EWNSBT}^{(k)} - \phi_{g,P}^{(k)} \right) \left[\frac{\partial \tilde{D}_{g,EWNSBT}}{\partial T_{M,P}} \left\{ \delta T_{M,P} \right\} + \frac{\partial \tilde{D}_{g,EWNSBT}}{\partial T_{D,P}} \left\{ \delta T_{D,P} \right\} \right] \\
&- V_p \phi_{g,P}^{(k)} \left(\left(\frac{\partial \Sigma_a}{\partial T_D} \right)^{n-1} \delta T_{D,P} + \left(\frac{\partial \Sigma_a}{\partial T_M} \right)^{n-1} \delta T_{M,P} \right) \\
&- V_p \phi_{g,P}^{(k)} \sum_{g'=1(\neq g)}^{G} \left(\left(\frac{\partial \Sigma_{s,gg'}}{\partial T_D} \right)^{n-1} \delta T_{D,P} + \left(\frac{\partial \Sigma_{s,gg'}}{\partial T_M} \right)^{n-1} \delta T_{M,P} \right) \\
&+ V_p \sum_{g'=1(\neq g)}^{G} \phi_{g',P}^{(k)} \left(\left(\frac{\partial \Sigma_{s,g'g}}{\partial T_D} \right)^{n-1} \delta T_{D,P} + \left(\frac{\partial \Sigma_{s,g'g}}{\partial T_M} \right)^{n-1} \delta T_{M,P} \right) \\
&+ V_p \chi_{gp} \Lambda^{(k)} \sum_{g'=1}^{G} \phi_{g',P}^{(k)} \left(\left(\frac{\partial \nu\Sigma_{f,g'}}{\partial T_D} \right)^{n-1} \delta T_{D,P} + \left(\frac{\partial \nu\Sigma_{f,g'}}{\partial T_M} \right)^{n-1} \delta T_{M,P} \right)
\end{aligned}
\tag{I.13}
$$

Collecting terms,

$$
\begin{aligned}
&= S^k + V_p \chi_{gp} \sum_{g'=1}^{G} \left(\nu \Sigma_{f,g'}^{(k)} \phi_{g',P}^{(k)} \right) \delta\Lambda \\
&+ \left(\tilde{D}_{g,E}^{(k)} \delta\phi_{g,E} + \tilde{D}_{g,W}^{(k)} \delta\phi_{g,W} + \tilde{D}_{g,N}^{(k)} \delta\phi_{g,N} + \tilde{D}_{g,S}^{(k)} \delta\phi_{g,S} + \tilde{D}_{g,B}^{(k)} \delta\phi_{g,B} + \tilde{D}_{g,T}^{(k)} \delta\phi_{g,T} \right) \\
&+ \left[- \left(\tilde{D}_{g,E}^{(k)} + \tilde{D}_{g,W}^{(k)} + \tilde{D}_{g,N}^{(k)} + \tilde{D}_{g,S}^{(k)} + \tilde{D}_{g,B}^{(k)} + \tilde{D}_{g,T}^{(k)} \right) - V_p \left(\Sigma_{a,g}^{(k)} + \sum_{g' \neq g}^{G} \Sigma_{s,gg'}^{(k)} \right) \right] \delta\phi_{g,P} \\
&+ V_p \left[\sum_{g'=1(\neq g)}^{G} \Sigma_{s,g'g}^{(k)} \delta\phi_{g',P} + \chi_{gp} \Lambda^{(k)} \sum_{g'=1}^{G} \nu \Sigma_{f,g'}^{(k)} \delta\phi_{g',P} \right] \\
&+ \left(\phi_{g,EWNSBT}^{(k)} - \phi_{g,P}^{(k)} \right) \left[\frac{\partial \tilde{D}_{g,EWNSBT}}{\partial T_{M,EWNSBT}} \left\{ \delta T_{M,EWNSBT} \right\} + \frac{\partial \tilde{D}_{g,EWNSBT}}{\partial T_{D,EWNSBT}} \left\{ \delta T_{D,EWNSBT} \right\} \right] \\
&+ \left(\phi_{g,EWNSBT}^{(k)} - \phi_{g,P}^{(k)} \right) \left[\frac{\partial \tilde{D}_{g,EWNSBT}}{\partial T_{M,P}} \left\{ \delta T_{M,P} \right\} + \frac{\partial \tilde{D}_{g,EWNSBT}}{\partial T_{D,P}} \left\{ \delta T_{D,P} \right\} \right] \\
&- V_p \phi_{g,P}^{(k)} \left(\left(\frac{\partial \Sigma_a}{\partial T_D} \right)^{n-1} + \left(\sum_{g'=1(\neq g)}^{G} \left(\frac{\partial \Sigma_{s,gg'}}{\partial T_D} \right)^{n-1} \right) \right) \delta T_{D,P} \\
&+ V_p \left(\sum_{g'=1(\neq g)}^{G} \phi_{g',P}^{(k)} \left(\frac{\partial \Sigma_{s,g'g}}{\partial T_D} \right)^{n-1} + \chi_{gp} \Lambda^{(k)} \left(\sum_{g'=1}^{G} \phi_{g',P}^{(k)} \left(\frac{\partial \nu \Sigma_{f,g'}}{\partial T_D} \right)^{n-1} \right) \right) \delta T_{D,P} \\
&- V_p \phi_{g,P}^{(k)} \left(\left(\frac{\partial \Sigma_a}{\partial T_M} \right)^{n-1} + \left(\sum_{g'=1(\neq g)}^{G} \left(\frac{\partial \Sigma_{s,gg'}}{\partial T_M} \right)^{n-1} \right) \right) \delta T_{M,P} \\
&+ V_p \left(\sum_{g'=1(\neq g)}^{G} \phi_{g',P}^{(k)} \left(\frac{\partial \Sigma_{s,g'g}}{\partial T_M} \right)^{n-1} + \chi_{gp} \Lambda^{(k)} \left(\sum_{g'=1}^{G} \phi_{g',P}^{(k)} \left(\frac{\partial \nu \Sigma_{f,g'}}{\partial T_M} \right)^{n-1} \right) \right) \delta T_{M,P}
\end{aligned}
\tag{I.14}
$$

If TH feedback is ignored in the initial development,

$$
\begin{aligned}
&+\left[\left(\tilde{D}_{g,E}^{(k)}+\tilde{D}_{g,W}^{(k)}+\tilde{D}_{g,N}^{(k)}+\tilde{D}_{g,S}^{(k)}+\tilde{D}_{g,B}^{(k)}+\tilde{D}_{g,T}^{(k)}\right)+V_p\left(\Sigma_{a,g}^{(k)}+\sum_{g'\neq g}^{G}\Sigma_{s,gg'}^{(k)}\right)\right]\delta\phi_{g,P}\\
&-V_p\left(\sum_{g'=1(\neq g)}^{G}\Sigma_{s,g'g}^{(k)}\delta\phi_{g',P}+\chi_{gp}\Lambda^{(k)}\sum_{g'=1}^{G}\nu\Sigma_{f,g'}^{(k)}\delta\phi_{g',P}\right)\\
&-V_p\chi_{gp}\sum_{g'=1}^{G}\left(\nu\Sigma_{f,g'}^{(k)}\phi_{g',P}^{(k)}\right)\delta\Lambda\\
&-\left(\tilde{D}_{g,E}^{(k)}\delta\phi_{g,E}+\tilde{D}_{g,W}^{(k)}\delta\phi_{g,W}+\tilde{D}_{g,N}^{(k)}\delta\phi_{g,N}+\tilde{D}_{g,S}^{(k)}\delta\phi_{g,S}+\tilde{D}_{g,B}^{(k)}\delta\phi_{g,B}+\tilde{D}_{g,T}^{(k)}\delta\phi_{g,T}\right)\\
&=S^k
\end{aligned}
\tag{I.15}
$$

Appendix J
Derivation: Eigenvalue

Beginning with a first order estimation for the eigenvalue,

$$\lambda^1 \sum_1^g \left(\nu\Sigma_{f,g}^1 \phi_g^1\right) = \lambda^0 \sum_1^g \left(\nu\Sigma_{f,g}^0 \phi_g^0\right) \tag{J.1}$$

Expanding the cross section, flux, and eigenvalue,

$$\lambda^1 \Rightarrow \lambda^{(k)} + \delta\lambda \tag{J.2a}$$
$$\phi_g^1 \Rightarrow \phi_g^{(k)} + \delta\phi_g \tag{J.2b}$$
$$\nu\Sigma_{f,g}^1 \Rightarrow \nu\Sigma_{f,g} + \delta\nu\Sigma_{f,g} \tag{J.2c}$$
$$\tag{J.2d}$$

Inserting these into the equation, assuming three groups,

$$\begin{aligned}\left(\lambda^{(k)} + \delta\lambda\right)\Bigg(\bigg(\Big(\phi_1^{(k)} + \delta\phi_1\Big)\left(\nu\Sigma_{f,1} + \delta\nu\Sigma_{f,1}\right)\bigg)& \\ + \bigg(\Big(\phi_2^{(k)} + \delta\phi_2\Big)\left(\nu\Sigma_{f,2} + \delta\nu\Sigma_{f,2}\right)\bigg) + \bigg(\Big(\phi_3^{(k)} + \delta\phi_3\Big)\left(\nu\Sigma_{f,3} + \delta\nu\Sigma_{f,3}\right)\bigg)\Bigg)& \\ = \lambda^0 \sum_1^g \left(\nu\Sigma_{f,g}^0 \phi_g^0\right)&\end{aligned} \tag{J.3}$$

Doing some algebra,

$$\begin{aligned}\left(\lambda^{(k)}+\delta\lambda\right)\left(\nu\Sigma_{f,1}\delta\phi_1+\nu\Sigma_{f,2}\delta\phi_2+\nu\Sigma_{f,3}\delta\phi_3\right.\\ +\delta\nu\Sigma_{f,1}\phi_1^{(k)}+\delta\nu\Sigma_{f,2}\phi_2^{(k)}+\delta\nu\Sigma_{f,3}\phi_3^{(k)}\\ \left.+\nu\Sigma_{f,1}\phi_1^{(k)}+\nu\Sigma_{f,2}\phi_2^{(k)}+\nu\Sigma_{f,3}\phi_3^{(k)}\right)\\ =\lambda^0\sum_1^g\left(\nu\Sigma_{f,g}^0\phi_g^0\right)\end{aligned} \tag{J.4}$$

More algebra,

$$\begin{aligned}\delta\lambda\left(\nu\Sigma_{f,1}\phi_1^{(k)}+\nu\Sigma_{f,2}\phi_2^{(k)}+\nu\Sigma_{f,3}\phi_3^{(k)}\right)\\ +\lambda^{(k)}\left(\nu\Sigma_{f,1}\delta\phi_1+\nu\Sigma_{f,2}\delta\phi_2+\nu\Sigma_{f,3}\delta\phi_3\right)\\ +\lambda^{(k)}\left(\delta\nu\Sigma_{f,1}\phi_1^{(k)}+\delta\nu\Sigma_{f,2}\phi_2^{(k)}+\delta\nu\Sigma_{f,3}\phi_3^{(k)}\right)\\ =\lambda^0\sum_1^g\left(\nu\Sigma_{f,g}^0\phi_g^0\right)-\lambda^{(k)}\left(\nu\Sigma_{f,1}\phi_1^{(k)}\nu\Sigma_{f,1}\phi_1^{(k)}+\nu\Sigma_{f,2}\phi_2^{(k)}\nu\Sigma_{f,2}\phi_2^{(k)}+\nu\Sigma_{f,3}\phi_3^{(k)}\nu\Sigma_{f,3}\phi_3^{(k)}\right)\end{aligned} \tag{J.5}$$

More algebra, expanding the cross sections with respect to Moderator and Doppler cross sections temperature

$$\begin{aligned}\left(\nu\Sigma_{f,1}\phi_1^{(k)}+\nu\Sigma_{f,2}\phi_2^{(k)}+\nu\Sigma_{f,3}\phi_3^{(k)}\right)\delta\lambda\\ +\nu\Sigma_{f,1}\delta\phi_1+\nu\Sigma_{f,2}\delta\phi_2+\nu\Sigma_{f,3}\delta\phi_3\\ +\lambda^{(k)}\left(\phi_1^{(k)}\frac{\partial\nu\Sigma_{f,1}}{\partial T_m}+\phi_2^{(k)}\frac{\partial\nu\Sigma_{f,2}}{\partial T_m}+\phi_3^{(k)}\frac{\partial\nu\Sigma_{f,3}}{\partial T_m}\right)\delta T_m\\ +\lambda^{(k)}\left(\phi_1^{(k)}\frac{\partial\nu\Sigma_{f,1}}{\partial T_d}+\phi_2^{(k)}\frac{\partial\nu\Sigma_{f,2}}{\partial T_d}+\phi_3^{(k)}\frac{\partial\nu\Sigma_{f,3}}{\partial T_d}\right)\delta T_d\\ =\lambda^0\sum_1^g\left(\nu\Sigma_{f,g}^0\phi_g^0\right)-\lambda^{(k)}\left(\nu\Sigma_{f,1}\phi_1^{(k)}+\nu\Sigma_{f,2}\phi_2^{(k)}+\nu\Sigma_{f,3}\phi_3^{(k)}\right)\end{aligned} \tag{J.6}$$

Appendix K
Derivation: 1D Conduction Solve

The 1-D conduction solve within the pebble is given as,

$$\frac{\partial}{\partial x}\left[kA\frac{\partial T}{\partial x}\right] + Q = \frac{\partial}{\partial t}\left[c_p T\right] \tag{K.1}$$

If we ignore the time dependent terms, and discretize in the radial direction,

$$k_{in}A_{in}\left(T_{sur} - T_{in}\right) + k_{out}A_{out}\left(T_{sur} - T_{out}\right) = q_{in}^{'''}V_{in} + q_{out}^{'''}V_{out} \tag{K.2}$$

The power can be represented in terms if the relative radial power and total pebble power,

$$k_{in}A_{in}\left(T_{sur} - T_{in}\right) + k_{out}A_{out}\left(T_{sur} - T_{out}\right) = \left(f_{in}V_{in} + f_{out}V_{out}\right)q_{peb}^{'''} \tag{K.3}$$

Expanding the primary variables,

$$\begin{aligned} k_{in}A_{in}&\left(T_{sur} + \delta T_{sur} - T_{in} + \delta T_{in}\right) \\ &+ k_{out}A_{out}\left(T_{sur} + \delta T_{sur} - T_{out} + \delta T_{out}\right) \\ &= \left(f_{in}V_{in} + f_{out}V_{out}\right)q_{peb}^{'''} \end{aligned} \tag{K.4}$$

Adding the coefficients,

$$\begin{aligned} \left(k_{in} + \delta k_{in}\right)A_{in}&\left(T_{sur} + \delta T_{sur} - T_{in} - \delta T_{in}\right) \\ &+ \left(k_{out} + \delta k_{out}\right)A_{out}\left(T_{sur} + \delta T_{sur} - T_{out} - \delta T_{out}\right) \\ &= \left(f_{in}V_{in} + f_{out}V_{out}\right)\left(q_{peb}^{'''} + \delta q_{peb}\right) \end{aligned} \tag{K.5}$$

Multiplying out, keeping 1st order terms,

$$\begin{aligned}
&A_{in}\left(T_{sur}-T_{in}\right)\delta k_{in}+k_{in}A_{in}\left(\delta T_{sur}-\delta T_{in}\right)\\
&\quad+A_{out}\left(T_{sur}-T_{out}\right)\delta k_{out}+k_{out}A_{out}\left(\delta T_{sur}-\delta T_{out}\right)-\left(f_{in}V_{in}+f_{out}V_{out}\right)\delta q_{peb}\\
&\qquad=\left(f_{in}V_{in}+f_{out}V_{out}\right)q_{peb}^{'''}-k_{in}A_{in}\left(T_{sur}-T_{in}\right)+k_{out}A_{out}\left(T_{sur}-T_{out}\right)
\end{aligned}\tag{K.6}$$

Expanding the conductivity, and heat generation, give the final form

$$\begin{aligned}
&A_{in}\left(T_{sur}-T_{in}\right)\left(\frac{\partial k_{in}}{\partial T_{sur}}\delta T_{sur}+\frac{\partial k_{in}}{\partial T_{in}}\delta T_{in}\right)\\
&\qquad+A_{out}\left(T_{sur}-T_{out}\right)\left(\frac{\partial k_{out}}{\partial T_{sur}}\delta T_{sur}+\frac{\partial k_{out}}{\partial T_{out}}\delta T_{out}\right)\\
&\quad-\left(f_{in}V_{in}+f_{out}V_{out}\right)\bar{P}_{core}\left(\left(\sum_{g=1}^{ngroup}\phi_g\frac{\partial\kappa\Sigma_f}{\partial T_m}\right)\delta T_m+\left(\sum_{g=1}^{ngroup}\phi_g\frac{\partial\kappa\Sigma_f}{\partial T_d}\right)\delta T_d\right)\\
&\qquad+k_{in}A_{in}\left(\delta T_{sur}-\delta T_{in}\right)+k_{out}A_{out}\left(\delta T_{sur}-\delta T_{out}\right)\\
&\quad=\left(f_{in}V_{in}+f_{out}V_{out}\right)q_{peb}^{'''}-k_{in}A_{in}\left(T_{sur}-T_{in}\right)+k_{out}A_{out}\left(T_{sur}-T_{out}\right)
\end{aligned}\tag{K.7}$$

In the first shell, closet to the center of the pebble, the "in" terms are equal to zero. The boundary with the fluid is treated differently,

$$\begin{aligned}
&k_{in}A_{in}\left(T_{sur}-T_{in}\right)+\alpha A_{peb}\left(T_{sur}-T_{flu}\right)\\
&\qquad+k_{eq,W}\left(T_{sur}-T_{sol,W}\right)+k_{eq,E}\left(T_{sur}-T_{sol,E}\right)\\
&\qquad+k_{eq,N}\left(T_{sur}-T_{sol,N}\right)+k_{eq,S}\left(T_{sur}-T_{sol,S}\right)\\
&\qquad+k_{eq,B}\left(T_{sur}-T_{sol,B}\right)+k_{eq,T}\left(T_{sur}-T_{sol,T}\right)\\
&\qquad\qquad=q_{in}^{'''}V_{in}
\end{aligned}\tag{K.8}$$

Similar to the solid energy equation, the neighbor solid temperature, fluid temperature, equivalent conductivity, and alpha are all expanded.

$$
\begin{aligned}
&A_{in}\left(T_{sur}-T_{in}\right)\left(\frac{\partial k_{in}}{\partial T_{sur}}\delta T_{sur}+\frac{\partial k_{in}}{\partial T_{in}}\delta T_{in}\right)\\
&+\alpha A_{peb}\left(\delta T_{sur}-\delta T_{flu}\right)+A_{peb}\left(\delta T_{sur}-\delta T_{flu}\right)\left(\frac{\partial \alpha}{\partial T_{sur}}\delta T_{sur}+\frac{\partial \alpha}{\partial T_{sol}}\delta T_{sol}+\frac{\partial \alpha}{\partial T_{flu}}\delta T_{flu}\right)\\
&+k_{eq,W}\left(\delta T_{sur}-\delta T_{sol,W}\right)+\left(T_{sur}-T_{sol,W}\right)\left(\frac{\partial k_{eq,W}}{\partial T_{sol,W}}\delta T_{sol,W}+\frac{\partial k_{eq,W}}{\partial T_{sol,P}}\delta T_{sol,P}\right)\\
&+k_{eq,E}\left(\delta T_{sur}-\delta T_{sol,E}\right)+\left(T_{sur}-T_{sol,E}\right)\left(\frac{\partial k_{eq,E}}{\partial T_{sol,E}}\delta T_{sol,E}+\frac{\partial k_{eq,E}}{\partial T_{sol,P}}\delta T_{sol,P}\right)\\
&+k_{eq,N}\left(\delta T_{sur}-\delta T_{sol,N}\right)+\left(T_{sur}-T_{sol,N}\right)\left(\frac{\partial k_{eq,N}}{\partial T_{sol,N}}\delta T_{sol,E}+\frac{\partial k_{eq,N}}{\partial T_{sol,P}}\delta T_{sol,P}\right)\\
&+k_{eq,S}\left(\delta T_{sur}-\delta T_{sol,S}\right)+\left(T_{sur}-T_{sol,S}\right)\left(\frac{\partial k_{eq,S}}{\partial T_{sol,S}}\delta T_{sol,E}+\frac{\partial k_{eq,S}}{\partial T_{sol,P}}\delta T_{sol,P}\right)\\
&+k_{eq,B}\left(\delta T_{sur}-\delta T_{sol,B}\right)+\left(T_{sur}-T_{sol,B}\right)\left(\frac{\partial k_{eq,B}}{\partial T_{sol,B}}\delta T_{sol,E}+\frac{\partial k_{eq,B}}{\partial T_{sol,P}}\delta T_{sol,P}\right)\\
&+k_{eq,T}\left(\delta T_{sur}-\delta T_{sol,T}\right)+\left(T_{sur}-T_{sol,T}\right)\left(\frac{\partial k_{eq,T}}{\partial T_{sol,T}}\delta T_{sol,T}+\frac{\partial k_{eq,T}}{\partial T_{sol,P}}\delta T_{sol,P}\right)\\
&=q_{in}^{'''}V_{in}-k_{in}A_{in}\left(T_{sur}-T_{in}\right)-\alpha A_{peb}\left(T_{sur}-T_{flu}\right)\\
&-k_{eq,W}\left(T_{sur}-T_{sol,W}\right)-k_{eq,E}\left(T_{sur}-T_{sol,E}\right)\\
&-k_{eq,N}\left(T_{sur}-T_{sol,N}\right)-k_{eq,S}\left(T_{sur}-T_{sol,S}\right)\\
&-k_{eq,B}\left(T_{sur}-T_{sol,B}\right)-k_{eq,T}\left(T_{sur}-T_{sol,T}\right)\qquad\text{(K.9)}
\end{aligned}
$$

Appendix L

Derivation: Moderator XS Temperature

The moderator temperature is a linear function of the 1-d conduction temperature

$$V_{peb}T_m = \sum_{sh=1}^{nshell} V_s \frac{T_{sh} + T_{sh+1}}{2} \tag{L.1}$$

Expanding the moderator and shell temperatures

$$V_{peb}\left(T_m + \delta T_m\right) = \sum_{sh=1}^{nshell} V_s \frac{\left(T_{sh} + \delta T_{sh}\right) + \left(T_{sh+1} + \delta T_{sh+1}\right)}{2} \tag{L.2}$$

Moving to LH and RHS,

$$V_{peb}\delta T_m - \sum_{sh=1}^{nshell} V_s \frac{\delta T_{sh} + \delta T_{sh+1}}{2} = \left(\sum_{sh=1}^{nshell} V_s \frac{T_{sh} + T_{sh+1}}{2}\right) - \left(V_{peb}T_m\right) \tag{L.3}$$

Appendix M

Derivation: Doppler XS Temperature

The Doppler temperature is a linear function of the 1-d conduction temperature, and also contains a heat generation term

$$V_{fuel}T_d = \sum_{sh=1}^{nshell(fuel)} \left(V_s \frac{T_{sh} + T_{sh+1}}{2} + V_s f_{pueb} Q_{peb} \right) \tag{M.1}$$

Expanding the Doppler and shell temperatures

$$V_{fuel}\left(T_d + \delta T_d\right) = \sum_{sh=1}^{nshell(fuel)} V_s \left(\frac{(T_{sh} + \delta T_{sh}) + (T_{sh+1} + \delta T_{sh+1})}{2} + V_s f_{pueb} Q_{peb} \right) \tag{M.2}$$

Moving to LH and RHS,

$$\begin{aligned} V_{fuel}\delta T_d - \sum_{sh=1}^{nshell(fuel)} V_s \frac{\delta T_{sh} + \delta T_{sh+1}}{2} \\ = \left(\sum_{sh=1}^{nshell(fuel)} \left(V_s \frac{T_{sh} + T_{sh+1}}{2} + V_s f_{pueb} Q_{peb} \right) \right) - \left(V_{fuel} T_d \right) \end{aligned} \tag{M.3}$$

Expanding the heat generation term,

$$V_{fuel}\delta T_d - \sum_{sh=1}^{nshell(fuel)} V_s \frac{\delta T_{sh} + \delta T_{sh+1}}{2} - \sum_{sh=1}^{nshell(fuel)} V_s f_{pueb} \delta Q_{peb}$$
$$= \left(\sum_{sh=1}^{nshell(fuel)} \left(V_s \frac{T_{sh} + T_{sh+1}}{2} + V_s f_{pueb} Q_{peb} \right) \right) - (V_{fuel} T_d) \tag{M.4}$$

The heat generation depends on kappa fission and flux,

$$\begin{aligned} V_{fuel}\delta T_d - \sum_{sh=1}^{nshell(fuel)} V_s \frac{\delta T_{sh} + \delta T_{sh+1}}{2} \\ - \left(\sum_{sh=1}^{nshell(fuel)} V_s f_{pueb} \right) \sum_{1}^{ngrp} \frac{\partial Q_{peb}}{\partial T_m} \delta T_m \\ - \left(\sum_{sh=1}^{nshell(fuel)} V_s f_{pueb} \right) \sum_{1}^{ngrp} \frac{\partial Q_{peb}}{\partial T_D} \delta T_D \\ - \left(\sum_{sh=1}^{nshell(fuel)} V_s f_{pueb} \right) + V_{fuel} \overline{P}_{n,fuel} \sum_{1}^{ngrp} \kappa \Sigma_g \delta \phi_g \\ = \left(\sum_{sh=1}^{nshell(fuel)} \left(V_s \frac{T_{sh} + T_{sh+1}}{2} + V_s f_{pueb} Q_{peb} \right) \right) - (V_{fuel} T_d) \end{aligned} \tag{M.5}$$

Bibliography

[1] J. Gan, Y. Xu, and T. J. Downar, "Efficient numerical solver for non-linear nuclear coupled codes," in *Nuclear Mathematical and Computaional Sciences: A Century in Review, A Century Anew*, 2003.

[2] B. R. Bandini, K. N. Ivanov, A. J. Baratta, and R. G. Steinke, "Verifcation of a three-dimensional nodal transient neutronics routine for the trace-pf1 / mod3 thermal-hydraulic system analysis code," *Nuclear Technology*, vol. 123, pp. 1–20, 1998.

[3] J. J. Jeong and H. C. No, "An improved numerical scheme with the fully-implicit two-fluid model for a fast-running system code," *Nulcear Engineering and Design*, vol. 104, pp. 145–153, 1987.

[4] J. C. Ragusa and V. S. Mahadevan, "Consistent and accurate schemes for coupled neutronics thermal-hydraulics reactor analysis," *Nuclear Engineering and Design*, vol. 239, pp. 566–579, 2009.

[5] H. Zhang, V. A. Mousseau, and H. Zhao, "Development of a high fidelity system analysis code for generation iv reactors," tech. rep., Idaho National Laboratory, 2008.

[6] V. S. Mahadeven and J. C. Ragusa, "Coupling schemes for multiphysics reactor simulation," tech. rep., Idaho National Laboratory, 2007.

[7] R. M. Miller, "The application of high performance computing to coupled thermal-hydraulic / neutronic reactor analysis," Master's thesis, Purdue University, 2000.

[8] V. A. Mousseau, "A generalized interface module for the coupling of spatial kinetics and thermal-hydraulics codes," in *9th International Topic Meeting on Nuclear Reactor Thermal Hydraulics. San Fransisco, CA*, 1998.

[9] V. A. Mousseau, "Application fo the generalized interface module to the coupling of parcs with both relap5 and trac-m," in *ANS Annual Meeting and Embedded Topical Meeting, Boston, MA*, 1999.

[10] M. Jelinek, "Study of higher order numerical schemes applied to full two-phase flow: A thesis," Master's thesis, Pennsylvania State University, 2005.

[11] A. Pautz and A. Birkhofer, "Coupling of time-dependent neutron transport theory with the thermal hydraulics code athlet and application to the research reactor frm-ii," *Nuclear Science and Engineering*, vol. 145, pp. 320–341, 2003.

[12] D. Viswanath, "Recurrent motions within plane couette turbulence," *Journal of Fluid Mechanics*, vol. 580, pp. 339–358, 2007.

[13] D. Viswanath, "The critical layer in piper flow at high reynolds number," *Philosophical Transactions of The Royal Society A*, vol. 367, pp. 561–576, 2009.

[14] K. Ivanov and M. Avramova, "Challenges in coupled thermal-hydraulics and neutronics simulation for lwr safety analysis," *Annals of Nuclear Energy*, vol. 34, pp. 501–513, 2007.

[15] A. V. Gulevich and O. F. Kukharchuk, "Methods for calculating coupled reactor systems," *Atomic Energy*, vol. 97, pp. 803–811, 2004.

[16] D. Gaston, C. Newman, G. Hansen, and D. Lebrun-Grandie, "Moose: A parallel computational framework for coupled systems of nonlinear equations," *Nuclear Engineering and Design*, vol. 239, pp. 1768–1778, 2009.

[17] D. Gaston, G. Hansen, S. Kadioglu, D. A. Knoll, C. Newman, H. Park, C. Permann, and W. Taitano, "Parallel multiphysics algorithmms and software for computational nuclear engineering," *Journal of Physics: Conference Series*, vol. 180, pp. 1–10, 2009.

[18] D. W. Jerng and H. C. No, "New computational method with a fully-implicit scheme for a real-time accident simulator," *Nuclear Engineering and Design*, vol. 99, pp. 101–107, 1987.

[19] D. A. Kastanya, *Implementaion of a Newton-Krylov Iterative Method to Address Strong Non-Linear Feedback Effects in Formosa-B BWR Core Simulator*. PhD thesis, North Carolina State University, 2002.

[20] D. A. Knoll and D. E. Keyes, "Jacobian-free newton-krylov methods: a survery of approaches and applications," *Journal of Computational Physics*, vol. 193, pp. 357–397, 2004.

[21] "Trace v5.0 (patch 01): Users manual," tech. rep., US NRC, Office of Nuclear Regulatory Research, 2008.

[22] "Relap5/mod3.3 code manual volume iv: Models and correlations," tech. rep., Information Systems Laboratories, Inc., 2001.

[23] J. H. McFadden and M. P. Paulsen, "Retran-03: A program for transient thermal-hydraulic analysis of complex fluid flow systems," tech. rep., Idaho: Computer Simulation & Analysis, Inc., 1992.

[24] J. W. Spore and J. H. Mahaffy, "Trac-m / fortran 90 (version 3.0) theory manual," tech. rep., Los Alamos National Laboratory, 2000.

[25] J. H. Mahaffy, "The advantages and limitations of the sets method," in *International Conference on Numerical Methods in Nuclear Engnieering. Montreal, Canada*, 1983.

[26] A. E. Aboanber, "Exact solution for the non-linear two point kinetic model of reflected reactors," *Progress in Nuclear Energy*, vol. 51, pp. 715–726, 2009.

[27] R. G. McClarren, J. P. Holloway, T. A. Brunner, and T. A. Melhorn, "A quasi-linear implicit riemann solver for the time-dependent pn equations," *Nuclear Science and Engineering*, vol. 155, pp. 290–299, 2007.

[28] "Simulate-3k, models and methodology," tech. rep., Studsvik Scandpower, 2009.

[29] T. J. Downar, Y. Xu, and V. Seker, "Parcs v3.0 u.s nrc core neutronics simulator theory manual," tech. rep., University of Michigan, 2010.

[30] B. Quintero-Leyva, "On the numerical solution of the integro-differential equation of the point kinetics of nuclear reactors," *Annals of Nuclear Energy*, vol. 36, pp. 1280–1284, 2009.

[31] F. D'Auria, A. B. Salah, G. Galassi, and J. Vedov, "Neutronics/thermal-hydraulics coupling in lwr technology, vol. 1 & 2," tech. rep., OECD, 2004.

[32] F. Merino, C. Ahnert, and J. M. Aragones, "Development and validation of the 3-d pwr core dynamics simtran code," *Mathematical Methods and Supercomputing in Nuclear Applications,*, vol. 1, p. 646, 1993.

[33] B. Davidson, *Neutron Transport Theory.* Oxford University Press, 1957.

[34] C. Frepoli, J. H. Mahaffy, and K. Ohkawa, "Notes on the implementaion of the fully-implicit numerical scheme for a two-phase three-field flow model," *Nuclear Engineering and Design*, vol. 225, pp. 191–217, 2003.

[35] V. A. Mousseau, "A fully implicit, second order in time, simulation of a nuclear reactor core," in *Proceedings of ICONE14, Miami, FL*, 2006.

[36] J. Watson, *Implicit Time-Integraion Method for Simultaneous Solution of a Coupled Non-Linear System.* PhD thesis, Penn State University, 2010.

[37] E. S. Lee, "A generalized newton-raphson method for nonlinear partial differential equations - packed bed reactors with axial mixing," *Chemical Engineering Science*, vol. 21, pp. 143–157, 1966.

[38] I. Limaiem, F. Damian, X. Raepsaet, and E. Studer, "Vhtr core modeling: Coupling between neutronic and thermal-hydraulics," in *MC+SNA. Avignon, France*, 2005.

[39] H. Park, D. A. Knoll, D. R. Gaston, and R. C. Martineau, "Tighly coupled multiphysics algorithms for pebble bed reactors," *Nuclear Science and Engineering*, vol. 166, pp. 118–133, 2010.

[40] F. D. Bramkamp, H. M. Bucker, and A. Rasch, "Using exact jacobian in an implicit newton-krylov method," *Computers and Fluids*, vol. 35, p. 1063, 2006.

[41] P. N. Brown and Y. Saad, "Hybrid krylov methods for nonlinear systems of equations," *SIAM J. Sci. Stat. Comput.*, vol. 11, pp. 450–481, 1990.

[42] T. F. Chan and K. R. Jackson, "Nonlinearly preconditioned krylov subspace methods for discrete newton algorithms," *SIAM J. Sci. Stat. Comput.*, vol. 5, pp. 533–542, 1984.

[43] J. I. Ramos, "Linearized methods for ordinary differential equations," *Applied Mathematics and Computation*, vol. 104, pp. 109–129, 1999.

[44] M. Rosa, J. S. Warsa, and J. H. Chang, "Fourier analysis of inexact block-jacobi splitting with transport synthetic acceleration," *Nuclear Science and Engineering*, vol. 164, pp. 248–263, 2010.

[45] J. F. Traub and H. Wozniakowski, "Convergence and complexity of newton iteration for operator equations," *Journal of the Association for Computing Machinery*, vol. 26, pp. 250–258, 1979.

[46] K. T. Chu, "A direct matrix method for computing analytical jacobians of discretized nonlinear integro-differential equations," *Journal of Computational Physics*, vol. 228, pp. 5526–5538, 2009.

[47] M. A. Fernandez and M. Moubachir, "A newton method using exact jacobians for solving fluid-structure coupled," *Computers and Fluids*, vol. 83, pp. 127–142, 2005.

[48] H. S. Abdel-Khalik and P. J. Turinsky, "Subspace methods for multi-scale / multi-physics calculations, part i: Theory," in *Transactions of the American Nuclear Society, Boston, MA*, pp. 548–550, 2007.

[49] H. S. Abdel-Khalik, P. J. Turinsky, and M. A. Jessee, "Efficient subspace methods-based algorithms for performing sensitivity, uncertainty, and adaptive simulation of large-scale computational models," *Nuclear Science and Engineering*, vol. 159, pp. 256–272, 2008.

[50] Y. Bang and H. S. Abdel-Khalik, "Verification tests for uncertainty quantification and sensitivity analysis studies," in *Transactions of the American Nuclear Society, Washington D.C.*, 2011.

[51] M. A. Jessee, H. S. Abdel-Khalik, and P. J. Turinsky, "Subspace methods for multi-scale / multi-physics calculations, part ii: Numerical experiments,," in *Transactions of American Nuclear Society, Boston, MA*, 2008.

[52] D. A. Knoll and W. J. Rider, "A multigrid preconditioned newton-krylov method," *SIAM Journal on Scientific Computation*, vol. 21, pp. 691–710, 1999.

[53] J. Gan, Y. Xu, and T. J. Downar, "A matrix-free newton method for coupled neutronics thermal-hydraulics reactor analyses," in *Nuclear Mathematical and Computational Sciences: A Century in Review, A Century Anew*, 2003.

[54] T. J. Downar and H. G. Joo, "A preconditioned krylov method for solution of the multi-dimensional, two-fluid, hydrodynamics equations," *Annals of Nuclear Energy*, vol. 28, pp. 1251–1267, 2000.

[55] Y. Xu, *A Matrix-Free Newton/Kyrlov Method for Coupling Complex Multiphysics Subsystem.* PhD thesis, Purdue University, 2004.

[56] S. Bellavia and B. Morini, "A globally convergent newton-gmres subspace method for systems of non-linear equations," *SIAM Journal on Scientific Computation*, vol. 3, pp. 940–960, 2001.

[57] G. J. V. Tuyle and J. C. Lee, "Linearized transient analysis of nuclear steam generators," *Nuclear Science and Engineering*, vol. 75, pp. 225–242, 1980.

[58] U. Graf, "Implicit coupling of fluid-dynamic systems: Application to multidimensional countercurrent two-phase flow of water and steam," *Nuclear Science and Engineering*, vol. 129, pp. 305–310, 1998.

[59] M. Liou, "A newton/upwind method and numerical study of shock wave / boundary layer interactions," *International Journal for Numerical Methods in Fluids*, vol. 9, pp. 747–761, 1989.

[60] V. A. Mousseau, "Implicitly balanced solution of the two-phase flow equatiosn coupled to nonlinear heat conduction," *Journal of Computational Physics*, vol. 200, pp. 104–132, 2004.

[61] J. Peter, "Non-linear implicit scheme using newton's method for the numerical solution of the navier-stokes equations," *Aerospace Science and Technology*, vol. 3, pp. 157–166, 1998.

[62] M. D. Tidiri, "Preconditing techniques for the newton-krylov solution of compressible flows," *Journal of Computational Physics*, vol. 132, pp. 51–61, 1997.

[63] I. Toumi and D. Caruge, "An implicit second-order numerical method for three-dimensional two-phase flow calculations," *Nuclear Science and Engineering*, vol. 130, pp. 213–225, 1998.

[64] C. O. Ober and J. N. Shadid, "Studies on the accuracy of time integration methods for the radiation-diffusion equations," *Journal of Computational Physics*, vol. 195, pp. 743–772, 2004.

[65] V. A. Mousseau, D. A. Knoll, and W. J. Rider, "Physics-based preconditioning and the newton-krylov method for non-equilibrium radiation diffusion," *Journal of Computational Physics*, vol. 160, pp. 734–765, 2000.

[66] M. P. Leonchuk, Z. V. Sivak, and Y. E. Shvetsov, "Implicit method of solving mass-transfer equations in the variables velocity-vorticity," *Atomic Energy*, vol. 58, pp. 166–170, 1985.

[67] J. M. Barry and J. P. Pollard, "Method of implicit non-stationary iteration for solving neutron diffusion linear equations," *Annals of Nuclear Energy*, vol. 4, pp. 485–493, 1977.

[68] E. D. Fichtl, J. S. Warsa, and J. D. Densmore, "The newton-krylov method applied to negative-flux fixup in sn transport calculations," *Nuclear Science and Engineering*, vol. 165, pp. 331–341, 2010.

[69] D. F. Gill, *Newton-Krylov Methods for the Solution of the k-eigenvalue problem in multigroup neutronics calculations: A Dissertation.* PhD thesis, Pennsylvania State University, 2009.

[70] A. Pautz and A. Birkhofer, "Dort-td: A transient neutron transport code with fully implicit time integration," *Nuclear Science and Engineering*, vol. 145, pp. 299–319, 2003.

[71] P. Ravette, A. M. Larosa, G. G. Coppa, G. Lapenta, and M. Carta, "Analysis and optimization of implicit methods for time-dependent transport calculations of source-driven systems," *Journal of Nuclear Science and Technology*, vol. 37, pp. 215–220, 2000.

[72] M. Geradin, S. Idelsohn, and M. Hogge, "Nonlinear structural dynamics via newton and quasi-newton methods," *Nuclear Engineering and Design*, vol. 58, pp. 339–348, 1980.

[73] W. S. Yang and H. G. Joo, "Lmr core temperature calculation based on implicit formulation of the energy model and a krylov subspace method," *Annals of Nuclear Energy*, vol. 26, pp. 629–640, 1999.

[74] V. A. Mousseau, D. A. Knoll, and J. M. Reisner, "An implicit nonlinearly consistent method for the two-dimensional shallow-water equations with coriolis force," *Monthly Weather Review*, vol. 130, pp. 2611–2625, 2002.

[75] D. A. Knoll, L. Chacon, L. G. Margolin, and V. A. Mousseau, "On balanced approximations for the time integration of multiple time scale systems," *Journal of Computational Physics*, vol. 185, pp. 583–611, 2003.

[76] R. Blaheta, "Convergence of newton-type methods in incremental return mappins analysis of elasto-plastic problems," *Computational Methods in Applied Mechanics and Engineering*, vol. 147, pp. 167–185, 1997.

[77] P. N. Brown and A. C. Hindmarsh, "Matrix-free methods for stiff systems of ode's," *SIAM Journal on Numerical Analysis*, vol. 23, pp. 610–638, 1986.

[78] P. N. Brown, "A local convergence theory for combined inexact-newton / finite-difference projection methods," *SIAM Journal on Numerical Analysis*, vol. 24, pp. 407–434, 1987.

[79] T. F. Chan, "An approximate newton method for coupled nonlinear systems," *SIAM Journal on Numerical Analysis*, vol. 22, pp. 904–913, 1985.

[80] R. S. Dembo, S. C. Eisenstat, and T. Steilhaug, "Inexact newton methods," *SIAM Journal on Numerical Analysis*, vol. 19, pp. 400–408, 1982.

[81] O. Zerkak, A. Manera, I. Gajev, and T. Kozlowski, "Review of multi-physics coupling techniques and suggestions of improvements in the context of nurisp," tech. rep., Paul Scherre Institute, 2010.

[82] Y. Saad and M. H. Schultz, "Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM J. Sci. Stat. Comput.*, vol. 7, pp. 856–869, 1986.

[83] R. Wienands, C. W. Oosterlee, and T. Washio, "Fourier analysis of gmres(m) preconditioned by multigrid," *SIAM Journal on Scientific Computation*, vol. 22, pp. 582–603, 2000.

[84] I. Moret, "A note on the superlinear convergence of gmres," *SIAM Journal on Numerical Analysis*, vol. 34, pp. 513–516, 1997.

[85] V. Seker, *Multiphysics Methods Development for High Temperature Gas Reactor*. PhD thesis, Purdue University, 2007.

[86] F. Reitsma, K. Ivanov, T. Downar, H. de Haas, S. Sen, G. Strydom, R. Mphahlele, B. Tyobeka, V. Seker, and H. D. Gougar, "Pbmr coupled neutronics/thermal hydraulics transient benchmark - the pbmr-400 core design, benchmark definition," tech. rep., NEA, 2005.

[87] M. Sosonkina, D. Allison, and L. Watson, "Scalability analysis of parallel gmres implementations," *Parallel Algorithms Applcations*, vol. 17, pp. 263–284, 2002.

[88] T. F. Chan, "Approximate newton method for coupled nonlinear systems," *SIAM Journal on Numerical Analysis*, vol. 22, pp. 904–913, 1985.