12-2017

Low-Order Multiphysics Coupling Techniques for Nuclear Reactor Applications

Erik Daniel Walker
University of Tennessee, ewalk@vols.utk.edu

To the Graduate Council:

I am submitting herewith a dissertation written by Erik Daniel Walker entitled "Low-Order Multiphysics Coupling Techniques for Nuclear Reactor Applications." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Nuclear Engineering.

Jess C. Gehin, Major Professor

We have read this dissertation and recommend its acceptance:

Charles Collins, G. Ivan Maldonado, Ronald Pevey, Benjamin Collins

Accepted for the Council:
Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

Low-Order Multiphysics Coupling Techniques for Nuclear Reactor Applications

A Dissertation Presented for the
Doctor of Philosophy
Degree
The University of Tennessee, Knoxville

Erik Daniel Walker
December 2017

This dissertation is dedicated to my wife and my family for their unending support.

Acknowledgements

First and foremost I would like to thank my advisor, Dr. Jess Gehin, for his guidance, mentorship, and for giving me the opportunity to pursue higher education while working at ORNL. I would also like to thank my mentor at ORNL, Dr. Benjamin Collins, for without his guidance and direction, this work would not have been possible. I would like to acknowledge the rest of my doctoral committee: Dr. Charles Collins, Dr. Ivan Maldonado, Dr. Ronald Pevey, for their support in the completion of this dissertation.

I would like to thank Dr. Shane Stimpson for always finding time in his busy schedule to help answer my unending questions. I would also like to thank Andrew Godfrey for is guidance during my first few years in graduate school and at ORNL. I also thank Dr. Lee Riedinger and the University of Tennessee Bredesen Center for Interdisciplinary Research and Graduate Education for providing funding for this work.

Finally, I would like to express my deep gratitude towards my parents, Richard and Sharon, who from an early age encouraged and reinforced my desire to learn. I wish to thank my sisters, Jacqui and Lauren, for always being able to make me laugh and for always reminding me that the answer is always C. And to my wife Kristen, none of this would have been possible were it not for your constant love and encouragement. Whether it was helping me get through the day with a sticky note, or editing this very document for errors, your support was what kept me going. Lastly, this would not have been possible were it not for the grace of our Lord and Savior, Jesus Christ, to whom all glory be given.

This research was supported by the Consortium for Advanced Simulation of Light Water Reactors (www.casl.gov), an Energy Innovation Hub (http://www.energy.gov/hubs) for Modeling and Simulation of Nuclear Reactors under U.S. Department of Energy Contract No. DE-AC05-00OR22725. This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

Abstract

The accurate modeling and simulation of nuclear reactor designs depends greatly on the ability to couple differing sets of physics together. Current coupling techniques most often use a fixed-point, or Picard, iteration scheme in which each set of physics is solved separately, and the resulting solutions are passed between each solver. In the work presented here, two different coupling techniques are investigated: a Jacobian-Free Newton-Krylov (JFNK) approach and a new methodology called Coarse Mesh Finite Difference Coupling (CMFD-Coupling). What both of these techniques have in common is that they are applied to the low-order CMFD system of equations. This allows for the multiphysics feedback effects to be captured on the low-order system without having to perform a neutron transport solve.

The JFNK and CMFD-Coupling approaches were implemented in the MPACT (Michigan Parallel Analysis based on Characteristic Tracing) neutron transport code, which is being developed for the Consortium for Advanced Simulation of Light Water Reactors (CASL). These methods were tested on a wide range of practical reactor physics problems, from a 2D pin cell to a massively parallel 3D full core problem. Initially, JFNK was implemented only as an eigenvalue solver without any feedback enabled. However this led to greatly increased runtimes without any obvious benefit. When multiphysics problems were investigated with both JFNK and CMFD-Coupling, it was concluded that CMFD-Coupling outperformed JFNK in terms of both accuracy and runtime for every problem. When applied to large full core problems with multiple sources of strong feedback enabled, CMFD-Coupling reduced the overall number of transport sweeps required for convergence.

Table of Contents

List of Tables

List of Figures

Abbreviations

1D	One-Dimensional
2D	Two-Dimensional
3D	Three-Dimensional
BOL	Beginning of Life
CASL	Consortium for Advanced Simulation of Light Water Reactors
CMFD	Coarse Mesh Finite Difference
DOE	U.S. Department of Energy
GMRES	Generalized Minimal Residuals
HFP	Hot Full Power
HO	High-Order
HZP	Hot Zero Power
JFNK	Jacobian-Free Newton-Krylov
LO	Low-Order
MOC	Method of Characteristics
MPACT	Michigan Parallel Analysis based on Characteristic Tracing
MPI	Message Passing Interface
NDA	Nonlinear Diffusion Acceleration
PETSc	Portable Extensible Toolkit for Scientific Computation
PWR	Pressurized Water Reactor
RMS	Root Mean Square
TH	Thermal-Hydraulics
VERA	Virtual Environment for Reactor Applications

1. Introduction

Computer modeling and simulation have become extremely valuable tools in nuclear reactor design analysis. The ability to accurately predict reactor performance is crucial for improving the safety and economic viability of a design. However, nuclear reactors are extremely complex systems that involve a variety of different physics, making them rather difficult to model. To address this, the U.S. Department of Energy (DOE) created the Consortium for Advanced Simulation of Light Water Reactors (CASL), an Energy Innovation Hub tasked with developing advanced modeling and simulation capabilities for the nuclear industry. The suite of capabilities being developed is known as the Virtual Environment for Reactor Applications (VERA) [2] and includes chemistry, neutronics, thermal-hydraulics (TH), and thermo-mechanics components. Coupling these different sets of physics together poses a unique challenge because the solution of one component often relies on the solution of another, and vice versa.

The ability to accurately predict coupled system behavior in a reasonable amount of time is critical for both steady state and transient calculations. The solution of a coupled multiphysics problem is most often solved using a fixed-point, or Picard, iteration in which each set of physics is solved separately, and the resulting outputs are passed between each solver. Generally, once two different codes are coupled into one, one set of physics is solved first while the other set of physics is solved only after convergence. These separate solvers treat one another as black boxes in that they only use information from the other as an input and do not share information between each other before convergence. In nuclear reactor applications, coupling neutronics to TH is no exception. A typical workflow is as follows: first, the neutronics equations are solved to calculate the fluxes in the problem, which in turn are used to calculate power. Then a TH solver takes these powers and uses them to determine the temperatures throughout the problem. These temperatures are then passed back to the neutronics solver where they are used to calculate new cross sections. This cycle continues until both the neutronics and TH solutions are stable. This Picard strategy is used for other forms of feedback as well. However, while this fixed-point method is easy to implement and solve, it can suffer from a slow convergence rate. There also is no guarantee that the solution will converge at all. This is due to the fact that only the local convergence within each solver is known and tested. The overall global convergence is not

actually known but is assumed from the local convergence of each solver. Therefore, it is possible under certain circumstances that global convergence is never reached, although each set of physics is locally converged.

Therefore, it is desired to develop methods that couple these different sets of physics more tightly and there are alternatives to the Picard iteration that may offer improvements. Newton-based iterative methods that utilize a Jacobian to provide gradient information have a quadratic convergence rate and are globally convergent [3]. Certain Newton-based methods avoid having to form the Jacobian, which is desirable when it is either expensive or impossible to compute. These methods are referred to as Jacobian-Free Newton-Krylov (JFNK) methods [4].

While JFNK has been extensively applied to accelerating the k-eigenvalue problem [5] [6] [7] [8] [9] [10], less attention has been paid towards its coupling abilities. While both Xu [11] and Ward [12] use JFNK to couple the neutronics equations to different TH codes, the neutronics calculations are performed using a simplified nodal code, rather than a true transport code. Similarly, Kastanya [13] solves the coupled neutronics-TH equations, but uses a two-group neutron diffusion approximation and achieves only a slight improvement in performance. Herman [14] mentions in his work the coupling advantages of using JFNK, but remarks that it was not used for such a purpose, despite being implemented to solve the k-eigenvalue problem. In research supported by CASL [15], the radiation transport equation was approximated using the SP_N angular approximation which was then coupled to the TH equations. However, despite investigating different implementations of JFNK, the improvements relative to the Picard iteration were only modest. Although the computational gains were small, this work yielded results that showed promise for the efficacy of JFNK in coupled reactor problems. Unlike these previous implementations, a unique contribution of this work is the fact that the nonlinear JFNK solver will be applied on the low-order condensed Coarse Mesh Finite Difference (CMFD) equations in an attempt to further accelerate the solution. This method can also be implemented to solve the critical boron search problem and transient problems, in addition to being a k-eigenvalue solver.

Another alternative to the Picard iteration involves tighter coupling between CMFD and other sets of physics. This method was first investigated by Herman [14] and was implemented in a Monte Carlo code along with a machine learning algorithm to couple CMFD to a TH solver. This method was modified for application in a deterministic transport code and can be used to couple any source of feedback with CMFD. In this document, this method is referred to as the CMFD-Coupling technique. CMFD-Coupling performs iterations between the low-order CMFD solver and the feedback operator before passing the updated solution back to the transport solver.

The objective of this work is to develop a multiphysics coupling strategy that increases the robustness of the solution while simultaneously reducing the number of transport sweeps required for convergence. This is achieved by implementing both a JFNK multiphysics solver and a CMFD-Coupling solver.

The remainder of this dissertation is organized into five major chapters. Chapter 2 details all of the background information that is required for the implementation of JFNK and CMFD-Coupling. First, the transport and CMFD equations are derived in detail in Section 2.1 and Section 2.2, respectively. Next, the nonlinear JFNK solver is derived along with all associated numerical solvers in Section 2.3. Finally, the various feedback models that were used in this work are outlined in Section 2.4. Chapter 3 outlines both the JFNK and CMFD-Coupling methodologies as well as details associated with their implementation. In Chapter 4, a series of smaller problems were explored and their results discussed to investigate the performance of JFNK as both a multiphysics solver in addition to an eigenvalue solver. CMFD-Coupling was also performed on these smaller problems in order to serve as a comparison against JFNK. However, in Chapter 5, CMFD-Coupling is tested on its own for a series of large scale full core problems. Both Watts Bar Unit 1 Cycle 1, and Cycle 2, are examined with different sources of multiphysics feedback enabled and their results discussed. Finally, the conclusions of this work, along with proposed work for the future, are given in Chapter 6.

2. Background Information

The following subsections review the fundamental mathematical and physical concepts required for implementing a CMFD accelerated transport solver. In addition, the algorithms embedded in the JFNK solver are outlined in detail. Lastly, the feedback models that are implemented in this work are discussed.

2.1 High-Order/Low-Order Acceleration

One method for accelerating the convergence of the neutron transport equation is to couple it to the neutron diffusion equation which is an approximation of the transport equation [16] [17]. These methods are known as High-Order/Low-Order (HOLO) Acceleration methods, or Moment-Based Acceleration methods. Recently, these methods have been given much attention and have been implemented successfully [18] [19] [20] [21] [22] [23] [24]. These methods work by solving the much simpler Low-Order (LO) diffusion equation and use its solution as an approximation to the High-Order (HO) transport equation solution. Acceleration is achieved by alternating between solving the HO transport equation and the LO diffusion equations, while the LO equations are chosen such that the HO and LO solutions are identical at convergence. This consistency is achieved by deriving the LO equations from the most recent HO transport sweep. Since the LO system of equations is much easier to solve and converges to the same solution, replacing every other HO solve with a LO solve reduces the total number of transport sweeps required to reach convergence. In addition to acceleration, the discrete consistency of the LO system can also be used to couple other physics [21].

The HOLO method that is most commonly implemented within the neutronics community is called Nonlinear Diffusion Acceleration (NDA), which is more commonly known as Coarse Mesh Finite Difference (CMFD). CMFD is applied to the HO Boltzmann transport equation which removes the angular dependence and leads to an angularly integrated scalar flux balance equation which is far less expensive to solve [19]. Additionally, CMFD is performed on a much coarser mesh than the HO problem, which makes the system of equations smaller and quicker to solve.

2.1.1 Derivation of HOLO from the Neutron Transport Equation

The steady state continuous form of the Boltzmann neutron transport equation is given by

$$\begin{aligned}\boldsymbol{\Omega}\cdot\nabla\psi(\boldsymbol{r},\boldsymbol{\Omega},E)+\Sigma_t(\boldsymbol{r},E)\psi(\boldsymbol{r},\boldsymbol{\Omega},E)&\\ =\frac{\chi(\boldsymbol{r},E)}{4\pi k_{\text{eff}}}\int_0^{\infty}\nu\Sigma_f(\boldsymbol{r},E')\int_0^{4\pi}\psi(\boldsymbol{r},\boldsymbol{\Omega}',E')d\Omega'dE'&\\ +\int_0^{\infty}\int_0^{4\pi}\Sigma_s(\boldsymbol{r},\boldsymbol{\Omega}'\cdot\boldsymbol{\Omega},E'\rightarrow E)\psi(\boldsymbol{r},\boldsymbol{\Omega}',E')d\Omega'dE'\,,&\end{aligned} \quad (1)$$

where $\psi(\boldsymbol{r},\boldsymbol{\Omega},E)$ is the angular neutron flux, such that $\psi(\boldsymbol{r},\boldsymbol{\Omega},E)dr\,d\Omega\,dE$ is the number of neutrons passing through volume element dr about $\boldsymbol{r}$, moving in solid angle $d\Omega$ about direction $\boldsymbol{\Omega}$, and with energies in dE about E. The variable $\chi(E)$ is the fission neutron energy distribution spectrum and k_{eff} is the effective multiplication factor. $\Sigma_t(\boldsymbol{r},E)$ is the total macroscopic cross section, $\nu\Sigma_f(\boldsymbol{r},E')$ is the macroscopic neutron production cross section, and $\Sigma_s(\boldsymbol{r},\boldsymbol{\Omega}'\cdot\boldsymbol{\Omega},E'\rightarrow E)$ is the macroscopic scattering cross section from direction $\boldsymbol{\Omega}'$ and energy E' to direction $\boldsymbol{\Omega}$ and energy E. Several approximations and substitutions can be made in order to make the transport equation easier to work with. First, the zeroth angular moment of the flux, also known as the scalar flux, is given by

$$\phi(\boldsymbol{r},E)=\int_{4\pi}\psi(\boldsymbol{r},\boldsymbol{\Omega},E)d\Omega\,. \quad (2)$$

An approximation is also made by assuming that neutrons are scattered isotopically:

$$\Sigma_s(\boldsymbol{r},\boldsymbol{\Omega}'\cdot\boldsymbol{\Omega},E'\rightarrow E)\approx\frac{\Sigma_s(\boldsymbol{r},E'\rightarrow E)}{4\pi}\,. \quad (3)$$

Substituting Equations 2 and 3 into Equation 1 yields

$$\boldsymbol{\Omega}\cdot\nabla\psi(\boldsymbol{r},\boldsymbol{\Omega},E)+\Sigma_t(\boldsymbol{r},E)\psi(\boldsymbol{r},\boldsymbol{\Omega},E)$$
$$=\frac{1}{4\pi}\left[\int_0^\infty \Sigma_s(\boldsymbol{r},E'\rightarrow E)\phi(\boldsymbol{r},E')dE'+\frac{\chi(\boldsymbol{r},E)}{k_{\text{eff}}}\int_0^\infty \nu\Sigma_f(\boldsymbol{r},E')\phi(\boldsymbol{r},E')dE'\right]. \quad (4)$$

2.1.2 Multigroup Approximation

In order to solve this equation for the dominant eigenvalue-eigenvector pair, (k_{eff},ϕ), some appropriate approximations must be made. The first of which is the multigroup approximation that discretizes the continuous energy variable, E, into energy groups in which the multigroup cross sections are a constant for a given group, g. These multigroup cross sections can be determined exactly for a given reaction type, x, using

$$\Sigma_{x,g}(\boldsymbol{r},\boldsymbol{\Omega})=\frac{\int_{E_g}^{E_{g-1}}\Sigma_x(\boldsymbol{r},E)\psi(\boldsymbol{r},\boldsymbol{\Omega},E)\,dE}{\int_{E_g}^{E_{g-1}}\psi(\boldsymbol{r},\boldsymbol{\Omega},E)\,dE}. \quad (5)$$

However, the angular neutron flux, $\psi(\boldsymbol{r},\boldsymbol{\Omega},E)$, is usually not known when making the multigroup approximation. Therefore an approximation is made assuming that $\psi(\boldsymbol{r},\boldsymbol{\Omega},E)$ is separable:

$$\psi(\boldsymbol{r},\boldsymbol{\Omega},E)\approx\Psi(\boldsymbol{r},E)f(\boldsymbol{r},\boldsymbol{\Omega}). \quad (6)$$

$\Psi(\boldsymbol{r},E)$ is a weighting factor in energy and should be selected to represent the neutron energy spectrum of the problem. Even though this is usually not known prior to solving the problem, this separation approximation is valid for collapsing the continuous energy cross sections as long as $\Psi(\boldsymbol{r},E)$ is reasonably consistent with the energy distribution in the problem. Substituting Equation 6 into Equation 5 removes the angular dependence, yielding

$$\Sigma_{x,g}(\boldsymbol{r})\approx\frac{\int_{E_g}^{E_{g-1}}\Sigma_x(\boldsymbol{r},E)\Psi(\boldsymbol{r},E)\,dE}{\int_{E_g}^{E_{g-1}}\Psi(\boldsymbol{r},E)\,dE}. \quad (7)$$

Similarly, the multigroup scattering cross section is calculated using

$$\Sigma_{s,g'\to g}(\boldsymbol{r}) \approx \frac{\int_{E_{g'}}^{E_{g'-1}} \int_{E_g}^{E_{g-1}} \Sigma_s(\boldsymbol{r}, E' \to E)\Psi(\boldsymbol{r}, E')\, dE\, dE'}{\int_{E_{g'}}^{E_{g'-1}} \Psi(\boldsymbol{r}, E')\, dE'}. \quad 8$$

The fission neutron energy distribution spectrum for a given group, g, is given by

$$\chi_g(\boldsymbol{r}) = \int_{E_g}^{E_{g-1}} \chi(\boldsymbol{r}, E)\, dE\,. \quad 9$$

Using these approximations with Equation 4 leads to the multigroup approximation of the transport equation given by

$$\boldsymbol{\Omega} \cdot \nabla\psi_g(\boldsymbol{r}, \boldsymbol{\Omega}) + \Sigma_{t,g}(\boldsymbol{r})\psi_g(\boldsymbol{r}, \boldsymbol{\Omega}) = \frac{1}{4\pi}\left[\sum_{g'=1}^{G} \Sigma_{s,g'\to g}(\boldsymbol{r})\phi_{g'}(\boldsymbol{r}) + \frac{\chi_g(\boldsymbol{r})}{k_{\text{eff}}} \sum_{g'=1}^{G} \nu\Sigma_{f,g'}(\boldsymbol{r})\phi_{g'}(\boldsymbol{r})\right], g = 1,2,3 \ldots G \quad 10$$

where

$$\psi_g(\boldsymbol{r}, \boldsymbol{\Omega}) = \int_{E_g}^{E_{g-1}} \psi(\boldsymbol{r}, \boldsymbol{\Omega}, E)\, dE \quad 11$$

and the multigroup scalar flux is given by

$$\phi_g(\boldsymbol{r}) = \int_{4\pi} \psi_g(\boldsymbol{r}, \boldsymbol{\Omega}) d\Omega = \int_{4\pi} \int_{E_g}^{E_{g-1}} \psi(\boldsymbol{r}, \boldsymbol{\Omega}, E)\, dE\, d\Omega\,. \quad 12$$

2.1.3 The Discrete Ordinates Approximation

The discrete ordinates approximation [25] discretizes the continuous angular variable $\boldsymbol{\Omega}$. This is done using a quadrature, which approximates the definite integral of a function of angle as a weighted sum of the function at specific values, given by

$$\int_{4\pi} f(\boldsymbol{\Omega})\, d\Omega \approx \sum_{m=1}^{M} w_m f(\boldsymbol{\Omega}_m)\,, m = 1,2,3 \ldots M \tag{13}$$

where w_m are the quadrature weights. Rewriting Equation 10 at discrete angles yields

$$\boldsymbol{\Omega}_m \cdot \nabla\psi_{g,m}(\boldsymbol{r}) + \Sigma_{t,g}(\boldsymbol{r})\psi_{g,m}(\boldsymbol{r}) = \frac{1}{4\pi}\left[\sum_{g'=1}^{G} \Sigma_{s,g'\to g}(\boldsymbol{r})\phi_{g'}(\boldsymbol{r}) + \frac{\chi_g(\boldsymbol{r})}{k_{\text{eff}}}\sum_{g'=1}^{G} \nu\Sigma_{f,g'}(\boldsymbol{r})\phi_{g'}(\boldsymbol{r})\right], g = 1,2,3 \ldots G \tag{14}$$

where

$$\psi_{g,m}(\boldsymbol{r}) = \psi_g(\boldsymbol{r}, \boldsymbol{\Omega}_m)\,. \tag{15}$$

Therefore, the multigroup scalar fluxes in Equation 12 are now represented as a weighted sum

$$\phi_g(\boldsymbol{r}) \approx \sum_{m=1}^{M} w_m \psi_{g,m}(\boldsymbol{r})\,. \tag{16}$$

The discrete ordinates approximation is found to be accurate as long as a sufficient number of angles are used along with an appropriate choice of w_m and $\boldsymbol{\Omega}_m$. Equation 14 is what will be referred to as the HO transport equation.

2.2 Coarse Mesh Finite Difference

CMFD is a type of NDA that utilizes second order multigroup diffusion equations on a spatial mesh that is coarser than the mesh used to solve the HO transport equation [14]. CMFD was first proposed by Smith in 1983 [26] and has been shown to reduce the number of transport sweeps by over a factor of 100 [17]. To apply CMFD, the HO equation must first be reduced to the easy to solve LO problem. This is done by using Equation 2 to take the zeroth angular moment of Equation 14, which results in the neutron continuity equation,

$$\nabla \cdot \boldsymbol{J}_g(\boldsymbol{r}) + \Sigma_{t,g}(\boldsymbol{r})\phi_g(\boldsymbol{r}) = \sum_{g'=1}^{G} \Sigma_{s,g'\to g}(\boldsymbol{r})\phi_{g'}(\boldsymbol{r}) + \frac{\chi_g(\boldsymbol{r})}{k_{\text{eff}}} \sum_{g'=1}^{G} \nu\Sigma_{f,g'}(\boldsymbol{r})\phi_{g'}(\boldsymbol{r}) \,, \tag{17}$$

where the neutron current, $\boldsymbol{J}_g$, is defined as the first angular moment of the flux,

$$\boldsymbol{J}_g(\boldsymbol{r}) = \int_{4\pi} \boldsymbol{\Omega}\psi_g(\boldsymbol{r},\boldsymbol{\Omega})\, d\Omega \,. \tag{18}$$

The standard neutron diffusion approximation is made using Fick's law, where the neutron current density is assumed to be proportional to the spatial gradient of the flux,

$$\boldsymbol{J}_g(\boldsymbol{r}) \approx -D_g(\boldsymbol{r})\nabla\phi_g(\boldsymbol{r}) \,, \tag{19}$$

where D_g is the standard neutron diffusion coefficient $\frac{1}{3\Sigma_{t,g}(\boldsymbol{r})}$. Substituting Equation 19 into Equation 17 yields the neutron diffusion equation,

$$\begin{aligned} &-\nabla \cdot D_g(\boldsymbol{r})\nabla\phi_g(\boldsymbol{r}) + \Sigma_{t,g}(\boldsymbol{r})\phi_g(\boldsymbol{r}) \\ &\qquad = \sum_{g'=1}^{G} \Sigma_{s,g'\to g}(\boldsymbol{r})\phi_{g'}(\boldsymbol{r}) + \frac{\chi_g(\boldsymbol{r})}{k_{\text{eff}}} \sum_{g'=1}^{G} \nu\Sigma_{f,g'}(\boldsymbol{r})\phi_{g'}(\boldsymbol{r}) \,. \end{aligned} \tag{20}$$

However, the diffusion approximation in Equation 19 can be improved by including a nonlinear drift term for the current [18],

$$\boldsymbol{J}_g(\boldsymbol{r}) = -D_g(\boldsymbol{r})\nabla\phi_g(\boldsymbol{r}) + \widehat{D}_g(\boldsymbol{r})\phi_g(\boldsymbol{r}) \,. \tag{21}$$

$\widehat{D}_g$ is a consistency term that ensures the HO and LO problems are discretely consistent upon convergence. This term not only forces consistency, but also causes the acceleration to be nonlinear because it is defined as a function of HO quantities from Equation 21

$$\widehat{D}_g(\boldsymbol{r}) = \frac{\boldsymbol{J}_g^{HO}(\boldsymbol{r}) + D_g(\boldsymbol{r})\nabla\phi_g^{HO}(\boldsymbol{r})}{\phi_g^{HO}(\boldsymbol{r})}. \tag{22}$$

Substituting Equation 21 into Equation 17 and moving the in-group scattering term, $\Sigma_{s,g\to g}(\boldsymbol{r})$, to the left hand side yields the LO system to be used by CMFD:

$$\begin{aligned}\nabla\cdot\left[-D_g(\boldsymbol{r})\nabla\phi_g(\boldsymbol{r}) + \widehat{D}_g(\boldsymbol{r})\phi_g(\boldsymbol{r})\right] + \left(\Sigma_{t,g}(\boldsymbol{r}) - \Sigma_{s,g\to g}(\boldsymbol{r})\right)\phi_g(\boldsymbol{r}) \\ = \sum_{\substack{g'=1 \\ g'\neq g}}^{G} \Sigma_{s,g'\to g}(\boldsymbol{r})\phi_{g'}(\boldsymbol{r}) + \frac{\chi_g(\boldsymbol{r})}{k_{\text{eff}}}\sum_{g'=1}^{G} \nu\Sigma_{f,g'}(\boldsymbol{r})\phi_{g'}(\boldsymbol{r}).\end{aligned} \tag{23}$$

With the HO and LO problems in hand, the CMFD algorithm is as follows:

1. Start with an initial guess for both the eigenvector and eigenvalue, $\phi_g^{(0)}$ and $k^{(0)}$
   Do until converged
2. Solve the HO equation (14) for $\phi_g^{HO}(r)$ and $\boldsymbol{J}_g^{HO}(r)$
3. Solve for $\widehat{D}$ using Equation 22
4. Solve the LO equation (23) for ϕ_g and k
   End Do

For now, the details of solving the HO and LO sets of equations have been overlooked, but these details will be examined in depth in the following sections.

2.2.1 Method of Characteristics

One of the neutron transport codes developed in CASL is Michigan Parallel Analysis based on Characteristic Tracing (MPACT) [27]. MPACT employs a 2D/1D approach to solving the neutron transport problem. The problem is broken up into a series of 2D axial planes whose axial transverse leakage is solved using a 1D axial calculation. Then each plane is solved independently using the Method of Characteristics (MOC), first proposed by J. R. Askew in 1972 [28]. MOC is a general mathematical technique for solving first-order partial differential equations and is an attractive neutron transport technique because it avoids some of the

drawbacks associated with other methods; Monte Carlo methods are very time consuming for large problems in which fine flux details are required, and the execution times and memory requirements of the method of collision probabilities increases with the square of the number of mesh. In contrast, MOC is relatively simple to implement while computation time and memory requirements scale linearly with the spatial and angular detail of the problem [29]. MOC is implemented by solving the characteristic form of the Boltzmann neutron transport equation along discrete tracks, oriented at different angles that are traced over the explicit problem geometry. Every unique angle is given a weight and the average angular flux along each track is calculated. The solutions from each track are combined to produce a very accurate description of the flux distribution throughout the problem.

To obtain the characteristic form of the transport equations, the spatial variable $\boldsymbol{r}$ from the HO equation, Equation 14, is transformed using a change of variables to represent the characteristic direction:

$$\boldsymbol{r} = \boldsymbol{r}_0 + s\boldsymbol{\Omega}_m \begin{cases} x(s) = x_0 + s\Omega_{m,x} \\ y(s) = y_0 + s\Omega_{m,y} \\ z(s) = z_0 + s\Omega_{m,z} \end{cases}, \tag{24}$$

where $\boldsymbol{r}_0$ is an arbitrary reference point and s is the characteristic segment length along the discreet direction $\boldsymbol{\Omega}_m$. Substituting into Equation 14 yields

$$\frac{d\psi_{g,m}}{ds}(\boldsymbol{r}_0 + s\boldsymbol{\Omega}_m) + \Sigma_{t,g}(\boldsymbol{r}_0 + s\boldsymbol{\Omega}_m)\,\psi_{g,m}(\boldsymbol{r}_0) = Q_g(\boldsymbol{r}_0 + s\boldsymbol{\Omega}_m), \tag{25}$$

where the right hand side has been rewritten as

$$Q_g(\boldsymbol{r}_0 + s\boldsymbol{\Omega}_m) = \frac{1}{4\pi}\left[\sum_{g'=1}^{G} \Sigma_{s,g'\to g}(\boldsymbol{r}_0 + s\boldsymbol{\Omega}_m)\,\phi_{g'}(\boldsymbol{r}_0 + s\boldsymbol{\Omega}_m) + \frac{\chi_g(\boldsymbol{r}_0 + s\boldsymbol{\Omega}_m)}{k_{\text{eff}}} \sum_{g'=1}^{G} \nu\Sigma_{f,g'}(\boldsymbol{r}_0 + s\boldsymbol{\Omega}_m)\phi_{g'}(\boldsymbol{r}_0 + s\boldsymbol{\Omega}_m)\right]. \quad (26)$$

Equation 25 can be solved analytically using the integrating factor

$$e^{-\int_0^s \Sigma_{t,g}(\boldsymbol{r}_0 + s'\boldsymbol{\Omega}_m)ds'}, \quad (27)$$

resulting in

$$\psi_{g,m}(\boldsymbol{r}_0 + s\boldsymbol{\Omega}_m) = \psi_{g,m}(\boldsymbol{r}_0)e^{-\int_0^s \Sigma_{t,g}(\boldsymbol{r}_0 + s'\boldsymbol{\Omega}_m)ds'} + \int_0^s Q_g(\boldsymbol{r}_0 + s\boldsymbol{\Omega}_m)e^{-\int_{s'}^s \Sigma_{t,g}(\boldsymbol{r}_0 + s''\boldsymbol{\Omega}_m)ds''}ds'. \quad (28)$$

Therefore, Equation 28 with Equation 26 is the steady state solution of the characteristics form of the Boltzmann neutron transport equation with isotropic scattering. However, in order to solve it numerically, the problem space must be divided into discrete regions. To further simplify the problem, the material properties in a given spatial region are assumed to be constant. With this simplification, Equation 28 and Equation 26 can be rewritten to describe a point s along a single characteristic ray, k, passing through a discrete region i:

$$\psi_{i,g,m,k}(s) = \psi^{in}_{i,g,m,k}e^{-\Sigma_{t,i,g}s} + \int_0^s Q_{i,g}(s')e^{-\Sigma_{t,i,g}(s_{i,k}-s')}ds', \quad (29)$$

$$Q_{i,g}(s) = \frac{1}{4\pi}\left[\sum_{g'=1}^{G} \Sigma_{s,i,g'\to g}\,\phi_{i,g'}(s) + \frac{\chi_{i,g}}{k_{\text{eff}}} \sum_{g'=1}^{G} \nu\Sigma_{f,i,g'}\phi_{i,g'}(s)\right], \quad (30)$$

where the incoming flux into a discrete region i is defined as

$$\psi_{i,g,m,k}^{in} = \psi_{i,g,m}(\boldsymbol{r}_0) = \psi_{i,g,m,k}(s = 0)\,.$$

The outgoing flux leaving a discrete region is found by substituting the total characteristic track length, $s_{i,k}$, into Equation 29:

$$\psi_{i,g,m,k}^{out} = \psi_{i,g,m,k}^{in} e^{-\Sigma_{t,i,g} s_{i,k}} + \int_0^{s_{i,k}} Q_{i,g}(s') e^{-\Sigma_{t,i,g}(s_{i,k} - s')} ds', \tag{31}$$

where

$$\psi_{i,g,m,k}^{out} = \psi_{i,g,m}\left(\boldsymbol{r}_0 + s_{i,k}\boldsymbol{\Omega}_m\right) = \psi_{i,g,m,k}\left(s = s_{i,k}\right).$$

The incoming and outgoing fluxes, $\psi_{i,g,m,k}^{in}$ and $\psi_{i,g,m,k}^{out}$ respectively, are coupled such that for two neighboring discrete regions i and $i+1$, the outgoing flux in region i equals the incoming flux in region $i+1$ along ray k, for energy group g, and in direction m.

To simplify the last remaining integral in Equation 29, it is assumed that the neutron source, $Q_{i,g}$, is constant within each discretized region. This assumption is called the flat source approximation. Applying the flat source approximation to Equation 29 allows for the remaining integral to be solved analytically, leading to:

$$\psi_{i,g,m,k}(s) = \psi_{i,g,m,k}^{in} e^{-\Sigma_{t,i,g} s} + \frac{Q_{i,g}}{\Sigma_{t,i,g}}\left[1 - e^{-\Sigma_{t,i,g} s}\right], \tag{32}$$

where the region averaged flat source, $Q_{i,g}$, is given by

$$Q_{i,g} = \frac{1}{4\pi}\left[\sum_{g'=1}^{G} \Sigma_{s,i,g' \to g}\, \phi_{i,g'} + \frac{\chi_{i,g}}{k_{\text{eff}}} \sum_{g'=1}^{G} \nu\Sigma_{f,i,g'} \phi_{i,g'}\right]. \tag{33}$$

The scalar flux, $\phi_{i,g}$, is defined as

$$\phi_{i,g} = \int_{4\pi} \bar{\psi}_{i,g}(\mathbf{\Omega}) d\Omega \approx \sum_{m=1}^{M} w_m \bar{\psi}_{i,g,m} \,. \tag{34}$$

The region averaged angular flux, $\bar{\psi}_{i,g,m}$, from Equation 34 is computed from

$$\bar{\psi}_{i,g,m} = \frac{\sum_{k \in i} \tilde{\psi}_{i,g,m,k} s_{i,k} \delta A_k}{\sum_{k \in i} s_{i,k} \delta A_k}, \tag{35}$$

where δA_k represents the cross sectional area of the characteristic ray k and $\tilde{\psi}_{i,g,m,k}$ is the segment averaged angular flux given by

$$\tilde{\psi}_{i,g,m,k} = \frac{\int_0^{s_{i,k}} \psi_{i,g,m,k}(s') ds'}{\int_0^{s_{i,k}} ds'} = \frac{1}{s_{i,k}} \left[\frac{\psi_{i,g,m,k}^{in} - \psi_{i,g,m,k}^{out}}{\Sigma_{t,i,g}} + \frac{Q_{i,g} s_{i,k}}{\Sigma_{t,i,g}} \right], \tag{36}$$

where $\psi_{i,g,m,k}(s)$ is from Equation 32. Therefore Equation 36 and Equation 32 must be solved at the endpoints of each characteristic ray in each discrete region in order to formulate the MOC solution for the flux:

$$\psi_{i,g,m,k}^{out} = \psi_{i,g,m,k}^{in} e^{-\Sigma_{t,i,g} s_{i,m,k}} + \frac{Q_{i,g,m}}{\Sigma_{t,i,g}} \left[1 - e^{-\Sigma_{t,i,g} s_{i,m,k}} \right], \tag{37}$$

and

$$Q_{i,g,m} = \frac{1}{4\pi} \left[\sum_{g'=1}^{G} \Sigma_{s,i,g' \to g} \bar{\phi}_{i,g'} + \frac{\chi_{i,g}}{k_{\text{eff}}} \sum_{g'=1}^{G} \nu \Sigma_{f,i,g'} \bar{\phi}_{i,g'} \right]. \tag{38}$$

2.2.2 Applying CMFD

The first step in implementing CMFD is to condense the cross sections and fluxes from the fine HO transport mesh to the LO coarse mesh. These quantities can not only be collapsed spatially, but the group structure can also be condensed to simplify the LO problem even more. This is done by flux weighting the cross sections and summing over the groups to be collapsed:

$$\Sigma_{x,\boldsymbol{g}}^{LO} = \frac{\int \sum_{g\in\boldsymbol{g}} \Sigma_{x,g}^{HO}(\boldsymbol{r})\phi_g^{HO}(\boldsymbol{r})dV}{\int \sum_{g\in\boldsymbol{g}} \phi_g^{HO}(\boldsymbol{r})dV}, \tag{39}$$

where x is a given reaction type, V is the fine mesh region volume, g are the HO energy groups, and $\boldsymbol{g}$ are the CMFD energy groups. The scattering cross section is treated the same way except with an additional summation over the initial particle energy group

$$\Sigma_{s,\boldsymbol{g}'\to\boldsymbol{g}}^{LO} = \frac{\int \sum_{g'\in\boldsymbol{g}'} \sum_{g\in\boldsymbol{g}} \Sigma_{s,g'\to g}^{HO}(\boldsymbol{r})\phi_g^{HO}(\boldsymbol{r})dV}{\int \sum_{g\in\boldsymbol{g}} \phi_g^{HO}(\boldsymbol{r})dV}. \tag{40}$$

In a similar fashion, the HO fine fluxes are volume weighted to collapse to the LO coarse fluxes

$$\phi_{\boldsymbol{g}}^{LO} = \frac{\int \sum_{g\in\boldsymbol{g}} \phi_g^{HO}(\boldsymbol{r})dV}{\int dV}. \tag{41}$$

In order to calculate the LO diffusion coefficient, the transport cross section is first flux weighted in space,

$$\Sigma_{tr,g}^{LO} = \frac{\int \Sigma_{tr,g}^{HO}(\boldsymbol{r})\phi_g^{HO}(\boldsymbol{r})dV}{\int \phi_g^{HO}(\boldsymbol{r})dV}, \tag{42}$$

and then this spatially collapsed transport cross section is used to flux weight the diffusion coefficient in energy:

$$D_{\boldsymbol{g}}^{LO} = \frac{\sum_{g\in\boldsymbol{g}} \frac{1}{3\Sigma_{tr,g}^{LO}} \phi_g^{HO}}{\sum_{g\in\boldsymbol{g}} \phi_g^{HO}}. \tag{43}$$

When expanding the coarse mesh fluxes back to the fine mesh fluxes the following discontinuity factor is used:

$$f_g = \frac{\phi_g^{HO}}{\phi_g^{LO}}. \tag{44}$$

In order to get the neutron balance in a given mesh cell, (i,j,k), the volumetric integral is taken of Equation 23, leading to

$$\begin{aligned}\int \nabla \cdot \boldsymbol{J}_g(\boldsymbol{r})\, dV + \Delta x^{i,j,k}\Delta y^{i,j,k}\Delta z^{i,j,k}\left(\Sigma_{t,g}^{i,j,k} - \Sigma_{s,g\to g}^{i,j,k}\right)\phi_g^{i,j,k} \\ = \Delta x^{i,j,k}\Delta y^{i,j,k}\Delta z^{i,j,k}\sum_{g'\neq g}^{G}\Sigma_{s,g'\to g}^{i,j,k}\,\phi_{g'}^{i,j,k} \\ + \Delta x^{i,j,k}\Delta y^{i,j,k}\Delta z^{i,j,k}\frac{\chi_g^{i,j,k}}{k_{\text{eff}}}\sum_{g'=1}^{G}\nu\Sigma_{f,g'}^{i,j,k}\phi_{g'}^{i,j,k}\end{aligned} \tag{45}$$

where $\Delta x^{i,j,k}$, $\Delta y^{i,j,k}$, and $\Delta z^{i,j,k}$ represent the thickness of cell (i,j,k) in the x, y, and z directions, respectively. When coupling two neighboring coarse mesh regions, the volume integral is replaced with the surface integral over the cell boundary using the divergence theorem,

$$\begin{aligned}\int \nabla \cdot \boldsymbol{J}_g(\boldsymbol{r})\, dV = \Delta y^{i,j,k}\Delta z^{i,j,k}\left(J_g^{i+\frac{1}{2},j,k} - J_g^{i-\frac{1}{2},j,k}\right) \\ + \Delta x^{i,j,k}\Delta z^{i,j,k}\left(J_g^{i,j+\frac{1}{2},k} - J_g^{i,j-\frac{1}{2},k}\right) + \Delta x^{i,j,k}\Delta y^{i,j,k}\left(J_g^{i,j,k+\frac{1}{2}} - J_g^{i,j,k-\frac{1}{2}}\right),\end{aligned} \tag{46}$$

where the $\frac{1}{2}$ superscripts correspond to the neighboring cell interfaces. An example of neighboring cells in the x direction is shown in Figure 1.

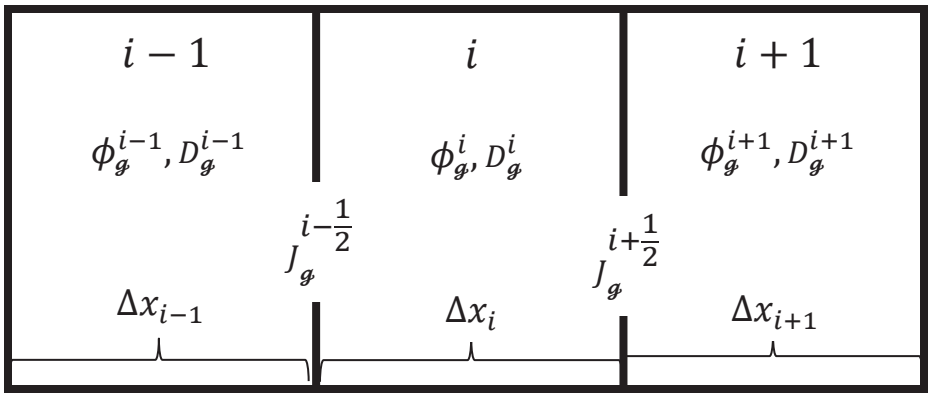

Figure 1. Neighboring CMFD cell boundaries in the $\boldsymbol{x}$ direction

The definition for the neutron current, $\boldsymbol{J_g}$, from Equation 21 is modified with the addition of finite difference approximations of the flux at the cell boundary using the flux in the neighboring cells:

$$\begin{aligned} J_g^{i\pm\frac{1}{2},j,k} &= -\widetilde{D}_g^{i\pm\frac{1}{2},j,k}\left(\pm\phi_g^{i\pm1,j,k}\mp\phi_g^{i,j,k}\right)+\widehat{D}_g^{i\pm\frac{1}{2},j,k}\left(\phi_g^{i\pm1,j,k}+\phi_g^{i,j,k}\right) \\ J_g^{i,j\pm\frac{1}{2},k} &= -\widetilde{D}_g^{i,j\pm\frac{1}{2},k}\left(\pm\phi_g^{i,j\pm1,k}\mp\phi_g^{i,j,k}\right)+\widehat{D}_g^{i,j\pm\frac{1}{2},k}\left(\phi_g^{i,j\pm1,k}+\phi_g^{i,j,k}\right). \\ J_g^{i,j,k\pm\frac{1}{2}} &= -\widetilde{D}_g^{i,j,k\pm\frac{1}{2}}\left(\pm\phi_g^{i,j,k\pm1}\mp\phi_g^{i,j,k}\right)+\widehat{D}_g^{i,j,k\pm\frac{1}{2}}\left(\phi_g^{i,j,k\pm1}+\phi_g^{i,j,k}\right) \end{aligned} \quad 47$$

The equation for the nonlinear diffusion coefficient correction factor, $\widehat{D}$, is found by rearranging Equation 47. It is because of this reason that the two fluxes that multiply the $\widehat{D}$ term are summed to avoid the potential of ever dividing by zero. It should be noted that the terms in Equation 47 are from the solution to the HO problem. The single diffusion coefficient in Equation 21 has been replaced with the $\widetilde{D}$ term that represents the linear coupling between the current and flux:

$$\widetilde{D}_{\boldsymbol{g}}^{i\pm\frac{1}{2},j,k} = \frac{2D_{\boldsymbol{g}}^{i\pm1,j,k}D_{\boldsymbol{g}}^{i,j,k}}{D_{\boldsymbol{g}}^{i\pm1,j,k}\Delta x^{i,j,k} + D_{\boldsymbol{g}}^{i,j,k}\Delta x^{i\pm1,j,k}}$$
$$\widetilde{D}_{\boldsymbol{g}}^{i,j\pm\frac{1}{2},k} = \frac{2D_{\boldsymbol{g}}^{i,j\pm1,k}D_{\boldsymbol{g}}^{i,j,k}}{D_{\boldsymbol{g}}^{i,j\pm1,k}\Delta y^{i,j,k} + D_{\boldsymbol{g}}^{i,j,k}\Delta y^{i,j\pm1,k}}. \quad (48)$$
$$\widetilde{D}_{\boldsymbol{g}}^{i,j,k\pm\frac{1}{2}} = \frac{2D_{\boldsymbol{g}}^{i,j,k\pm1}D_{\boldsymbol{g}}^{i,j,k}}{D_{\boldsymbol{g}}^{i,j,k\pm1}\Delta z^{i,j,k} + D_{\boldsymbol{g}}^{i,j,k}\Delta z^{i,j,k\pm1}}$$

Substituting Equation 46 into Equation 45 yields the full 3D CMFD neutron balance equation for an interior cell:

$$\begin{aligned}
&\Delta y^{i,j,k}\Delta z^{i,j,k}\left(J_{\boldsymbol{g}}^{i+\frac{1}{2},j,k} - J_{\boldsymbol{g}}^{i-\frac{1}{2},j,k}\right) + \Delta x^{i,j,k}\Delta z^{i,j,k}\left(J_{\boldsymbol{g}}^{i,j+\frac{1}{2},k} - J_{\boldsymbol{g}}^{i,j-\frac{1}{2},k}\right) \\
&\quad + \Delta x^{i,j,k}\Delta y^{i,j,k}\left(J_{\boldsymbol{g}}^{i,j,k+\frac{1}{2}} - J_{\boldsymbol{g}}^{i,j,k-\frac{1}{2}}\right) \\
&\quad + \Delta x^{i,j,k}\Delta y^{i,j,k}\Delta z^{i,j,k}\left(\Sigma_{t,\boldsymbol{g}}^{i,j,k} - \Sigma_{s,\boldsymbol{g}\to\boldsymbol{g}}^{i,j,k}\right)\phi_{\boldsymbol{g}}^{i,j,k} \\
&\quad = \Delta x^{i,j,k}\Delta y^{i,j,k}\Delta z^{i,j,k}\sum_{\boldsymbol{g}'\neq\boldsymbol{g}}^{G}\Sigma_{s,\boldsymbol{g}'\to\boldsymbol{g}}^{i,j,k}\phi_{\boldsymbol{g}'}^{i,j,k} \\
&\quad + \Delta x^{i,j,k}\Delta y^{i,j,k}\Delta z^{i,j,k}\frac{\chi_{\boldsymbol{g}}^{i,j,k}}{k_{\text{eff}}}\sum_{\boldsymbol{g}'=1}^{G}\nu\Sigma_{f,\boldsymbol{g}'}^{i,j,k}\phi_{\boldsymbol{g}'}^{i,j,k}.
\end{aligned} \quad (49)$$

After fully substituting Equation 47 into Equation 49, it can be rewritten in operator notation as

$$\mathbb{D}\boldsymbol{\phi} + \mathbb{T}\boldsymbol{\phi} - \mathbb{S}\boldsymbol{\phi} = \frac{1}{k_{\text{eff}}}\mathbb{F}\boldsymbol{\phi}, \quad (50)$$

where $\mathbb{D}$ is a matrix containing all of the diffusion streaming terms, $\mathbb{T}$ is a matrix containing all of the total cross section terms, $\mathbb{S}$ is a matrix containing all of the scattering terms, and $\mathbb{F}$ is the fission operator, or matrix, which contains the fission neutron production terms. Equation 50 can be rewritten as a generalized eigenvalue problem given by

$$\mathbb{M}\boldsymbol{\phi} = \frac{1}{k_{\text{eff}}}\mathbb{F}\boldsymbol{\phi} \tag{51}$$

where

$$\mathbb{M} = \mathbb{D} + \mathbb{T} - \mathbb{S}\,. \tag{52}$$

This problem can be solved using a wide variety of numerical solvers, some of which will be described in the following section.

2.3 Numerical Solvers

In modern reactor simulation codes, the most widespread method used for solving the k-eigenvalue problem is the power method [30]. However, the power method can be very slow solving problems that are common in reactor core simulations. An alternative would be to solve for the eigenvalue using a Jacobian-Free Newton-Krylov method. Additionally, JFNK can solve for the eigenvalue while simultaneously solving coupled multiphysics problems.

The following sections are devoted to reviewing the background details and formulation of numerical solvers used in this work. First the standard power method is outlined, followed by all of the necessary components to build up a Jacobian-Free Newton-Krylov solver.

2.3.1 Power Method

In order to solve the eigenvalue problem in Equation 51, an iterative method must be used. In reactor physics applications, the simplest and most common method used to find the eigenvalue-eigenvector pair is the power method, or power iteration. In addition to its simplicity, the power method is an attractive option because it only converges to an eigenvector that corresponds to the largest, or dominant, eigenvalue, k_{eff}. In reactor applications, only the dominant eigenvalue leads to a physical answer: one where the flux distribution is non-negative everywhere throughout the problem [30]. The steps of the power iteration are shown in Algorithm 1.

Algorithm 1. Power Method

1. Select initial k_0 and ϕ_0
   for $n = 0, 1, 2, \ldots$
2. Solve $\mathbb{M}\phi_{n+1} = \frac{1}{k_n}\mathbb{F}\phi_n$ for ϕ_{n+1}
3. $k_{n+1} = k_n \frac{\|\mathbb{F}\phi_{n+1}\|}{\|\mathbb{F}\phi_n\|}$

Upon convergence, ϕ_n is the eigenvector that corresponds to the dominant eigenvalue k_{eff}. The convergence rate of the power iteration is linear and is determined by the ratio of the second largest eigenvalue, k_2, to the dominant eigenvalue, known as the dominance ratio: $|k_2/k_{\text{eff}}|$. Therefore, the power iteration can converge very slowly if $k_2 \approx k_{\text{eff}}$ [22]. In practical reactor applications, this is a common occurrence in physically large systems [31]. However, the dominance ratio of a problem can be reduced using an eigenvalue deflation method.

The power iteration is also inefficient in solving problems that have a high scattering ratio in addition to a large dominance ratio. As a result, there are different methods that can accelerate the power method which are commonly used in reactor applications. However, these methods are not discussed here, because the goal of this work is to replace the power method with a JFNK method that solves the k-eigenvalue problem. The different components needed for building a JFNK solver framework are examined in detail in the subsequent sections.

2.3.2 Krylov Subspaces

Many well-known modern iterative methods utilize Krylov subspaces: Arnoldi, Generalized Minimal Residuals (GMRES), Lanczos, Conjugate Gradients (CG), and Biconjugate Gradient Stabilized (BiCGSTAB) methods. Given a $m \times m$ matrix A and a vector b of dimension m, then the nth-dimensional Krylov subspace is defined as

$$\mathcal{K}_n(A, b) = span\{b, Ab, A^2b, \ldots, A^{n-1}b\}. \quad 53$$

Krylov subspace methods project an m-dimensional problem onto a lower-dimensional Krylov subspace. When solving a linear system of equations $Ax = b$, where x is the solution vector, the residual for any approximate solution vector x_n is defined as

$$r_n = b - Ax_n \,. \tag{54}$$

A property of Krylov subspace methods is that, for an initial approximation of the solution x_0 and residual r_0, the solution vector x lies in the affine space given by

$$x \in x_0 + \mathcal{K}_n(A, r_0) \,. \tag{55}$$

Therefore, the goal of Krylov subspace methods is to generate a series of approximate solution vectors such that their corresponding residuals converge to zero [32]. When the residual reaches zero, then $x_n = x$. In order for these methods to converge in a finite number of steps, it is required that the residuals be linearly independent.

2.3.3 Arnoldi Iteration

The Arnoldi iteration is an iterative method that uses Krylov subspaces to reduce a non-Hermitian matrix, A, to an upper Hessenberg form, H, by a series of orthogonal similarity transformations:

$$AQ = QH \,, \tag{56}$$

where Q is an orthogonal matrix [33]. The step-by-step process is shown below in Algorithm 2.

Algorithm 2. Arnoldi Iteration

1. $q_1 = b/\|b\|_2$, where b is arbitrary
   for $k = 1$ to m
2. $\quad v = Aq_k$
   $\quad$ for $i = 1$ to k
3. $\quad\quad h_{i,k} = q_i^* v$
4. $\quad\quad v = v - h_{i,k} q_i$
5. $\quad h_{k+1,k} = \|v\|_2$
6. $\quad q_{k+1} = v/h_{k+1,k}$

The columns of the matrix Q generated in the Arnoldi algorithm form an orthonormal basis for the Krylov subspace $\mathcal{K}_k$. Step 4 in the Arnoldi algorithm is where the vector v is orthogonalized to all the previous basis vectors using a standard Gram-Schmidt method. Therefore, the Arnoldi iteration can be thought of as the procedural formulation of orthonormal bases for successive Krylov subspaces. If Q_k is the $m \times k$ matrix whose columns are the first k columns of Q, and $\widetilde{H}_k$ is the upper-left $(k+1) \times k$ portion of H then

$$AQ_k = Q_{k+1}\widetilde{H}_k \,. \tag{57}$$

Therefore, after k Arnoldi iterations, the kth column of this matrix is

$$Aq_k = h_{1,k}q_1 + h_{2,k}q_2 + \cdots + h_{k,k}q_k + h_{k+1,k}q_{k+1} \,. \tag{58}$$

The vector q_{k+1} comes from a recurrence relation involving itself and all of the previous Krylov vectors. However, when the Arnoldi iteration is performed for $k = m$ iterations, the final vector v already lies in the Krylov subspace $\mathcal{K}_k$. As a result, orthogonalizing it to all of the previous Krylov vectors results in $v = 0$. Therefore, there is no $h_{m+1,m}$ value or q_{m+1} vector to be calculated in steps 5 and 6 of the Arnoldi algorithm, and Equation 57 becomes Equation 56.

2.3.4 GMRES

One of the linear solvers used in this work is the Generalized Minimal Residuals (GMRES) method, which utilizes Arnoldi's method to solve a linear system of equations $Ax = b$. GMRES was first presented by Youcef Saad and Martin Schultz in 1986 [34]. The idea is that, after each Arnoldi iteration, a least squares problem is solved to determine the minimum 2-norm of the residual over the affine space $x_0 + \mathcal{K}_k$ [35]. This is done in order to form a good approximate solution vector x_k without having to carry out the Arnoldi iteration to completion, i.e. $k = m$. In GMRES the arbitrary vector b in Step 1 on the Arnoldi process is chosen to be the initial residual r_0 such that the initial Krylov vector is given by

$$q_1 = r_0 / \|r_0\|_2 \,. \tag{59}$$

Since Q_k from the Arnoldi iteration forms an orthonormal basis for the subspace $\mathcal{K}_k$, then any vector x_k in $x_0 + \mathcal{K}_k$ can be written as

$$x_k = x_0 + Q_k y_k \,, \tag{60}$$

where y_k is a k-vector. Using Equation 57 and Equation 60 the residual r_k can be rewritten as

$$\begin{aligned} r_k &= b - Ax_k \\ &= b - A(x_0 + Q_k y_k) \\ &= r_0 - AQ_k y_k \\ &= r_0 - Q_{k+1}\widetilde{H}_k y_k \,. \end{aligned} \tag{61}$$

Since the first column of Q_{k+1} is defined by Equation 59, the residual r_0 can be rewritten as

$$r_0 = Q_{k+1}\eta \,, \tag{62}$$

where η is the $(k+1)$-vector:

$$\eta = \begin{bmatrix} \|r_0\|_2 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{63}$$

Substituting Equation 62 into Equation 61 yields

$$r_k = Q_{k+1}\left(\eta - \widetilde{H}_k y_k\right). \tag{64}$$

Since the columns of Q_{k+1} are orthonormal, the 2-norm of the residual becomes

$$\|r_k\|_2 = \left\|\eta - \widetilde{H}_k y_k\right\|_2 . \tag{65}$$

Therefore, it is desired to solve for the vector y_k that minimizes Equation 65, or, in other words, to solve the least squares problem given by

$$\min_{y_k \in \mathcal{K}_k} \left\| \eta - \widetilde{H}_k y_k \right\|_2 . \qquad 66$$

Since the Hessenberg matrix $\widetilde{H}_k$ is nearly upper triangular, solving the least squares problem via a QR factorization is relatively inexpensive. The algorithm used in this work to solve the least squares problem is a Householder QR factorization followed by back substitution. Once y_k is determined the approximate solution x_k is calculated using Equation 60. The steps of the GMRES algorithm are shown in Algorithm 3.

Algorithm 3. GMRES

1. $q_1 = r_0 / \|r_0\|_2$
   for $k = 1, 2, 3, \ldots$
2. Complete iteration k of Arnoldi iteration, Algorithm 2
3. Solve least squares problem $\min_{y_k \in \mathcal{K}_k} \left\| \eta - \widetilde{H}_k y_k \right\|_2$
4. $x_k = x_0 + Q_k y_k$

It may have been noted by the reader that there is a potential breakdown of the Arnoldi iteration within GMRES at Step 6 of Algorithm 2, when $h_{k+1,k} = \|v\|_2 = 0$. However, this only happens when the residual vector is zero for step k. Therefore, if A is nonsingular, GMRES breaks down in the kth iteration if and only if the approximate solution is exact, i.e., $x_k = x$ [36]. One of the drawbacks for GMRES is the fact that, in the kth iteration, the Arnoldi procedure must orthogonalize the vector v to all k previous basis vectors, and as a result they must all be stored in memory. If the size of A is large, this could be computationally prohibitive. One possible solution is to restart GMRES after a certain number of iterations and use x_k as the initial guess for a new GMRES iteration.

2.3.5 Newton's Method

Newton's method, or the Newton-Raphson method, is an iterative method for finding the roots of a real-valued function, i.e., $f(x) = 0$. If the current root approximation is given by x_k, and the

subsequent approximation is given by x_{k+1}, then the method can be derived from a Taylor series expansion of $f(x_{k+1})$ about x_k:

$$f(x_{k+1}) = f(x_k) + f'(x_k)(x_{k+1} - x_k) + \cdots \tag{67}$$

Since it is desired that as k gets large, $f(x_{k+1})$ will approach zero, the right hand side of Equation 67 is set to zero and the higher order terms are ignored, leading to

$$0 = f(x_k) + f'(x_k)(x_{k+1} - x_k)\,. \tag{68}$$

Solving for x_{k+1} yields Newton's method:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}. \tag{69}$$

Provided the initial guess is sufficiently close to a root, Newton's method has a quadratic convergence rate which is a desirable feature in numerical linear algebra. Newton's method can also be extended to solve an m-dimensional system of nonlinear equations, $F(x) = 0$, where x is now a vector of length m. This version of Newton's method has the same form as Equation 69, but is usually written as the linear system

$$J(x_k)\delta x_k = -F(x_k)\,, \tag{70}$$

where $\delta x_k = x_{k+1} - x_k$ and $J(x_k) \equiv F'(x_k)$ is the Jacobian matrix

$$J(x_k) = \begin{bmatrix} \dfrac{\partial F_1(x_k)}{\partial x_1} & \cdots & \dfrac{\partial F_1(x_k)}{\partial x_m} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial F_m(x_k)}{\partial x_1} & \cdots & \dfrac{\partial F_m(x_k)}{\partial x_m} \end{bmatrix}. \tag{71}$$

Therefore, the vector δx_k is the solution to Equation 70 and is added to the current root approximation, x_k, in order to obtain the approximation at the next step. The initial guess, δx_0, is typically set to zero since as the method converges, δx_0 should approach zero [4]. This process is outlined in Algorithm 4.

Algorithm 4. Newton's Method

1. Select x_0 sufficiently close to a root
   for $k = 0, 1, 2, \ldots$
2. Solve $J(x_k)\delta x_k = -F(x_k)$
3. $x_{k+1} = x_k + \delta x_k$

Step 2 of Newton's method can be solved using a wide variety of linear solvers, including a Krylov subspace method like GMRES. Although Equation 70 reduced a nonlinear system of equations to a series of linear equations, the Jacobian must be evaluated and stored at each iteration step. This can be prohibitively expensive or impossible if the derivatives aren't available in a closed form. The next section looks at a way to approximate the Jacobian without sacrificing the quadratic convergence rate.

2.3.6 Jacobian-Free Newton-Krylov Methods

A Jacobian-Free Newton-Krylov (JFNK) method solves a nonlinear system of equations using Newton iterations without explicitly forming the Jacobian. It should be noted that in the Arnoldi iteration of GMRES, in Step 2 of Algorithm 2, the explicit elements of matrix A are not needed to be known; only the action of the matrix on a vector is required. Therefore, to avoid forming the Jacobian explicitly, a finite difference is used to approximate this matrix-vector product using

$$J(x)v \approx \frac{F(x + \varepsilon v) - F(x)}{\varepsilon}, \tag{72}$$

where ε is a small perturbation. The error in this approximation is proportional to ε. This is the basis for JFNK, whose full algorithm is shown in Algorithm 5.

Algorithm 5. JFNK

Newton's method

1. Select x_0 sufficiently close to a root

for $k = 0, 1, 2, \ldots$

Solve $J(x_k)\delta x_k = -F(x_k)$ *using GMRES*

2. $q_1 = r_0/\|r_0\|_2$, where $r_0 = -F(x_k) - J(x_k)\delta x_k \approx -F(x_k) - \frac{F(x_k+\varepsilon\delta x_k)-F(x_k)}{\varepsilon}$

for $j = 1, 2, 3, \ldots$

Arnoldi

3. $v = J(x_k)q_j \approx \frac{F(x_k+\varepsilon q_j)-F(x_k)}{\varepsilon}$

for $i = 1$ to j

4. $h_{i,j} = q_i^* v$
5. $v = v - h_{i,j}q_i$

Stop if $v = 0$

6. $h_{j+1,j} = \|v\|_2$
7. $q_{j+1} = v/h_{j+1,j}$
8. Solve least squares problem $\min_{y_j \in \mathcal{K}_j} \left\|\eta - \tilde{H}_j y_j\right\|_2$
9. $\delta x_j = \delta x_0 + Q_j y_j$

$\delta x_k = \delta x_j$

10. $x_{k+1} = x_k + \delta x_k$

While JFNK has the obvious advantage of applying the quadratically convergent Newton's method on a nonlinear system of equations without the need to form or store the Jacobian, it does have a drawback: it is only feasible on large scale problems with the use of an effective preconditioner [4]. Therefore the study of preconditioners will be a critical part of this work.

2.4 Feedback Models

Different forms of multiphysics feedback were tested in this work. These various feedback operators all change the cross sections of the problem, whether it is through changing the temperatures of the materials in the problem, or changing the material composition of the problem by altering the number densities of isotopes of interest. The following subsections discuss each of the multiphysics feedback operators that were tested in this work.

2.4.1 Thermal Hydraulic Feedback Model

Within MPACT an internal Simplified TH solver was incorporated [37]. This model is 'simplified' compared to CTF, which is the sub-channel TH code currently coupled to MPACT. CTF employs a two-fluid solution method over three different flow fields: fluid film, vapor, and liquid droplets [38]. While CTF delivers very detailed sub-channel results, the Simplified TH solver executes much faster, making it appropriate for certain applications. The node-based approach utilized by the Simplified TH solver is comparable to what many industry codes use today.

The Simplified TH model approximates thermal hydraulic feedback using 1D conservation of mass and energy. The mass flow rate through a given flow region is approximated by assuming uniform flow throughout the entire problem. By default, a flow region is a full assembly. Therefore, the mass flow rate in a given region, $\dot{m}_{reg}$, is given by

$$\dot{m}_{reg} = \frac{A_{reg}}{A_{total}} \dot{m}_{total} , \tag{73}$$

where A_{reg} is the cross sectional area of the flow region under consideration, A_{total} is the total cross sectional area of flow over the whole problem, and $\dot{m}_{total}$ is the total mass flow rate of the core.

Once the neutronics calculation has been performed, the resulting flux distribution is used to calculate the power deposited in each flow region. These flow region powers, P_{reg}, are then used to calculate the outlet flow region enthalpies for a given axial region using

$$h_{out} = h_{in} + \frac{P_{reg}}{\dot{m}_{reg}} , \tag{74}$$

where h_{out} and h_{in} are the outlet and inlet enthalpies, respectively, for that axial region. For the bottommost axial regions, the inlet enthalpy is calculated from the inlet coolant conditions. The outlet enthalpy for that flow region is then calculated using Equation 74, which is then used as

the inlet enthalpy for the above axial region. This process continues for all axial levels of the problem. Along the way, the average coolant temperatures and densities are calculated for each flow region. These values are then used to determine the fuel, gap, and cladding temperatures for each pin in the region. These temperatures are calculated using fuel temperature tables, which are described in more detail in the next section.

2.4.2 Fuel Temperature Tables

Accurately predicting the temperatures of the fuel is essential in reactor calculations since changes in the fuel temperatures lead to changes in the material cross sections. In order to capture this important phenomenon, tables can be generated that are used to calculate the fuel temperatures of a given problem. The fuel temperature tables in VERA were generated using the BISON fuel performance code [39]. BISON captures a number of important thermomechanical processes that impact the calculation of the fuel temperatures, such as fission gas release and the closure of the fuel-clad gap. A number of different fuel pins were simulated using BISON over a wide range of operating conditions. The results of these simulations were used to construct a table which could be used to lookup a fuel temperature, T_{fuel}, as a quadratic function of power and burnup using

$$T_{fuel} = T_{bulk} + \alpha(Bu)P + \beta(Bu)P^2 \,, \tag{75}$$

where T_{bulk} is the bulk coolant temperature, P is the local linear heat rate, and $\alpha(Bu)$ and $\beta(Bu)$ are both functions of burnup, Bu, that are obtained from the temperature table. The cladding temperature, T_{clad}, is set to a value between that of the fuel and coolant temperatures using

$$T_{clad} = f_{clad}T_{fuel} + (1 - f_{clad})T_{bulk} \,, \tag{76}$$

where f_{clad} is set to 0.2 by default. Meanwhile the temperature of the fuel-clad gap is approximated as the fuel temperature.

2.4.3 Xenon-135 Feedback Model

Xenon-135 is a fission product that builds up during reactor operation and has a large neutron absorption cross section. Therefore, the xenon concentration throughout the core has a significant impact on the result of the problem. Xenon-135 is produced both directly from fission and as a decay product of Iodine-135 via beta decay. The xenon decay chain is given by

$$^{135}\text{Te} \xrightarrow[11\ sec]{} {}^{135}\text{I} \xrightarrow[6.7\ hr]{} {}^{135}\text{Xe} \xrightarrow[9.2\ hr]{} {}^{135}\text{Cs} \xrightarrow[2.3\times10^{6}\ yr]{} {}^{135}\text{Ba(stable)}\ , \tag{77}$$

where the half-life of each decay is shown. Assuming the atomic number density of ^{135}I, N_I, changes only with additions from fission and losses from decay, the time rate of change of N_I is given by

$$\frac{dN_I}{dt} = \gamma_I \sum_{g=1}^{G} \Sigma_{f,g}\phi_g - \lambda_I N_I\ , \tag{78}$$

where γ_I is the effective fraction of fission products that are ^{135}I and λ_I is the beta decay constant for ^{135}I. Similarly, the time rate of change of the ^{135}Xe atomic number density, N_{Xe}, is given by

$$\frac{dN_{Xe}}{dt} = \lambda_I N_I + \gamma_X \sum_{g=1}^{G} \Sigma_{f,g}\phi_g - \lambda_X N_{Xe} - \sum_{g=1}^{G} \sigma_{a,g}^{X} N_{Xe}\phi_g\ , \tag{79}$$

where γ_X is the effective fraction of fission products that are ^{135}Xe and λ_X is the beta decay constant for ^{135}Xe. The first two terms of Equation 79 are the additions due to decay from ^{135}I and fission, while the last two terms are the losses due to β-decay and neutron absorption. If a reactor is operated with a constant neutron flux for an extended period of time, the number densities of these fission product poisons will eventually saturate to equilibrium values. Therefore, to calculate the equilibrium xenon concentration, X_{eq}, the left hand sides of Equations 78 and 79 are set to zero, and the coupled system of equations are combined to find

$$X_{eq} = \frac{(\gamma_I + \gamma_X)\sum_{g=1}^{G}\Sigma_{f,g}\phi_g}{\lambda_X + \sum_{g=1}^{G}\sigma_{a,g}^{X}\phi_g}. \quad (80)$$

2.4.4 Critical Boron Search Feedback Model

In Pressurized Water Reactors (PWR), boric acid is typically dissolved in the coolant to act as uniform excess reactivity control because boron has a large neutron absorption cross section. As the fuel depletes over time, the boron concentration in the coolant is reduced to compensate for the loss of reactivity. Therefore, in reactor analysis, it is often desired to calculate the soluble boron concentration that yields an eigenvalue of one. This operation is called a critical boron search. This process is relatively straight forward: if the dominant eigenvalue of the system is greater than 1.0 the soluble boron concentration is increased, while if the eigenvalue is less than 1.0 the boron concentration is reduced. This process repeats until the boron concentration converges and the eigenvalue is exactly one.

2.5 Summary

In this chapter, the background information necessary for understanding the implementation of both JFNK and CMFD-Coupling was discussed. First the high-order transport equation was derived along with all of the approximations and discretizations that were applied in order to simplify its solution. Similarly, the low-order CMFD system of equations was derived in detail. Additionally the numerical solvers used in this work were derived and discussed in detail. Specifically, all of the fundamental algorithms required in the JFNK solver were presented. Finally, the various forms of multiphysics feedback implemented in this work were discussed. These include thermal-hydraulic, equilibrium xenon, and critical boron search feedback models. The next chapter explains, in depth, both the JFNK and the CMFD-Coupling methodologies.

3. Methodology

The methodology for multiphysics coupling methods considered in this work will now be described. First, the current multiphysics coupling method based on the Picard iteration will be discussed. Next, the two multiphysics coupling methods implemented in this research are described, including the JFNK coupling method and the CMFD-Coupling method. For JFNK, the differences between the eigenvalue implementation and the multiphysics coupling implementation are pointed out. Additionally, the preconditioners used to accelerate JFNK are discussed.

3.1 Current Multiphysics Coupling

The standard multiphysics coupling methodology used in the industry today employs a Picard, or fixed-point, iteration scheme. First, the individual physics are solved independently of the neutronics solver. With the feedback effects from these solutions now captured, the macroscopic cross sections are updated. Next the eigenvalue problem is solved using CMFD.

The standard method for solving the k-eigenvalue problem is the power iteration scheme outlined in Section 2.3.1. First the transport cross sections and transport fluxes are reduced to a low order CMFD system using the equations defined in Section 2.2.2. Then the CMFD system is solved using power iterations for updated flux and eigenvalue estimations. The second step of the power method process outlined in Algorithm 1 is solved using the GMRES iterative solver. This yields updated fluxes which are then used to calculate an updated eigenvalue. This process continues until the relative difference in successive eigenvalues is below a predefined tolerance. Upon convergence the coarse mesh fluxes are then projected back onto the fine transport mesh using Equation 44.

With updated approximations of the transport fluxes and the eigenvalue from CMFD as a starting point, the neutron transport problem is solved. Once a transport solution is obtained, the fission source distribution is compared to the previous solution to determine convergence. If the 2-norm of the difference between successive fission source distributions is less than a defined

convergence criteria, typically $5x10^{-5}$, the fluxes are considered converged. Similarly, the eigenvalue calculated from the CMFD solve is compared to that from the previous iteration. If the eigenvalue difference is less than a defined convergence criteria, typically $1x10^{-6}$, the eigenvalue is considered converged. Once the fission source distribution and the eigenvalue are converged, the problem stops. Otherwise the solver loops back and solves the coupled set of physics again and the whole process repeats. A flowchart of this current coupling methodology is shown in Figure 2.

3.2 JFNK Implementation

JFNK was implemented in MPACT using the Portable Extensible Toolkit for Scientific Computation (PETSc) [40]. All PETSc routines support the Message Passing Interface (MPI) standard for message-passing communication. MPI is used to distribute matrices and vectors across multiple processors in order to utilize parallel computing.

3.2.1 JFNK Convergence Criteria within PETSc

Since the linear system being solved by GMRES within JFNK is of the form $Ax = b$ given by Equation 75, the residual for the k-th iteration is given by

$$r_k = -F(x_k) - J(x_k)\delta x_k \,. \tag{81}$$

Within PETSc, the default convergence criterion of the linear GMRES solver is determined by a decrease of the residual norm relative to the right hand side:

$$\|r_k\|_2 < 10^{-5} \times \|F(x_k)\|_2 \,. \tag{82}$$

Similarly, the default convergence criterion for the nonlinear Newton's method within JFNK is determined by

$$\|F(x_k)\|_2 < 10^{-8} \times \|F(x_0)\|_2 \,, \tag{83}$$

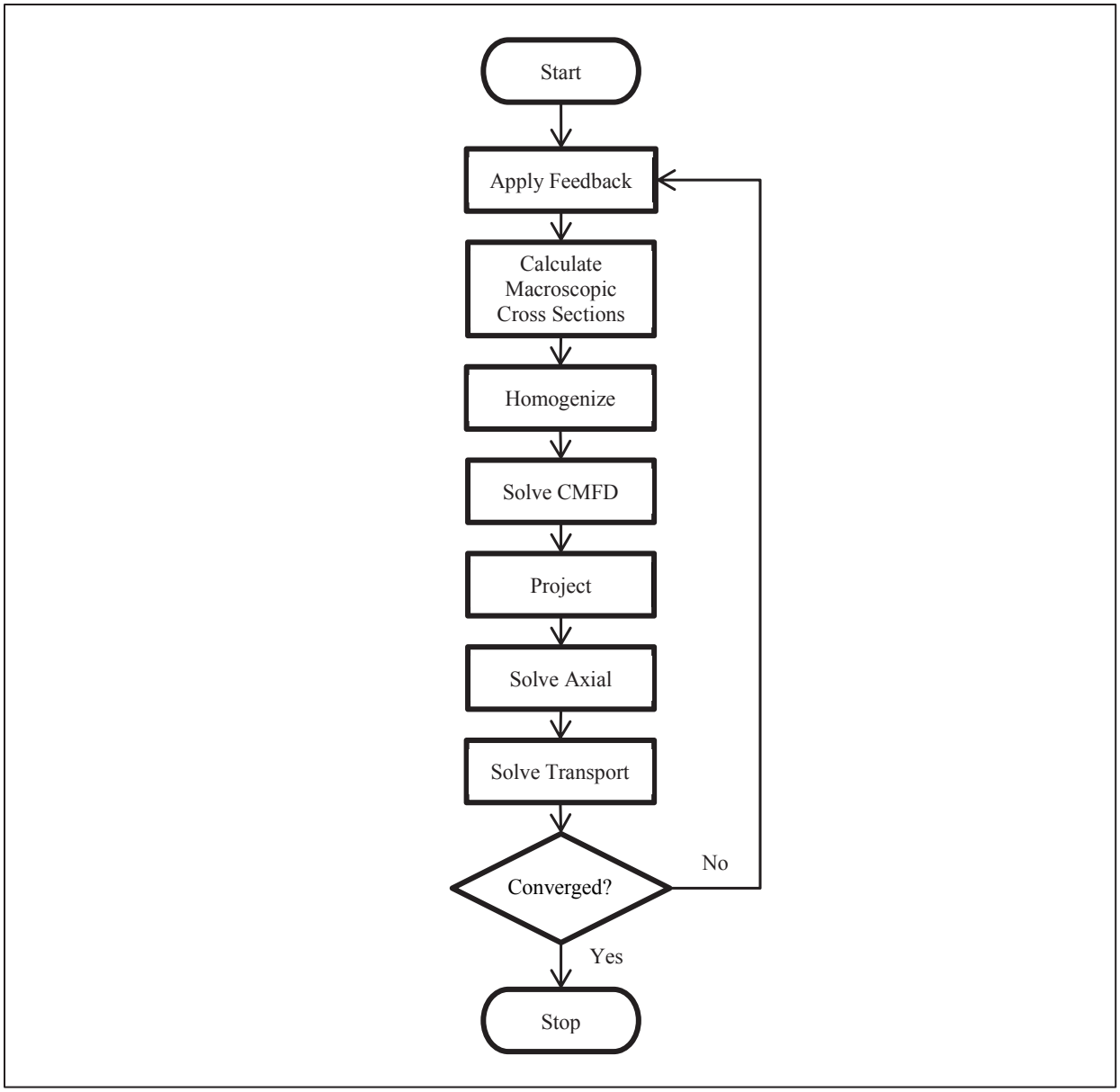

Figure 2. Flowchart of the Picard coupling technique

where x_0 is the initial guess of the solution vector.

3.2.2 JFNK Preconditioning

In order to accelerate the convergence of JFNK, preconditioners are often used. Preconditioners speed up the JFNK method by reducing the number of linear iterations needed to reach convergence. A preconditioner is a matrix that, when properly chosen, efficiently clusters the eigenvalues of the system. For any nonsingular system given by $Ax = b$, the system

$$P^{-1}Ax = P^{-1}b \tag{84}$$

has the same solution given a nonsingular preconditioner matrix P. Applying the preconditioner in this fashion is called left preconditioning. If $P = A$, applying the preconditioner is as difficult as solving the original system, while if $P = I$, applying the preconditioner is trivial but does not do anything. Therefore, useful preconditioners lie somewhere between these two extremes, where P is similar to A but structured so that it is easily invertible. Similarly, right preconditioning can be used to transform the nonsingular system $Ax = b$ into

$$AP^{-1}y = b \tag{85}$$

where

$$x = P^{-1}y \tag{86}$$

While JFNK can use either right or left preconditioning, right preconditioning is most often used because left preconditioning alters the norm of the residual, which is how convergence of the linear solver is measured [4]. Therefore, only right preconditioning was examined in this work. As a result, the preconditioned linear system solved by GMRES within JFNK is given by

$$J(x_k)P^{-1}P\delta x_k = -F(x_k)\,. \tag{87}$$

This is done using a two-step process by first solving

$$J(x_k)P^{-1}y = -F(x_k)\,. \tag{88}$$

for y and then solving

$$\delta x_k = P^{-1}y\,. \tag{89}$$

for δx_k. Using right preconditioning, the Jacobian-matrix approximation from Equation 72 becomes

$$J(x)P^{-1}v \approx \frac{F(x + \varepsilon P^{-1}v) - F(x)}{\varepsilon}\,. \tag{90}$$

While the usefulness of JFNK lies in its ability to not need an explicitly formed Jacobian, effective preconditioners typically require some knowledge of the Jacobian. However, the preconditioner can use a much simpler version of the Jacobian and can use approximations to make its formulation easier.

3.2.3 JFNK Eigenvalue Implementation

In addition to being used as a multiphysics coupling technique, JFNK can also be used as a method for solving the generalized eigenvalue problem given in Equation 51. Even though the eigenvalue problem is a linear problem, the nonlinear JFNK algorithm in Section 2.3.6 can be used to solve it. JFNK was first implemented as an eigenvalue solver without feedback enabled. This allowed for an easier initial implementation, as well as allowing for error checking against existing eigenvalue solvers already implemented.

In order to replace the power iteration scheme described in Section 3.1 with a JFNK eigenvalue solver, Equation 51 must first be rewritten in residual form

$$F(\boldsymbol{\phi}) = 0 = \mathbb{M}\boldsymbol{\phi} - \frac{1}{k_{\text{eff}}}\mathbb{F}\boldsymbol{\phi}\,, \tag{91}$$

where $F(\boldsymbol{\phi})$ is referred to as the residual equation. With Equation 91 in hand, the JFNK algorithm outlined in Section 2.3.6 can be used as an eigenvalue solver. The flowchart for the JFNK eigenvalue solver is the exact same as that shown in Figure 2, except now the 'Solve CMFD' block uses JFNK rather than the default power method.

In addition, a preconditioner was investigated in order to determine its effects on convergence. Looking at Equation 91, the exact Jacobian would be given by

$$J(\boldsymbol{\phi}) = \mathbb{M} - \frac{1}{k_{\text{eff}}}\mathbb{F}\,. \tag{92}$$

However, since the migration matrix, $\mathbb{M}$, is sparse and easily invertible, the fission matrix, $\mathbb{F}$, term was neglected. Therefore, the preconditioner was simply chosen to be

$$P = \mathbb{M} \tag{93}$$

for all JFNK eigenvalue implementations.

3.2.4 JFNK Coupled Multiphysics Implementation

The JFNK eigenvalue solver described in the previous section can be extended to solving coupled problems. When coupling to the Simplified TH solver, solution vector, $\boldsymbol{x}$, now contains the fuel temperatures, T, in addition to the fluxes,

$$\boldsymbol{x} = \begin{bmatrix} \phi_1 \\ \vdots \\ \phi_{NF} \\ T_1 \\ \vdots \\ T_{NT} \end{bmatrix}, \tag{94}$$

where the NF subscript refers to the total number of coarse mesh fluxes in the problem, and NT refers to the total number of fuel temperatures being solved for. With fluxes that are properly normalized, the eigenvalue can be calculated by summing the fission source over all regions, I, given by

$$k_{\text{eff}}(\boldsymbol{\phi},T)=\sum \mathbb{F}\boldsymbol{\phi}=\sum_{i=1}^{I}\sum_{g=1}^{G}V^{i}\chi_{g}^{i}(T)\sum_{g'=1}^{G}\nu\Sigma_{f,g'}^{i}(T)\phi_{g'}^{i}, \tag{95}$$

where V^i is the volume of region i. In addition, the residual calculation must now account for the fuel temperature and eigenvalue residuals as well as the coarse mesh flux residuals, leading to the nonlinear system of equations given by

$$F=\begin{pmatrix} f_{\phi}(\boldsymbol{\phi},T,k_{\text{eff}})=\mathbb{M}\boldsymbol{\phi}-\dfrac{1}{k_{\text{eff}}}\mathbb{F}\boldsymbol{\phi} \\ f_{T}(T,\boldsymbol{\phi})=T_{bulk}+\alpha(Bu)P+\beta(Bu)P^{2}-T_{fuel} \\ f_{k}(\boldsymbol{\phi},T,k_{\text{eff}})=k_{\text{eff}}-\sum \mathbb{F}\boldsymbol{\phi} \end{pmatrix}. \tag{96}$$

The third equation from Equation 96, $f_k(\boldsymbol{\phi},T,k_{\text{eff}})$, can be eliminated by substituting the function evaluation of k_{eff} from Equation 95 into the first equation:

$$F=\begin{pmatrix} f_{\phi}(\boldsymbol{\phi},T,k_{\text{eff}})=\mathbb{M}\boldsymbol{\phi}-\dfrac{1}{k_{\text{eff}}(\boldsymbol{\phi},T)}\mathbb{F}\boldsymbol{\phi} \\ f_{T}(T,\boldsymbol{\phi})=T_{bulk}+\alpha(Bu)P+\beta(Bu)P^{2}-T_{fuel} \end{pmatrix}. \tag{97}$$

It should be noted that as JFNK is solving this coupled system, the fuel temperatures are updated with every linear GMRES iteration. Therefore the cross sections need to be updated as well in order to capture this temperature feedback on the fluxes. A flowchart of this JFNK coupling scheme is shown in Figure 3.

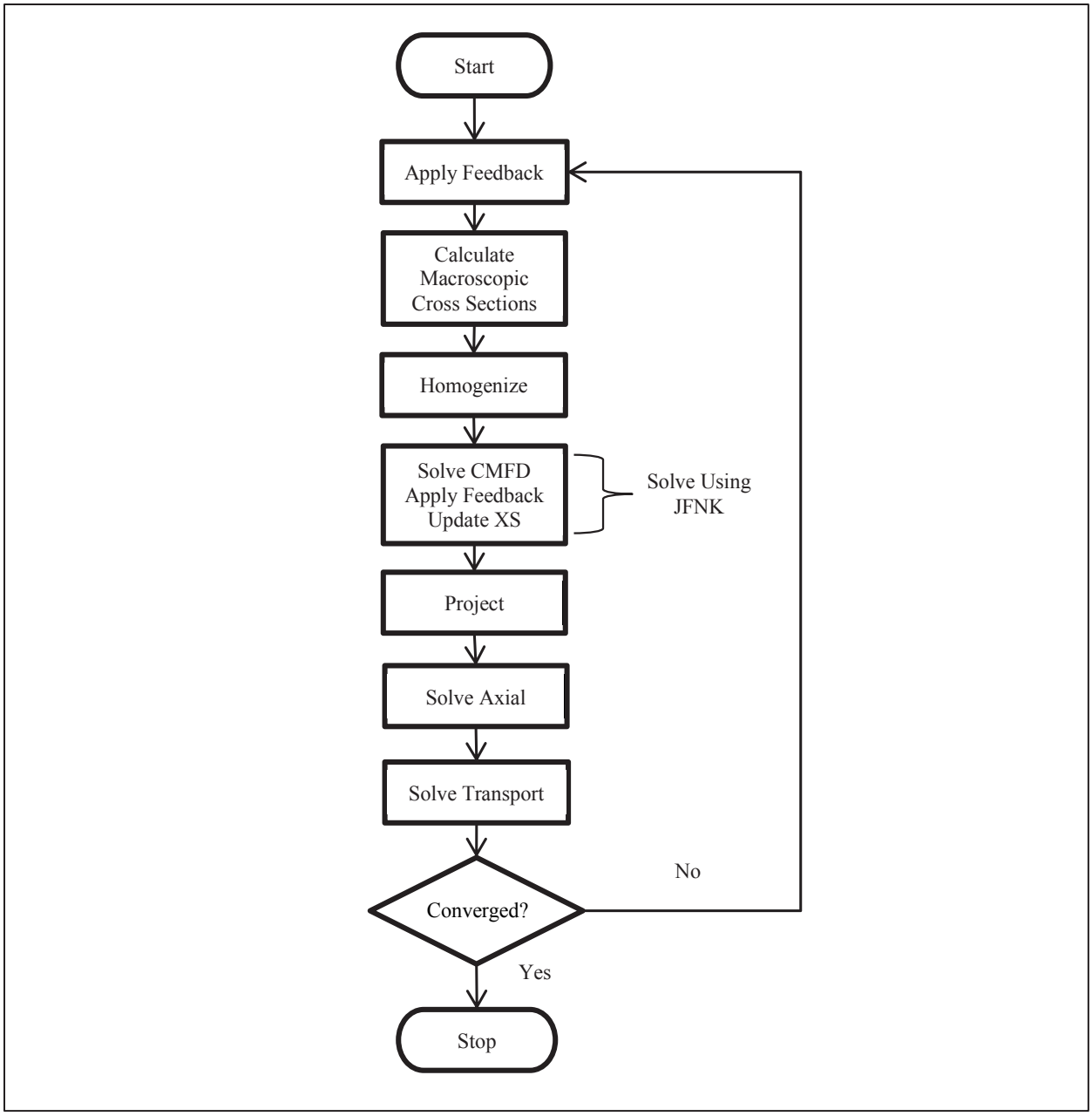

Figure 3. Flowchart of the JFNK coupling technique

Because the coupled JFNK implementation incorporates TH feedback, Equation 75 is now represented in a block-matrix notation given by

$$\begin{bmatrix} J_{\phi,\phi} & J_{\phi,T} \\ J_{T,\phi} & J_{T,T} \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{\phi} \\ \delta\boldsymbol{T} \end{bmatrix} = \begin{bmatrix} F_{\phi}(\boldsymbol{\phi},\boldsymbol{T}) \\ F_{T}(\boldsymbol{\phi},\boldsymbol{T}) \end{bmatrix}. \quad 98$$

Therefore, when approximating the block-Jacobian to act as a preconditioner, the upper left portion is chosen to be the migration matrix, $\mathbb{M}$, as was previously discussed in Section 3.2.3. The lower right block of the Jacobian is simply the identity matrix, since the fuel temperature of a given region does not depend on the fuel temperatures of any other region. As discussed in Section 2.4.2, the fuel temperatures are calculated using a pre-calculated fuel temperature table. Since the local linear heat rate, P, is a function of the flux solution, the lower left block of the Jacobian is obtained by taking the derivative of Equation 75.

The upper right block of the Jacobian was never fully implemented into the preconditioner. While it may appear to be incomplete, the purpose of the preconditioner is to reduce the number of linear GMRES iterations required for convergence, and has no impact on the total number of nonlinear iterations or transport sweeps. Therefore, two different preconditioners were implemented in the coupled JFNK application: a Diagonal Preconditioner that contains only the diagonal $J_{\phi,\phi}$ and $J_{T,T}$ terms, and a Lower Triangular Preconditioner that includes the $J_{T,\phi}$ term.

This concludes the derivation for the JFNK approach. The next section will explain in depth the details of the CMFD-Coupling method.

3.3 CMFD-Coupling Implementation

As an alternative to the coupled JFNK implementation, a new technique called CMFD-Coupling was employed. The idea behind this alternative method is to solve the low-order CMFD eigenvalue system using the power method, but then apply the coupled physics feedback again before solving the MOC transport problem. The feedback effects of the coupled physics system are captured since the cross sections are updated within this loop. This process can be iterated

multiple times before a transport sweep is performed. A flowchart of this CMFD-Coupling scheme is shown in Figure 4.

Two strategies were implemented to determine when MOC should be performed again. The first is a simple counter: the CMFD-Coupling loop is executed a set number of times before the transport solver. This method will be referred to as the CMFD-N implementation, where N is the number of CMFD-Coupling iterations completed before performing a full transport solve. The second method has the CMFD-Coupling loop iterate until the maximum difference in temperature between two successive iterations is below a certain threshold. This method will be referenced as $\Delta T < tol$, where tol is the tolerance under which the maximum temperature difference between iterations must fall. Typical values for tol ranged from 0.1 K for loose convergence and 0.001 K for very tight convergence.

The default eigenvalue convergence criteria of $1\text{x}10^{-6}$ and the default fission source convergence criteria of $5\text{x}10^{-5}$ are kept the same. However, the differences are not taken between successive CMFD-Coupling iterations, but instead are taken between successive MOC iterations.

3.4 Summary

In this chapter, the different methodologies explored in this work were described in depth. First the current multiphysics coupling strategy, the Picard iteration, was discussed. Next the JFNK method was explored. The specifics of how PETSc implements JFNK were discussed before the details of the preconditioners used were examined. JFNK was implemented as both an eigenvalue solver as well as a coupled multiphysics solver. The details of both of these methods were laid out. Finally, the CMFD-Coupling method was discussed. The following chapter explores the application of all of these methods on a series of smaller problems.

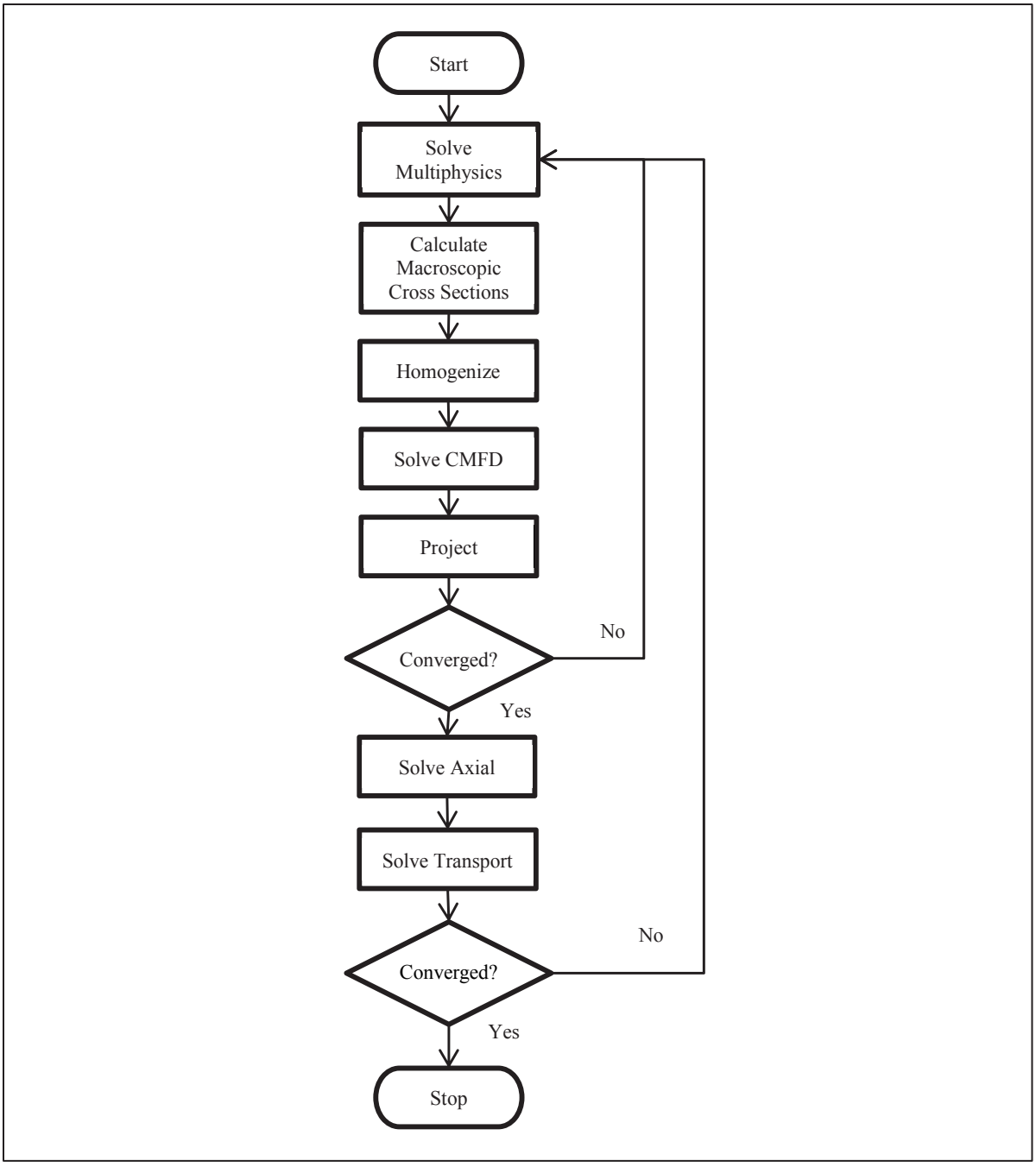

Figure 4. Flowchart of the CMFD-Coupling technique

4. JFNK Investigation

As a proof of concept, two simplified reactor problems were developed: a multigroup infinite homogeneous medium problem, and a one-group, one-dimensional homogeneous slab problem. The one-dimensional slab problem looks at the potential impact of implementing a JFNK nonlinear solver on a problem with spatial dependencies while the infinite homogeneous problem examines the impact on a problem with an energy dependence. With information gained from these simplified problems, a 2D pin cell, 2D fuel lattice, a 3D fuel pin, and a 3D 7x7 fuel assembly were investigated. All of these problems model Pressurized Water Reactor (PWR) fuel.

4.1 Infinite Homogeneous Medium

The benefit of testing a method on an infinite homogeneous medium problem is that the problem has no spatial dependence. Instead, an energy dependence was incorporated through the use of a multigroup framework. Since the problem is thought of as an infinite material with constant properties, the cross sections are constant throughout the problem. With no spatial dependence of the problem, the neutron continuity equation in Equation 17 reduces to

$$\Sigma_{t,g}(T)\phi_g = \sum_{g'=1}^{G} \Sigma_{s,g'\to g}(T)\,\phi_{g'} + \frac{\chi_g(T)}{k_{\text{eff}}} \sum_{g'=1}^{G} \nu\Sigma_{f,g'}(T)\phi_{g'}\,. \qquad 99$$

Because there is no spatial dependence, MOC-CMFD is not applicable and is replaced with a cross section table lookup. The cross sections were generated from 2D pin cell calculations at varying fuel temperatures from 565 K to 1500 K in five degree increments. These cross sections are then used in a table look up procedure. A linear interpolation was used to determine the cross section between data points. The temperature feedback used in this problem was chosen using the solution of one of the 2D pin cell problems such that the problem would converge to a predetermined solution. Arbitrarily choosing the 1365 K case and using its eigenvalue leads to the TH feedback used:

$$T = 565K + 688.534K \times k_{\text{eff}} \,. \tag{100}$$

While not a physically correct relationship, Equation 100 ensures that as the eigenvalue increases or decreases, the temperature follows suit. It also has the added benefit of knowing what the solution should converge to upon completion, allowing for error checking. The multigroup equation for calculating k_{eff} is found from

$$k_{\text{eff}}(\boldsymbol{\phi}, T) = \sum_{g=1}^{G} \nu\Sigma_{f,g}(T)\phi_g \,. \tag{101}$$

Equations 99, 100, and 101 form a nonlinear system of equations that can be solved using the JFNK method outlined in Algorithm 5:

$$F = \begin{pmatrix} f_\phi(\boldsymbol{\phi}, T, k_{\text{eff}}) = \Sigma_{t,g}(T)\phi_g - \sum_{g'=1}^{G} \Sigma_{s,g'\to g}(T)\,\phi_{g'} + \dfrac{\chi_g(T)}{k_{\text{eff}}} \sum_{g'=1}^{G} \nu\Sigma_{f,g'}(T)\phi_{g'} \\ f_T(T, k_{\text{eff}}) = T - 565K - 688.534K * k_{\text{eff}} \\ f_k(\boldsymbol{\phi}, T, k_{\text{eff}}) = k_{\text{eff}} - \sum_{g=1}^{G} \nu\Sigma_{f,g}(T)\phi_g \end{pmatrix}. \tag{102}$$

The third equation, $f_k(\boldsymbol{\phi}, T, k_{\text{eff}})$, can be eliminated from Equation 102 by substituting the functional evaluation of k_{eff} from Equation 101 into the first two equations:

$$F = \begin{pmatrix} f_\phi(\boldsymbol{\phi}, T) = \Sigma_{t,g}(T)\phi_g - \sum_{g'=1}^{G} \Sigma_{s,g'\to g}(T)\,\phi_{g'} + \dfrac{\chi_g(T)}{k_{\text{eff}}(\boldsymbol{\phi}, T)} \sum_{g'=1}^{G} \nu\Sigma_{f,g'}(T)\phi_{g'} \\ f_T(T) = T - 565K - 688.534K * k_{\text{eff}}(\phi, T) \end{pmatrix}. \tag{103}$$

Therefore, the solution vector of these equations is of the form

$$\boldsymbol{x} = \begin{bmatrix} \phi_1 \\ \vdots \\ \phi_G \\ T \end{bmatrix}. \tag{104}$$

The iteration scheme for solving this infinite homogeneous medium problem is shown in Algorithm 6.

Algorithm 6. Infinite Homogeneous Medium Iterations

1. Select appropriate $k_{\text{eff}}^{(0)}$, $\phi_g^{(0)}$, $T^{(0)}$
   for $n = 1, 2, \ldots$
2. Use table to look up cross sections at $T^{(n)}$
3. Solve Equation 103 using JFNK for updated $\phi_g^{(n+1)}$ and $T^{(n+1)}$
4. $k_{\text{eff}} = \sum_{g=1}^{G} \nu\Sigma_{f,g}(T^{(n+1)})\phi_g^{(n+1)}$

Steps 2 and 3 are repeated until the difference between sequential eigenvalues is sufficiently small. If the process of updating the cross sections through the table lookup is thought of as a surrogate for MOC, this iteration scheme is similar to the JFNK coupled multiphysics implementation described in Section 3.2.4.

It is possible to accelerate the convergence of the temperature and multigroup fluxes if some information about the cross section dependence on temperature is used in the JFNK iterations. If Algorithm 6 is simply implemented as written, while Newton's method within JFNK is iterating towards a solution, the cross sections are held constant. Even though the temperature is changing, the cross sections are not updated until after the JFNK solution has converged. This is analogous to the Picard iteration. In order to give Newton's method the ability to update the cross sections, the cross section derivative with respect to temperature, $d\Sigma_x/dT$, is calculated in Step 2 of Algorithm 6 at the current temperature using a forward finite difference approximation. These derivatives are then passed into JFNK and are used to linearly extrapolate the cross sections as the temperature is converging. The comparison of the convergence of these two methods is shown in Figure 5.

Figure 5. Convergence of a table lookup (Algorithm 6) vs the addition of a linear update

As seen in Figure 5, the inclusion of a linear update of the cross sections within the Newton iteration offers significant improvement. The linear update curve appears to have quadratic convergence while the table lookup only curve is linear. While this speedup appears to come without much effort, one must think of larger full-scale reactor problems. In this simple problem, there is only one material and therefore one temperature. Using this method on a full-core problem would be prohibitively expensive because one would have to store each cross section derivative for every flat source region in the problem. Therefore, an analysis was performed to determine how to achieve the most acceleration without the large memory requirements.

First, to determine which of the cross sections had the largest impact on speedup, each was linearly updated individually, while the others were only updated outside of the JFNK iteration. The results from this test are given in Figure 6. The curves from Figure 5 were included in Figure 6 because they act as bounding limits for this study. It is clear from Figure 6 that updating χ_g, $\nu\Sigma_{f,g}$, and $\Sigma_{s,g}$ has little, if any, effect on the convergence rate. Updating $\Sigma_{a,g}$ however, has a very significant impact on the problem convergence. Therefore, only the absorption cross section update will be considered for the remainder of this section.

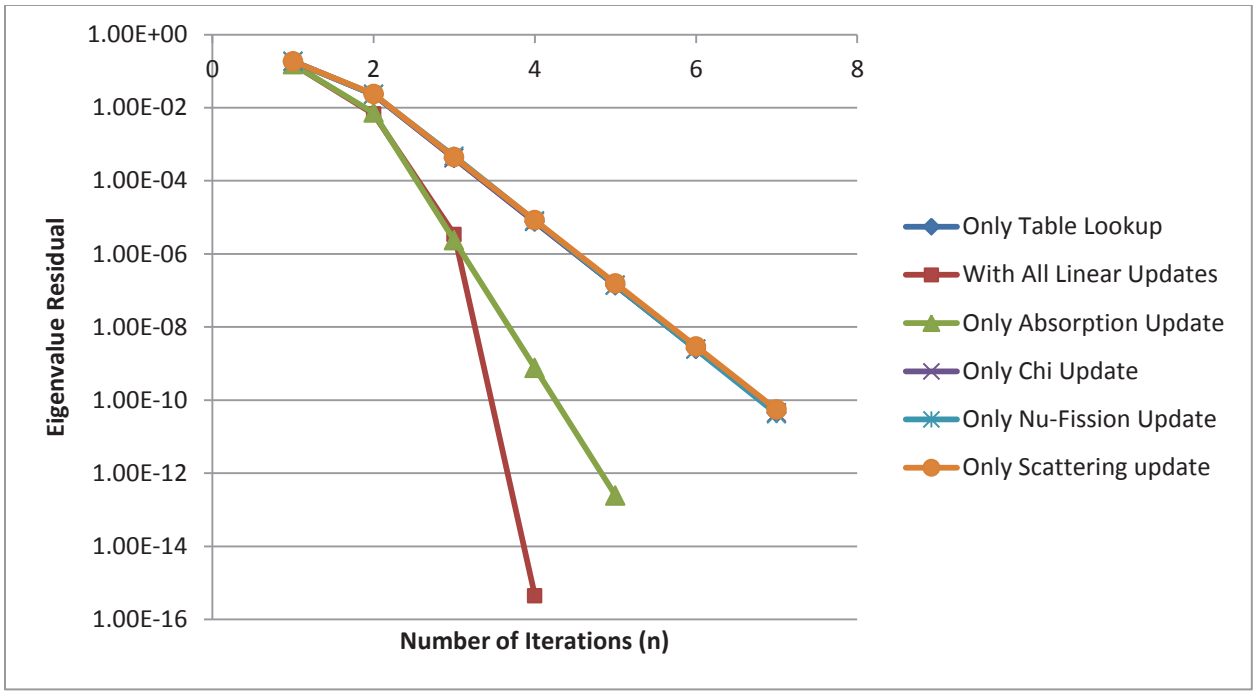

Figure 6. Convergence plots for each cross section being linearly updated individually

While removing the other linear cross section updates does relieve some of the memory burden, having to determine the absorption cross section derivative for every region in the core may still be too expensive. Therefore, a study was performed to determine how exact the absorption cross section derivatives need to be and what approximations could be made. The first approximation made removes the temperature dependence of the derivative and uses an average value per group instead. This was done by calculating $d\Sigma_{a,g}/dT$ for all temperatures for each group. An average was then calculated by summing all of the derivatives for a given group and then dividing by the number of derivatives summed. This approximation is referred to as the ***Average Groupwise Derivative Approximation***. The next approximation tested looked at removing the group dependence and only using a one-group temperature dependent derivative. The first step was to collapse the multigroup cross sections to a one-group cross section using Equation 39. Then $d\Sigma_a/dT$ was calculated using a forward finite difference for a given temperature. This approximation is referred to as the ***One-Group Derivative Approximation***. The final simplification examined involves combining the two previous approximations to remove both the temperature and group dependencies of the derivative. Like the One-Group Derivative approximation, the cross sections are initially collapsed to one-group. Then like the Average Groupwise Derivative approximation, these one-group cross sections are used to calculate the

derivatives using a forward finite difference, which are then averaged together. Therefore, $d\Sigma_a/dT$ becomes simply a constant value. This method is called the ***Average One-Group Derivative Approximation***. Each of these different methods were implemented independently and their convergence plots are shown in Figure 7.

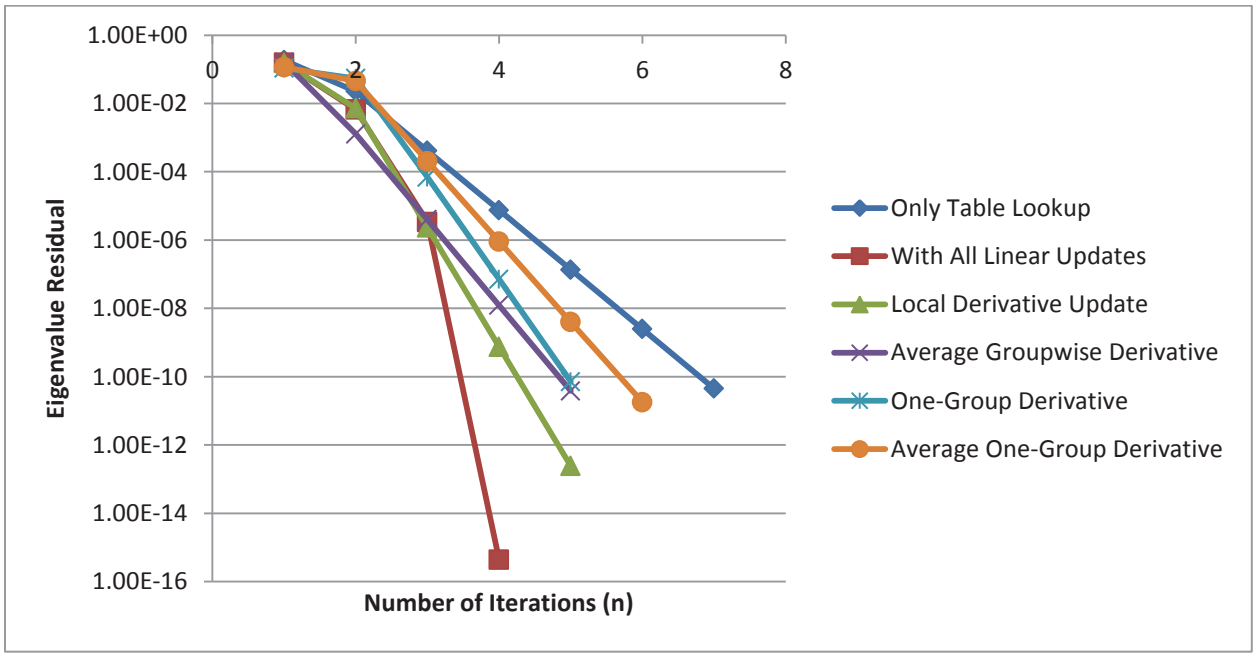

Figure 7. Convergence plots for different absorption cross section derivative approximations

As seen in Figure 7, updating the absorption cross sections fully, shown as the ***Local Derivative Update***, performed the best while the constant Average One-Group Derivative method performed the worst. However all possible approximations are improvements upon the method that only updates the cross sections after GMRES has converged. Therefore, depending on computational requirements, more than one of these approximations might be appropriate on largescale calculations.

4.2 One-Dimensional, One-Group Homogeneous Slab

The one-dimensional (1D), one-group homogeneous slab problem is a common problem in reactor analysis from which important physics can be gleaned. This simplified problem allowed for methods to be developed and tested without the added complexity and computational burden of a multidimensional heterogeneous problem with multiple energy groups. The slab is finite in

the x dimension but is infinite in the y and z dimensions, thus removing their dependencies. The boundary conditions for this problem are those of a vacuum, meaning that there is no incident neutron flux on the edges of the problem. A depiction of this 1D slab problem is shown in Figure 8. The first step in solving the 1D slab problem is to simplify the multigroup MOC and CMFD equations to their 1D, one-group form.

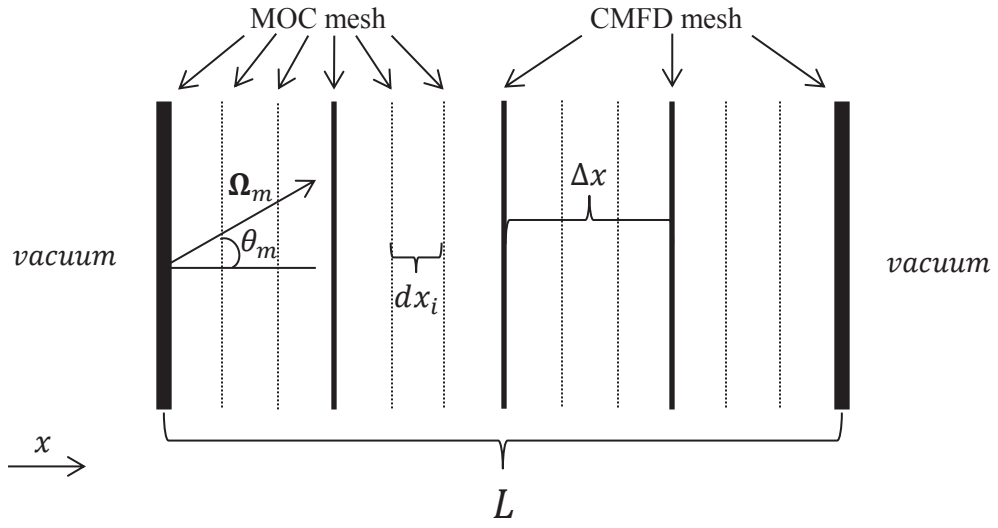

Figure 8. 1D homogeneous slab problem showing MOC and CMFD meshes

4.2.1 Simplified MOC Equations

As outlined in Section 2.2.1, the transport equation in this work is solved using MOC. Since MOC already solves the transport equation along a 1D characteristic, the form of Equation 32 does not change apart from losing the group subscript:

$$\psi_{i,m,k}(s) = \psi_{i,m,k}^{in} e^{-\Sigma_t s} + \frac{Q_i}{\Sigma_t}\left[1 - e^{-\Sigma_t s}\right]. \tag{105}$$

It should be noted that since the slab is homogeneous, the cross sections do not vary as a function of flat source regions, i, and as a result the cross sections are now constant values. Likewise, the definition of the region averaged source in Equation 33 becomes

$$Q_i = \frac{1}{2}\left[\Sigma_s \phi_i + \frac{1}{k_{\text{eff}}} \nu\Sigma_f \phi_i\right] \quad 106$$

in 1D. In a similar fashion, the formula for the segment averaged angular flux in Equation 36 becomes

$$\tilde{\psi}_{i,m,k} = \frac{1}{s_{i,k}}\left[\frac{\psi_{i,m,k}^{in} - \psi_{i,m,k}^{out}}{\Sigma_t} + \frac{Q_i s_{i,k}}{\Sigma_t}\right], \quad 107$$

while the region averaged angular flux from Equation 35 is now defined as

$$\bar{\psi}_{i,m} = \tilde{\psi}_{i,m,k}\,, \quad 108$$

since there is only one segment averaged angular flux per angle per region in 1D. The formula for the scalar flux in Equation 34 also loses the group subscript and becomes

$$\phi_i = \int_{4\pi} \bar{\psi}_i(\mathbf{\Omega}) d\widehat{\Omega} \approx \sum_{m=1}^{M} w_m \bar{\psi}_{i,m}\,. \quad 109$$

To solve these 1D MOC equations for the flux and k_{eff}, a fixed-point power iteration is traditionally used. Note that the equation for the region averaged flat source, Q_i, depends on the scalar flux, ϕ_i, which in turn depends on the source Q_i. In order to break this recursive relationship, initial guesses are made for both the scalar flux and k, and are used to create an initial guess for the source,

$$Q_i{}^{(0)} = \frac{1}{2}\left[\Sigma_s \phi_i{}^{(0)} + \frac{1}{k^{(0)}} \nu\Sigma_f \phi_i{}^{(0)}\right], \quad 110$$

where the superscript indicates an iteration count. With an initial guess of the source, the outgoing angular fluxes are calculated using Equation 105 for each MOC region. Once the angular fluxes are calculated for each region at a given angle in the forward direction, the

process is repeated again in the reverse direction. With the incoming and outgoing angular fluxes known for each interface, the segment averaged angular flux is calculated for each segment using Equation 107 along with the initial source guess. Then the region averaged angular flux and the updated scalar flux are calculated using Equation 108 and Equation 109, respectively. Once the updated scalar fluxes are known for each discrete region, an updated k can be calculated from

$$k^{(n+1)} = \frac{\int \nu\Sigma_f {\phi_i}^{(n+1)} dV}{\frac{1}{k^{(n)}} \int \nu\Sigma_f {\phi_i}^{(n)} dV} = \frac{\sum_i \nu\Sigma_f {\phi_i}^{(n+1)} dx_i}{\frac{1}{k^{(n)}} \sum_i \nu\Sigma_f {\phi_i}^{(n)} dx_i}, \tag{111}$$

where dx_i is the thickness of region i. For this simplified problem, the mesh is uniform so the i subscript can be dropped. In the CMFD equations that are used to accelerate this MOC solution, the current, $\boldsymbol{J}$, is required. Therefore, the MOC solution is used to calculate the current at each boundary interface using a combination of Equation 18 and Equation 13:

$$J_{i+\frac{1}{2}} = \sum_{m=1}^{M} w_m \cos(\theta_m) \psi_{i,m,k}^{out}. \tag{112}$$

However, this is only the current in the forward direction. To get the total current this quantity must be added to that computed in the reverse direction. These equations can now be used to form an iterative algorithm to solve for the scalar flux and eigenvalue. This process is commonly referred to as outer or source iterations, and is outlined step by step in Algorithm 7. However, as discussed in Section 2.1, performing Algorithm 7 by itself is extremely inefficient. Therefore CMFD was also implemented in the 1D one group slab problem.

4.2.2 Simplified CMFD Equations

The CMFD equations are also greatly simplified in 1D. The 3D CMFD neutron balance equation in Equation 49 becomes

$$\left(\boldsymbol{J}_{i+\frac{1}{2}} - \boldsymbol{J}_{i-\frac{1}{2}}\right) + \Delta x(\Sigma_t - \Sigma_s)\phi_i = \frac{\Delta x}{k_{eff}} \nu\Sigma_f \phi_i, \tag{113}$$

Algorithm 7. MOC Outer Iteration

1. Select appropriate $k^{(0)}$ and $\phi_i{}^{(0)}$
   Do until converged
2. $\phi_i = 0$
3. $\boldsymbol{J}_{i+\frac{1}{2}} = 0$
   for $m = 1, 2, \dots, M$ *(angle)*
4. $s_m = \frac{dx}{\cos(\theta_m)}$
   Forward Sweep
5. $\psi_{1,m,k}^{in} = 0$ *(vacuum BCs)*
   for $i = 1, 2, \dots, I$ *(region)*
6. $Q_i{}^{(n)} = \frac{1}{2}\left[\Sigma_s \phi_i{}^{(n)} + \frac{1}{k^{(n)}} \nu\Sigma_f \phi_i{}^{(n)}\right]$
7. $\psi_{i,m,k}^{out} = \psi_{i,m,k}^{in} e^{-\Sigma_t s_m} + \frac{Q_i}{\Sigma_t}\left[1 - e^{-\Sigma_t s_m}\right]$
8. $\tilde{\psi}_{i,m,k} = \frac{1}{s_m}\left[\frac{\psi_{i,m,k}^{in} - \psi_{i,m,k}^{out}}{\Sigma_t} + \frac{Q_i s_m}{\Sigma_t}\right]$
9. $\bar{\psi}_{i,m} = \tilde{\psi}_{i,m,k}$
10. $\phi_i = \phi_i + w_m \bar{\psi}_{i,m}$
11. $\psi_{i+1,m,k}^{in} = \psi_{i,m,k}^{out}$
12. $\boldsymbol{J}_{i+\frac{1}{2}} = \boldsymbol{J}_{i+\frac{1}{2}} + w_m \cos(\theta_m) \psi_{i,m,k}^{out}$
    Backward Sweep
13. $\psi_{I,m,k}^{in} = 0$ *(vacuum BCs)*
    for $i = I, I-1, \dots, 1$ *(region)*
    Repeat Steps 6-11
14. $\boldsymbol{J}_{i-\frac{1}{2}} = \boldsymbol{J}_{i-\frac{1}{2}} - w_m \cos(\theta_m) \psi_{i,m,k}^{out}$
15. $k^{(n+1)} = \frac{\sum_i \nu\Sigma_f \phi_i{}^{(n+1)} dx}{\frac{1}{k^{(n)}} \sum_i \nu\Sigma_f \phi_i{}^{(n)} dx}$

where the region superscripts, i, have been changed to subscripts to keep consistent with the MOC equations. The definition of the currents now comes from a simplified version of Equation 47:

$$\boldsymbol{J}_{i\pm\frac{1}{2}} = -\widetilde{D}_{i\pm\frac{1}{2}}(\pm\phi_{i\pm1} \mp \phi_i) + \widehat{D}_{i\pm\frac{1}{2}}(\phi_{i\pm1} + \phi_i)\,. \tag{114}$$

The equation for the nonlinear diffusion coefficient correction factor, $\widehat{D}$, is found by rearranging Equation 114, while the linear coupling term, $\widetilde{D}$, is now given by

$$\widetilde{D}_{i\pm\frac{1}{2}} = \frac{D_i}{\Delta x}, \tag{115}$$

since the cross sections are constant between regions. Using Equation 114 to substitute into the left-hand side of Equation 113 yields

$$\begin{aligned} -\widetilde{D}_{i+\frac{1}{2}}(\phi_{i+1} - \phi_i) + \widehat{D}_{i+\frac{1}{2}}(\phi_{i+1} + \phi_i) + \widetilde{D}_{i-\frac{1}{2}}(-\phi_{i-1} + \phi_i) \\ - \widehat{D}_{i-\frac{1}{2}}(\phi_{i-1} + \phi_i) + \Delta x(\Sigma_t - \Sigma_s)\phi_i = \frac{\Delta x}{k_{\text{eff}}}\nu\Sigma_f\phi_i \,. \end{aligned} \tag{116}$$

If Equation 116 is rearranged, it can be rewritten in operator notation as

$$\mathbb{M}\phi = \frac{1}{k_{\text{eff}}}\mathbb{F}\phi \,. \tag{117}$$

where $\mathbb{M}$ is a tridiagonal matrix of the form

$$\mathbb{M} = \begin{bmatrix} \widetilde{D}_{1-\frac{1}{2}} + \widehat{D}_{1+\frac{1}{2}} + \widetilde{D}_{1-\frac{1}{2}} - \widehat{D}_{1-\frac{1}{2}} + \Delta x(\Sigma_t - \Sigma_s) & -\widetilde{D}_{1+\frac{1}{2}} + \widehat{D}_{1+\frac{1}{2}} & 0 & 0 \\ -\widetilde{D}_{2-\frac{1}{2}} - \widehat{D}_{2-\frac{1}{2}} & \ddots & \ddots & 0 \\ 0 & \ddots & \ddots & -\widetilde{D}_{(I-1)+\frac{1}{2}} + \widehat{D}_{(I-1)+\frac{1}{2}} \\ 0 & 0 & -\widetilde{D}_{I-\frac{1}{2}} - \widehat{D}_{I-\frac{1}{2}} & \widetilde{D}_{I-\frac{1}{2}} + \widehat{D}_{I+\frac{1}{2}} + \widetilde{D}_{I-\frac{1}{2}} - \widehat{D}_{I-\frac{1}{2}} + \Delta x(\Sigma_t - \Sigma_s) \end{bmatrix}, \tag{118}$$

and where

$$\phi = \begin{bmatrix} \phi_1 \\ \vdots \\ \phi_I \end{bmatrix}, \tag{119}$$

and

$$\mathbb{F} = \begin{bmatrix} \Delta x\, \nu\Sigma_f & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \Delta x\, \nu\Sigma_f \end{bmatrix}. \quad (120)$$

Equation 117 can be solved using a variety of numerical solvers, including the ones described in Section 2.3. For simply accelerating the MOC solution, GMRES is used. Similar to what was done in Algorithm 7, the right-hand side of Equation 117 is set to a constant, calculated from an initial guess of the flux and k. This puts it in the $Ax = b$ form used in GMRES. Once GMRES has converged to a new estimate of the flux, the new estimate of k is calculated from Equation 111. This process is repeated until the difference between sequential iterations is sufficiently small. Upon convergence, the coarse CMFD fluxes are expanded back out to the fine mesh MOC fluxes using the factor calculated in Equation 44. These iterations are called inner iterations as they are performed within the outer MOC iterations. This iteration scheme is laid out in Algorithm 8.

Algorithm 8. CMFD Inner Iteration

1. Select appropriate $k^{(0)}$ and $\phi_i{}^{(0)}$
   Do until converged
2. $b = \frac{1}{k^{(n)}} \nu\Sigma_f^{LO} \phi_i^{LO\,(n)} \Delta x$
3. Solve $\mathbb{M}\phi_i^{LO\,(n+1)} = b$ for $\phi_i^{LO\,(n+1)}$ using GMRES
4. $k^{(n+1)} = \frac{\sum_i \nu\Sigma_f^{LO} \phi_i^{LO\,(n+1)} \Delta x}{\frac{1}{k^{(n)}} \sum_i \nu\Sigma_f^{LO} \phi_i^{LO\,(n)} \Delta x}$
5. $\phi_i^{HO} = \phi_i^{LO} f_i$

When used to accelerate the MOC solution in Algorithm 7, the CMFD iterations are put before Step 2 and replace the k update in Step 15. In very general terms, the MOC iteration generates the currents that are used in the CMFD iteration, which then gives MOC an updated flux and eigenvalue based on these currents. This process repeats until the fine mesh MOC flux and eigenvalue converge sufficiently. For this 1D slab problem with tight 1x10-10 convergence criteria, the speedup of this combined method is drastic, and is shown in Figure 9.

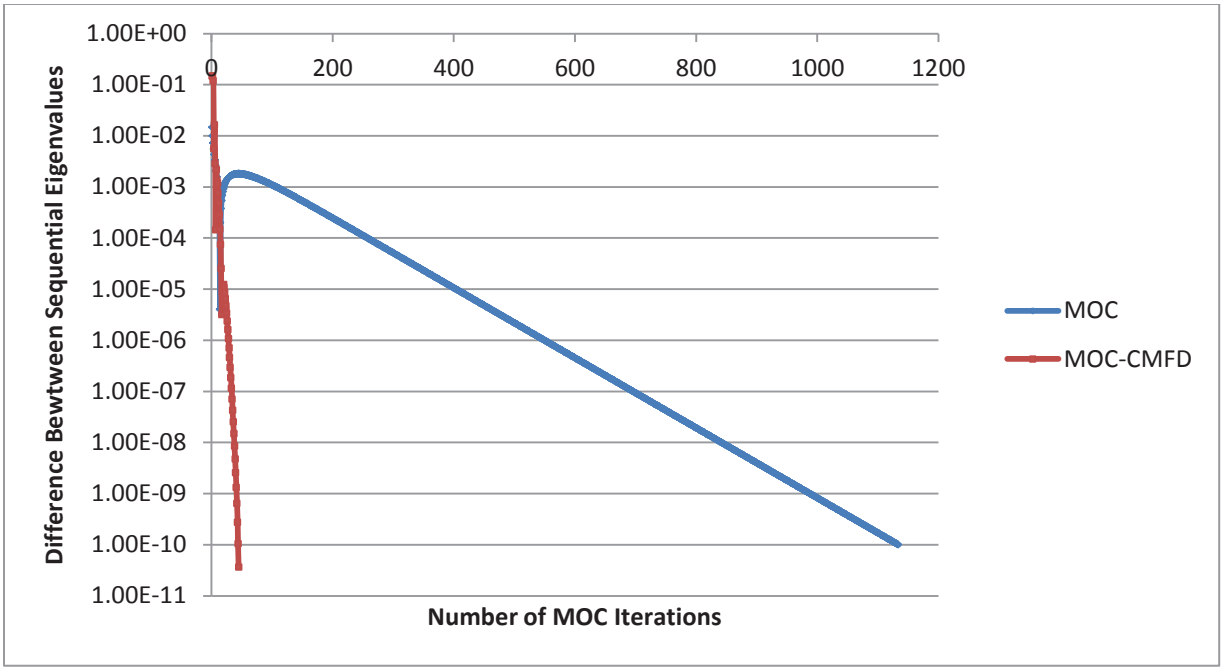

Figure 9. Number of MOC iterations for straight MOC vs CMFD accelerated MOC

It is obvious that CMFD should always be used in tandem along with MOC. It will now be investigated how well JFNK performs as a coupled multiphysics solver for this problem.

4.2.3 Thermal-Hydraulics

When coupling neutronics to thermal-hydraulics, most codes treat each set of physics as a black box. Typically, a neutronics code is used to calculate the fluxes in the problem, which in turn are used to calculate power. A TH code then takes these powers and uses them to determine the heat generated, and therefore the temperatures throughout the problem. These temperatures are then passed back to the neutronics code where they are used to calculate new cross sections. This cycle continues until both the neutronics and TH solutions are stable. However, the transport calculation in the neutronics code is very computationally expensive. Therefore it is desired to minimize the number of transport sweeps performed in a given problem. As seen in the previous section, CMFD is an acceleration technique that greatly reduces the number of transport calculations. Therefore, if the TH calculations could be performed in the CMFD iteration, then the temperatures would converge with the low-order diffusion solution and might help avoid some costly MOC iterations. Since the coupled neutronics-TH equations are nonlinear, JFNK is a good solution method candidate.

For this problem, a simple 1D steady state heat conduction model was used for TH feedback. For this we assume a fuel pin geometry with a fixed surface temperature and constant thermal conductivity. Assuming no axial conduction, the average temperature at each location is given by:

$$T_i = T_s + \frac{\kappa \Sigma_{f,i} \phi_i R^2}{4\pi K}, \tag{121}$$

where κ is the energy released per fission, R is the radius and K is the thermal conductivity of the material. The TH feedback can be thought of as a pin cell where the heat generated within each axial slice travels out towards the outer radius, R, beyond which there is a constant surface temperature T_s. A sketch of this layout is shown in Figure 10. Therefore, if the fluxes are known for each region, then they lead to a realistic description of the temperature in each region using Equation 121. Therefore, the TH problem treats the geometry different than the neutronics problem of Figure 8. These can be reconciled if one simply imagines the neutronics problem as a pin cell as well, but with a reflective boundary condition.

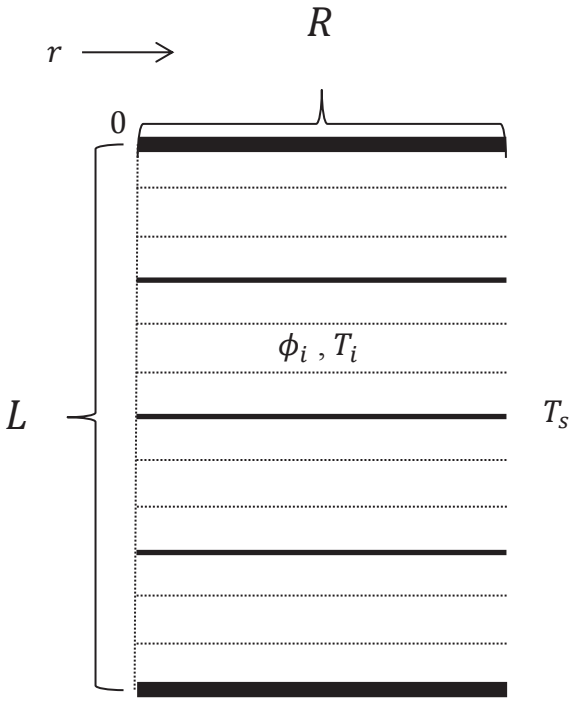

Figure 10. Geometry for TH feedback problem

The cross section update for this problem is performed using

$$\Sigma(\mathrm{T}) = \Sigma_0(T_0)\sqrt{\frac{T_0}{T}}, \tag{122}$$

where T_0 and Σ_0 are the reference temperature and corresponding cross section, respectively. With Equations 121 and 122 in hand, the TH feedback can be implemented in the MOC-CMFD iteration before Step 2 of Algorithm 7. Therefore, new temperatures and cross sections are generated from the newly converged CMFD fluxes, which are then passed on to MOC. However, as discussed before, a more attractive option is to update the cross sections and temperatures along with the fluxes in JFNK.

When performing the temperature updates in JFNK, these calculations are performed using the coarse CMFD mesh. Therefore, the temperatures must first be condensed from the fine MOC mesh to the coarse CMFD mesh. This is achieved using a straightforward average of the fine regions within a coarse mesh. When projecting the coarse temperature back onto the fine mesh, the temperature is assumed the same for every fine region within a coarse mesh.

4.2.4 JFNK System of Equations

The equation for the temperature-dependent eigenvalue is given by summing the fission source over all regions, i,

$$k_{\mathrm{eff}}(\boldsymbol{\phi}, \boldsymbol{T}) = \sum_{i=1}^{I} \nu\Sigma_{f,i}(T_i)\phi_i\Delta x\,. \tag{123}$$

Equations 117, 121, and 123 form a nonlinear system of equations that can be solved using the JFNK method outlined in Algorithm 5:

$$F = \begin{pmatrix} f_\phi(\boldsymbol{\phi}, \boldsymbol{T}, k_{\text{eff}}) = \frac{1}{k_{\text{eff}}} \mathbb{F}\boldsymbol{\phi} - \mathbb{M}\boldsymbol{\phi} \\ f_T(\boldsymbol{T}, \boldsymbol{\phi}) = T_0 + \frac{\kappa \Sigma_f(T)\boldsymbol{\phi} R^2}{4\pi K} - T \\ f_k(\boldsymbol{\phi}, \boldsymbol{T}, k_{\text{eff}}) = k_{eff} - \sum_{i=1}^{I} \nu \Sigma_{f,i}(T_i)\phi_i \Delta x \end{pmatrix}. \quad 124$$

The third equation, $f_k(\phi, T, k)$, can be eliminated from Equation 124 by substituting the functional evaluation of k_{eff} from Equation 123 into the first equation:

$$F = \begin{pmatrix} f_\phi(\boldsymbol{\phi}, \boldsymbol{T}) = \frac{1}{k_{\text{eff}}(\boldsymbol{\phi}, \boldsymbol{T})} \mathbb{F}\boldsymbol{\phi} - \mathbb{M}\boldsymbol{\phi} \\ f_T(\boldsymbol{T}, \boldsymbol{\phi}) = T_0 + \frac{\kappa \Sigma_f(T)\boldsymbol{\phi} R^2}{4\pi K} - T \end{pmatrix}. \quad 125$$

Since the fluxes and temperatures are solved simultaneously in JFNK, the solution vector is of the form

$$\boldsymbol{x} = \begin{bmatrix} \phi_1 \\ \vdots \\ \phi_I \\ T_1 \\ \vdots \\ T_I \end{bmatrix}. \quad 126$$

Therefore, in Step 2 and Step 3 of Algorithm 5, the residual F is evaluated using Equation 126, which performs the temperature and cross section update.

4.2.5 Results

The various coupling techniques outlined in Chapter 3 were applied to this 1D problem and compared against each other to assess their performance. Because it is the current default method implemented in MPACT, the Picard iteration scheme was implemented to serve as a control against which the other methods would be compared. The other methods applied were a coupled JFNK case as well as two different CMFD-Coupling cases: CMFD-2 and CMFD-10. CMFD-1 is

the same as the Picard implementation, where CMFD is performed and feedback is applied only once per MOC transport sweep. As a result, CMFD-2 adds the least amount of additional computational work to the problem while CMFD-10 adds nine additional CMFD-TH loops. A plot showing the eigenvalue residual as a function of MOC sweeps is shown for all four approaches in Figure 11 while a plot showing the 2-norm of the flux residual is shown in Figure 12. The convergence criteria for these cases were made extremely tight: $1x10^{-10}$ for both the eigenvalue and the 2-norm of the flux residual.

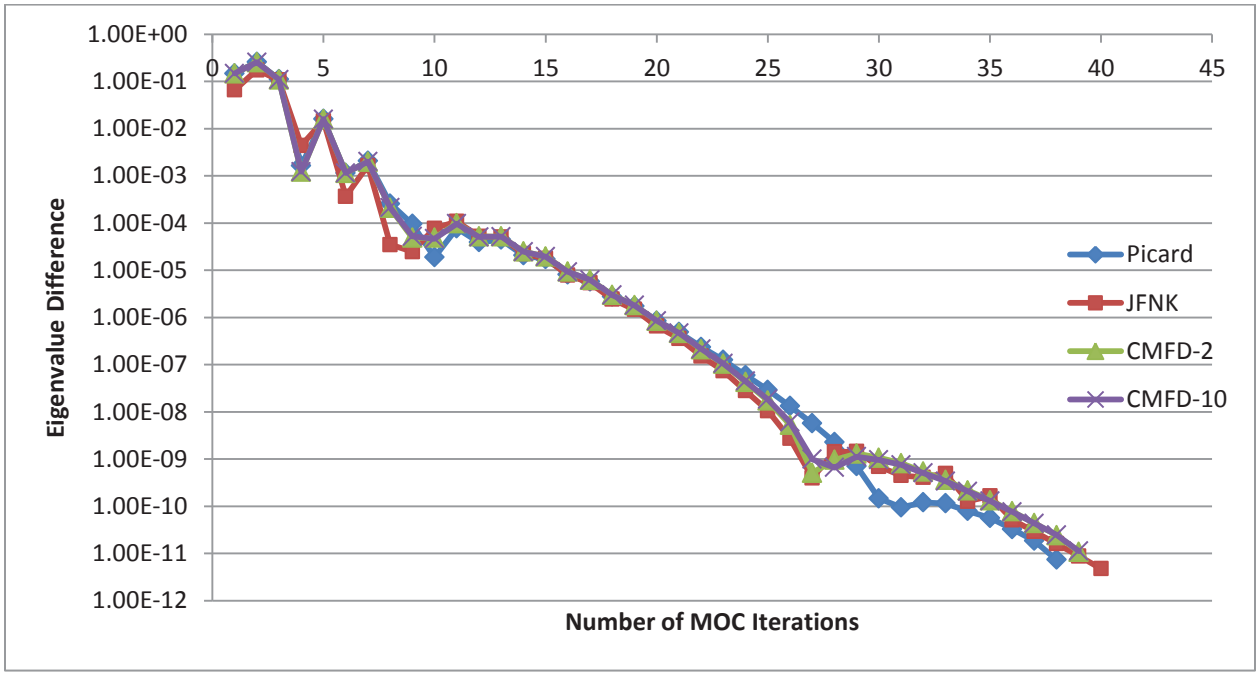

Figure 11. Convergence of eigenvalue for the 1D, one-group homogenous slab problem

As seen in Figure 11 and Figure 12, the Picard approach takes 38 MOC iterations to converge, JFNK takes 40, and both CMFD-Coupling implementations take 39. It can be seen from Figure 12 that, with convergence criteria of $1x10^{-10}$ for both the eigenvalue difference and the 2-norm of the difference in flux, the flux residual is the limiting factor for these problems. Figure 11 shows that, despite the eight extra CMFD-TH iterations, the CMFD-10 implementation differs only slightly from the CMFD-2 case. Additionally, they are comparable to the Picard residuals until they begin to deviate significantly around MOC iteration 25. At a more realistic convergence criterion of $1x10^{-6}$, all methods take 20 iterations for the eigenvalue

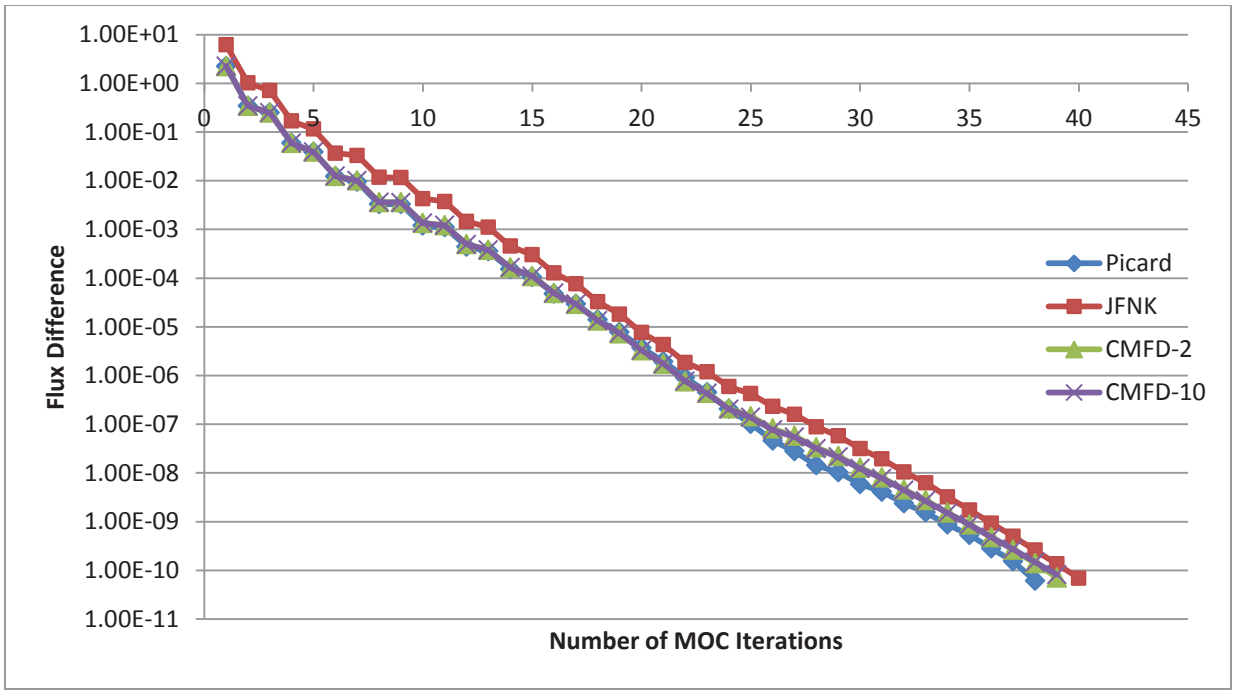

Figure 12. Convergence of the 2-norm of the flux difference for the 1D, one-group homogenous slab problem

to converge. In Figure 12 it should be noted that the JFNK flux residual is consistently lagging behind the residuals of the other three cases. Also, similar to the eigenvalue convergence plot of Figure 11, the CMFD-2 and CMFD-10 flux differences are nearly identical. If a more realistic convergence criterion of $1x10^{-6}$ was used, the Picard, CMFD-2, and CMFD-10 implementations would have taken 22 MOC iterations to converge the fluxes while JFNK would have taken 24.

When looking at Figure 11 and Figure 12 together, it is clear that performing the TH update within the JFNK iterations does not offer a significant speedup compared to the standard Picard iteration. Additionally, the CMFD-Coupling methods do not offer a speedup either, though they perform marginally better than JFNK. This could be due to the fact that this problem is too simple and easy to solve so the benefits of the alternative methods are not apparent. However, despite the additional MOC sweeps, the JFNK and CMFD-Coupling methods are more tightly coupled than the standard Picard method for multiphysics coupling.

4.3 2D PWR Pin Cell

One of the simplest problems in reactor analysis is a 2D pin cell. While these problems tend to be small in size and fast to solve, they can offer significant insight into method development. Unlike the simplified problems discussed in the previous sections, this 2D pin cell has both a spatial and a multigroup energy dependence.

4.3.1 Problem Description

The pin cell model was CASL VERA Core Physics Benchmark Problem 1A [1], which is a single 2D Hot Zero Power (HZP) pin cell at Beginning of Life (BOL). Due to reflective boundary conditions, this 2D pin cell model can be thought of as being a single, infinitely tall fuel rod in a square coolant channel within an infinite array of pins. This simple model consists of four regions consisting of standard materials: a UO_2 fuel pellet, a helium gap, a Zircaloy-4 cladding, and borated water as the surrounding coolant and moderator. The operating conditions and input specifications are shown in Table 1. These input parameters were taken from the VERA Core Physics Benchmark Problem Specifications [1]. A 2D representation of the pin cell is shown in Figure 13.

Table 1: 2D pin cell input specifications **[1]**

Parameter	Value
Moderator Temperature	565 K
Moderator Density	0.743 g/cc
Fuel Temperature	565 K
Fuel Density	10.257 g/cc
Fuel Enrichment	3.10%
Power	0.0%
Boron Concentration	1300 ppm
Pressure	2250 psia
Pin Pitch	1.26 cm
Fuel Radius	0.4096 cm
Gap Thickness	0.0084 cm
Clad Thickness	0.057 cm

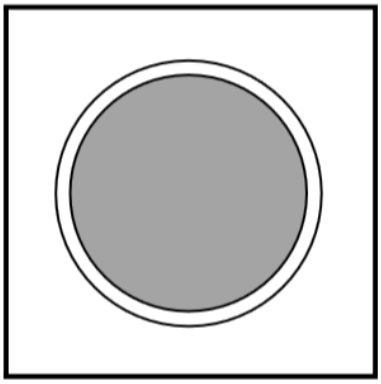

Figure 13. 2D representation of a pin cell **[1]**

4.3.2 JFNK Eigenvalue Solver Implementation

As stated before, PETSc was used to implement JFNK in parallel within MPACT to solve the eigenvalue problem as described in Section 3.2.3. This 2D pin cell problem was executed on four processors in parallel, one for each coarse mesh quarter of the full geometry. A 51-group cross section library was used to capture the energy dependence of the problem. The results comparing the standard power method to both the preconditioned and unpreconditioned forms of JFNK are shown in Table 2.

Table 2: Results of the 2D pin cell JFNK eigenvalue problem

Case	Eigenvalue	MOC Iterations	JFNK GMRES Iterations	Runtime (s)
Power Method	1.1869334	39	-	6.16
JFNK-Unpreconditioned	1.1869333	39	13345	18.38
JFNK-Preconditioned	1.1869332	39	13647	20.03

As seen in Table 2, both JFNK versions and the power method converge to the same eigenvalue while taking the same number of transport sweeps. However, both JFNK implementations took approximately three times longer to solve. This is due to the extra work required to carry out a JFNK solve. The size of the coarse mesh system for this problem is 204x204: the four coarse mesh regions multiplied by the 51 energy groups. For each of the 39 iterations during each of the JFNK solves, Newton's method required only two steps to converge. However, each of those Newton steps took approximately 170 GMRES iterations to converge that step, almost the full size of the matrix.

Also it should be noted that while the goal of preconditioning is to reduce the total number of linear GMRES iterations, in this case, the addition of a preconditioner actually slightly increased the number of GMRES iterations. This is likely due to the fact that this 2D pin cell problem is highly unstable. This is evidenced not only by the large number of transport sweeps required for convergence, but also by the fact that, on average, 170 GMRES iterations were required for each nonlinear Newton step when the size of the entire system is 204.

4.3.3 Coupled JFNK Implementation

After using JFNK as an eigenvalue solver, the same methodology was reworked to incorporate the feedback effects from the TH solver within JFNK as described in Section 3.2.4. However, unlike the eigenvalue solver, the coupled implementation was not applied in parallel. Because of this fact, the 51-group library causes the problem to have very long runtimes. Alternatively, an 8-group cross section library was used for the JFNK coupled cases instead. While the use of a few-group library will lead to solutions that may be inaccurate when compared to a library with a large number of energy groups, the purpose of this investigation is to assess the convergence behavior of JFNK when used to solve a coupled multiphysics problem. To that end, the 8-group library will be sufficient.

This pin cell was modeled in full symmetry, with the CMFD portion being 32x32 and the additional TH portion being 4x4. With TH feedback now turned on, the problem was modeled at Hot Full Power (HFP) with a rated power of 0.268 kW and a flow rate of 0.00236 Mlbs/hr. This problem incorporated the two preconditioners discussed in Section 3.2.4 as well as no preconditioner. The results of these cases, as well as those for the standard Picard iteration, are shown in Table 3.

Table 3: Results of the coupled JFNK 2D pin cell problem

Case	Eigenvalue	MOC Iterations	JFNK GMRES Iterations	Runtime (s)
Picard	1.1702266	15	-	1.61
JFNK-Unpreconditioned	1.1702266	15	426	2.53
JFNK-Diag Preconditioner	1.1702266	15	199	1.67
JFNK-LT Preconditioner	1.1702266	15	198	1.72

Turning TH feedback on for this problem is actually slightly misleading: since there is no axial or radial temperature gradient, the Simplified TH solver converges the temperatures during the first iteration, after which the problem becomes an eigenvalue search. Despite this, Table 3 still demonstrates the behavior of the block Jacobian preconditioners. As expected, the unpreconditioned version required the most GMRES iterations to converge, while the two preconditioners tested performed comparably to one another.

It should be noted that the results of the coupled cases in Table 3 require significantly fewer transport sweeps than the eigenvalue cases shown in Table 2. This is due, in part, to the fact that the 51-group cross section library was replaced for the smaller 8-group library. This change greatly reduces the size of the coupled problem. Also, the addition of the TH feedback acts like a damping factor, easing the convergence of the problem. Additionally the runtimes for the coupled cases in Table 3 are comparable across all methods while the eigenvalue JFNK cases in Table 2 take significantly longer. This is again due to the smaller problem size as a result of using the 8-group library. The 1-2 second runtimes of the coupled cases are too short to establish any meaningful assessment of the methods tested. Therefore larger problems are tested in the following sections to gain additional insight.

While it is possible to implement the CMFD-Coupling technique outlined in Section 3.3 to this coupled problem, it would be for nothing. As stated above, after the first iteration the problem becomes an eigenvalue problem. There is nothing to be gained by using the CMFD-Coupling technique since there is no significant TH feedback after the first iteration.

4.4 2D PWR Fuel Lattice

This 2D fuel lattice expands on the techniques used for the 2D pin cell case and now applies them to a problem with more radial variability. Additionally, this larger problem allows for the comparison of the normalized fission reaction rate distribution. These fission rates, when normalized, can be used to represent the pin power distribution.

4.4.1 Problem Description

This 2D fuel lattice model was chosen to be CASL VERA Core Physics Benchmark Problem 2A [1], which is a single 2D HZP fuel lattice at BOL. The individual fuel pins within the problem are identical to the single pin cell case described in Section 4.3.1. These fuel pins are arranged in a 17x17 array along with 24 guide tubes and a central instrument tube with reflective boundary conditions on all sides. A 2D representation of the lower right quadrant of the fuel lattice is shown in Figure 14.

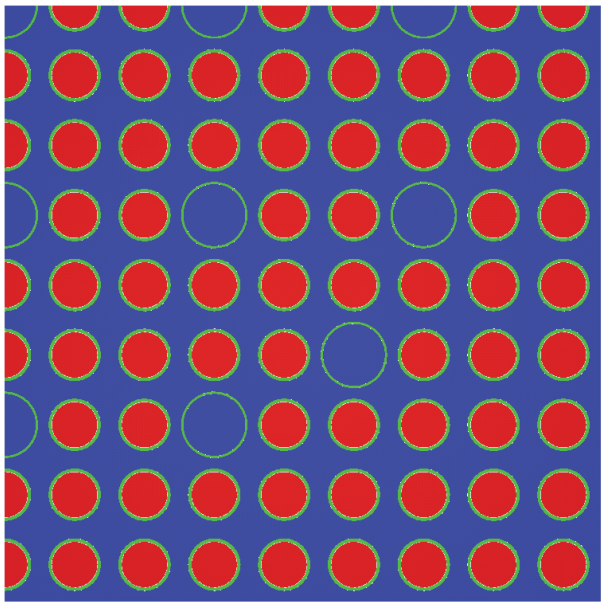

Figure 14. 2D representation of 17x17 fuel lattice in quarter symmetry **[1]**

This model consists of four standard materials: UO_2 fuel, helium, Zircaloy-4 cladding, and borated water as the surrounding coolant and moderator. The operating conditions are the same as those shown in Table 1 with the addition of the guide tube and instrumentation tube specifications shown in Table 4. These input parameters were taken from the VERA Core Physics Benchmark Problem Specifications [1].

4.4.2 JFNK Eigenvalue Solver Implementation

Similar to the 2D pin cell eigenvalue problem, this fuel lattice was modeled in full symmetry and executed on four processors, where each processor handled a quarter of the lattice. This problem also used a 51-group cross section library. In addition, the approximate preconditioner in

Table 4: 2D fuel lattice input specifications **[1]**

Parameter	Value
Inner Guide Tube Radius	0.561 cm
Outer Guide Tube Radius	0.602 cm
Inner Instrument Tube Radius	0.559 cm
Outer Instrument Tube Radius	0.605 cm
Tube Materials	Zircaloy-4
Assembly Pitch	21.50 cm

Equation 93 was used such that the migration matrix, $\mathbb{M}$, is the preconditioner matrix. The size of the linearsystem for this problem is 16,524x16,524. The results comparing the standard power method to both the preconditioned and unpreconditioned forms of JFNK are shown in Table 5.

Table 5: Results of the 2D fuel lattice JFNK eigenvalue problem

Case	Eigenvalue	MOC Its.	PP RMS	PP Max	JFNK GMRES Its.	Runtime (s)
Power Method	1.1821254	10	-	-	-	33.25
JFNK-Unpreconditioned	1.1821254	10	0.000%	0.000%	7548	45.43
JFNK-Preconditioned	1.1821255	10	0.000%	0.000%	1738	36.78

As seen in Table 5, both JFNK versions and the power method converge to the same eigenvalue while taking the same number of transport sweeps. Additionally, the pin powers throughout the problem were compared to those generated using the default power method. Both JFNK implementations converged to the same pin powers as evidenced by the nonexistent Root Mean Square (RMS) and max pin power differences. The power method resulted in the fastest runtime, while JFNK with preconditioning was only slightly slower. The case that implemented an unconditioned JFNK routine was significantly slower than the standard power method. This is because the total number of linear GMRES iterations required for convergence more than quadrupled for the unpreconditioned case. The average number of GMRES iterations performed per Newton step was 58 for the preconditioned case and 252 for the unpreconditioned case. For both the preconditioned and the unpreconditioned cases, Newton's method required three iterations to converge during each of the ten outer iterations.

The results from this 2D fuel lattice case also reinforce some of the conclusions reached for the 2D pin cell eigenvalue case in Section 4.3.2. In that section it was concluded that the 2D pin cell problem was highly unstable. The fact that a full symmetry 17x17 fuel lattice calculation took only 10 transport sweeps, while a singular pin cell took 39, helps to uphold that finding. In addition, the 204x204 pin cell case took an average of 170 GMRES iterations to converge, while the 16,524x16,524 fuel lattice took at most 252 GMRES iterations to converge.

4.4.3 Coupled JFNK Implementation

The same 2D fuel lattice was investigated at HFP with TH feedback enabled. The problem was modeled with a rated power of 0.0708 MW and a flow rate of 0.682 Mlbs/hr. The cases were executed in serial in full symmetry with an 8-group cross section library. The CMFD portion of the problem is 2592x2592 and the additional TH portion is 288x288. The results of the coupled JFNK solver are shown in Table 6 along with the standard Picard iteration case.

Table 6: Results of the coupled JFNK 2D fuel lattice problem

Case	Eigenvalue	MOC Its.	PP RMS	PP Max	JFNK GMRES Its.	Runtime (s)
Picard	1.1675591	8	-	-	-	23.76
JFNK-Unpreconditioned	1.1675591	8	0.000%	0.000%	904	50.29
JFNK-Diag Preconditioned	1.1675592	8	0.000%	0.000%	154	28.65
JFNK-LT Preconditioned	1.1675592	8	0.000%	0.000%	154	28.92

While all of the JFNK cases shown in Table 6 converge to the same eigenvalue and pin powers as Picard, they are all slower. The unpreconditioned case takes the longest by far because of the extra GMRES iterations needed for convergence. The cases that applied both forms of the preconditioner used the same number of GMRES iterations and essentially took the same amount of time to execute. Most importantly, the coupled JFNK solver did not reduce the number of MOC transport sweeps.

4.4.4 CMFD-Coupling Comparison

As discussed in Section 3.3, two methods for controlling the CMFD-Coupling iteration scheme were developed. A number of different implementations were tried and their results are shown in Table 7.

Table 7: Various CMFD-Coupling results for the 2D fuel lattice problem

Case	Eigenvalue	MOC Its.	PP RMS	PP Max	CMFD Solves	Runtime (s)
Picard	1.1675591	8	-	-	8	23.76
CMFD-2	1.1675591	8	0.000%	0.000%	15	26.15
CMFD-3	1.1675591	8	0.000%	0.000%	22	28.5
CMFD-10	1.1675591	8	0.000%	0.000%	71	44.81
ΔT<0.1	1.1675591	8	0.000%	0.000%	19	28.08
ΔT<0.001	1.1675591	8	0.000%	0.000%	32	31.98

Since the Picard iteration scheme is the same as CMFD-1, CFMD-2 was chosen as a solution method because performing the CMFD-TH loop one additional time per transport sweep introduces the least amount of additional work. Similarly, CMFD-3 was investigated to see what additional benefits, if any, are gained from looping between CMFD and the TH solver one additional time. CMFD-10 was considered as an extreme case in order to see if looping a large number of times offers any benefit in terms of transport iteration reduction. The $\Delta T < 0.1$ case was performed as a realistic maximum temperature difference threshold while the $\Delta T < 0.001$ case was investigated as an extreme case, with intentions similar to those for CMFD-10.

Table 7 shows all methods tested converging to the exact same eigenvalue and pin powers as the Picard method. The only variance between the methods is the total number of times CMFD is solved, which closely corresponds to the problem runtime. For this 2D fuel lattice, there is no obvious benefit to using either CMFD-Coupling or JFNK.

4.5 3D PWR Fuel Pin

In order to test the performance of JFNK on a 3D problem with little radial heterogeneity, a simple 3D fuel pin problem was created and tested.

4.5.1 Problem Description

In order to create a 3D fuel pin problem, the 2D pin cell problem from Section 4.3.1 was used as a starting point. This 2D pin cell was then extended axially 250 cm. Since MPACT utilizes a 2D/1D approach to solving the neutron transport problem, the problem is broken up into a series of 28 2D axial planes at differing heights. Each plane is solved independently using 2D MOC while their axial transverse leakage is solved using a 1D axial calculation. The problem also no longer has reflective top and bottom boundary conditions and instead has a vacuum boundary condition for both the top and bottom. The radial boundary condition is still reflective.

4.5.2 JFNK Eigenvalue Solver Implementation

Like the previous eigenvalue problems, the 3D fuel pin was modeled in full symmetry and used a 51-group cross section library. However, this case was executed in parallel on 28 processors. The preconditioner used to accelerate convergence was chosen to be the migration matrix, $\mathbb{M}$. The size of the linear system for this problem is 5,712x5,712. The results comparing the standard power method to both the preconditioned and unpreconditioned forms of JFNK are shown in Table 8.

Table 8: Results of the 3D fuel pin JFNK eigenvalue problem

Case	Eigenvalue	MOC Its.	PP RMS	PP Max	JFNK GMRES Its.	Runtime (s)
Power Method	1.1729369	17	-	-	-	10.8
JFNK-Unpreconditioned*	1.1729373	17	0.011%	0.016%	26383	166.42
JFNK-Preconditioned	1.1729370	17	0.002%	0.003%	10721	25.15

*Turning off GMRES restart was required for convergence

Table 8 shows that all three methods converged to roughly the same eigenvalue and pin powers within the same number of MOC iterations. However, in order to get the unpreconditioned JFNK case to converge at all, the GMRES restart capability had to be turned off in PETSc. By default, PETSc restarts GMRES after 30 iterations in order to reduce the number of vectors stored in memory. Even without restarts, the unpreconditioned case leads to a worse estimate of the eigenvalue and pin powers when compared to the preconditioned case. Similarly, the unpreconditioned case took significantly longer to converge than the other two cases. However, while the preconditioned case offered a better solution estimate with a faster runtime when

compared to the unpreconditioned case, it was over two times slower than the standard power method.

The preconditioned case required four nonlinear Newton steps to converge during the first outer iteration but required only three for each of the 16 other outer iterations. On average, each Newton iteration needed 206 GMRES iterations to converge. However, the unpreconditioned case required four Newton iterations four different times, helping to increase the total number of GMRES iterations even further. On average each Newton step required 480 GMRES iterations to converge.

4.5.3 Coupled JFNK Implementation

The same 3D fuel pin described in the previous sections was investigated at HFP with TH feedback enabled. The coupled problem was modeled with a rated power of 0.0669 MW and a flow rate of 0.00263 Mlbs/hr. These cases were executed in serial in full symmetry with an 8-group cross section library. The CMFD portion of the problem is 896x896 and the additional TH portion is 112x112. The results of the coupled JFNK solver, along with the standard Picard iteration case, are shown in Table 9.

Table 9: Results of the coupled JFNK 3D fuel pin problem

Case	Eigenvalue	MOC Its.	PP RMS	PP Max	JFNK GMRES Its.	Runtime (s)
Picard	1.1544792	15	-	-	-	21.85
JFNK-Unpreconditioned	1.1544996	15	0.173%	0.245%	31734	364.16
JFNK-Diag Preconditioned	1.1544823	15	0.027%	0.040%	8592	111.51
JFNK-LT Preconditioned	1.1544823	15	0.027%	0.040%	6300	86.18

Both preconditioned cases converged to the same eigenvalue which differs from the Picard eigenvalue by only 0.31 percent mille (pcm). The unpreconditioned case, however, had an eigenvalue estimate which differed from the Picard eigenvalue by 2.04 pcm. In addition to the larger eigenvalue difference, the unpreconditioned JFNK case also had significantly worse estimates of the pin powers than the preconditioned cases when compared to the Picard case. These differences are likely caused by the fact that JFNK calculates the eigenvalue and fluxes

differently than Picard. It is expected that with tighter convergence criteria, the solutions should converge to the same answer. Without preconditioning, the coupled JFNK solver took over 16 times longer to converge than the standard Picard iteration. This is due to the large number of linear GMRES iterations required to converge each nonlinear Newton step within JFNK. Including the block-diagonal preconditioner significantly reduces the total number of GMRES iterations performed and the addition of the lower left preconditioner block reduces this number even further. However, despite these reductions, all coupled JFNK implementations required significantly longer runtimes than Picard to converge.

The eigenvalue difference between successive iterations is shown as a function of MOC iteration count in Figure 15. Likewise, the 2-norm of the fission source residual as a function of MOC iteration count is shown in Figure 16. As seen in Figure 15, the convergence of the eigenvalue remains roughly constant across all methods. The two preconditioned cases have the same convergence and follow very closely with the convergence of Picard. However, the unpreconditioned case differs only slightly. In Figure 16 the fission source residual of the preconditioned cases follow each other very closely. For the most part, all three coupled JFNK methods perform only slightly better than the Picard implementation over all MOC iterations.

4.5.4 CMFD-Coupling Comparison

As discussed in Section 3.3, two methods for controlling the CMFD-Coupling iteration scheme were developed. A number of different implementations were tried and their results are shown in Table 10.

Table 10: Various CMFD-Coupling results for the 3D pin cell problem

Case	Eigenvalue	MOC Its.	PP RMS	PP Max	CMFD Solves	Runtime (s)
Picard	1.1544792	15	-	-	15	21.85
CMFD-2	1.1544797	15	0.007%	0.012%	29	29.75
CMFD-3	1.1544797	15	0.007%	0.012%	43	34.47
CMFD-10	1.1544797	15	0.007%	0.011%	141	61.34
$\Delta T<0.1$	1.1544797	15	0.006%	0.010%	41	34.26
$\Delta T<0.001$	1.1544797	15	0.007%	0.012%	122	56.16

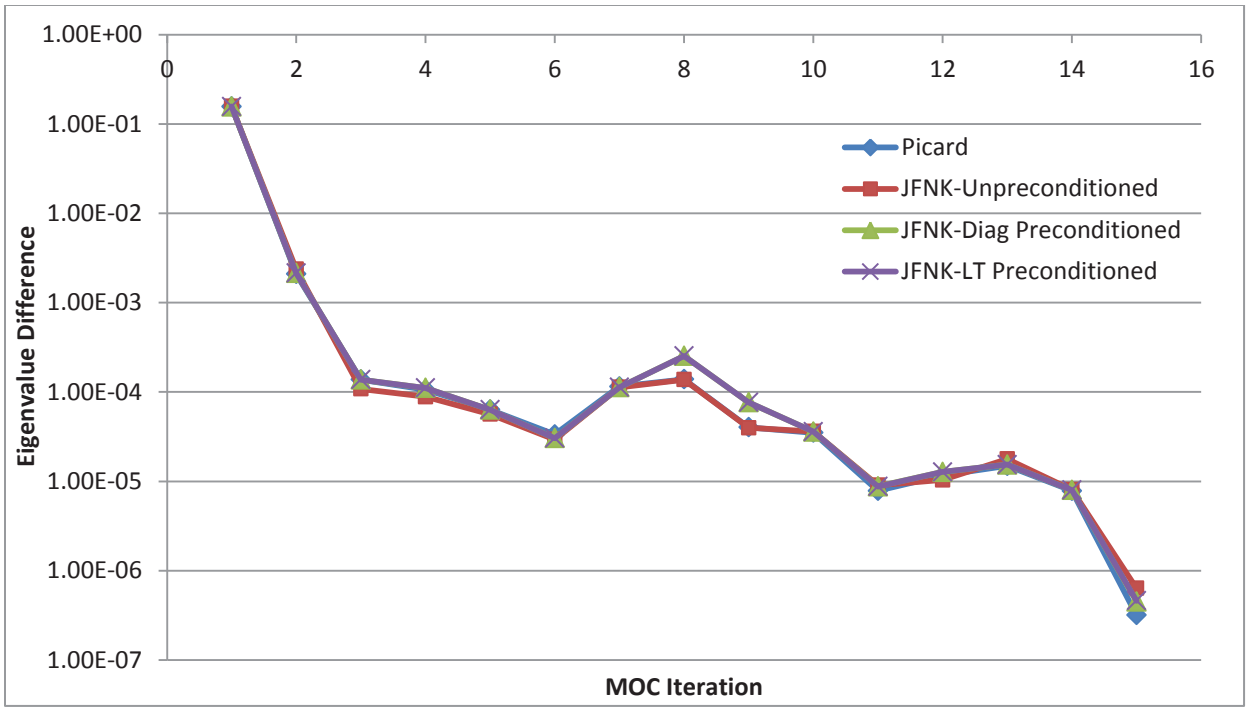

Figure 15. Comparison of the eigenvalue differences for each coupled JFNK implementation applied to the 3D fuel pin problem

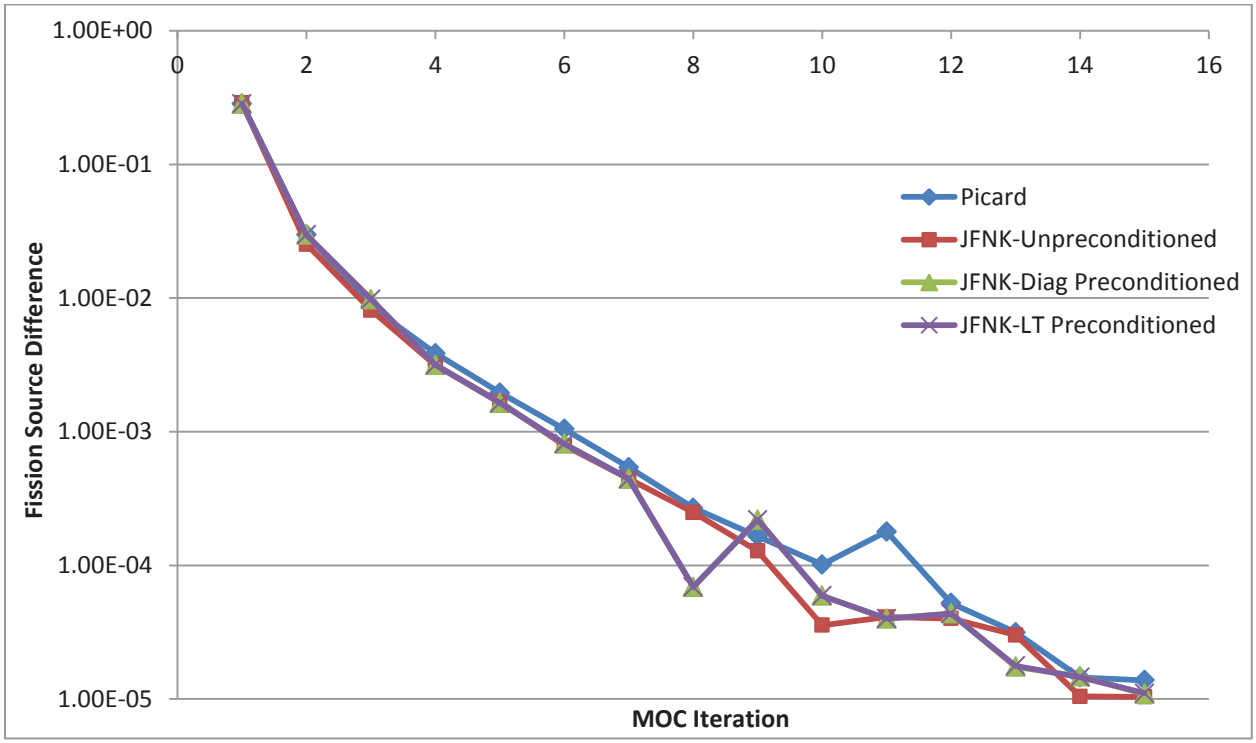

Figure 16. Comparison of the fission source differences for each coupled JFNK implementation applied to the 3D fuel pin problem

All of the various CMFD-Coupling approaches shown in Table 10 converge to the same eigenvalue. Similarly all of their calculated pin powers are very comparable. The only major difference between these methods when applied to the 3D fuel pin problem is the runtime, which is closely correlated to the number of CMFD solves required for convergence. When compared to Table 9 it is seen that all CMFD-Coupling strategies outperform every JFNK coupling technique in terms of eigenvalue and pin power accuracy as well as runtime. Even the CMFD-10 and $\Delta T < 0.001$ cases, which were intended to be overkill, ran faster than all of the coupled JFNK strategies.

The eigenvalue difference between successive iterations is shown as a function of MOC iteration count in Figure 17. Similarly, Figure 18 shows the 2-norm of the fission source residual as a function of MOC iteration count. Figure 17 shows that the eigenvalue convergence for all CMFD-Coupling strategies investigated follow the same trend. While these methods differ from the Picard convergence initially, all methods eventually line up and converge simultaneously. However, as seen in Figure 18, the fission source residual for all CMFD-Coupling strategies is universally lower than that for the Picard implementation. Despite the reduced fission source residual, there is no reduction in the number of MOC iterations required for convergence or the runtime.

4.6 3D 7x7 Assembly

In order to test JFNK on a problem similar to those found in real world applications, a miniaturized 3D assembly problem was tested.

4.6.1 Problem Description

This 3D 7x7 assembly is a miniaturized version of CASL VERA Core Physics Benchmark Problem 3A [1], which is a single 3D 17x17 HZP fuel assembly at BOL. This 7x7 version contains 40 fuel pins and nine guide tubes. Their configuration is shown in quarter symmetry in Figure 19.

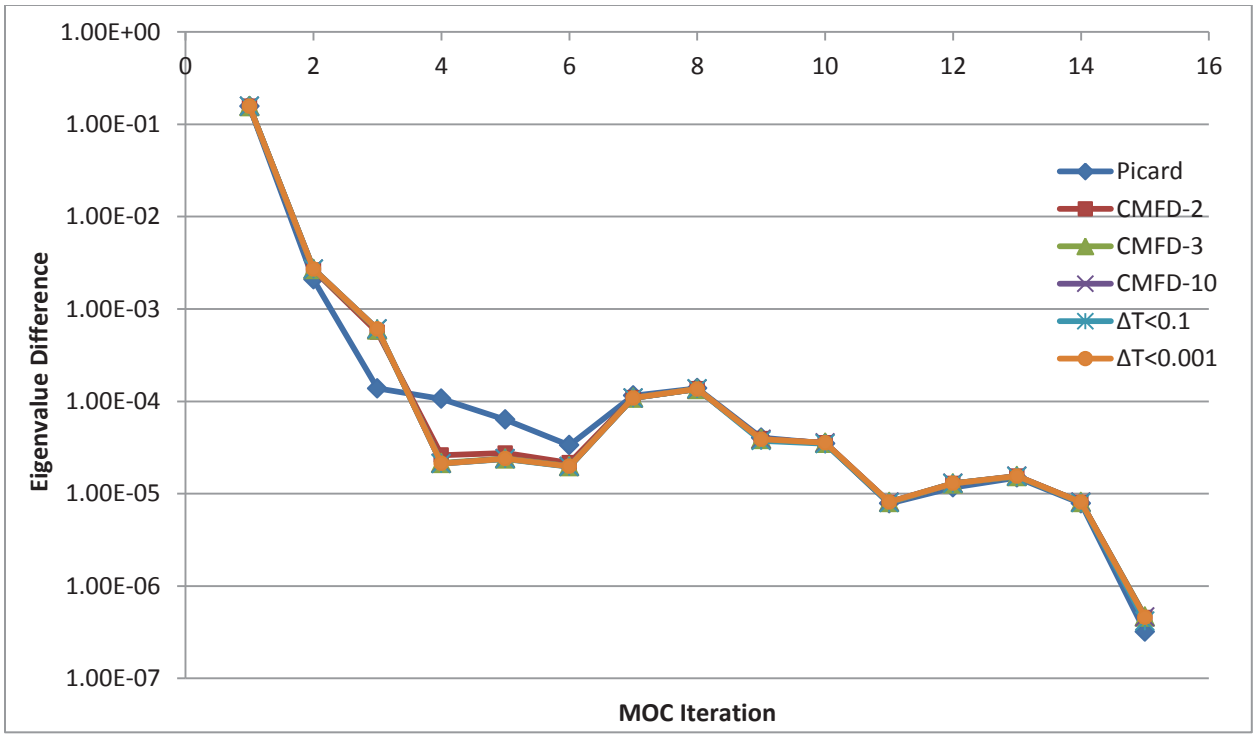

Figure 17. Comparison of the eigenvalue differences for each CMFD-Coupling implementation applied to the 3D fuel pin problem

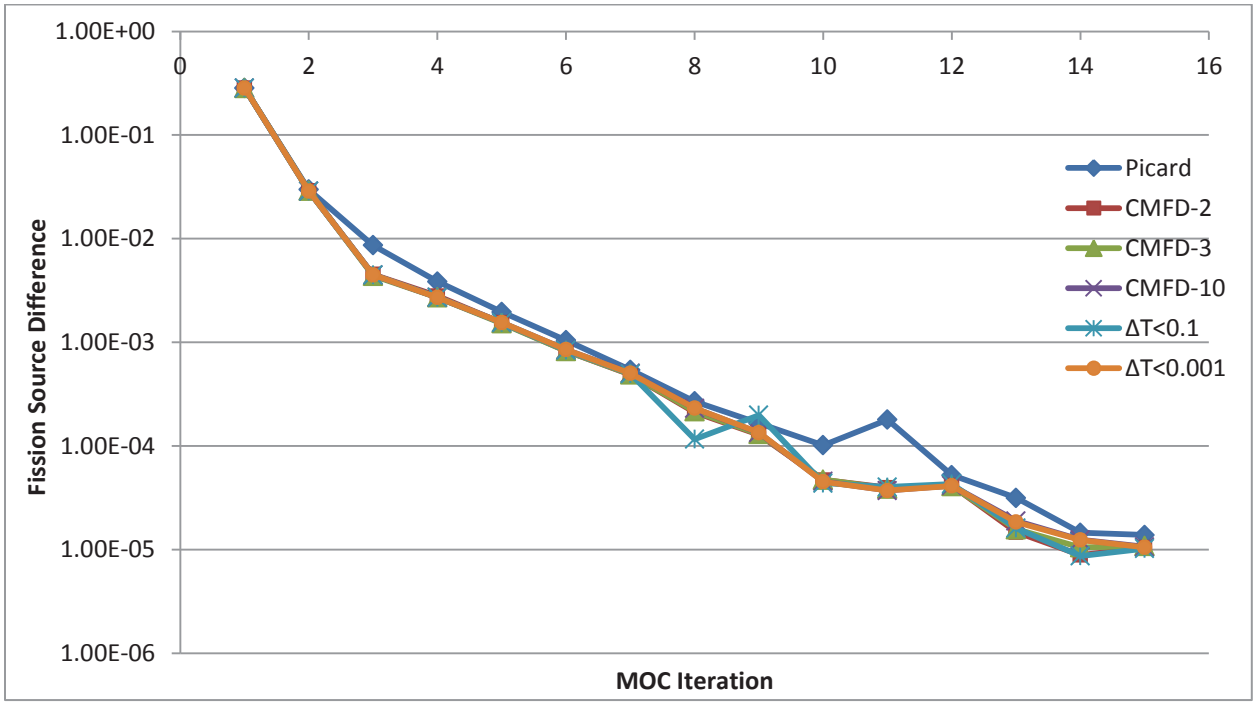

Figure 18. Comparison of the fission source differences for each CMFD-Coupling implementation applied to the 3D fuel pin problem

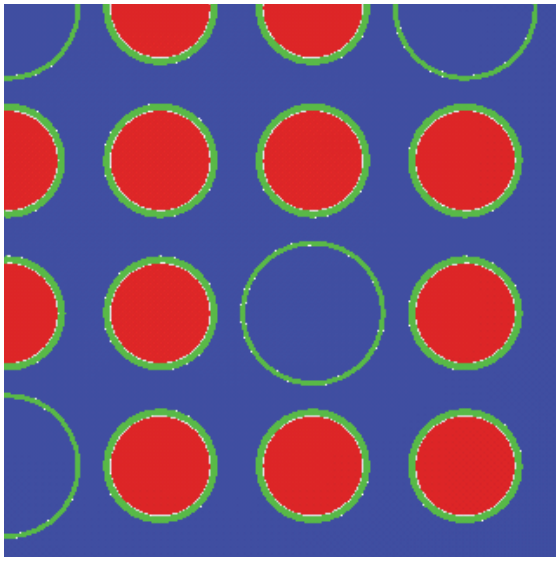

Figure 19. 2D representation of 7x7 fuel assembly in quarter symmetry

This problem uses the same input specifications as described in Table 1 and Table 4. Like the 3D pin cell problem described in Section 4.5, the active fuel height for this problem is 250 cm. This problem is broken up into a series of 35 2D axial planes at differing heights and was modeled in quarter symmetry with reflective radial boundary conditions.

4.6.2 JFNK Eigenvalue Solver Implementation

This 3D fuel assembly was executed in parallel on 35 processors to solve the eigenvalue problem. This problem also used a 51-group cross section library. In addition, the approximate preconditioner in Equation 93 was used such that the migration matrix, $\mathbb{M}$, is the preconditioner matrix. The size of the linear system for this problem is 28,560x28,560. The results comparing the standard power method to both the preconditioned and unpreconditioned forms of JFNK are shown in Table 11.

Table 11: Results of the 3D 7x7 fuel assembly JFNK eigenvalue problem

Case	Eigenvalue	MOC Its.	PP RMS	PP Max	JFNK GMRES Its.	Runtime (s)
Power Method	1.1082356	10	-	-	-	20.99
JFNK-Unpreconditioned*	1.1082321	39	0.012%	0.048%	99676	828.18
JFNK-Preconditioned	1.1082500	13	0.207%	0.312%	6224	28.99

*GMRES restart of 500 was required for convergence

As seen in Table 11, the unpreconditioned and the preconditioned cases converged to eigenvalues within 0.35 and 1.44 pcm of the power method eigenvalue, respectively. However, the unpreconditioned case would not converge with the default GMRES restart capability in PETSc so the restart value was increased to 500 in order to have the problem converge. Turning off restarts entirely was not possible with such a large linear system due to memory constraints. Even with the larger restart value, the unpreconditioned case required 29 more transport sweeps than did the standard power iteration. While the preconditioned case performed better, it still required an additional three MOC iterations beyond what was required by the power method. Because of these additional transport sweeps, neither JFNK eigenvalue solver implementation was faster than the power method.

The unpreconditioned case required at most 5 nonlinear Newton steps to converge a given iteration and as few as three. On average each of these Newton steps required 733 linear GMRES iterations to converge. The preconditioned case only required three Newton steps per iteration with an average of 160 GMRES per step.

4.6.3 Coupled JFNK Implementation

The 3D 7x7 fuel assembly was then investigated at HFP with TH feedback enabled. The coupled problem was modeled with a rated power of 2.945 MW and a flow rate of 0.1157 Mlbs/hr. These cases were executed in serial, in quarter symmetry, and with an 8-group cross section library. The CMFD portion of the problem is 4480x4480 and the additional TH portion is 392x392. The results of the coupled JFNK solver, along with the standard Picard iteration case, are shown in Table 12.

Table 12: Results of the coupled JFNK 3D 7x7 fuel assembly problem

Case	Eigenvalue	MOC Its.	PP RMS	PP Max	JFNK GMRES Its.	Runtime (s)
Picard	1.1000484	10	-	-	-	44.79
JFNK-Unpreconditioned	1.1000559	9	0.059%	0.108%	32654	1652.17
JFNK-Diag Preconditioned	1.1000596	9	0.092%	0.129%	1087	85.99
JFNK-LT Preconditioned	1.1000596	9	0.092%	0.129%	1087	86.02

The unpreconditioned case converged to an eigenvalue that differed from the eigenvalue calculated using the standard Picard iteration by 0.75 pcm. Both versions of the preconditioned case converged to an eigenvalue that differed by 1.12 pcm from the Picard eigenvalue. The pin power differences for all three cases were comparable and were all reasonably close to the Picard case. All three JFNK implementations converged in one less MOC iteration than the Picard iteration. However, despite requiring one fewer transport sweep, all JFNK implementations took significantly longer to converge.

The eigenvalue convergence behavior of all three JFNK implementations as well as the standard Picard method is shown in Figure 20. Similarly, the convergence of the 2-norm of the fission source residual as a function of MOC iteration is shown in Figure 21. Figure 20 shows that the eigenvalue convergence of the unpreconditioned JFNK case is more oscillatory than those for the other three cases. Both preconditioned cases follow the same trend, which follows closely with the Picard eigenvalue convergence until about the 7^{th} MOC iteration, after which the JFNK methods converge faster. As seen in Figure 21, all three coupled JFNK implementations follow the same general fission source convergence. All three of these JFNK implementations fall below the $5x10^{-5}$ fission source convergence criteria one MOC iteration sooner than the Picard method.

4.6.4 CMFD-Coupling Comparison

As discussed in Section 3.3, two methods for controlling the CMFD-Coupling iteration scheme were developed. A number of different implementations were tried and their results are shown in Table 13.

Table 13: Various CMFD-Coupling results for the 3D 7x7 fuel assembly problem

Case	Eigenvalue	MOC Its.	PP RMS	PP Max	CMFD Solves	Runtime (s)
Picard	1.1000484	10	-	-	10	44.79
CMFD-2	1.1000486	9	0.006%	0.010%	17	61.13
CMFD-3	1.1000485	9	0.007%	0.012%	25	71.54
CMFD-10	1.1000485	10	0.012%	0.020%	91	128.86
$\Delta T<0.1$	1.1000489	10	0.002%	0.003%	29	76.17
$\Delta T<0.001$	1.1000485	10	0.012%	0.020%	83	122.41

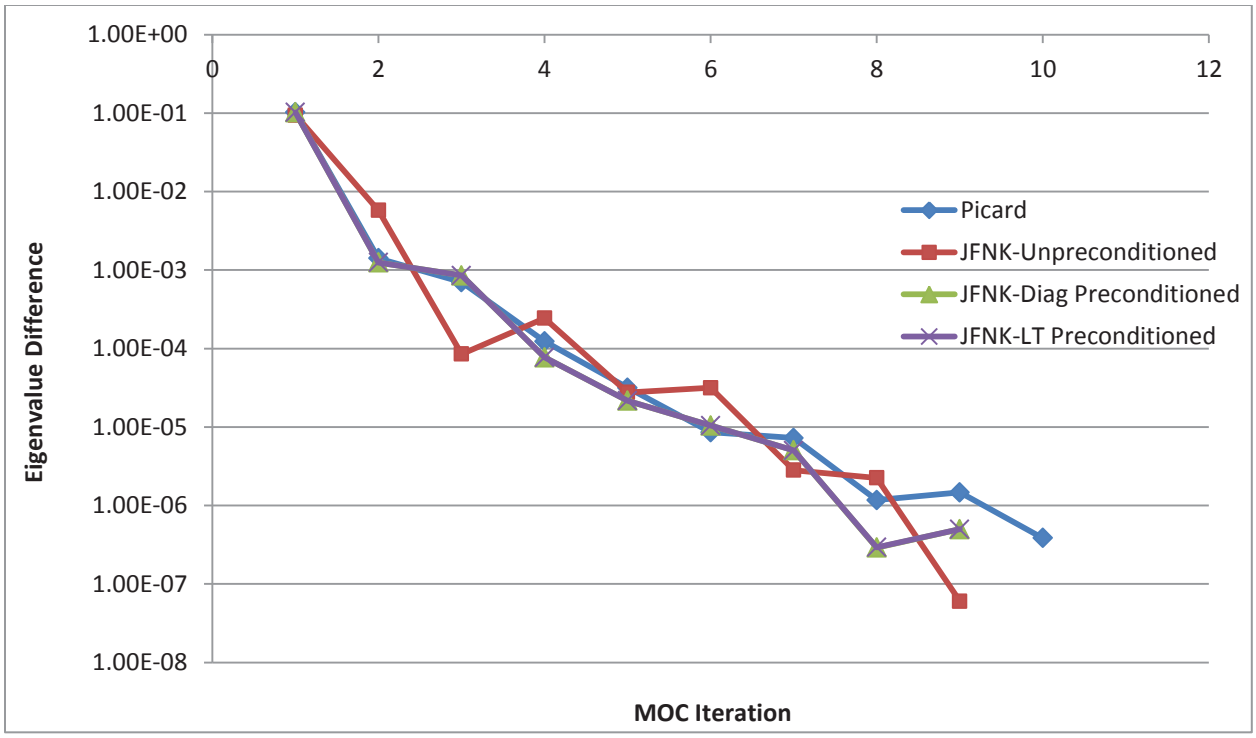

Figure 20. Comparison of the eigenvalue differences for each coupled JFNK implementation applied to the 3D 7x7 fuel assembly problem

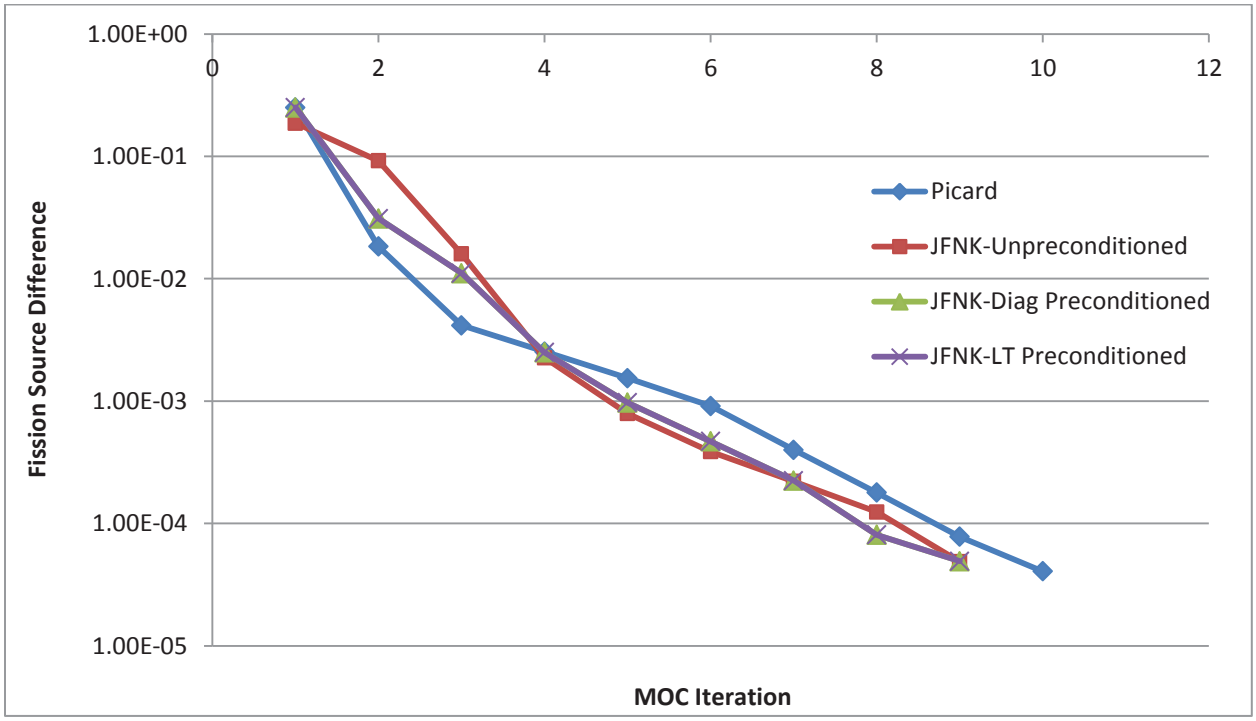

Figure 21. Comparison of the fission source differences for each coupled JFNK implementation applied to the 3D 7x7 fuel assembly problem

All of the CMFD-Coupling approaches shown in Table 13 converge to within 0.1 pcm of the eigenvalue calculated using the Picard iteration. Similarly, all of the CMFD-Coupling approaches converged the pin powers remarkably well compared to those calculated using the standard Picard iteration. The CMFD-2 and CMFD-3 methods were the only cases to offer any reduction in the number of total MOC sweeps, while the others offered no change. Despite this fact, none of the CMFD-Coupling approached offered any speedup in terms of runtime.

The convergence of the eigenvalue for these CMFD-Coupling cases is shown as a function of MOC iteration count in Figure 22. The 2-norm of the fission source residual as a function of MOC iteration count is shown in Figure 23. As seen in Figure 22, the eigenvalue residuals for all CMFD-Coupling strategies implemented are lower than that for the Picard method at convergence. Although only the CMFD-2 and CMFD-3 cases were completed after nine MOC sweeps, all CMFD-Coupling strategies have eigenvalue residuals below the 1E-6 convergence criteria in the 9th MOC iteration. In Figure 23, the fission source residuals for all CMFD-Coupling implementations are universally lower than those for the Picard iteration scheme.

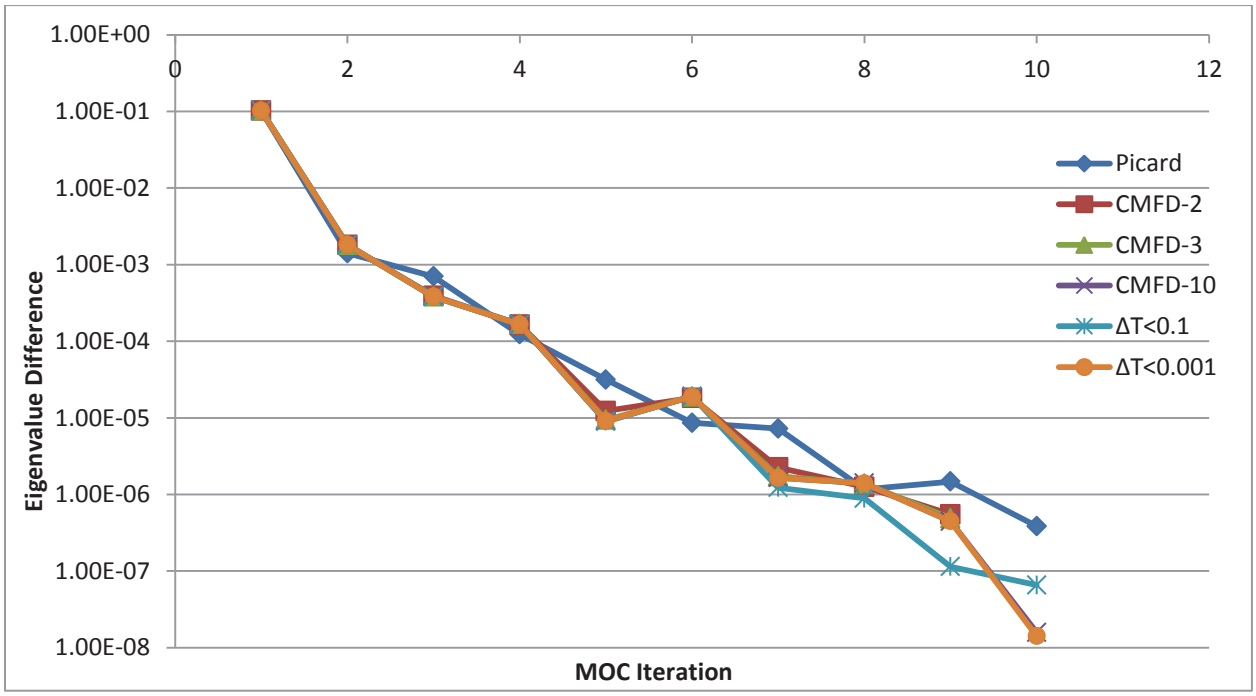

Figure 22. Comparison of the eigenvalue differences for each CMFD-Coupling implementation applied to the 3D 7x7 fuel assembly problem

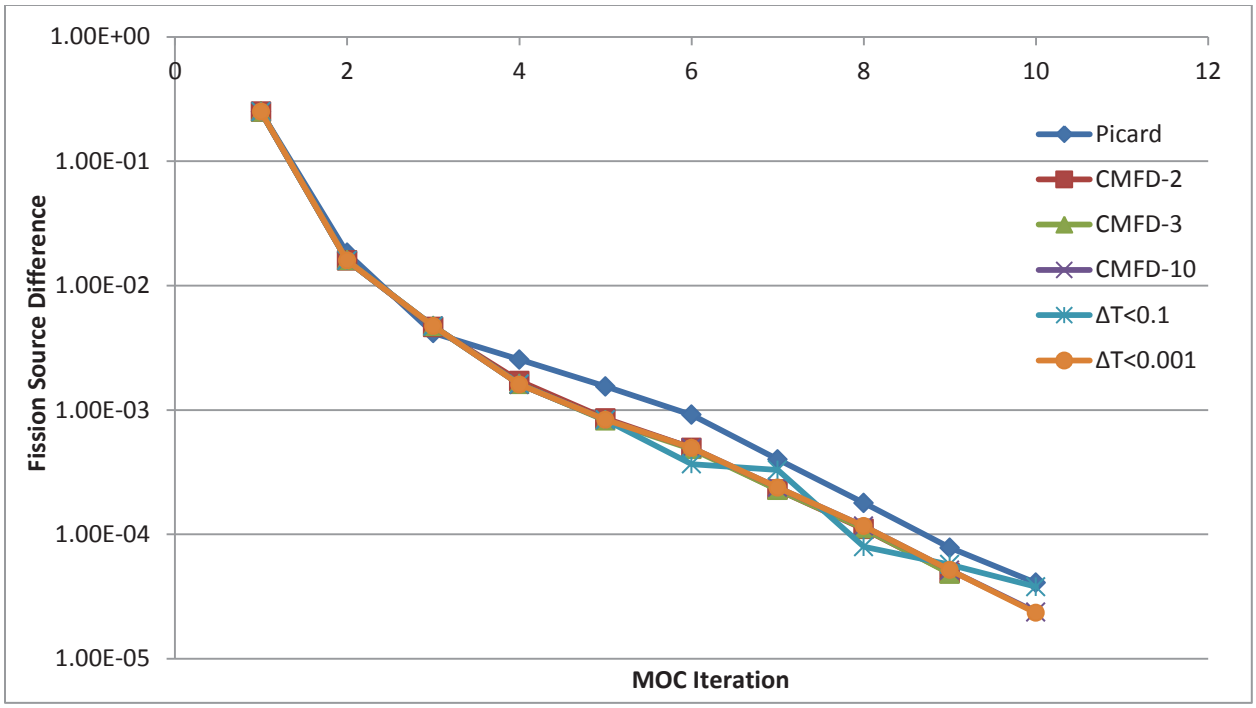

Figure 23. Comparison of the fission source differences for each CMFD-Coupling implementation applied to the 3D 7x7 fuel assembly problem

4.7 Summary

Both a JFNK based nonlinear solver and a new iteration strategy called CMFD-Coupling were successfully implemented and applied to various reactor problems. For the 1D, one-group homogeneous slab problem, neither coupling methodology performed better than the default Picard iteration, with JFNK performing the worst. When used as an eigenvalue solver, it was confirmed that preconditioning is a requirement for the viability of JFNK. However, even with appropriate preconditioning, it offered no improvement in reducing the number of transport sweeps required for convergence for any of the problems tested. In fact, the JFNK eigenvalue solver performed slower than the default power iteration for every test case.

Similarly, when used as a TH coupling technique, JFNK was consistently slower than the Picard iteration. Again, preconditioners were found to be a crucial part of the implementation of JFNK as a coupled solver. While both the diagonal and the lower triangular preconditioners were effective in reducing the total number of linear GMRES iterations required for convergence, neither preconditioner preformed significantly better than the other. In the larger 3D fuel pin and

3D 7x7 fuel assembly problems, the eigenvalue and fission source residuals were consistently lower for the coupled JFNK cases than those for the Picard iteration.

The five CMFD-Coupling techniques implemented were also consistently slower than the default Picard iteration. However, when compared to the JFNK coupling implementation, all CMFD-Coupling strategies provided more accurate eigenvalue and pin power estimates. Additionally the runtimes for the CMFD-Coupling cases were generally faster than those for the JFNK coupled cases.

Therefore, based on these results, JFNK will not be considered for the remainder of this document. However, because of its superior performance and its ability to run in parallel, CMFD-Coupling will be further investigated as a coupling strategy when applied to larger problems.

5. CMFD-Coupling Investigations

Since JFNK is no longer being considered as either a multiphysics coupling method or as an eigenvalue solver, larger problems are now investigated. Therefore, CMFD-Coupling is further evaluated as a multiphysics coupling technique. Because it was implemented in parallel, CMFD-Coupling was used on a series of large scale full core problems. First, Watts Bar Unit 1 Cycle 1 is investigated with various forms of feedback enabled and finally Watts Bar Unit 1 Cycle 2 is tested using the CMFD-Coupling method.

5.1 3D Full Core Problem – Cycle 1

In order to test CMFD-Coupling on a realistic problem that is of interest to reactor engineers, a large full core problem was modeled in 3D. This problem will demonstrate the performance for realistic problems that would be expected for real-world applications of the CMFD-Coupling method.

5.1.1 Problem Description

The 3D full core problem modeled is CASL VERA Core Physics Benchmark Problem 7 [1], which is a representation of Watts Bar Unit 1 Cycle 1. The problem consists of 193 Westinghouse 17x17-type fuel assemblies at BOL with HFP conditions. Control rod banks, instruments, and radial support structures are included in the model. The core layout is shown in Figure 24. The core consists of three different fuel enrichments: 2.11%, 2.619%, and 3.10%. Some fuel assemblies contain burnable poison rods in the form of Pyrex rods of borosilicate glass. The layout of the fuel assemblies, burnable poisons, and control rods are shown in Figure 25 in quarter symmetry.

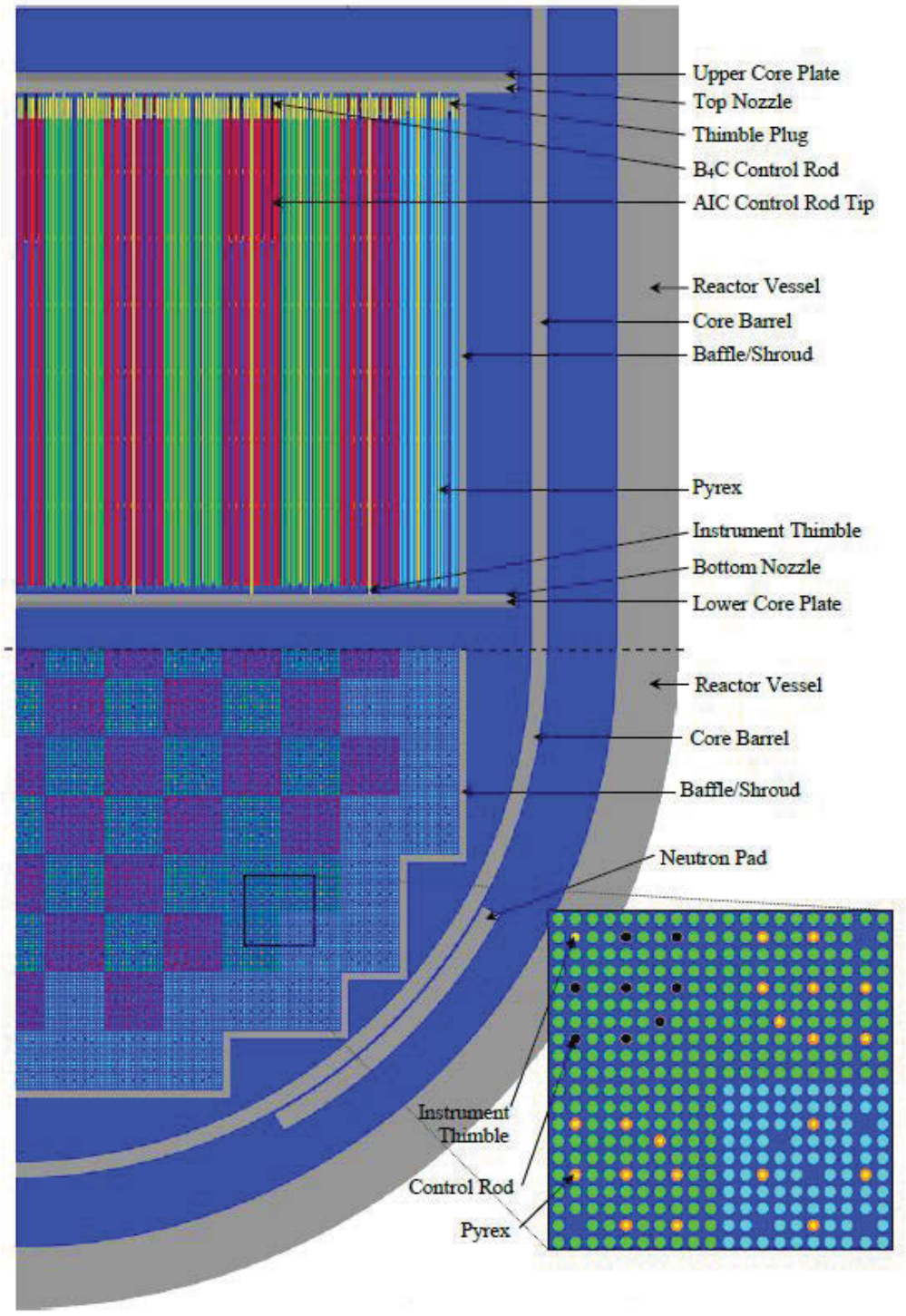

Figure 24. Core geometry of VERA Core Physics Progression Problem 7 **[1]**

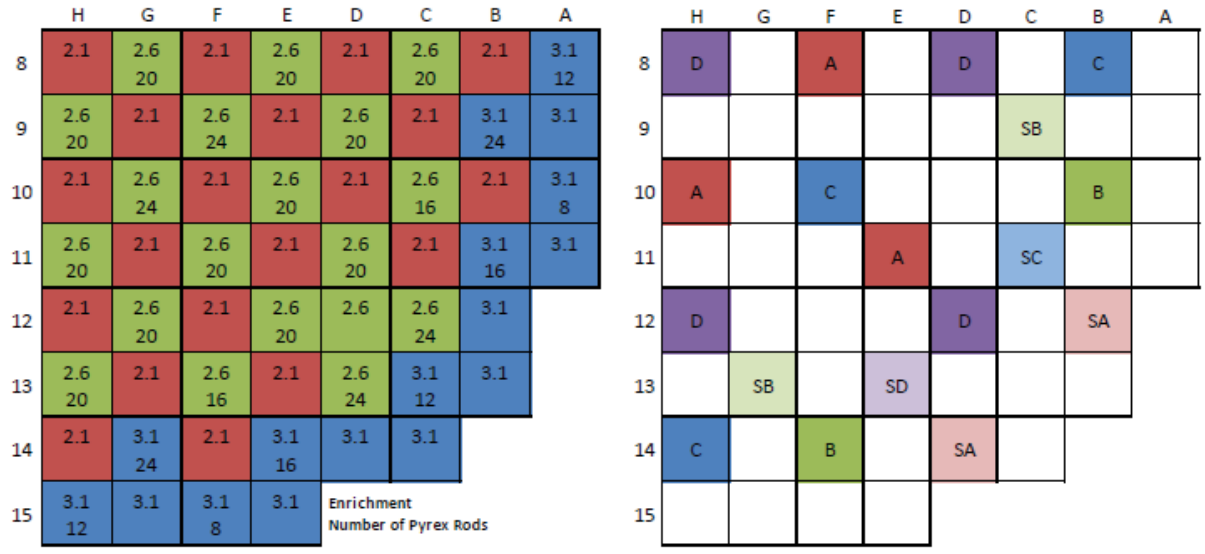

Figure 25. Fuel loading, poison, and control bank layout in quarter symmetry **[1]**

5.1.2 TH Feedback

While the CASL VERA Core Physics Benchmark Problem 7 has equilibrium xenon and critical boron search feedback effects enabled, the same problem geometry was examined with only TH feedback enabled. Four different CMFD-Coupling methodologies were implemented in addition the standard Picard iteration: CMFD-2, CMFD-3, $\Delta T < 0.1$, and $\Delta T < 0.001$ and their results are shown in Table 14.

Table 14: CMFD-Coupling results for the 3D full core problem with TH feedback

Method	Eigenvalue	MOC Iterations	CMFD Iterations	Runtime (h:mm:ss)
Picard	1.1294727	12	12	0:48:55
CMFD-2	1.1294568	19	37	1:32:08
CMFD-3	1.1294482	15	43	1:30:29
ΔT<0.1	1.1294547	12	59	1:47:53
ΔT<0.001	1.1294476	13	264	4:59:17

While the CMFD-2 case was intended to minimize the additional amount of work added to the Picard iteration, it took seven more MOC iterations to converge. Adding one additional CMFD-TH loop in the CMFD-3 case helped reduce the number of transport sweeps relative to the CMFD-2 case but still required a larger number than Picard. While $\Delta T < 0.1$ was the only case

to match the number of Picard MOC iterations, it required an additional hour of compute time. The $\Delta T < 0.001$ case added a large amount of extra work and compute time with no reduction in the overall number of transport sweeps. The convergence of the eigenvalue for each of these CMFD-Coupling cases is shown as a function of MOC iteration count in Figure 26. The 2-norm of the fission source residual as a function of MOC iteration count is shown in Figure 27.

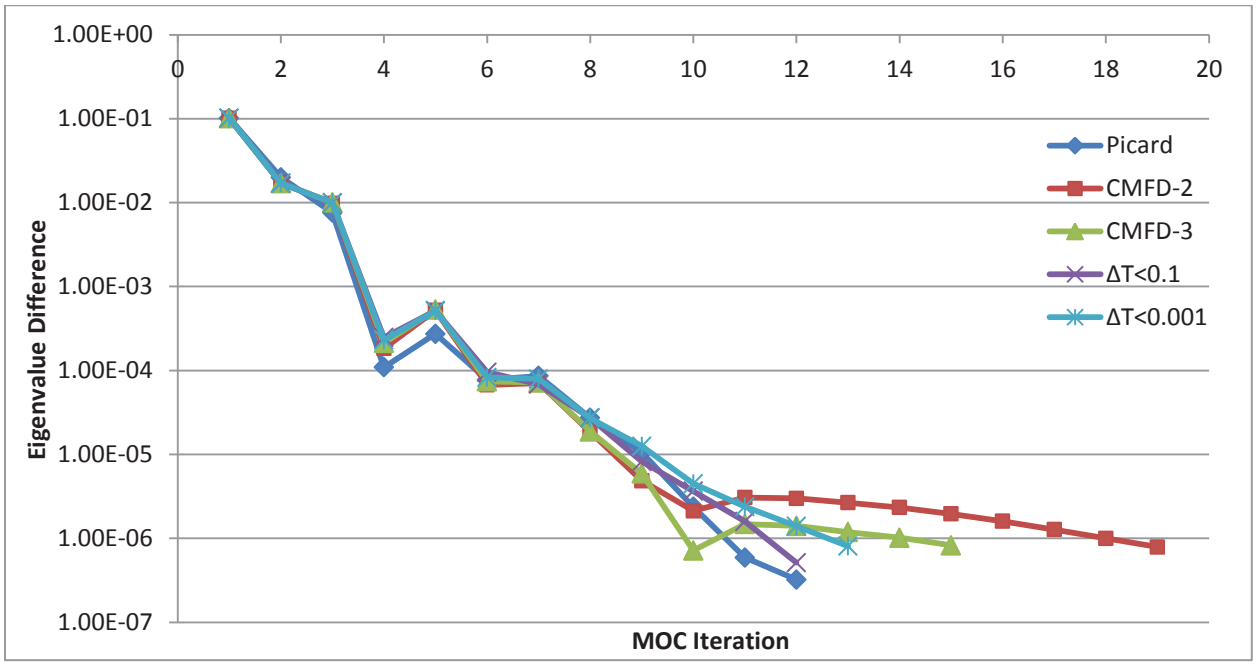

Figure 26. Comparison of the eigenvalue differences for each CMFD-Coupling implementation applied to the 3D full core problem

Figure 26 shows that the eigenvalue residual of all cases follow roughly the same pattern until about the 10th MOC iteration, after which they begin to diverge from one another. At convergence, the Picard iteration has the lowest eigenvalue residual. When examining the fission course convergence in Figure 27, it is seen that both $\Delta T < tol$ cases have better convergence rates than Picard. Despite this fact, these cases do not perform better than the Picard iteration in terms of runtime or MOC iteration reduction.

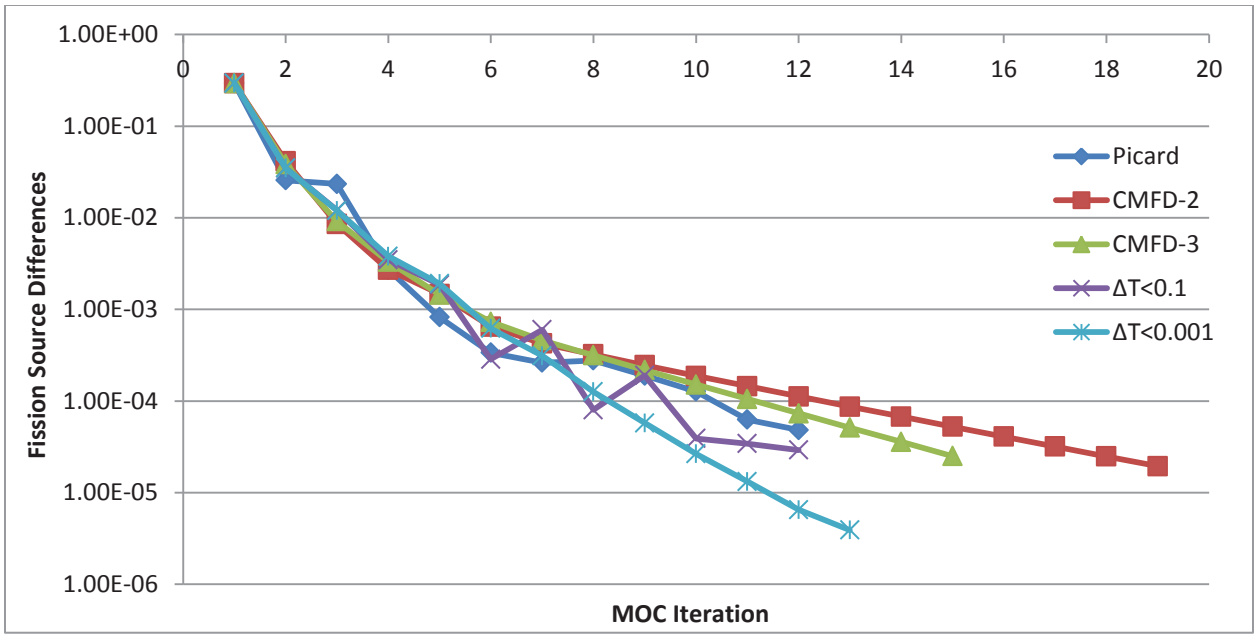

Figure 27. Comparison of the fission source differences for each CMFD-Coupling implementation applied to the 3D full core problem

If this same 3D full core problem is assessed with no TH feedback turned on, it takes Picard 13 MOC iterations to converge in just less than 38 minutes. This means that, while TH feedback adds an additional 10 minutes to the runtime, it actually reduces the total number of MOC sweeps by one iteration. This is similar to what was seen in Chapter 4 where the TH solver acts like a damping factor, easing the convergence of the problem. Therefore, it can be concluded that the addition of TH feedback to this problem adds no significant computational burden. As a result, tighter coupling of CMFD with the TH solver has no benefit, as seen in Table 14. For that reason, other forms of feedback were considered in addition to TH: xenon feedback and critical boron search. These other sources of feedback were considered because they are commonly included in modern nuclear reactor analysis.

5.1.3 Equilibrium Xenon

The same 3D full core problem was retested with both TH and equilibrium xenon feedback enabled. The feedback effects from the xenon calculation are applied to the problem in the same fashion that TH feedback effects are applied: both feedback calculations are performed and their results are used to update the material cross sections of the problem. The results of modeling this problem with CMFD-Coupling are shown in Table 15.

Table 15: CMFD-Coupling results for the 3D full core problem with TH and equilibrium xenon feedback

Method	Eigenvalue	MOC Iterations	CMFD Iterations	Runtime (h:mm:ss)
Picard	1.0931780	16	16	0:58:29
CMFD-2	1.0931718	12	23	1:13:34
CMFD-3	1.0931635	12	34	1:24:22
ΔT<0.1	1.0931661	13	50	1:44:15
ΔT<0.001	1.0931638	13	204	4:24:58

As seen in Table 15, all of the CMFD-Coupling approaches reduced the total number of MOC iterations required for convergence. However, despite this fact, none of these methods have a faster runtime than the Picard iteration. The best performer was the CMFD-2 case that reduced the number of MOC iterations by four while increasing the number of CMFD iterations by only seven. Yet it still required an additional 15 minutes to converge. The convergence of the eigenvalue for these CMFD-Coupling cases is shown as a function of MOC iteration in Figure 28. The 2-norm of the fission source residual as a function of MOC iteration count is shown in Figure 29.

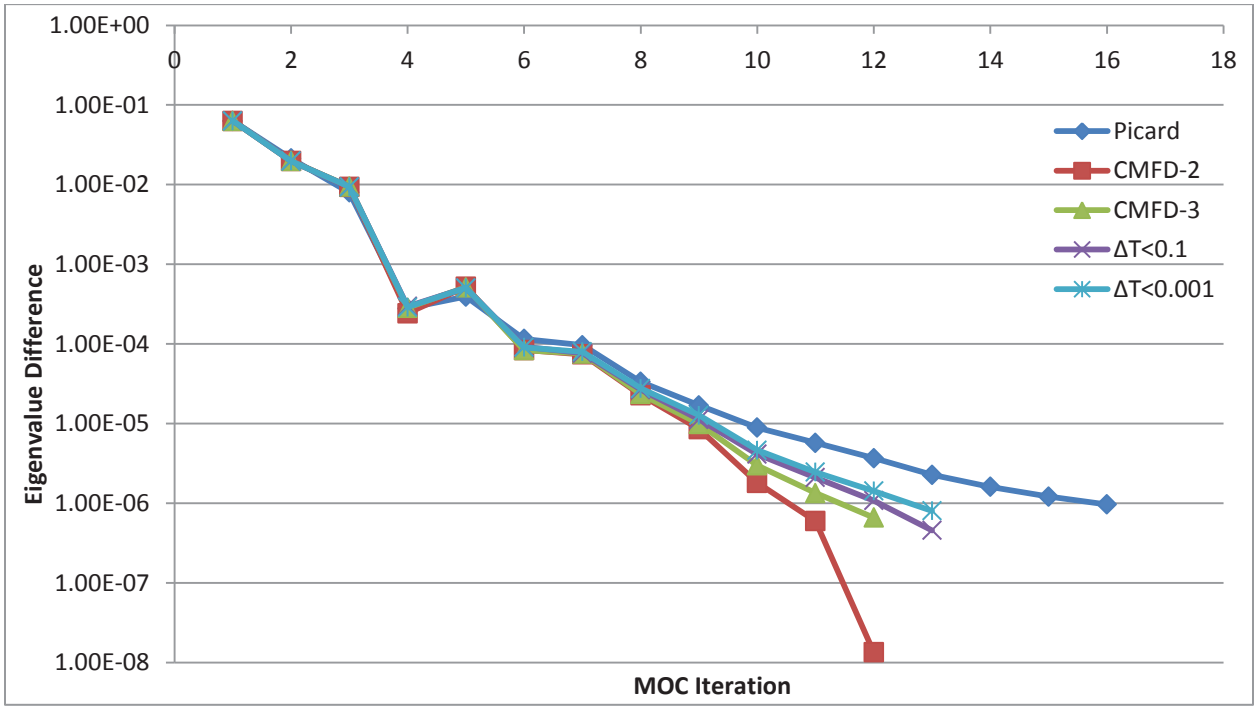

Figure 28. Comparison of the eigenvalue differences for each CMFD-Coupling implementation applied to the 3D full core problem with equilibrium xenon feedback enabled

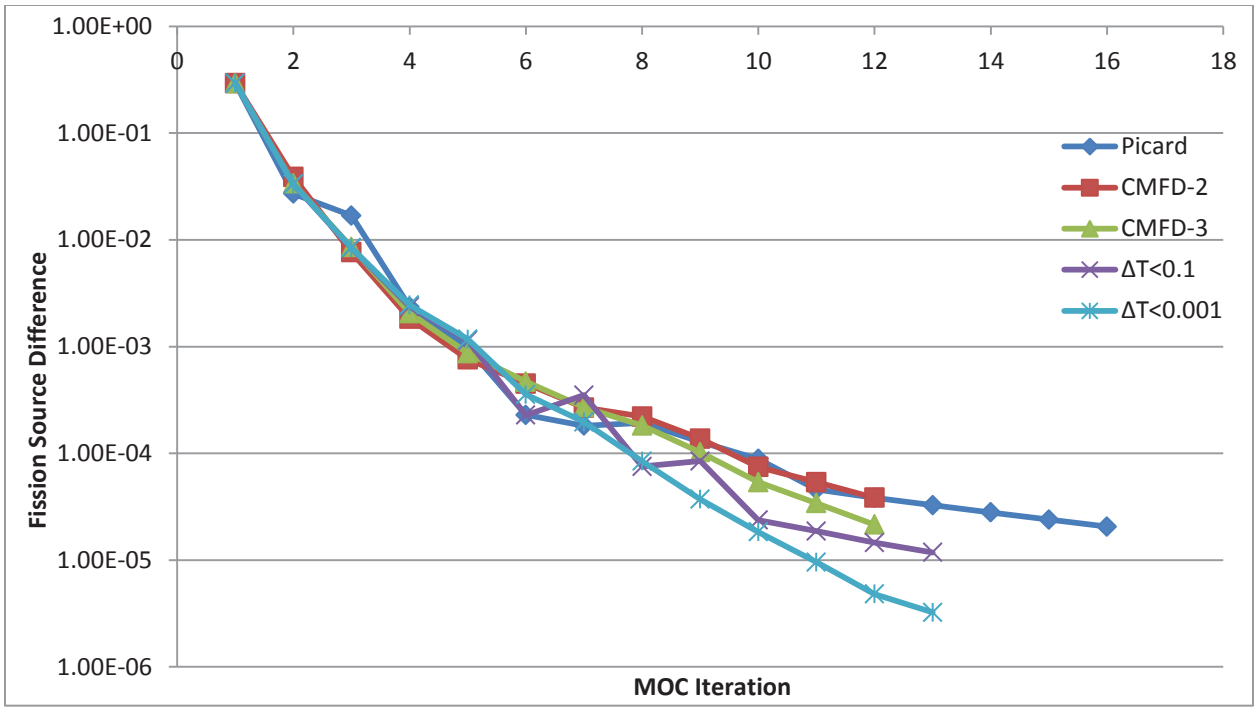

Figure 29. Comparison of the fission source differences for each CMFD-Coupling implementation applied to the 3D full core problem with equilibrium xenon feedback enabled

Figure 28 shows the eigenvalue residual for the CMFD-2 case converging much faster than the others once the solution is close to the convergence criteria. In addition, all CMFD-Coupling cases converge the eigenvalue faster than the standard Picard iteration. When looking at Figure 29 as well, it can be seen that all CMFD-Cases converge the fission source either as good as, or better than, the Picard iteration. The $\Delta T < 0.001$ case converges the fission source much faster than the other cases; however this does not translate into fewer transport sweeps. This is because these cases are eigenvalue limited, and the problem must continue until the eigenvalue difference reaches the convergence criteria. Therefore, even though the CMFD-2 case converges the fission source the same as Picard, it converges the eigenvalue much faster resulting in the greatest reduction in MOC iterations.

5.1.4 Critical Boron Search

Unlike the eigenvalue problems performed earlier, critical boron search calculations adjust the isotopics of the problem in order to converge to a predetermined eigenvalue and is applied as a feedback effect. Therefore, it is possible to couple the CMFD solver to the critical boron

calculations more tightly using CMFD-Coupling. The results of modeling this problem with both TH feedback and a critical boron search are shown in Table 16.

Table 16: CMFD-Coupling results for the 3D full core problem with TH and critical boron search feedback

Method	Boron Concentration	MOC Iterations	CMFD Iterations	Runtime (h:mm:ss)
Picard	1147.07	14	14	1:08:23
CMFD-2	1147.12	12	23	1:12:52
CMFD-3	1147.11	11	31	1:15:59
ΔT<0.1	1147.05	15	57	1:48:41
ΔT<0.001	1147.1	9	245	4:36:52

The addition of CMFD-Coupling reduces the number of MOC iterations for every case except the $\Delta T < 0.1$ case. The $\Delta T < 0.001$ case required the fewest number of MOC iterations to converge but took almost 3.5 hours longer than the Picard iteration. This is due to the significant increase in the number of CMFD iterations performed while converging the norm of the differences in temperature to less than 0.001 K. Similar to the problem with xenon feedback enabled, CMFD-2 performed the best in terms of overall runtime, but was still slightly slower than Picard.

The eigenvalue convergence for these CMFD-Coupling cases is shown as a function of MOC iteration in Figure 30. The 2-norm of the fission source residual as a function of MOC iteration count is shown in Figure 31. As shown in Figure 30, every CMFD-Coupling implementation except for the $\Delta T < 0.1$ case greatly improved upon the eigenvalue convergence when compared to the Picard iteration. The behavior of the $\Delta T < 0.1$ case can be explained by the fact that, for the first four MOC iterations, there were multiple CMFD iterations for every MOC iteration. After that point, the temperature differences between successive MOC sweeps were less than 0.1 K so a transport solve was performed without any CMFD-Coupling iterations in between. This is why the eigenvalue residuals begin to converge similarly to Picard after this point. In addition, the CMFD-2 and CMFD-3 cases smoothed out the convergence with respect to Picard. In Figure 31 it is seen that all cases had better fission source convergence than the standard

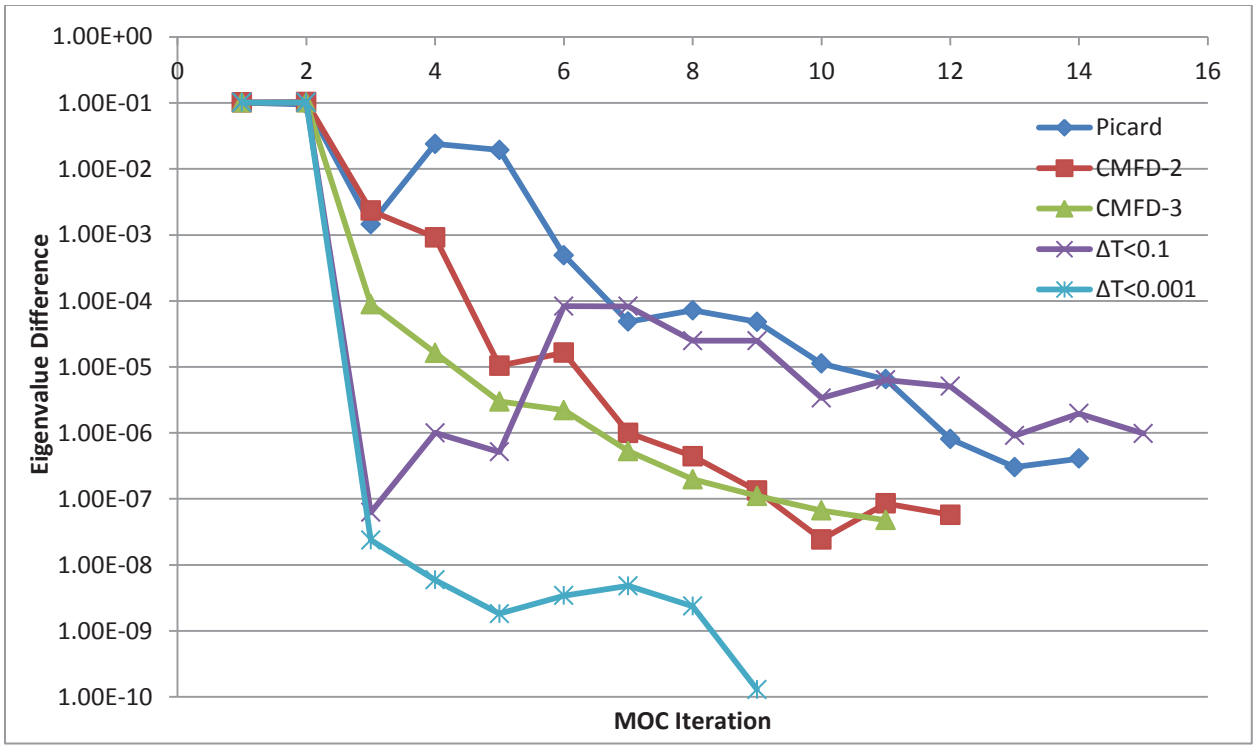

Figure 30. Comparison of the eigenvalue differences for each CMFD-Coupling implementation applied to the 3D full core problem with critical boron search enabled

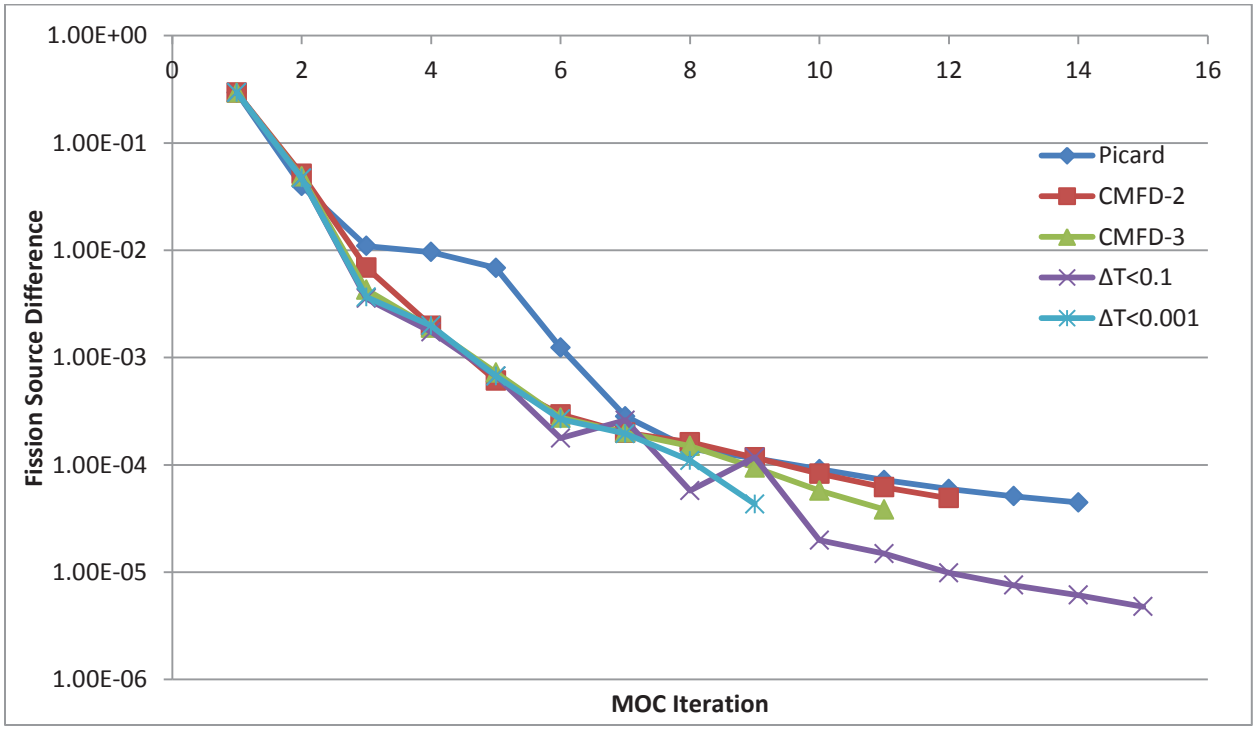

Figure 31. Comparison of the fission source differences for each CMFD-Coupling implementation applied to the 3D full core problem with critical boron search enabled

Picard iteration. Most of these cases were fission source limited, and so this improved convergence is why there is a reduction in the total number of MOC iterations as seen in Table 16.

5.1.5 3D Full Core with All Feedback Enabled

Finally, the CASL VERA Core Physics Benchmark Problem 7 was tested in full: with TH feedback, equilibrium xenon, and critical boron search enabled. This is typical of the simulations performed for reactor operation. However, unlike the previous problems, the $\Delta T < tol$ cases are ignored due to the fact that they were consistently ranked last in terms of runtime for completion. In their place an additional CMFD-5 case was tested. The results of these CMFD-Coupling implementations are shown in Table 17.

Table 17: CMFD-Coupling results for the 3D full core problem with TH feedback, equilibrium xenon, and critical boron search enabled

Method	Boron Concentration	MOC Iterations	CMFD Iterations	Runtime (h:mm:ss)
Picard	848.12	14	14	0:55:31
CMFD-2	848.15	10	19	1:11:26
CMFD-3	848.15	10	28	1:27:31
CMFD-5	848.10	9	41	1:44:14

All CMFD-Coupling techniques were shown to reduce the total number of MOC iterations. While the CMFD-5 case reduced the number of transport sweeps the most, it took the longest runtime to complete. Like the previous problems, the CMFD-2 case preformed the best in terms of runtime. However, it still took significantly longer to converge when compared to the Picard iteration. This is due to the fact that, while MOC was performed fewer times, the increased number of iterations through the CMFD-Coupling loop offset that gain. The convergence of the eigenvalue differences for these cases is shown in Figure 32. Similarly, the convergence of the fission source differences is shown in Figure 33. Figure 32 shows that the eigenvalue residual for all CMFD-Coupling cases is greatly reduced when compared to that for the Picard iteration. However the convergence of the fission source shows little to no improvement. It should be noted that while the Picard iteration is eigenvalue limited, all of the CMFD-Coupled cases are fission source limited due to the increased convergence of the eigenvalue.

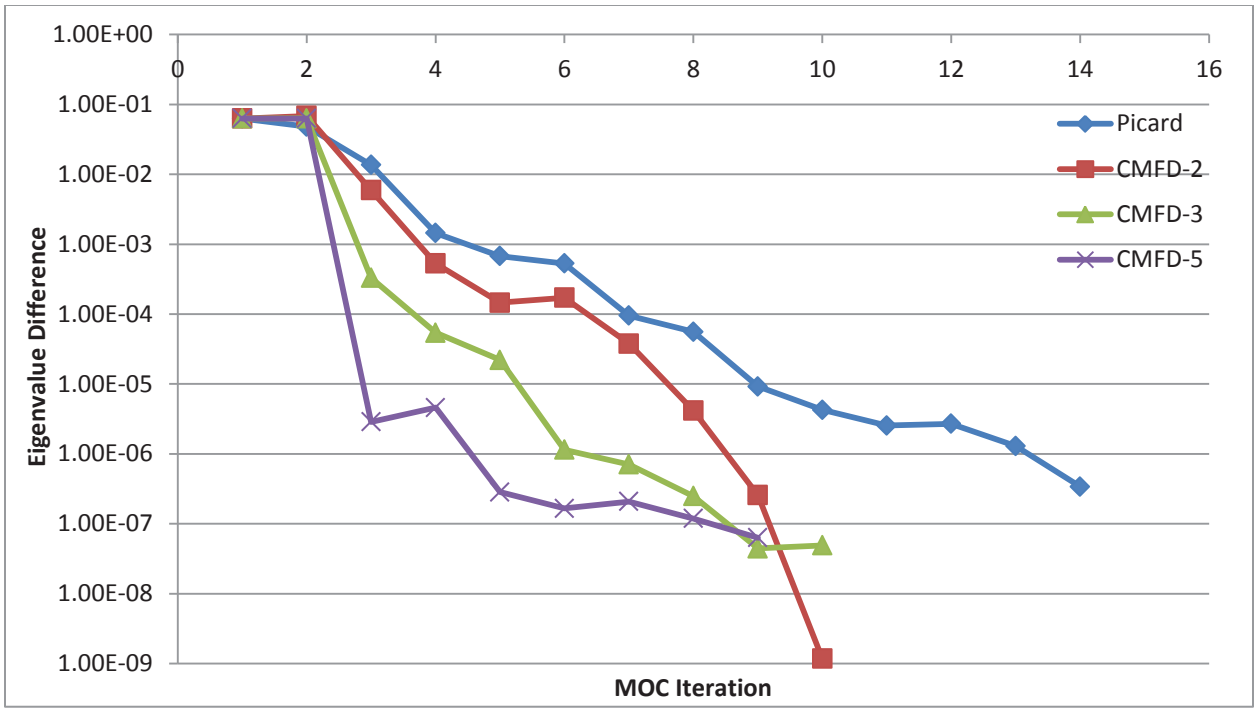

Figure 32. Comparison of the eigenvalue differences for each CMFD-Coupling implementation applied to the 3D full core problem with TH feedback, equilibrium xenon, and critical boron search enabled

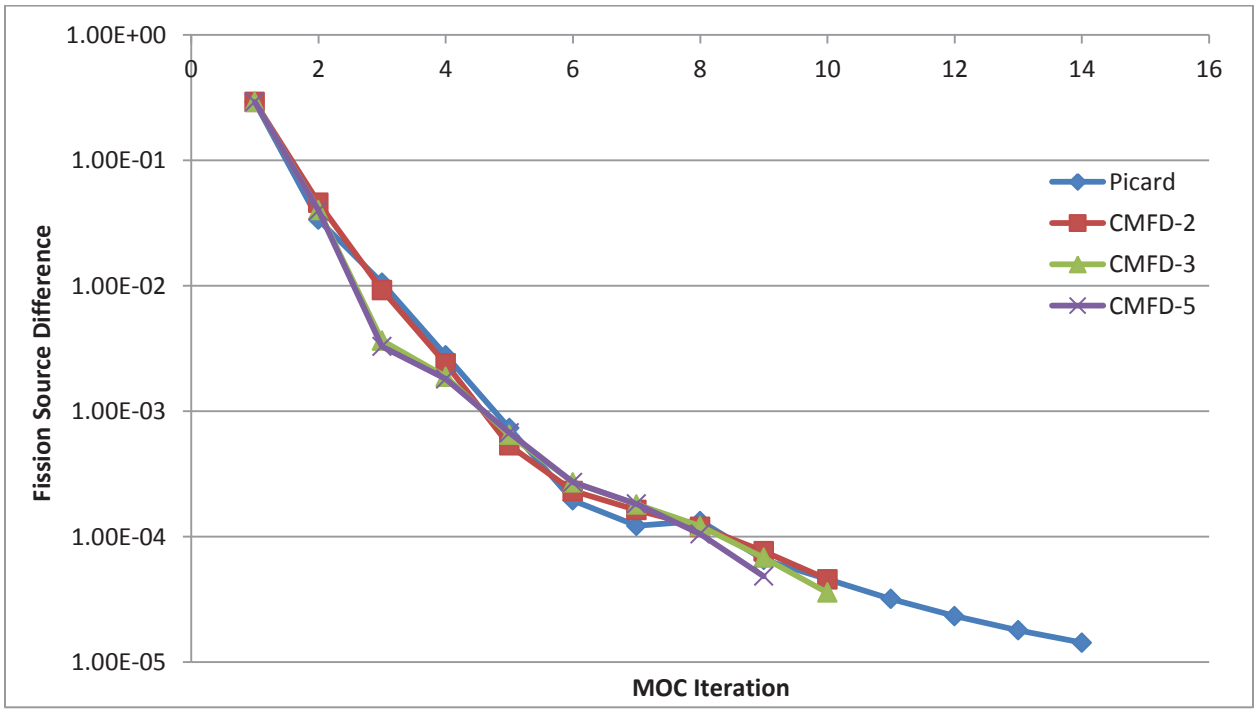

Figure 33. Comparison of the fission source differences for each CMFD-Coupling implementation applied to the 3D full core problem with TH feedback, equilibrium xenon, and critical boron search enabled

5.2 3D Full Core Problem – Cycle 2

The CASL VERA Core Physics Benchmark Problem 10 [1] was modeled to examine the potential improvements using CMFD-Coupling. This problem is geometrically the same as that described in Section 5.1 but at the start of the second cycle. This full core problem was depleted over the course of an 18-month fuel cycle. Once this depletion is completed, approximately one third of the fuel was removed from the problem and replaced with fresh fuel. The remaining fuel elements were rearranged to new locations to begin cycle two. This problem is modeled at HZP and therefore neither TH nor xenon feedback are enabled. However a critical boron search is still performed. Cycle two was examined because the current Picard methodology for solving this problem introduces large instabilities during convergence likely due to isotopic oscillations. The results for the Picard iteration scheme along with three CMFD-Coupling implementations are shown in Table 18.

Table 18: CMFD-Coupling results for Cycle 2 of the 3D full core problem with critical boron search enabled

Method	Boron Concentration	MOC Iterations	CMFD Iterations	Runtime (h:mm:ss)
Picard	1436.69	51	51	3:20:01
CMFD-2	1436.69	35	69	3:44:19
CMFD-3	1436.69	28	82	4:03:32
CMFD-5	1436.69	22	106	4:44:59

The instability of the Picard iteration is evidenced by the large number of transport sweeps required for convergence. All three CMFD-Coupling methods tested helped to reduce the total number of MOC iterations. However, this was accomplished at the cost of adding more CMFD iterations. Therefore, despite the significant reduction in the number of MOC sweeps, all CMFD-Coupling cases had longer runtimes to convergence. Like in the previous problems, CMFD-2 was the fastest case tested. A plot showing the convergence of the eigenvalue differences for these cases is shown in Figure 34. Likewise, a plot showing the convergence of the fission source differences is shown in Figure 35.

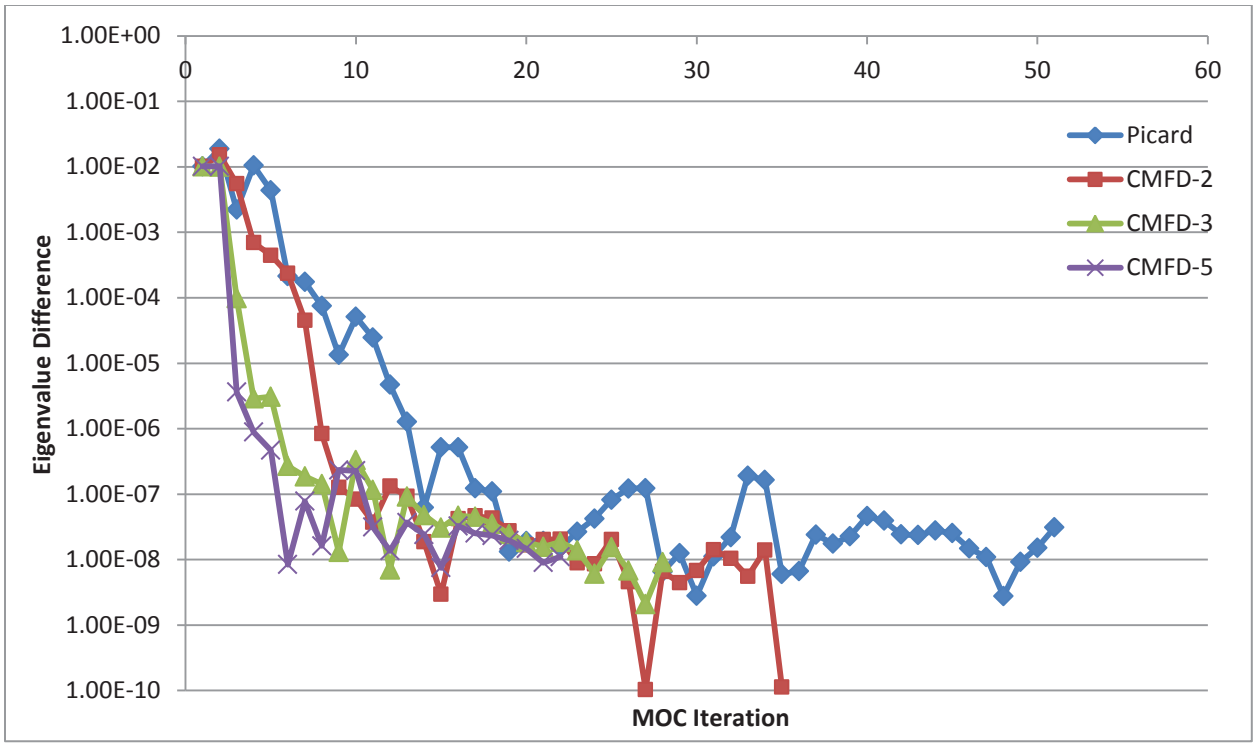

Figure 34. Comparison of the eigenvalue differences for each CMFD-Coupling implementation applied to Cycle 2 of the 3D full core problem with critical boron search enabled

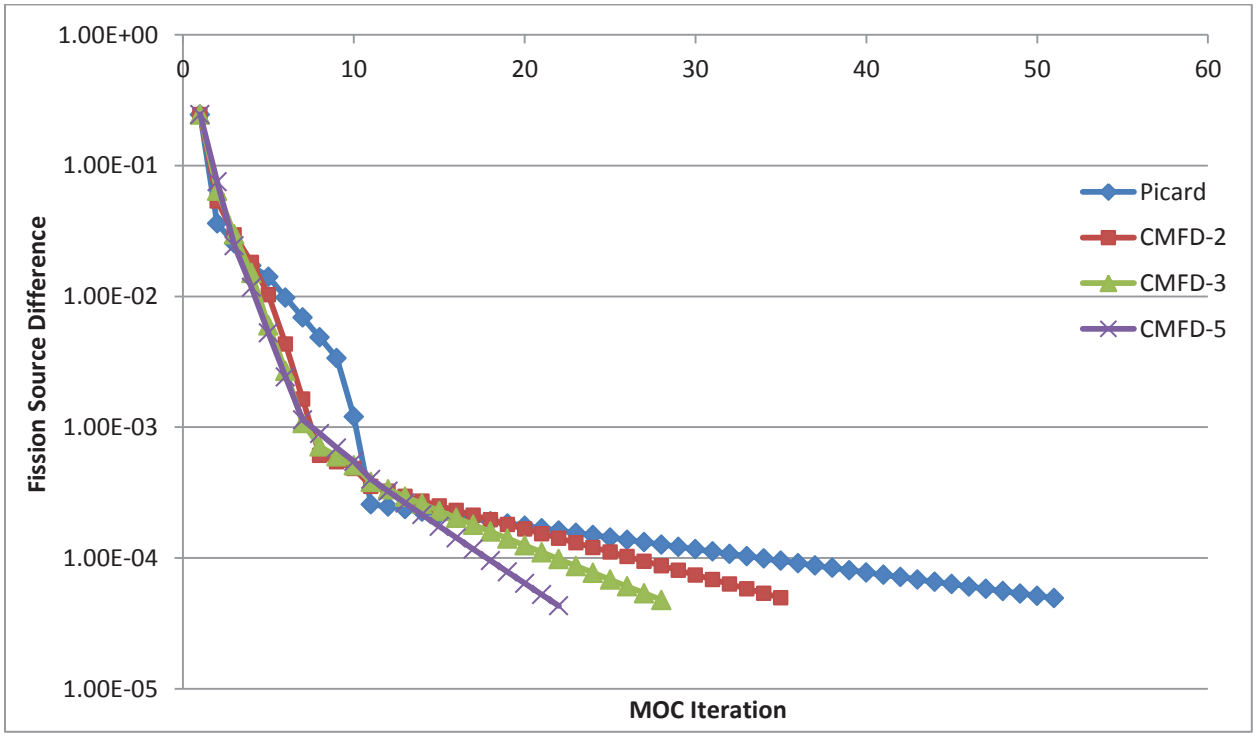

Figure 35. Comparison of the fission source differences for each CMFD-Coupling implementation applied to Cycle 2 of the 3D full core problem with critical boron search enabled

While Figure 34 shows large instabilities in the eigenvalue residual, it should be noted that these oscillations only begin to happen once the residual is below the $1x10^{-6}$ convergence criteria. Looking at Figure 35 it is seen that all cases are fission source limited. Every CMFD-Coupling case had fission source residuals that were improvements when compared to the Picard iteration scheme, but, as mentioned before, took significantly longer.

5.3 Summary

Various CMFD-Coupling strategies were successfully implemented and applied to 3D full core reactor problems. For the core problem at BOL, different sets of feedback were coupled to the neutronics problem to assess the performance of the CMFD-Coupling methods. When only coupled to the TH solver, all methods tested greatly increased the overall runtime with no reduction in the total number of MOC iterations. When equilibrium xenon feedback was included in addition to the TH solver, CMFD-Coupling offered a transport sweep reduction of 25%. However, despite this reduction in the number of transport sweeps, the best performing method, CMFD-2, led to a 25% increase in runtime. The equilibrium xenon feedback solver was then replaced with the critical boron search solver. Again, the best performer was CMFD-2 with a 17% reduction in MOC iterations but with only a 6% increase in the overall runtime. Finally, all three sets of feedback were coupled to the low order CMFD equations. Again the best performing method was CMFD-2. It offered a 29% reduction in the total number of transport sweeps required while increasing the total runtime by 29%. When applied to the same full core geometry at the start of Cycle 2, CMFD-Coupling reduces the total number of transport sweeps for all strategies tested. The best performing case was CMFD-2, with a 31% reduction in MOC iterations with only a 12% increase in the overall runtime.

Even though CMFD-Coupling reduced the number of MOC iterations in most of these cases, the overall increase in the number of CMFD iterations resulted in the increase in runtime.

6. Conclusions

Two different multiphysics coupling methods were investigated that incorporate coupling to the low-order CMFD equations: JFNK and a new iteration strategy called CMFD-Coupling. JFNK incorporates the nonlinear Newton's method along with a finite difference approximation to approximate the action of the Jacobian on a vector. This method combines multiple sets of physics in the same solution vector and solves the low-order coupled problem simultaneously. Conversely, the CMFD-Coupling method iterates between the CMFD solver and the multiphysics solvers multiple times before entering into a transport solve.

Initially, JFNK was implemented as an eigenvalue solver, both preconditioned and unpreconditioned. It was found that, for JFNK to be computationally competitive with other methods, preconditioning was a necessity. However, regardless of preconditioning, JFNK offered no improvement in reducing the number of transport sweeps required for convergence for any of the problems tested. In fact, the JFNK eigenvalue solver performed slower than the default power iteration for every test case due to the extra computational requirements of the method.

Before being applied to more sophisticated problems, JFNK was implemented as a coupled neutronics-TH solver for an infinite homogeneous medium problem. The MOC-CMFD neutronics solver was replaced with a cross section table lookup. The cross section table was generated from a series of 2D pin cell calculations at varying fuel temperatures. In order to maximize the effectiveness of JFNK on this problem, the cross sections within a given Newton step needed to be updated. This was done by calculating the cross section derivative with respect to temperature before each Newton step, and then using that value to linearly update the cross sections during each linear GMRES iteration. In addition to calculating the cross section derivatives fully, a series of three cross section derivative approximations were tested: one removed the temperature dependence, one removed the energy dependence, and one removed both, leading to a constant value. While all of these approximations performed worse than the fully calculated cross section derivative, they all performed better than the case in which the cross sections were not updated within each Newton step. This study showed that updating the

cross sections within JFNK, even approximately, is better than having them remain constant, which is what is done using a Picard iteration scheme.

When used as a multiphysics coupling technique, JFNK and CMFD-Coupling, along with the standard Picard iteration, were applied to a series of simplified reactor problems with coupled TH feedback. A 1D one-group homogeneous slab, a 2D pin cell, a 2D lattice, a 3D fuel rod, and a 3D 7x7 assembly problem were investigated. Again it was realized that preconditioning is a crucial part of JFNKs implementation as a coupled solver. In all of the problems tested, JFNK was consistently slower than the Picard iteration and offered little to no improvement in the total number of transport solves required for convergence. However, in the larger 3D fuel pin and 3D 7x7 fuel assembly problems, the eigenvalue and fission source residuals were consistently lower for the coupled JFNK cases than those for the Picard iteration. When the same five problems were solved with CMFD-Coupling instead of JFNK, some improvements were seen. When compared to the JFNK coupling implementation, all CMFD-Coupling strategies provided more accurate eigenvalue and pin power estimates. Additionally, the runtimes for the CMFD-Coupling cases were generally faster than those for the JFNK coupled cases. However, the CMFD-Coupling runtimes were still slower than the default Picard iteration. This was due to the fact that the reduced number of transport sweeps was offset by an even larger increase in the number of CMFD iterations required for convergence.

Because of its inferior performance when compared to CMFD-Coupling, JFNK was abandoned and was never attempted on problems coupled to physics other than TH. However, CMFD-Coupling was applied in parallel to large 3D full core problems. These problems modeled both Cycle 1 and the start of Cycle 2 while coupling different combinations of TH, equilibrium xenon and critical boron search feedback. The best performing method on these problems was CMFD-2, in which the coupled physics and CMFD solvers are performed one additional time per transport sweep. For the problem at BOL, all methods tested greatly increased the overall runtime with no reduction in the total number of MOC iterations when coupled only to the TH solver. This increase in runtime was caused by the increase in the number of CMFD iterations performed along with no decrease in the number of MOC sweeps. However, when equilibrium xenon feedback was included in addition to the TH solver, CMFD-Coupling offered a transport

sweep reduction of 25%. Despite this reduction, there was still a 25% increase in runtime due to all of the additional CMFD iterations required. When the xenon solver was replaced with a critical boron search solver, CMFD-2 offered a 17% reduction in MOC iterations with a 6% increase in the overall runtime. When all three sets of feedback were applied at the same time, the use of CMFD-2 led to a 29% reduction in the total number of transport sweeps required while increasing the total runtime by 29%. The same problem at the start of Cycle 2 experiences xenon oscillations as the result of the isotopes buildup during the first cycle depletion. When CMFD-Coupling was applied to this problem with only critical boron feedback turned on, CMFD-2 led to a 31% reduction in MOC iterations with only a 12% increase in the overall runtime.

While neither JFNK nor CMFD-Coupling offered any improvement in terms of runtime, they do offer limited improvements in the convergence of the eigenvalue and fission source residual. Therefore, the tighter coupling of feedback to the low-order CMFD equations does offer some benefits. Additionally, the implementation of CMFD-Coupling shifts the computational burden from the transport solver to CMFD. If any new developments lead to the significant acceleration of CMFD, CMFD-Coupling may become computationally cheaper than the Picard iteration scheme in terms of runtime.

6.1 Future Work

6.1.1 Stronger TH Feedback

When coupled only to the TH solver, both JFNK and CMFD-Coupling offered no decrease in the transport iteration count. This was because the TH feedback in these cases was not very strong and did not add any MOC sweeps when compared to a non-coupled case. Therefore problems with stronger feedback effects should be further examined using both JFNK and CMFD-Coupling. For example, Boiling Water Reactor (BWR) cases in which the void feedback is very strong lead to slow coupled convergence. These cases could potentially benefit from tighter coupling with JFNK or CMFD-Coupling.

6.1.2 Transient Problems

All of the cases tested in this work were single state problems. During multistate time-dependent problems, such as transients, large feedback effects can occur over a short period of time requiring a large number of time-steps in order to accurately capture these feedback effects. Cases such as these might benefit from tighter multiphysics coupling using either JFNK or CMFD-Coupling and should be investigated.

6.1.3 Adaptive Coupling

For all of the problems in this work that were tested with CMFD-Coupling, the CMFD-Coupling was enabled for the entire duration of the problems execution. However, there may be problems that require tighter multiphysics coupling during only part of its execution. Therefore, methods should be developed for adaptively turning on the low-order coupling when it is needed, and avoiding the extra computational burden when it is not. The first step in this process would be identifying key parameters that would be used to trigger the low-order coupling.

6.1.4 CMFD Acceleration

The full core problems tested in this work benefited from the use of CMFD-Coupling in the form of reduced transport sweeps required for convergence. However, these problems also took longer to execute because of the larger number of CMFD iterations which offset the savings gained by the reduction in transport iterations. Essentially the computational burden was shifted from the transport solver to the low-order CMFD system. Therefore, if methods were developed for further accelerating the CMFD solution, the application of CMFD-Coupling might prove to be faster than the Picard iteration scheme. One possible method to be investigated is a multilevel CMFD scheme in which the low-order CMFD system is further reduced to an even coarser mesh. The idea being that this even smaller system would be used to further accelerate the CMFD solution. Another possible method for accelerating CMFD would be to collapse the group structure in addition to the spatial collapse. Though the equations in Section 2.2.2 include a group collapse, this is not done in practice within MPACT. Collapsing in energy will further reduce the size of the CMFD system, therefore requiring less computational expense to solve it.

Bibliography

[1] J. A. Turner, K. Clarno, M. Sieger, R. Bartlett, B. Collins, R. Pawlowski, R. Schmidt and R. Summers, "The Virtual Environment for Reactor Applications (VERA): Design and architecture," *Journal of Computational Physics,* vol. 326, pp. 544-568, 2016.

[2] C. T. Kelley, Iterative Methods for Linear and Nonlinear Equations, Philadelphia: Society for Industrial and Applied Mathematics, 1995.

[3] D. Knoll and D. Keyes, "Jacobian-free Newton-Krylov methods: a survey of approaches and applications," *Journal of Computational Physics,* vol. 193, pp. 357-397, 2004.

[4] D. Gill and Y. Azmy, "A Jacobian-Free Newton_krylov Iterative Scheme for Criticality Calculations Based on the Neutron Diffusion Equation," in *International Conference on Mathematics, Computational Methods & Reactor Physics*, LaGrange Park, 2009.

[5] D. Knoll, H. Park and C. Newman, "Acceleration of k-Eigenvalue/Criticality Calculations Using the Jacobian-Free Newton-Krylov Method," *Nuclear Science and Engineering,* vol. 167, pp. 133-140, 2011.

[6] J. A. Willert, C. T. Kelley, D. A. Knoll and H. Park, "Hybrid Deterministic/Monte Carlo Neutronics," *Journal of Scientific Computing,* vol. 35, no. 5, pp. S62-S83, 2013.

[7] D. Gill and Y. Azmy, "Newton's Method for Solving k-Eigenvalue Problems in Neutron Diffusion Theory," *Nuclear Science and Engineering,* vol. 167, pp. 141-153, 2011.

[8] E. Fichtl, J. Warsa and M. Calef, "Nonlinear Acceleration of SN Transport Calculations," in *International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering*, Rio de Janeiro, 2011.

[9] M. Calef, E. Fichtl, J. Warsa, M. Berndt and N. Carlson, "Nonlinear Krylov Acceleration Applied to a Discrete Ordinates Formulation of the k-Eigenvalue Problem," *Journal of Computational Physics,* vol. 238, pp. 188-209, 2013.

[10] Y. Xu, "A Matrix-Free Newton/Krylov Method for Coupling Complex Multiphysics Subsystems," Purdue University, Ph.D. Thesis, 2004.

[11] A. Ward, "A Newton-Krylov Solution to the Coupled Neutronics-Porous Medium Equations," The University of Michigan, Ph.D. Thesis, 2012.

[12] D. F. Kastanya, "Implementation of a Newton-Krylov Iterative Method to Address Strong Non-Linear Feedback Effects in FORMOSA-B BWR Core Simulator," North Carolina State University, Ph.D. Thesis, 2002.

[13] B. Herman, "Monte Carlo and Thermal Hydraulic Coupling using Low-Order Nonlinear Diffusion Acceleration," Massachusetts Institute of Technology, Ph.D. Thesis, 2009.

[14] M. Berrill, K. Clarno, S. Hamilton and R. Pawlowski, "Evaluation of Coupling Approaches," Consortium for Advanced Simulation of LWRs, CASL-U-2014-0081-000, 2014.

[15] R. E. Alcouffe, "Diffusion Synthetic Acceleration Methods for the Diamond-Differenced Discrete-Ordinates Equations," *Nuclear Science and Engineering,* vol. 64, pp. 344-355, 1977.

[16] K. S. Smith and J. D. Rhodes, "Full Core, 2-D, LWR Core Calculations with CASMO-4E," in *Proceedings of PHYSOR*, Seoul, 2002.

[17] D. A. Knoll, H. Park and K. Smith, "Application of the Jacobian-Free Newton-Krylov Method to Nonlinear Acceleration of Transport Source Iteration in Slab Geometry," *Nuclear*

Science and Engineering, vol. 167, pp. 122-132, 2011.

[18] H. Park, D. A. Knoll and C. K. Newman, "Nonlinear Acceleration of Transport Criticality Problems," *Nuclear Science and Engineering,* vol. 172, pp. 52-65, 2012.

[19] J. A. Willert and C. T. Kelley, "Efficient Solutions to the NDA-NCA Low-Order Eigenvalue Problem," in *Proceedings of the International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering*, Sun Valley, 2013.

[20] H. Park, D. A. Knoll, R. M. Rauenzahn, C. K. Newman, J. D. Densmore and A. B. Wollaber, "An Efficient and Time Accurate, Moment-Based Scale-Bridging Algorithm for Thermal Radiative Transfer Problems," *Journal of Scientific Computing,* vol. 35, no. 5, pp. S18-S41, 2013.

[21] J. A. Willert, H. Park and D. A. Knoll, "A Comparison of Acceleration Methods for Solving the Neutron Transport k-Eigenvalue Problem," *Journal of Computational Physics,* vol. 274, pp. 681-694, 2014.

[22] J. A. Willert and H. Park, "Applying Nonlinear Diffusion Acceleration to the Neutron Transport k-Eigenvalue Problem with Anisotropic Scattering," *Nuclear Science and Engineering,* vol. 181, pp. 1-10, 2015.

[23] J. A. Willert, "Hybrid Deterministic/Monte Carlo Methods for Solving the Neutron Transport Equation and k-Eigenvalue Problem," PhD Thesis, North Carolina State University, 2013.

[24] B. Carlson and K. Lathrop, "Transport Theory: The Method of Discrete Ordinates," Los Alamos Scientific Laboratory Report LA-3251-MS, 1965.

[25] MPACT Development Team, "MPACT Theory Manual," University of Michigan, Ann Arbor, 2014.

[26] J. Askew, "A Characteristics Formulation of the Neutron Transport Equation in Complicated Geometries," AEEW-R-1108, 1972.

[27] D. Knott and A. Yamamoto, Handbook of Nuclear Engineering Vol. 2 - Lattice Physics Computations, ISBN: 978-0-387-98130-7: Springer, 2010.

[28] K. S. Smith, "Nodal Method Storage Reduction by Nonlinear Iteration," *Transactions of the American Nuclear Society,* vol. 44, pp. 265-266, 1983.

[29] J. Duderstadt and L. Hamilton, Nuclear Reactor Analysis, John Wiley & Sons, 1976.

[30] E. Lewis and W. Miller, Computational Methods of Neutron Transport, La Grange Park: American Nuclear Society, 1993.

[31] M. Gutknecht, "A Brief Introduction to Krylov Space Methods for Solving Linear Systems," in *International Symposium on Frontiers of Computational Science*, Nagoya, 2005.

[32] L. Trefethen and D. Bau, Numerical Linear Algebra, Philadelphia: Society for Industrial and Applied Mathematics, 1997.

[33] Y. Saad and M. H. Schultz, "A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems," *SIAM Journal on Scientific and Statistical Computing,* vol. 7, no. 3, pp. 856-869, 1986.

[34] R. LeVeque, Finite Difference Methods for Ordinary and Partial Differential Equations, Philadelphia: Society for Industrial and Applied Mathematics, 2007.

[35] Y. Saad, Iterative Methods for Sparse Linear Systems, Philadelphia: Society for Industrial

and Applied Mathematics, 2003.

[36] A. Graham, T. Downar, B. Collins, R. Salko and S. Palmtag, "Assessment of Thermal-Hydraulic Feedback Models," in *PHYSOR*, Sun Valley, 2016.

[37] R. K. Salko and M. N. Avramova, "CTF Theory Manual," The Pennsylvania State University, 2016.

[38] R. Williamson, J. Hales, S. Novascone, M. Tonks, D. Gaston, C. Permann, D. Andrs and R. Martineau, "Multidimensional multiphysics simulation of nuclear fuel behavior," *Journal of Nuclear Materials,* vol. 423, no. 1-3, pp. 149-163, 2012.

[39] "PETSc Users Manual - Revision 3.6," Argonne National Laboratory, Argonne, 2016.

[40] A. Godfrey, "VERA Core Physics Benchmark Progression Problem Specifications," Consortium for Advanced Simulation of LWRs, CASL-U-2012-0131-004, 2013.

[41] D. Y. Anistratov, "Multilevel NDA Methods for Solving Multigroup Eigenvalue Neutron Transport Problems," *Nuclear Science and Engineering,* pp. 150-162, 2013.

[42] J. A. Willert, C. T. Kelley, D. A. Knoll and H. Park, "A Hybrid Approach to the Neutron Transport k-Eigenvalue Problem Using NDA-based Algorithms," in *Proceedings of the International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering*, Sun Valley, 2013.

[43] J. Lu, J. Guo, H. Zhang and F. Li, "A Preconditioning Algorithm for JFNK Method Applied To Multi-Physics Calculations," in *PHYSOR*, Sun Valley, 2016.

Vita

Erik Daniel Walker was born December 15, 1989 in Fort Smith, Arkansas to Richard and Sharon Walker. He is the youngest of three children: Jacqueline and Lauren. Erik attended Farmington High School in Farmington, Michigan where he would meet his future wife, Kristen. After graduating in 2008, he went on to study Nuclear Engineering and Radiological Science at the University of Michigan. While studying at Michigan, Erik worked as a research assistant in the field of radiation detection and measurement. He graduated in 2012 with a Bachelor of Science Degree.

After graduating, Erik worked as an intern for a summer at Knolls Atomic Power Laboratory in the Advanced Reactors Program. He then accepted a fellowship to the University of Tennessee through the Bredesen Center for Interdisciplinary Research and Graduate Education. Erik worked as a graduate research assistant at Oak Ridge National Laboratory for the Consortium for Advanced Simulation of Light Water Reactors. He was awarded his Master of Science Degree in Nuclear Engineering in December of 2014 and went on to earn his Ph.D. in Nuclear Engineering, along with a concentration in Energy Science Engineering, in December of 2017. Erik is looking forward to pursuing a career involving research and perhaps eventually teaching as a professor.