The NILO-CMFD Method for Iteratively Solving Coupled Neutron Transport-Thermal Hydraulics Problems

by

Nickolas Jordan Adamowicz

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Nuclear Engineering and Radiological Sciences)
in the University of Michigan
2022

Doctoral Committee:

Professor Emeritus Edward Larsen, Co-Chair
Professor Annalisa Manera, Co-Chair
Dr. David Aumiller
Associate Professor Krzysztof Fidkowski
Assistant Professor Brendan Kochunas
Dr. Yuxuan Liu

Nickolas J. Adamowicz
nicka@umich.edu
ORCID iD: 0000-0001-7053-564X

Dedication

To Anna (domestic partner),
Tony (father), Karry (mother), Anthony (sibling),
Pearl (large dog / roommate?), Toby, and Montey (small dogs).

Table of Contents

Dedication ii

List of Figures vi

List of Tables vii

List of Appendices viii

List of Symbols ix

Abstract x

Chapter 1. Introduction 1
- 1.1. Motivation . . . 1
  - 1.1.1. Nonlinear Interaction Between Radiation and Matter . . . 1
  - 1.1.2. Nuclear Reactors . . . 3
- 1.2. Historical Review . . . 6
  - 1.2.1. Transport Iterative Methods (Single-Physics) . . . 7
  - 1.2.2. Multiphysics Iterative Methods . . . 12
- 1.3. Thesis Outline . . . 18

Chapter 2. Problem Definition 20
- 2.1. Neutron Macroscopic Cross Sections . . . 20
- 2.2. Neutron Transport . . . 24
- 2.3. Thermal Hydraulics . . . 28

Chapter 3. Transport Iterative Methods 32
- 3.1. Transport Problem (Single-Physics) . . . 34
- 3.2. Source Iteration (SI) . . . 35
- 3.3. Coarse-Mesh Finite Difference (CMFD) . . . 38

Chapter 4. Multiphysics Iterative Methods 42
- 4.1. Terminology and Notation . . . 44
- 4.2. Neutron Transport – Thermal Hydraulics Problem . . . 44
- 4.3. Multiphysics CMFD (M-CMFD) & Relaxed CMFD (R-CMFD) . . . 47
  - 4.3.1. NOPC-CMFD . . . 54

4.3.2. Summary . . . 54
4.4. Theoretical Methods . . . 55
4.4.1. Theoretical Method 0 (TM-0) . . . 55
4.4.2. Theoretical Method 1 (TM-1) . . . 57
4.5. X-CMFD . . . 61
4.5.1. Inner Iteration . . . 64
4.5.2. Summary Remarks . . . 68
4.6. Nonlinearly Implicit Low-Order CMFD (NILO-CMFD) . . . 70
4.6.1. NILO-A . . . 71
4.6.2. NILO-CMFD . . . 74
4.7. NILO-CMFD Outlines . . . 76
4.7.1. NILO-A . . . 76
4.7.2. NILO-CMFD . . . 81
4.8. Discussion . . . 81
4.9. NILO-CMFD Modifications to Include Density . . . 83

Chapter 5. Continuous 1-D Model **86**
5.1. Problem Geometry . . . 87
5.2. Neutron Transport . . . 88
5.3. Thermal Hydraulics . . . 91
5.4. Nuclear Data (Cross Sections) . . . 97
5.5. 1-D Model : Reorganization and Summary . . . 99

Chapter 6. Discrete 1-D Model **102**
6.1. Introduction . . . 102
6.2. Continuous 1-D Model . . . 102
6.3. Discrete 1-D Model . . . 104
6.3.1. Neutron Transport . . . 105
6.3.2. Transport-Corrected Diffusion . . . 108
6.3.3. Thermal Hydraulics . . . 111
6.3.4. Nuclear Data . . . 114
6.4. Multiphysics Iterative Methods . . . 114
6.4.1. Multiphysics CMFD (M-CMFD) & Relaxed CMFD (R-CMFD) 114
6.4.2. X-CMFD, NILO-A, & NILO-CMFD . . . 121
6.5. Discrete Model Problem Parameters . . . 128
6.6. Numerical Results . . . 130
6.6.1. Single-Physics Model Solutions . . . 130
6.6.2. (Multiphysics) Discrete Model Solutions . . . 134
6.6.3. Multiphysics Iterative Performance . . . 140
6.7. Conclusions . . . 144

Chapter 7. NILO-CMFD in MPACT **146**
7.1. Introduction . . . 146
7.2. MPACT Overview . . . 149

7.3. NILO-A Overview 153
7.4. Numerical Results 155
7.4.1. Reactor Assembly Calculations 155
7.4.2. Quarter Core 175
7.5. Step 4 Iteration Performance 181
7.6. Conclusions 182

Chapter 8. Summary, Conclusions, & Future Work **184**
8.1. Summary & Conclusions 184
8.2. Future Work 188

Appendices **192**

Bibliography **203**

List of Figures

2.1. Sample Thermal Hydraulics Problem Domain. 30

5.1. 1-D Model Geometry. 87

6.1. Discrete 1-D Model Spatial Mesh. 106
6.2. Single-Physics Model Scalar Flux Solutions. 131
6.3. Single-Physics Model Fluid Temperature Solutions. 132
6.4. Single-Physics Model Doppler Temperature Solutions. 133
6.5. Discrete 1-D Model Scalar Flux Solutions. 135
6.6. Discrete 1-D Model Fluid Temperature Solutions. 136
6.7. Discrete 1-D Model Doppler Temperature Solutions. 137
6.8. R-CMFD Spectral Radius vs. Relaxation Factor. 140
6.9. Modified Model Spectral Rad. vs. Axial Thickness (N $\Rightarrow$ NILO-A). 143
6.10. Discrete 1-D Model Spectral Radius vs. Spatial Mesh. 144

7.1. Pin-Cell Fine- and Coarse- Mesh Cells (MPACT Theory Manual). . 151
7.2. Fresh Fuel Pin Power Radial Profiles. 159
7.3. Fresh Fuel Coolant Temperature [°C] Radial Profiles. 160
7.4. Fresh Fuel Pellet Temperature [°C] Radial Profiles. 162
7.5. Fresh Fuel Pin Power Axial Profiles. 163
7.6. Fresh Fuel Coolant Temperature [°C] Axial Profiles. 164
7.7. Fresh Fuel Pellet Temperature [°C] Axial Profiles. 165
7.8. Feedback-Depletion k-Eigenvalue vs. Burnup. 168
7.9. Feedback-Depletion k-Eigenvalue vs. Burnup. 169
7.10. Outer Iterations / Feedback-Depletion Timestep (1/2). 171
7.11. Outer Iterations / Feedback-Depletion Timestep (2/2). 172
7.12. Quarter-Core 3-D Solution Profiles. 178
7.13. Quarter Core (P9) Critical Boron Conc. vs. Burnup. 179
7.14. Quarter Core (P9) Outer Iters. / Statepoint vs. Burnup. 180

List of Tables

6.1. Assembly Compositions. 128
6.2. Nuclear Data Parameters. 129
6.3. Constant Nuclear Data Parameters. 129
6.4. Additional Model Parameters. 129
6.5. Single-Physics Model Eigenvalue Solutions. 134
6.6. Single-Physics Model Spectral Radii (ρ) and Iterations (N). 134
6.7. Discrete 1-D Model Eigenvalue Solutions. 139
6.8. Discrete Model Spec. Radii (ρ) and Iterations (N) (R-CMFD $\alpha = 0.7$). 142

7.1. 3-D Assembly Problem Identifiers & Characteristics. 156
7.2. Problem 2a-3d Convergence Table. 158
7.3. Problem 2f-3d Convergence Table. 158
7.4. Assembly Feedback-Depletion Outer Iterations (Total). 173
7.5. Feedback-Depletion Problem Timing, 2a-3d. 173
7.6. Feedback-Depletion Problem Timing, 2f-3d. 174
7.7. P9 (Quarter Core) Statepoint Information (1/2). 176
7.8. P9 (Quarter Core) Statepoint Information (2/2). 176
7.9. Feedback-Depletion Problem Timing, P9 (Quarter Core). 181
7.10. Linear Solver Iterations / Outer Iteration. 182

List of Appendices

Appendix A. VERA Input Deck **192**

Appendix B. Estimating NILO-CMFD Runtime **200**

List of Symbols

microscopic cross section	σ	macroscopic cross section	Σ
spatial coordinate	$\boldsymbol{r}$	neutron energy	E
material temperature	T	material density	ρ
angular flux	ψ	eigenvalue	λ
direction-of-flight	$\hat{\boldsymbol{\Omega}}$	fission energy distribution	χ
neutrons / fission	ν	energy deposition / fission	κ
system thermal power	P	heat source distribution	h
scalar flux	ϕ	current	$\boldsymbol{J}$
outer iteration index	n	inner iteration index	p
diffusion coefficient	D	transport-correction vector	$\hat{\boldsymbol{D}}$
high-order thermal hydraulics	$\mathcal{L}$	low-order thermal hydraulics	L
relaxation factor	α	relaxed heat source distribution	$\tilde{h}$
polar cosine	μ	azimuthal angle	ω
linear heat generation	q'	heat flux	q''
volumetric heat generation	q'''	Doppler temperature	T_d
fluid temperature	T_b	thermal resistance constant	β
centered cross section	$\Sigma_{u,0}$	Doppler temperature coefficient	$\Sigma_{u,1,d}$
fluid temperature coefficient	$\Sigma_{u,1,b}$	centered Doppler temperature	$T_{d,0}$
centered fluid temperature	$T_{b,0}$	cell height	Δz
discrete ordinate	μ_m	angular weight	w_m
Wielandt shift	λ_s		

Abstract

The interactions between neutron radiation and matter are nonlinear. Specifically, certain neutron nuclear reactions (e.g., fission events) generate heat, which modifies the temperature and density of the surrounding material, which in turn modify the neutron interaction probabilities (through the macroscopic cross sections). This nonlinear process is referred to as *neutron transport with thermal hydraulics feedback*.

Linear transport problems (without feedback) are accurately described by the *Neutron Transport Equation* (NTE). Discretized forms of the NTE are routinely solved using iterative transport acceleration schemes, for example, the standard *Coarse Mesh Finite Difference* (CMFD) method. CMFD and related acceleration schemes for linear transport problems are reliable and well-understood. However, when these same iterative methods are applied to nonlinear (i.e. multiphysics) problems, significant performance and stability issues often occur.

In this thesis, we propose modifications to the CMFD procedure, so that it can more robustly and efficiently solve loosely-coupled neutron transport – thermal hydraulics multiphysics problems. We refer to the new method as *Nonlinearly Implicit Low-Order CMFD* (NILO-CMFD). In this method, the transport-corrected diffusion equation – a foundational component of the CMFD method – is modifed to include approximate thermal hydraulics and nuclear data update physics.

To begin, we provide a general derivation of the 3-D NILO-CMFD method. Then, to initially analyze and test the method, we develop an approximate 1-D multiphysics model of a 3-D nuclear reactor fuel pin. The 1-D model consists of: (i) a 1-D neutron transport equation describing neutron transport, (ii) a 1-D advection-diffusion equation describing fluid temperature variation, (iii) a fuel temperature that depends on the 1-D fission heat source, and (iv) model equations that define the neutron cross sections and related nuclear data in terms of the fluid and fuel temperatures. The implementation of this model in a 1-D test code shows that the NILO-CMFD method overcomes the instabilities that occur in the standard *Relaxed-CMFD* (R-CMFD) method.

To test the new NILO-CMFD method on realistic 3-D problems, we implemented an incomplete version of the method, NILO-A, in the 3-D neutron transport code MPACT. For multiphysics problems, MPACT is loosely coupled to the 3-D COBRA-TF (CTF) code, which performs subchannel thermal hydraulics calculations. In every 3-D problem that we ran, NILO-CMFD successfully reduced the number of outer iterations to the low number required by CMFD for single-physics calculations. Unfortunately, due to lack of time, we could not implement the full NILO-CMFD method, which would have significantly reduced the wall clock time of the simulations. Consequently, some of our 3-D simulations required more wall clock time to converge than R-CMFD.

Nonetheless, our results show that the NILO-CMFD method is a legitimate generalization of single-physics CMFD to multiphysics problems. The number of outer iterations required by NILO-CMFD to achieve convergence is the same as that of single-physics CMFD, and when the NILO-CMFD nonlinear diffusion solve is well-optimized, we expect the cost of each NILO-CMFD outer iteration to be minimal – slightly greater than the cost of a single R-CMFD outer iteration.

Overall, our results indicate that the new NILO-CMFD method should be a robust, efficient, and easy-to-implement iteration scheme for solving loosely-coupled neutron transport – thermal hydraulics multiphysics problems.

Chapter 1.

Introduction

In this introductory chapter, we first motivate our interest in studying neutron transport – thermal hydraulics multiphysics problems by reviewing the nonlinear interaction between neutron radiation and matter. We explain how this physical process plays an essential role in the study of nuclear reactors. Next, we give a brief historical overview of iterative methods for solving both linear and nonlinear (multiphysics) transport problems. We then explain how the new methods introduced in this thesis fit within this broader context. Finally, we summarize the remainder of the thesis.

1.1. Motivation

1.1.1. Nonlinear Interaction Between Radiation and Matter

In this thesis, we develop iterative methods for the robust and efficient simulation of neutron transport problems with nonlinear thermal hydraulic feedback. We refer to this class of problems as *neutron transport – thermal hydraulics multiphysics problems*. At a fundamental level, the physical process underlying these multiphysics problems is the nonlinear interaction between energetic neutron radiation and the more-or-less stationary (aside from thermal motion) *background material* through which these *free neutrons* propagate. Next, we discuss the origins and consequences of this nonlinear relationship.

When describing systems with neutron radiation, we distinguish between two classes of neutrons, *bound* and *free*. Bound neutrons are held in nuclei of the background material by the nuclear strong force. The background material is the typically vast

collection of atoms through which energetic free neutrons propagate. Free neutrons are unbound; they are not attached to a nucleus. Owing to their neutral charge, free neutrons propagate through the system in straight-line paths until they interact with a nucleus of the background material in a discrete *neutron nuclear reaction*. For the purposes of this thesis, free neutrons are more relevant than bound neutrons. In order to simplify the text, we refer to "free neutrons" simply as "neutrons".

Neutron macroscopic cross sections describe neutron nuclear interaction probabilities (per unit distance of neutron flight). For a given reaction type, the macroscopic cross section depends on properties of both the neutron and the background material. For example, a neutron's energy affects macroscopic cross sections, as do the thermal motion, atomic number density, and isotopic composition of the background material.

Neutron nuclear reactions take several forms, with their relative likelihood of occurrence depending on the relative magnitudes of macroscopic cross sections for each potential reaction type. Below, we list a few neutron nuclear reactions relevant to our work:

(i) In an *elastic scattering* event, the *incident neutron* may undergo a change in both energy and direction of flight. In addition, the *target nucleus* may receive kinetic energy from the incident neutron. This *neutron downscatter energy* is eventually transferred to the bulk thermal energy of the background material.

(ii) In a *capture* event, a target nucleus absorbs the incident neutron. The once free neutron becomes a bound neutron. This increases the target nucleus' mass number (M) by one, without affecting its atomic number (Z). Thus, a capture event results in the formation of a new isotope of the same element as the original target nucleus. After formation, the new isotope may be unstable, which can lead to a spontaneous nuclear decay or emission of a nuclear de-excitation gamma-ray.

(iii) In a *fission* event, an incident neutron splits an unstable heavy nucleus into two *daughter nuclei*, producing a variable number of energetic free neutrons and gamma-rays in the process. Fission product daughter nuclei are imparted with a significant kinetic energy. As they slow down over a short distance, this excess kinetic energy is deposited into the thermal energy of the background material.

The above list of neutron nuclear reactions is not exhaustive. Rather, these examples were chosen to illustrate a key concept. In addition to changing the neutron

distribution, neutron nuclear reactions may also modify background material properties. When neutrons modify the background material in ways that affect neutron macroscopic cross sections (neutron interaction probabilities), a nonlinear feedback loop is created.

As free neutrons interact, their number, direction, and energy change. For example, (i) scattering events modify neutron energy and direction-of-flight. (ii) capture events remove neutrons from the system, and (iii) fission events remove a single neutron and produce a variable number of isotropically emitted fission-product neutrons that follow a fission energy spectrum.

Additionally, some neutron nuclear reactions affect the background material. For example, (i) scattering events that impart kinetic energy on the target nucleus increase background material temperature, (ii) capture events result in direct isotope transmutation, and (iii) fission events result in the creation of energetic daughter isotopes that increase material temperature through a slowing-down process. As material temperature increases, density changes through equations of state. These background material effects in turn modify neutron interaction probabilities through macroscopic cross sections. As a reminder, macroscopic cross sections depend on background material isotopic composition, temperature, and density, all of which may be perturbed by certain neutron nuclear reactions. For example, those listed above.

These examples help to illustrate the fact that neutron interaction with matter is a nonlinear process. There is a nonlinear feedback link between free neutrons and the background material through which the neutrons propagate. As neutrons interact with matter, the neutron distribution and neutron-material interaction probabilities (through macroscopic cross sections) are both modified. The inherent nonlinearity of this physical process poses numerous challenges to its numerical simulation. In mathematics, nonlinear problems tend to be more difficult to solve than linear ones. In this thesis, we address some of these nonlinearity-borne challenges.

1.1.2. Nuclear Reactors

To motivate our interest in the nonlinear interaction between neutron radiation and matter, we give a real-world example in which this physical process plays an important role: the study of nuclear reactors.

A nuclear reactor is designed to control fission chain reactions. As mentioned

earlier, certain target nuclei may fission upon interaction with a free neutron, for example, those of the nuclides uranium-235 ($^{235}_{92}$U) and plutonium-239 ($^{239}_{94}$Pu). The heavy fission daughter fragments formed in a fission event are imparted with significant kinetic energy ($\approx$ 200 [MeV]). This energy is absorbed into the bulk thermal energy of the material immediately surrounding the fission site, modifying its thermodynamic state (temperature, density, etc.). Fissions also produce additional neutrons following a probabilistic multiplicity (i.e. number emitted) and energy distribution.

To summarize, a fission both locally deposits thermal energy and produces additional energetic neutrons. As one might expect, the newly minted energetic fission neutrons may eventually initiate other fission events. This is the basis of the fission chain reaction. Given an initial collection of energetic neutrons and a system containing fissionable nuclides, some of the original neutrons may initiate fissions, generating additional free neutron(s) in place of the originals. Other neutrons may have shorter-lived lineages; for example, a neutron may prematurely exit the fission chain reaction by leaking out of the system, or by being absorbed in a capture reaction. We think of the *fission-initiating* neutron as being *killed* and the *fission-produced* neutron(s) as being *born*.

Over time, all of the neutrons from the first generation will have died, via fission or otherwise (e.g., leakage, capture, etc.), and a new generation of neutrons (born from fission events) will have taken their place. The population of the new neutron generation may be greater than, less than, or equal to that of the previous generation. A nuclear reactor increases, decreases, or maintains the neutron population depending on the desired reactor operating condition. Control of the neutron population also gives influence over the thermal power generation in the core, since the neutron distribution influences fission reaction rates and, by proxy, thermal energy deposition by heavy fission daughter nuclei.

The control over the neutron population and thermal power generation facilitated by a nuclear reactor lead to many useful applications. As one example, high reactor core neutron fluxes can be used to irradiate samples in order to test radiation-induced material degradation. These experiments provide vital data used to develop more resilient reactor materials [71, 101]. In the medical physics [108] and national security [67] communities, high in-core neutron reaction rates are used for isotope production.

Most frequently, nuclear reactors are used for thermal power generation. In some cases, the reactor heat source is used directly, for example, in water desalination [77, 79] or hydrogen fuel production [23, 121]. More often, the core thermal output

is converted into electrical energy through a thermodynamic cycle. Nuclear reactors provide a significant portion of the electricity generation in many developed countries. Approximately 20% of the United States power grid is supplied by nuclear reactors. In France, the majority of electricity is generated by nuclear (fission) power ($\approx 77\%$) [41].

Numerical simulation is an important tool for nuclear reactor design and operation. In order to better understand the fission chain reaction that drives a reactor, we require sophisticated simulation capabilities. We must computationally model the nonlinear interaction between the core neutron distribution and the background material (nuclear fuel, structural material, coolant / moderator, etc.) Simulation is an indispensable tool for both reactor designers and operators, allowing them to:

(i) Better understand the state of an operating core, especially if certain physical quantities are difficult or impossible to measure. An active reactor core is a harsh environment. An inserted physical measurement probe must survive in an unwelcoming, hot, corrosive, and radioactive environment. Numerical simulations give another, safer, way to "see" inside of a reactor.

(ii) Develop improved (tighter) safety margins in order to push reactors towards their physical limit, while maintaining acceptable risk.

(iii) Study new reactor core designs without the need to construct expensive physical prototypes.

(iv) Increase reactor performance by decreasing operating costs and maximizing energy production. This is important to make nuclear reactors cost-effective.

Unfortunately, the accurate numerical simulation of a reactor core is not a simple task. Reactor simulation requires the careful consideration of neutron particle transport and its interaction with the core background material. As mentioned earlier, this is an inherently nonlinear problem. Since the physical system is nonlinear, its accurate mathematical description requires a nonlinear formulation. In addition to neutron transport physics, we must also consider nuclide accretion / depletion, material degradation, fission gas release, and thermal hydraulics, among other physical processes that are both driven by and drive, through nonlinear feedback, the neutron distribution. In this thesis, rather than focusing on all core feedback mechanisms, we narrow our focus to the effects of thermal hydraulics on the neutron transport

process. Often, this is the dominant feedback mechanism for steady-state reactor operation and short-timescale transients.

A four-step *simulation pipeline* is required to model a reactor. First, it must be decided which physics will be included in the simulation. In this thesis, this step is predetermined; since we will only focus on neutron transport – thermal hydraulics multiphysics problems. Second, one must formulate the governing mathematical equations for each physics of interest. In this thesis, the neutron transport process is described by the Neutron Transport Equation (NTE), and the thermal hydraulics physics is given a general form. These are explained in further detail in Ch. 2. Third, one must discretize each governing equation, transforming from differential or integro-differential forms into an algebraic form suitable for solution on a computer. For the most part, this thesis eschews discussion of discretization. We rely on standard techniques. The final step in the simulation pipeline is the solution of the discretized equations. In this thesis, we focus on this step. Since the neutron transport – thermal hydraulics problem is nonlinear, one must solve a system of nonlinearly coupled algebraic equations. Oftentimes, direct solution of this system is unfeasible, and iterative techniques are required. Iterative methods decompose the full, nonlinear problem into a sequence of simpler, usually linear, sub-problems solved repeatedly until suitable convergence criteria are met.

Standard iterative techniques for solving some neutron transport – thermal hydraulics multiphysics problems in nuclear reactors lack robustness and efficiency. In this thesis, we propose a new iterative method to improve upon the flawed multiphysics iterative methods in use today. In the following section, we set the stage for the development of this new technique by giving a brief historical review of the research that has influenced its design.

1.2. Historical Review

To contextualize the iterative methods discussed later in this thesis, we now provide a brief historical review of iterative techniques for solving single-physics and multiphysics neutron transport problems.

1.2.1. Transport Iterative Methods (Single-Physics)

Before covering multiphysics-focused research, we first review the history of iterative method development for single-physics neutron transport problems. One of the single-physics methods we cover, *Coarse Mesh Finite Difference* (CMFD), will act as the base upon which we build the multiphysics iterative methods introduced later in the thesis.

The description of particle transport phenomena requires a high-dimensional phase space. The neutron distribution in a steady-state (time-independent) system lives in a six-dimensional phase space consisting of three spatial dimensions, two angular dimensions, and one energy dimension. The high dimensionality of transport problems oftentimes results in significant computer memory requirements to store even the transport solution itself. Rather than requiring storage of the full six-dimensional transport solution, transport iterative methods usually work with angularly integrated solution quantities that live in a smaller phase space. This is accomplished by using *transport sweeps*, which are discussed in greater detail in Ch. 3. For now, we simply mention that sweeps are a low-memory-cost means of updating angular moments of the transport solution. Since memory usage is of great concern in computing environments, steady-state transport iterative methods are frequently based on a foundation of iterated transport sweeps.

Source Iteration (SI) is the simplest transport iterative method based on *sweeping* [2, 6, 24, 62, 63, 122]. In the literature, SI may alternatively be called "*Richardson iteration*" or "*iteration on the scattering source*", among other aliases. In SI, transport sweeps are performed repeatedly until convergence. SI is slowly-converging in scattering-dominated, optically-thick problems. Put another way, SI converges slowly for systems characterized by a long average neutron lifetime. For such problems, solutions obtained with SI may take hundreds, thousands, or even millions of transport sweeps to converge.

In practical calculations, transport sweeps are computationally expensive. Iterative methods that require a large number of sweeps to converge are inefficient. As such, it was quickly realized that in order to solve real-world transport problems with reasonable computational effort and in a sensible amount of time, methods were required to *accelerate* the iterative convergence of Source Iteration (SI). This set off decades' worth of research into *transport acceleration methods*. This research began in the mid 20^{th} century and continues to this day. A thorough review of acceleration

method research up until ≈ 2002 is catalogued by Adams and Larsen in [2]. Notably, their review does not mention Coarse Mesh Finite Difference (CMFD), an acceleration scheme that takes center stage in this thesis.

One of the earliest transport acceleration schemes, *Chebyshev acceleration* [12, 34, 62, 109], falls into a class of methods we refer to as being *algebraic*. Algebraic acceleration schemes are mathematically based. Generally, they do not consider the physics of the underlying transport problem being solved. Chebyshev acceleration is a fixed-point acceleration scheme at its heart. The method can be straightforwardly copied from a numerical methods textbook (e.g., [109]) and applied to transport problems with relative ease. This ease-of-implementation is a benefit of algebraic acceleration techniques. However, by not using physical information about the problem being solved, algebraic acceleration schemes are often not the most efficient or robust choice.

In recent years, algebraic acceleration has seen a minor resurgence, especially in problems with complications due to negative flux fix-ups and unstructured meshes with cycles. As two specific examples, *Anderson acceleration* ([4, 58, 103, 105, 119]) and *Dynamic Mode Decomposition*-based acceleration ([72, 73]) have both recently received attention.

While the ease-of-implementation of algebraic acceleration is attractive, early researchers quickly moved on from Chebyshev acceleration and instead looked towards what we call *physics-based acceleration* techniques, which make use of the physics of the underlying transport problem. *Coarse and Fine-Mesh Rebalance* are early examples of physics-based approaches [13, 63, 75, 83]. Rebalance saw wide adoption for a number of years. In Rebalance, sweep-determined solution estimates are multiplied by cell-dependent scalar values in an attempt to preserve global neutron balance. Rebalance was found to be a quite brittle acceleration scheme. A successful application of Rebalance often required a skilled code-user at the helm of a simulation. Improper use of Rebalance could result in slow convergence or even divergence. Eventually, Rebalance fell out of favor and was supplanted by diffusion-based accelerators, covered next. Later on, the brittleness of Rebalance was revisited through a theoretical lens in [13] using a Fourier stability analysis.

While rebalance faded away, another physics-based acceleration method, *Diffusion Synthetic Acceleration* (DSA), proved to have tremendous staying power. We do mention that DSA is primarily used to accelerate the solution of fixed-source transport problems. While DSA can be modified to work with eigenvalue problems, a problem-

type of major interest to reactor physicists, the required adjustments to the method are rather awkward [3].

We mention that the name "Diffusion Synthetic Acceleration (DSA)" was not introduced until Alcouffe's seminal work on developing a consistent diffusion discretization for the Diamond-Differenced (DD) transport equations in the 1970's [3]. Regardless, we use the term DSA to refer to a general class of methods that use a diffusion solve to construct an additive correction to the transport sweep solution estimate, even when discussing research that predates Alcouffe's work.

Diffusion Synthetic Acceleration (DSA) research began with Kopp's introduction of the more-general *synthetic method* in the 1960's [51]. Kopp originally proposed the synthetic method to iteratively solve linear systems of equations. The basic idea of the synthetic method (or, *synthetic acceleration*) is to avoid the expensive direct solution of a difficult, *high-order* problem by instead *synthesizing* (i.e. combining) the solutions of a sequence of, preferably simpler, *low-order* problems. For a particular high-order problem, a spectrum of synthetic methods are available that vary the definition of the low-order accelerator. DSA is a specific implementation of the synthetic method in which the high-order problem is neutron transport and the low-order problem is neutron diffusion.

Kopp presented the first application of the synthetic method towards particle transport problems using diffusion (P_1) as the low-order component [51]. Gelbard and Hageman explored the synthetic acceleration of X-Y geometry, *discrete ordinates* (S_N) transport using both diffusion (P_1) and low angular quadrature order S_N (specifically, S_2) [29]. In [83], Reed used both analysis and numerical experiment to show that when a slab geometry, diamond-differenced, transport equation is synthetically accelerated by a standard diffusion discretization, as mesh cell optical thickness is increased beyond a certain point, performance degrades and the scheme eventually becomes unstable. In [3], Alcouffe introduced the name "*Diffusion Synthetic Acceleration*" (DSA) to describe a synthetic acceleration scheme for the diamond-differenced transport equation in which the low-order diffusion equation was carefully discretized, removing the stability issues experienced by Reed and others. Researchers describe Alcouffe's diffusion discretization, as well as other diffusion discretizations derived in a similar manner, as being *consistent*. Consistent diffusion discretizations are derived by directly manipulating transport discretizations.

In [56, 74], Larsen and McCoy extended Alcouffe's work by deriving and numerically testing consistent, slab geometry, diffusion discretizations for general weighted-

diamond, linear characteristic, linear discontinuous, and linear moments spatial discretizations for discrete-ordinates (S_N) transport. Larsen developed a *"four-step"* procedure to derive consistent diffusion discretizations. Unfortunately, for some spatial discretizations, especially those used in multidimensional problems, Larsen's four step procedure results in a discretized system of P_1 equations. This system cannot always be collapsed into a discretized diffusion equation without imposing extra ad hoc closure relationships.

In [1], Adams and Martin built off of Larsen's work and proposed the *modified four step* (M4S) procedure. Using M4S, one can obtain a "mostly consistent" diffusion discretization, even when Larsen's four-step procedure fails. In their original work, Adams and Martin showed the success of M4S by deriving a mostly consistent diffusion discretization for a bilinear discontinuous finite-element transport discretization. Other diffusion discretizations that pair well with DSA can be derived using thick diffusion limit asymptotic expansions [118].

(We pause to mention the *Quasidiffusion* method [5], also known as the *Variable Eddington Factor* (VEF) method. In our view, and the view of others [2], Quasidiffussion is not a transport acceleration scheme, since the numerical solutions produced by the discretized Quasidiffusion equations are different from those of the discretized transport equations they are applied to. All acceleration methods in this thesis, both single-physics and multiphysics, preserve the discretized transport solution.)

Next, we consider *Coarse Mesh Finite Difference* (CMFD), a transport acceleration method that has become widely used in deterministic reactor physics codes. Smith proposed the original formulation of CMFD as an acceleration method for nodal diffusion calculations in [98]. The description of CMFD presented in Smith's original paper [98] is somewhat detached from the description of the method as understood by modern practitioners. For a clearer, more in-depth picture of CMFD, we refer the reader to Smith & Rhodes later publication in [99].

Similar to DSA, CMFD accelerates iterated transport sweep convergence using a low-order, diffusion-based accelerator. However, while DSA computes a diffusion-based additive correction to the transport sweep-determined solution estimate, CMFD's transport-corrected diffusion equation calculates the scalar flux update itself. Unlike Quasidiffusion (a.k.a. VEF), the CMFD diffusion scalar flux solution is made consistent with the discrete transport solution by the proper definition of a *transport correction term*.

Frequently, CMFD is implemented using a two-mesh approach. A fine spatial mesh

is used for the high-order transport sweeps, while a coarser, usually block, mesh is used for the low-order CMFD diffusion problem. Simple restriction and prolongation operations are used to map problem parameters and solution information between meshes.

(The name *Nonlinear Diffusion Acceleration* (NDA) [21, 88, 117] is sometimes used in the literature to refer to variants of CMFD. In our view, NDA is a special case of CMFD in which the same spatial mesh is used for both the transport and diffusion discretizations. That is, in NDA, there is no coarse-mesh. We also use the term NDA to refer to the CMFD methodology when it is applied to a continuous (i.e. undiscretized) transport problem.)

CMFD has been implemented in numerous deterministic reactor physics codes: MPACT [55], OpenMOC [7], CASMO [84, 99] to name a few. CMFD has also been applied to accelerate fission source convergence in Monte Carlo (stochastic) transport codes [59, 120].

Along with its record of successful applications in production transport codes, CMFD is also well-understood theoretically. Fourier analyses have thoroughly explored the stability and convergence properties of CMFD [26, 38, 42, 97, 115]. Also, a strong theoretical link has been shown to exist between DSA and CMFD [57]. The two approaches are algebraically equivalent upon linearization. That is, ignoring nonlinear effects, the methods have equivalent convergence properties.

A significant drawback of the standard CMFD and (inconsistent) DSA formulations is the appearance of instabilities in the presence of optically thick spatial cells. A number of publications have proposed simple modifications to CMFD to avoid these instabilities. As a few examples, the *partial-current based CMFD* (pCMFD) [14, 15], *optimally diffusive CMFD* (odCMFD) [126], and *linear prolongation-based CMFD* (lpCMFD) [114] methods each improve upon standard CMFD.

Owing to CMFD's strong theoretical foundation and pervasiveness in modern reactor physics codes, we have chosen to exclusively consider CMFD-based multiphysics iterative methods in this thesis. Originally, CMFD was developed as an acceleration scheme for single-physics neutronics problems. Later in the thesis, we will see that stability and convergence rate difficulties appear when multiphysics feedback is naïvely included in the standard CMFD algorithm. A primary goal of this thesis is to develop approaches to effectively and efficiently address these stability issues.

Next, we move away from single-physics transport acceleration and switch to a literature review of iterative methods for multiphysics problems. We pay special attention

to CMFD-based methods, since this thread of research is built upon throughout the remainder of this thesis.

1.2.2. Multiphysics Iterative Methods

Multiphysics iterative methods are broadly categorized as being either *tightly-coupled* or *loosely-coupled* [116]. In our work, we focus exclusively on loosely-coupled neutron transport – thermal hydraulics iteration schemes. Before introducing loose coupling, we briefly discuss tight coupling.

In tight coupling, the discretized equations governing each single-physics in a multiphysics problem are combined to form a large, monolithic, nonlinear algebraic system to be iteratively solved, usually using some residual minimization technique. In tight coupling, each single-physics is *tightly* integrated with one another in a single system of equations. An excellent review of tight coupling is given in [47], with a focus on the popular *Jacobian-Free Newton Krylov* (JFNK) method. Tightly-coupled methods often employ *Newton's method*, a common function minimization technique [44]. Newton's method can be used to minimize multivariate, vector-valued functions. Tightly-coupled multiphysics methods rely on this capability. In tight coupling, a (usually approximate) Newton-based method is used to find inputs that minimize a function returning the residuals of a discrete multiphysics problem. Newton-based methods iteratively search for the inputs to this function that produce a residual output vector closer and closer to the zero vector. If all residuals are identically zero, a solution to the multiphysics problem has been reached. Usually, iteration is halted when the residual vector is sufficiently close to the zero vector, according to some user-specified convergence criteria.

Newton's method converges quadratically, but requires the Jacobian (i.e. derivative information) of the residual function. In practice, derivative information may not be easily available, in which case a "*derivative-free*" approximation to Newton's method can be used, for example, Jacobian-Free Newton Krylov (JFNK) [8, 43, 47]. While JFNK avoids the need to explicitly evaluate the Jacobian, it requires a suitable preconditioner for an efficient implementation.

In practice, tightly-coupled methods are more demanding to implement than loosely-coupled methods (introduced next). Even when explicit derivative evaluations are avoided through the use of techniques like JFNK, residual information for the full nonlinear system is still required. In order to leverage existing single-physics soft-

ware by connecting them together in a tightly-coupled iterative framework, we require residual information from each solver. If a solver does not provide residuals, it cannot be used in a Newton-based method without modifying its source-code. In some cases, source-code modification may be difficult or time-consuming. To avoid this scenario, we choose to ignore tightly-coupled methods. Instead, we focus our efforts towards improving loosely-coupled approaches that can more-easily accommodate the linking together of existing single-physics applications (regardless of whether they provide residuals).

In tight coupling, a nonlinear system is constructed from each of the single-physics involved in a multiphysics problem. In effect, tightly-coupled methods construct simultaneous updates to the entire multiphysics problem. This is done iteratively, usually with some Newton-based method. In loose coupling, each single-physics is treated separately. Single-physics codes are independently solved and their associated solution information is updated and passed to other single-physics codes. This is repeated for each single-physics in the multiphysics problem in a round robin fashion (Picard iteration).

While tight coupling methods are based on Newton's method, loose coupling methods are based on *Picard iteration* [35, 54]. Picard iteration amounts to repeatedly (i) linearizing nonlinear terms by lagging them with the most recent solution estimates, and then (ii) solving the resulting linear sub-problem for improved solution estimates. To solve a multiphysics problem with Picard iteration, we decompose the problem into its single-physics components and then solve a sequence of single-physics problems with lagged parameter estimates.

From this description, it is clear that loosely-coupled methods based on Picard iteration can easily accommodate existing single-physics codes. In order to do this, we just need to interface each single-physics solver so that it can send and receive updated solution estimates to and from each of the other solvers. This data exchange can be accomplished through simple text-based file input-output (IO) or by using more efficient *in-memory coupling* [16, 113]. Although loose coupling is "simple" to implement, a naïve implementation is not always performant. In fact, Picard-based loose coupling often gives rise to instabilities. A major goal of this thesis is to propose modifications to standard loose-coupling techniques that address these instabilities.

Next, we provide a few examples from the literature of loosely-coupled, Picard-based schemes used to solve neutron transport – thermal hydraulics reactor physics problems. Although we mostly avoid discussing stochastic transport in this thesis,

we mention that several Monte Carlo transport codes have been loosely coupled to thermal hydraulics solvers. For example, the OpenMC [78], Serpent [22], MCNP [37], and MC21 [45] codes each have proof-of-concept implementations.

Loosely-coupled deterministic transport codes have also been used to solve neutron transport – thermal hydraulics multiphysics problems. Since these codes primarily use Coarse Mesh Finite Difference (CMFD) for transport acceleration, their implementations of Picard iteration frequently use a paired transport sweep and transport-corrected diffusion solve. After performing a sweep and solving the CMFD diffusion equations, the transport code then calls out to a (likely external) thermal hydraulics solver to update material temperatures and densities, and then to a nuclear data library to update cross sections. We refer to this basic approach as *Multiphysics CMFD* (M-CMFD). As a concrete example, the M-CMFD strategy is implemented in the Michigan Parallel Characteristics-based Transport (MPACT) code [55, 80] allowing it to couple to a variety of thermal hydraulics solvers, e.g., an internal simplified thermal hydraulics (*simTH*) solver [32], the COBRA-TF (CTF) subchannel code [49, 65, 87], and the CFD code STAR-CCM+ [113]. MPACT's ability to couple to this diverse set of thermal hydraulics codes lends itself as evidence to the ease of code coupling that loosely-coupled methods facilitate.

While Picard-based schemes are common, they are seldom implemented without the addition of some form of *relaxation* to address stability and convergence rate issues. Relaxation is a simple "engineering-approach" used to stabilize iterative methods. With relaxation, a scalar relaxation factor is introduced to dampen iterative solution updates. The relaxation factor itself is a user-chosen "knob". The optimal relaxation factor often differs from problem to problem. We use the name *Relaxed CMFD* (R-CMFD) to refer to the Multiphysics CMFD (M-CMFD) method described above when relaxation is included. R-CMFD is the default multiphysics iterative method used in the MPACT code for steady-state eigenvalue problems [55, 80]. To be clear, M-CMFD is just R-CMFD without relaxation.

In [50], Kochunas et al. show, using the Fourier analysis of loosely-coupled iterative methods applied to a multiphysics transport problem with an approximate thermal hydraulics feedback mechanism, that M-CMFD promotes instabilities in low-spatial-frequency (i.e. flat) error modes. This was a surprising result, since in single-physics transport problems, CMFD is designed to effectively eliminate these same flat error modes. Kochunas shows that relaxation can help avoid these issues, but that it is not a panacea. For problems with a sufficiently strong feedback intensity, R-CMFD

remains slowly-convergent or unstable, regardless of the chosen relaxation factor.

Issues with R-CMFD and related relaxation-based methods led researchers to explore other approaches less reliant on a properly chosen relaxation factor. One thread of research focused on surrounding R-CMFD-like methods with an algebraic accelerator. Another campaign looked into removing the relaxation factor and replacing it with a more-performant, easier-to-choose relaxation-like parameter. We refer to both of these efforts as "*pseudo-relaxation techniques*", since they do not fundamentally alter the structure of R-CMFD-like methods. We cover a few of these next.

Some researchers looked into wrapping relaxation-based, loosely-coupled methods in fixed-point accelerators for nonlinear problems. In some ways, this is similar to the algebraic acceleration of transport sweeps that was explored using Chebyshev acceleration at the dawn of single-physics transport iterative method development. Toth's work covers Anderson acceleration of CMFD-based Picard-schemes for nonlinear reactor multiphysics problems [103]. Anderson acceleration [4, 112] is a popular fixed-point accelerator used in a number of fields. Like other algebraic techniques, the implementation of Anderson acceleration is relatively simple, owing to its mathematical- rather than physics-based nature. A downside of Anderson acceleration is its requirement to store solution information over several iterations in order to compute its accelerated updates. Unfortunately, Anderson acceleration does not entirely do away with user-specified iteration parameters (knobs). For example, implementation of Anderson acceleration requires the choice of a storage depth (how many iterations worth of data to save). Also, Anderson acceleration may include its own relaxation factor. Toth's work on Anderson acceleration [103, 104] was mainly applied to Pressurized Water Reactor (PWR) problems. Recently, the technique has also been used for Boiling Water Reactor (BWR) simulation [39].

Another avenue of pseudo-relaxation research is based on *partial convergence*, which is used to avoid residual over-solving. The basic idea of partial convergence is to avoid wasting time fully solving a single-physics problem in a Picard iteration, when it is known that this problem has incorrect, lagged, nonlinear terms. Senecal and Ji explored reducing residual over-solving in [89, 90]. In [92], Shen proposed a partial-convergence scheme in which the CMFD diffusion equation was purposefully loosely converged between thermal hydraulics evaluations and nuclear data updates. Shen showed that partial convergence of the CMFD diffusion equations behaves like a modified form of relaxation. However, partial convergence still involves a user-chosen iteration parameter (knob). A user must choose the degree of partial convergence to ap-

ply in solving the CMFD transport-corrected diffusion equations. That is, how loose should diffusion inner iteration convergence tolerances be made? Shen later extended his work by developing the Nearly Optimally Partially Converged (NOPC)-CMFD method in [91, 94, 96], which sought to algorithmically determine an approximation to the optimal partial convergence tolerance. This was shown to be successful when applied to Pressurized Water Reactor (PWR) multiphysics problems. NOPC-CMFD was also successfully applied to Boiling Water Reactor (BWR) problems [17].

Next, we move on from pseudo-relaxation and instead focus on a recent thread of research looking to couple nonlinear feedback directly into the CMFD transport-corrected diffusion equation. The general idea here is to minimize M/R-CMFD transport sweeps by introducing thermal hydraulics feedback at the (transport-corrected) diffusion inner iteration level rather than transport sweep outer iteration level. We refer to this general idea as *diffusion-level coupling*.

In [36], Herman considered diffusion-level coupling between a CMFD-accelerated Monte Carlo transport code and a thermal hydraulics code. In [110, 111], Walker explored diffusion-level coupling of deterministic transport; he referred to this approach as "Low-Order Coupling". In Walker's work, nuclear data in the CMFD diffusion equation were updated through either Picard-based or JFNK-based iteration with a thermal hydraulics solver. Walker found that as one tightens the tolerance of this nonlinear neutron diffusion – thermal hydraulics iteration, the total number of transport sweeps required to converge a problem is reduced. In other words, by performing more feedback work at the diffusion-level, the outer-iteration spectral radius at the transport-sweep-level is improved. However, Walker also found that the added cost imposed by diffusion – thermal hydraulics inner iterations was sometimes unjustifiably expensive.

Following Walker and Herman's work, Shen et al. proposed the *X-CMFD* method [93], which took diffusion-level coupling to the extreme by fully converging the nonlinear neutron diffusion – thermal hydraulics problem in Walker's Low-Order Coupling methods. In [91, 95], Shen calls this method the *Nonlinearly Fully Coupled Diffusion Acceleration* (NFCDA) method and instead uses the term X-CMFD to refer to a practical means of solving the NFCDA nonlinear diffusion equations using a modified power iteration approach. In this thesis, when we refer to X-CMFD, we are referring to what Shen calls NFCDA. In Ch. 4, we provide a full outline of what we view as X-CMFD.

In our view, X-CMFD is not a practical multiphysics iterative method because

it does not specify an efficient means of solving its nonlinearly-coupled neutron diffusion – thermal hydraulics problem. Straightforward iterative techniques do exist for solving this problem, for example, Walker or Shen's Picard-style inner iteration schemes do work, but they necessarily require multiple evaluations of the thermal hydraulics solver and nuclear data update routines. If the thermal hydraulics solves or nuclear data evaluations are expensive, the cost of solving X-CMFD's nonlinear diffusion problem would be unfeasibly large. This could occur, for example, if we were using X-CMFD to solve a problem in which both the thermal hydraulics is high-fidelity Computational Fluid Dynamics (CFD) and the nuclear data evaluations are expensive due to the presence of hundreds of distinct isotopes in the system (i.e. as could occur in a depletion calculation).

A push is being made towards using higher-fidelity and, accordingly, higher-cost thermal hydraulics physics to more accurately model the thermal hydraulics physics in nuclear reactors. By using these expensive thermal hydraulics solvers in X-CMFD (Shen's NFCDA), or other diffusion-level coupling approaches, one may overburden the thermal hydraulics solver. That is, while diffusion-level coupling schemes help to minimize the number of transport sweeps required to solve a multiphysics problem, they require many extra evaluations of the thermal hydraulics solver. This can become computationally expensive. A similar argument holds for the excess reliance of diffusion-level coupling on nuclear data updates. In this thesis, our goal is to modify diffusion-level schemes so that their outer iteration convergence and stability is maintained without requiring multiple high-order thermal hydraulics solves or nuclear data updates each iteration. We believe that the Nonlinearly Implicit Low Order (NILO)-CMFD method, introduced in Ch. 4, meets this goal.

The use of approximate operators to describe the neutron transport equation is a common approach among transport acceleration techniques. This is the foundation of Kopp's Synthetic Method [51], which in turn influenced the development of both Diffusion Synthetic Acceleration (DSA) and Nonlinear Diffusion Acceleration (NDA) / Coarse Mesh Finite Difference (CMFD). In NILO-CMFD, we use a neutron diffusion operator to accelerate the slow convergence of iterated transport sweeps. This is standard. In addition, we use low-order (i.e. approximate) thermal hydraulics operators and nuclear data update expressions to improve the convergence rate and robustness of CMFD-based Picard schemes for multiphysics neutron transport – thermal hydraulics problems. Ch. 4 provides full detail on this approach.

1.3. Thesis Outline

The remainder of this thesis is organized as follows:

In Ch. 2, we outline the underlying mathematical equations of the neutron transport – thermal hydraulics multiphysics problem in which we are interested. Specifically, we cover (i) how neutron macroscopic cross sections depend on background material temperature and density, (ii) the neutron transport equation, specifically it's power (k or λ)-eigenvalue formulation, and (iii) the general thermal hydraulics description we use throughout this thesis.

Ch. 3 takes a brief detour into the realm of single-physics neutron transport iterative methods. As mentioned above, the multiphysics iterative methods covered in this thesis are built upon the widely-used Coarse-Mesh Finite Difference (CMFD) transport acceleration scheme. CMFD works exceptionally well for single-physics transport problems, but it tends to break down when multiphysics feedback is naïvely included. In Ch. 3, we first introduce Source Iteration (SI), the iterative method upon which CMFD is built, and then we move into a derivation of Coarse Mesh Finite Difference (CMFD), taking special note of the role CMFD plays in combating the poor convergence properties of Source Iteration (SI). (In reality, rather than deriving Coarse Mesh Finite Difference (CMFD) in Ch. 3, we instead derive Nonlinear Diffusion Acceleration (NDA), since we work with the continuous transport equation. We use the name CMFD throughout the thesis for consistency's sake, even when it is technically not proper to do so.)

In Ch. 4, we add multiphysics feedback to CMFD in a straightforward way to produce the Multiphysics CMFD (M-CMFD) and Relaxed CMFD (R-CMFD) methods. We discuss the strengths and weaknesses of these approaches. Next, we introduce two purely-theoretical methods, *Theoretical Method 0* (TM-0) and *Theoretical Method 1* (TM-1), both of which possess an attractive property: they converge in a single iteration. Unfortunately, TM-0 and TM-1 are purely theoretical, they are not implementable in a computer code. We introduce the X-CMFD method as an approximation to TM-1 and discuss how it overcomes the stability and convergence rate issues of both M-CMFD and R-CMFD, but at the cost of placing a significant burden on the thermal hydraulics solver and nuclear data update routines. Finally, at the conclusion of Ch. 4, we introduce a series of approximations to X-CMFD in order to form the *Nonlinearly Implicit Low-Order CMFD* (NILO-CMFD) method,

the main thrust of this thesis. We derive two versions of this approach; the "incomplete" NILO-A and "complete" NILO-CMFD methods. NILO-A is a simplification of NILO-CMFD designed to address problems in which thermal hydraulics solves are expensive but nuclear data updates are relatively cheap.

In Chs. 5 and 6, we consider the application of NILO-A and NILO-CMFD to a simplified, 1-D multiphysics model problem. In Ch. 5, we derive this model in its continuous form by manipulating and simplifying the neutron transport – thermal hydraulics equations governing a 3-D reactor pin cell. In Ch. 6, we (i) discretize the 1-D model, (ii) describe the R-CMFD, X-CMFD, NILO-A, and NILO-CMFD methods applied to the discrete model, and (iii) compare the relative performance of each iterative method through numerical tests on a set of 1-D problems. The numerical results presented in Ch. 6 suggest that, at least for the 1-D model, NILO-A and NILO-CMFD have an equivalent outer iteration convergence rate as X-CMFD. This is our desired result.

In Ch. 7, we discuss the implementation of NILO-A in the 3-D *Michigan Parallel Characteristics-based Transport* (MPACT) reactor physics code coupled to the subchannel thermal hydraulics code COBRA-TF (CTF). (Our implementation of the full NILO-CMFD method in MPACT is incomplete. Therefore, we save its discussion for the future work section of Ch. 8.) We compare the relative performance of NILO-A and R-CMFD on a sequence of assembly and quarter-core problems. On some problems, we see that R-CMFD suffers from a poor convergence rate. In extreme cases, R-CMFD diverges. In contrast, NILO-A converges optimally on all problems tested. Unfortunately, NILO-A's use of full nuclear data evaluations in inner iterations may impose a significant runtime penalty. In some cases, this penalty overrides the positive effects of NILO-A's excellent outer iteration convergence rate. On some problems, we observe NILO-A runtimes greater than R-CMFD, even though NILO-A converges in far fewer outer iterations. We believe that an implementation of the full NILO-CMFD method in MPACT would combat this issue and provide an unambiguous improvement over R-CMFD. Although, due to time constraints, we have not yet implemented the full NILO-CMFD method in MPACT.

In Ch. 8, we summarize our conclusions and propose a bevy of possible future work (including the implementation of the full NILO-CMFD method in MPACT). The NILO-A and NILO-CMFD approaches are quite general, and we hope that it will be possible to extend them to a diverse set of more complicated problems.

Chapter 2.

Problem Definition

In this chapter, we discuss the single-physics components that interact in neutron transport – thermal hydraulics multiphysics problems. We introduce (i) neutron macroscopic cross sections (neutron interaction probabilities), (ii) the Neutron Transport Equation (NTE), and (iii) thermal hydraulics equations. We pay special attention to the interrelation between these single-physics components in multiphysics problems (i.e. how the solution fields and equations of each single-physics are coupled to one another). In particular, we make note of the dependence of macroscopic cross sections on the dependent variables of the thermal hydraulics equations, namely, material temperature and density.

This chapter also introduces notation, the physical rationale behind equations, and a few key simplifying assumptions we make use of throughout this thesis. This chapter *does not* cover iterative solution techniques for solving multiphysics systems; that is done later. Also, for the sake of generality, we focus on the continuous equations (without discretization).

2.1. Neutron Macroscopic Cross Sections

To begin, we discuss *neutron macroscopic cross sections*, which can be understood as neutron interaction probabilities. There exists a multitude of references on this topic (e.g., [25, 61], etc). Here, we present only a brief introduction focused on our immediate needs and refer the reader to the aforementioned references for a more complete treatment.

Since neutrons are neutral (uncharged) particles, they travel through a system in straight paths until they interact with a nuclide comprising the background-material

of the system. (Neutron densities in nuclear reactor applications are sufficiently low that neutron-neutron interaction rates can be ignored.) To clarify, the term "background-material" refers to the physical material in which the neutrons propagate. For example, in a Pressurized Water Reactor (PWR) core, the background material could be water, cladding (zirconium alloy), fuel (uranium oxide), structural material (steel), etc., depending on the position of a neutron.

Macroscopic cross sections, written as Σ_u^n, give the probability per unit neutron-path-length traveled that a neutron will interact through neutron nuclear reaction pathway $u \in (s, \gamma, f, \ldots)$ with a nuclide $n \in ({}^{1}_{1}\mathrm{H}, {}^{2}_{1}\mathrm{H}, \ldots, {}^{12}_{6}\mathrm{C}, \ldots, {}^{235}_{92}\mathrm{U}, {}^{238}_{92}\mathrm{U}, \ldots)$. ($n$ may refer to any known isotope of any known element, e.g., see the chart of the nuclides [82].) If a neutron interacts with a nuclide n, there are many reaction pathways u through which the interaction can proceed, some more probable than others. As a few examples: incident neutrons can (i) scatter off the target nucleus, changing direction and energy, in a *scattering reaction* ($u = s$); (ii) be "absorbed" by the target nucleus in a *capture reaction* ($u = \gamma$); or even (iii) initiate a particularly large and unstable target nucleus to split apart (ejecting a variable number of neutrons and γ-rays in the process) in a *fission reaction* ($u = f$).

The macroscopic cross section Σ_u^n [cm^{-1}] for reaction pathway u and nuclide n can be written as the product of the microscopic cross section σ_u^n (in [cm^2] or [*barns*], with 1 [barn] $\equiv 10^{-24}$ [cm^2]) with the nuclide n number density N_n [cm^{-3}]:

$$\Sigma_u^n = \sigma_u^n N_n \ . \tag{2.1}$$

A *microscopic cross section* σ can be thought of as the effective cross-sectional area of a target nucleus as "seen" by a neutron. (This description can be somewhat misleading, as it omits important quantum mechanical effects that determine the actual microscopic cross section value.)

Eq. (2.1) does not explicitly state the dependence of a macroscopic cross section Σ_u^n on position $\boldsymbol{r} \equiv x\,\hat{\boldsymbol{i}} + y\,\hat{\boldsymbol{j}} + z\,\hat{\boldsymbol{k}}$ (with x, y, and z in [cm], multiplying mutually orthogonal spatial coordinate vectors $\hat{\boldsymbol{i}}$, $\hat{\boldsymbol{j}}$, and $\hat{\boldsymbol{k}}$), neutron energy E (in [J] or [eV]), background material temperature $T(\boldsymbol{r})$ (in [K] or [°C]), and background material density $\rho(\boldsymbol{r})$ [$\mathrm{g{\cdot}cm}^{-3}$]. Rewriting Eq. (2.1) with these dependencies explicitly included, we obtain:

$$\Sigma_u^n(\boldsymbol{r}, T(\boldsymbol{r}), \rho(\boldsymbol{r}), E) = \sigma_u^n(T(\boldsymbol{r}), E) N_n(\boldsymbol{r}, \rho(\boldsymbol{r})) \ . \tag{2.2}$$

In Eq. (2.2), the microscopic cross section σ_u^n depends explicitly on background ma-

terial temperature $T(\boldsymbol{r})$ and neutron energy E. Microscopic thermal motion of the individual atoms comprising the background material leads to the dependence on temperature. This is known as the "Doppler effect". For brevity, we gloss over the complex physics involved in determining microscopic cross sections and simply state that given a nuclide n, reaction pathway u, background material temperature $T(\boldsymbol{r})$, and neutron energy E, the microscopic cross section $\sigma_u^n(T(\boldsymbol{r}), E)$ is available from a nuclear data library that we treat as a black-box. For further information on cross section evaluation, we refer the reader to [11, 28, 100].

The dependence of nuclide n number density N_n on position $\boldsymbol{r}$ and background material density $\rho(\boldsymbol{r})$ is straightforwardly derived. We can rewrite N_n [cm^{-3}] in terms of background material density $\rho(\boldsymbol{r})$ [$\text{g} \cdot \text{cm}^{-3}$], the molar mass of the background material $M(\boldsymbol{r})$ [$\text{g} \cdot \text{mol}^{-1}$], Avogadro's number $N_A \approx 6.02214 \cdot 10^{23}$ [mol^{-1}], and the nuclide n abundance $a_n(\boldsymbol{r})$ [unitless]:

$$N_n(\boldsymbol{r}, \rho(\boldsymbol{r})) = \rho(\boldsymbol{r}) M(\boldsymbol{r})^{-1} N_A \, a_n(\boldsymbol{r}) \; . \tag{2.3}$$

We provide a quick example to illustrate the logic behind Eq. (2.3). Let us consider a Pressurized Water Reactor (PWR) at a given position $\boldsymbol{r}_c$. We are tasked with finding the nuclide number density of deuterium (i.e. hydrogen-2, ${}^2_1\text{D}$) at $\boldsymbol{r}_c$, $N_{{}^2_1\text{D}}(\boldsymbol{r}_c)$ [no. ${}^2_1\text{D} \cdot \text{cm}^{-3}$]. We are told that $\boldsymbol{r}_c$ is located in a reactor coolant channel, and that the background material at $\boldsymbol{r}_c$ is water (H_2O) with a density $\rho(\boldsymbol{r}_c) = 1$ [$\text{g} \cdot \text{cm}^{-3}$], a molar mass of $M(\boldsymbol{r}_c) = 18$ [g $\text{H}_2\text{O} \cdot \text{mol}^{-1}\text{H}_2\text{O}$], and a deuterium abundance in water of $a_{{}^2_1\text{D}}(\boldsymbol{r}_c) = 0.01$ [no. ${}^2_1\text{D} \cdot \text{no.}^{-1}\text{H}_2\text{O}$]. With this information, and using Eq. (2.3), we find:

$$\begin{aligned} N_{{}^2_1\text{D}}(\boldsymbol{r_c}) &= \rho(\boldsymbol{r}_c) M(\boldsymbol{r}_c)^{-1} N_A \, a_{{}^2_1\text{D}}(\boldsymbol{r}_c) \\ &\approx 1 \left[\frac{\text{g } \text{H}_2\text{O}}{\text{cm}^3}\right] \cdot \frac{1}{18} \left[\frac{\text{mol } \text{H}_2\text{O}}{\text{g } \text{H}_2\text{O}}\right] \cdot 6 \times 10^{23} \left[\frac{\text{no.}}{\text{mol}}\right] \cdot 0.01 \left[\frac{\text{no. } {}^2_1\text{D}}{\text{no. } \text{H}_2\text{O}}\right] \\ &\approx 3 \times 10^{20} \left[\frac{\text{no. } {}^2_1\text{D}}{\text{cm}^3}\right] . \end{aligned} \tag{2.4}$$

At this point, we have shown how macroscopic cross sections Σ_u^n depend on both background material temperature $T(\boldsymbol{r})$ and density $\rho(\boldsymbol{r})$ (as well as position $\boldsymbol{r}$ and neutron energy E). Next, we make our first simplifying assumption of this thesis. For the remainder of this document we assume, unless otherwise stated, that we have in our possession an explicit dependence of density $\rho(\boldsymbol{r})$ on temperature $T(\boldsymbol{r})$. That

is, we let ourselves write $\rho(T(\boldsymbol{r}))$, ignoring the blatant abuse of notation. In general, this assumption is not entirely accurate. In reality, thermodynamic state variables (e.g., pressure p, temperature T, density ρ, internal energy u, etc.) are related to one another by equations of state, which usually fix any *one* thermodynamic state variable in terms of any other *two*. Nonetheless, we will assume $\rho(T(\boldsymbol{r}))$ for our discussion of neutron transport – thermal hydraulics multiphysics problems. This assumption is valid in systems where the fluid is mostly single-phase and held at a near constant pressure, e.g., Pressurized Water Reactor (PWR) cores. Using this relationship, we can rewrite Eq. (2.2) as:

$$\Sigma_u^n(\boldsymbol{r}, T(\boldsymbol{r}), E) = \sigma_u^n(T(\boldsymbol{r}), E) N_n(\boldsymbol{r}, \rho(T(\boldsymbol{r}))) \ . \tag{2.5}$$

We note that in moving from Eq. (2.2) to Eq. (2.5), the explicit dependence of the macroscopic cross section Σ_u^n on material density ρ has been removed. The density ρ dependence is now *implied* by the temperature T dependence! Now, the only thermodynamic state variable our macroscopic cross sections explicitly depend on is temperature T. The above simplification is not required by NILO-CMFD, nor any of the other multiphysics iterative methods we discuss later, but it greatly simplifies notation and is reasonable and sufficiently accurate to describe Pressurized Water Reactor (PWR) cores. So, we use it!

Our work is primarily focused on steady-state multiphysics problems. We do not concern ourselves with time-dependent burn-up (i.e. depletion) problems (except for feedback-depletion numerical results presented in Ch. 7). Owing to this, we do not need to know the specific nuclide n in the background material with which a neutron interacts. That kind of information is too detailed for our needs. By summing the macroscopic cross sections Σ_u^n for reaction pathway u over all nuclides n at position $\boldsymbol{r}$, we obtain Σ_u, the probability per unit neutron-path-length traveled that a neutron will have a reaction through pathway u with *any* nuclide at position $\boldsymbol{r}$:

$$\Sigma_u(\boldsymbol{r}, T(\boldsymbol{r}), E) = \sum_{n \in N_{\text{nuc}}(\boldsymbol{r})} \Sigma_u^n(\boldsymbol{r}, T(\boldsymbol{r}), E) \ . \tag{2.6}$$

(In Eq. (2.6), we have already started using Eq. (2.5) in place of Eq. (2.2)). The dependence of Σ_u on density ρ is *implied* by its dependence on temperature T.) In Eq. (2.6), we have used $N_{\text{nuc}}(\boldsymbol{r})$ to refer to the set of nuclides in the background material located at position $\boldsymbol{r}$ (not to be confused with the nuclide number density N_n

appearing in Eqs. (2.1), (2.2), etc.). For example, if we were considering a position $\boldsymbol{r}_c$ located in water (H_2O) containing the oxygen isotopes ${}^{16}_{8}\mathrm{O}$ and ${}^{18}_{8}\mathrm{O}$ and the hydrogen isotopes ${}^{1}_{1}\mathrm{H}$ and ${}^{2}_{1}\mathrm{D}$, we would write $n \in N_{\mathrm{nuc}}(\boldsymbol{r}_c)$ as $n \in ({}^{16}_{8}\mathrm{O}, {}^{18}_{8}\mathrm{O}, {}^{1}_{1}\mathrm{H}, {}^{2}_{1}\mathrm{D})$.

In the upcoming section on neutron transport, we will see other nuclear data, in addition to macroscopic cross sections Σ_u, that depend explicitly on temperature $T(\boldsymbol{r})$ (with an implied density ρ dependence). These include (i) the double differential scattering cross section $\Sigma_s(\boldsymbol{r}, T(\boldsymbol{r}), \hat{\boldsymbol{\Omega}}' \cdot \hat{\boldsymbol{\Omega}}, E' \to E)$, (ii) the fission neutron spectrum $\chi(\boldsymbol{r}, T(\boldsymbol{r}), E)$, (iii) the mean number of neutrons produced in a fission reaction ν multiplied by the macroscopic fission cross section Σ_f: $\nu\Sigma_f(\boldsymbol{r}, T(\boldsymbol{r}), E)$, and (iv) the amount of thermal energy deposited locally by heavy fission products per fission reaction κ [J] multiplied by the macroscopic fission cross section Σ_f: $\kappa\Sigma_f(\boldsymbol{r}, T(\boldsymbol{r}), E)$. [We never treat ν or κ individually, these symbols will always appear multiplying the macroscopic fission cross section (i.e. as $\nu\Sigma_f$ and $\kappa\Sigma_f$).]

2.2. Neutron Transport

Next, we introduce the *Neutron Transport Equation* (NTE). The NTE governs the distribution of energetic free neutrons in a system in which neutrons interact with the system's background material according to prescribed interaction probabilities (e.g., macroscopic cross sections). The NTE assumes that neutrons traveling through the system do not interact with one another. The NTE may be written in several forms, with the chosen form usually depending on the characteristics of, or requested information from, the physical system of interest. As a few examples, time-dependent, steady-state, α-eigenvalue, and k- (or λ-) eigenvalue variants of the NTE are used in a diverse assortment of engineering applications. For our particular area of focus, the study of nuclear reactors, the k-eigenvalue formulation of the NTE is convenient and commonly used.

In a fissioning (i.e. multiplying) system, such as a nuclear reactor core, a steady-state, nonzero physical distribution of neutrons does not necessarily exist. To visualize this, we give a contrived example. Imagine a cube of purely fissioning material with a fission neutron multiplicity of $\nu = 2$. In other words, the only interaction neutrons are able to have with the background material is to initiate a fission reaction, thereby removing the interacting neutron and forming two neutrons in its place (for a net gain of one neutron per interaction). We also assume that the fissioning material (i.e. fuel) in the cube never depletes (i.e. never runs out). Finally, we assume the cube is

surrounded by a perfect reflector. Every neutron that attempts to leave the system is scattered back in (i.e. returned).

A steady-state, nonzero physical distribution of neutrons is not possible for this system. The system is *supercritical*, meaning that any nonzero neutron population in the system increases in time without bound. There cannot be an equilibrium (i.e. steady-state) distribution of neutrons because the system will always tend towards generating more neutrons than were initially present. *The steady-state NTE cannot describe this system nor many others like it (including those of interest to the nuclear reactor community).*

Instead of using the steady-state NTE, we must resort to another NTE formulation to describe this system. We could use the time-dependent NTE to describe the dynamics of the system; however, this equation is much too costly to solve over the time-scales of interest for most engineering applications. Instead, we settle on either the α or k-eigenvalue formulations, both of which approximately describe the dynamics of a system, importantly, without including a time-derivative! For nuclear reactor applications, the k-eigenvalue formulation is preferred, so it is our focus.

In Eqs. (2.7), we write the continuous, three-dimensional, energy-dependent, k (or rather, λ, with $\lambda = k^{-1}$)-eigenvalue neutron transport equation with general anisotropic scattering and temperature dependent cross sections. (As per our discussion in the previous section, the macroscopic cross sections also have an implied dependence on density through their explicit temperature dependence, e.g., Eq. (2.5).)

$$\begin{aligned}
&\hat{\boldsymbol{\Omega}} \cdot \boldsymbol{\nabla} \psi(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E) + \Sigma_t(\boldsymbol{r}, T(\boldsymbol{r}), E) \psi(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E) \\
&\quad = \int_0^\infty \int_{4\pi} \Sigma_s(\boldsymbol{r}, T(\boldsymbol{r}), \hat{\boldsymbol{\Omega}}' \cdot \hat{\boldsymbol{\Omega}}, E' \to E) \psi(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E') \, \mathrm{d}\Omega' \, \mathrm{d}E' \\
&\qquad + \frac{\lambda}{4\pi} \chi(\boldsymbol{r}, T(\boldsymbol{r}), E) \int_0^\infty \int_{4\pi} \nu \Sigma_f(\boldsymbol{r}, T(\boldsymbol{r}), E') \psi(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E') \, \mathrm{d}\Omega' \, \mathrm{d}E' \, , \\
&\qquad\qquad \boldsymbol{r} \in V \, , \quad \hat{\boldsymbol{\Omega}} \in 4\pi \, , \quad 0 < E < \infty \, ,
\end{aligned} \tag{2.7a}$$

$$\begin{aligned}
\psi(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E) &= g(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E) \, , \\
\boldsymbol{r} \in \partial V \, , \quad \hat{\boldsymbol{\Omega}} \cdot \hat{\boldsymbol{n}}(\boldsymbol{r}) &< 0 \, , \quad 0 < E < \infty \, ,
\end{aligned} \tag{2.7b}$$

$$P = \int_0^\infty \int_{4\pi} \int_V \kappa \Sigma_f(\boldsymbol{r}', T(\boldsymbol{r}'), E') \psi(\boldsymbol{r}', \hat{\boldsymbol{\Omega}}', E') \, \mathrm{d}r' \, \mathrm{d}\Omega' \, \mathrm{d}E' \, . \tag{2.7c}$$

Two aspects of Eqs. (2.7) are nonstandard. First, for notational convenience, we use the λ- rather than the k-eigenvalue, with $\lambda = k^{-1}$. Second, several parameters are

written with an explicit dependence on the background material temperature $T(\boldsymbol{r})$, owing to our interest in transport problems with thermal hydraulic feedback. We note that the density dependence of Σ_t, Σ_s, χ, $\nu\Sigma_f$, and $\kappa\Sigma_f$ are implied by the temperature dependence, similarly to how Eq. (2.2) was converted to Eq. (2.5) in the previous section. As a reminder, this simplification is possible because we assume that we have an explicit dependence of material density on temperature available, $\rho(T(\boldsymbol{r}))$. The temperature dependence of macroscopic cross sections and related nuclear data in Eqs. (2.7) acts as the main influence from the thermal hydraulics solution, discussed in the next section, influencing the transport solution. Otherwise, Eqs. (2.7) are written in standard notation for the scalar eigenvalue λ and the neutron angular flux eigenfunction ψ , a function of position $\boldsymbol{r}$ in the arbitrary volume V, direction $\hat{\boldsymbol{\Omega}}$ on the unit sphere 4π, and any positive energy E.

To provide physical context to Eqs. (2.7), we note that the angular flux $\psi(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E)$ represents the neutron path-length-generation rate per differential volume dr about $\boldsymbol{r}$, differential solid angle dΩ about $\hat{\boldsymbol{\Omega}}$, and differential energy dE about E. For *critical* systems $\lambda = 1$, for *subcritical* systems $\lambda > 1$, and for *supercritical* systems $\lambda < 1$, by definition.

Eq. (2.7b) specifies the angular flux boundary condition on the external surface ∂V of the problem domain V for each incoming direction $\hat{\boldsymbol{\Omega}} \cdot \hat{\boldsymbol{n}}(\boldsymbol{r}) < 0$ (with $\hat{\boldsymbol{n}}(\boldsymbol{r})$ as the outward-pointing surface normal at position $\boldsymbol{r}$ on ∂V) and all energies $0 < E < \infty$. The function g specifies the boundary condition type at location $\boldsymbol{r}$ on ∂V. For example,

(i) vacuum boundary : $g(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E) = 0$,

(ii) reflective boundary : $g(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E) = \psi(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}_r(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}), E)$,
with $\hat{\boldsymbol{\Omega}}_r(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}) = \hat{\boldsymbol{\Omega}} - 2(\hat{\boldsymbol{\Omega}} \cdot \hat{\boldsymbol{n}}(\boldsymbol{r}))\hat{\boldsymbol{n}}(\boldsymbol{r})$ as the outgoing direction corresponding to the incoming reflected direction $\hat{\boldsymbol{\Omega}}$ at $\boldsymbol{r}$ on ∂V.

Since Eq. (2.7a) is in eigenvalue form, we require a normalization condition to specify a unique, positive (i.e. dominant) eigenfunction solution $\psi(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E)$. In Eq. (2.7c), we normalize by specifying the total system thermal power P (generated by the local energy deposition of heavy fission products).

An important feature of *nonlinear* (multiphysics) λ-eigenvalue problems (as compared to their *linear* single-physics counterparts) is that varying the normalization constant (in this case, P in Eq. (2.7c)) preserves neither the eigenvalue λ nor the shape of the corresponding eigenfunction $\psi(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E)$. In a physical nuclear reactor,

it would be undesirable, and potentially catastrophic from a reactor-safety point of view, if increasing the reactor power did not affect the reactor's dominant eigenvalue λ. Instead, we want the eigenvalue λ to increase (tend towards sub-criticality) as the system thermal power (and, in turn, the average system temperature) increases. This is actually a design requirement of most modern reactors. This is sometimes referred to as a core having a "*negative temperature coefficient of reactivity*". Here, reactivity is defined as $\rho \equiv (k-1)/k = 1 - \lambda$ (we note that as λ increases, ρ decreases).

To proceed, we introduce the *neutron scalar flux* ϕ, defined by:

$$\phi(\boldsymbol{r}, E) \equiv \int_{4\pi} \psi(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E) \, \mathrm{d}\Omega' \, . \tag{2.8}$$

The scalar flux ϕ appears in the low-order, angularly-integrated acceleration equations we introduce in Chs. 3 and 4 when describing both single-physics (pure transport) and multiphysics iterative methods. At the moment, for our description of the components of the neutron transport – thermal hydraulics multiphysics problem, the usefulness of the scalar flux is that it acts as a main component of the link from the transport solution influencing the thermal hydraulics solution.

To be more specific, the product of scalar flux ϕ and macroscopic cross section Σ_u integrated over energy gives neutron interaction rates with the background material. For a given reaction pathway u, the quantity:

$$\int_0^\infty \Sigma_u(\boldsymbol{r}, T(\boldsymbol{r}), E') \phi(\boldsymbol{r}, E') \, \mathrm{d}E' \, ,$$

gives the expected rate of reactions of type u occurring per differential volume $\mathrm{d}r$ about $\boldsymbol{r}$. From this description, and noting the definition of κ as the thermal energy deposited locally by heavy fission products per fission reaction, we see that the rate of thermal energy deposited locally by fission reactions per differential volume $\mathrm{d}r$ about $\boldsymbol{r}$ is given by the *fission heat source distribution* $h(\boldsymbol{r})$, defined by:

$$h(\boldsymbol{r}) \equiv \int_0^\infty \kappa \Sigma_f(\boldsymbol{r}, T(\boldsymbol{r}), E') \phi(\boldsymbol{r}, E') \, \mathrm{d}E' \, . \tag{2.9}$$

In our discussion of neutron transport – thermal hydraulics multiphysics problems, we only consider the local (heavy fission product) fission heat source distribution's ($h(\boldsymbol{r})$) influence on the thermal hydraulics equations. (We ignore nonlocal energy deposition via fission-produced γ-ray transport, for example.)

2.3. Thermal Hydraulics

With the heat source distribution $h(\boldsymbol{r})$ defined in Eq. (2.9), we now move on to a description of the thermal hydraulics equations. Our discussion of thermal hydraulics, presented below, is more abstract than our previous discussion of neutron macroscopic cross sections and neutron transport. There is a practical reason for this abstractness, which we discuss next.

State-of-the-art nuclear reactor research codes, both deterministic and stochastic (e.g., MPACT [55], etc.), use the neutron transport equation for their neutronics description. These codes favor the accuracy and generality of neutron transport over the simulation of simpler, easier-to-solve, approximate neutronics equations, e.g., neutron diffusion, point kinetics, etc. The accuracy of the Neutron Transport Equation (NTE) is an especially convenient feature. In theory, the λ-eigenvalue formulation of the NTE [Eqs. (2.7)] can be used to simulate *any* fissioning system, e.g., Pressurized Water Reactors (PWR)'s, Boiling Water Reactors (BWR)'s, Molten Salt Reactors (MSR)'s, High-Temperature Gas-cooled Reactors (HTGR)'s, etc. Therefore, one can use Eqs. (2.7) to describe the neutronic treatment in most state-of-the-art reactor physics codes, applied to any reactor (of any type, e.g., PWR, BWR, etc.) sufficiently near criticality $\lambda \approx 1$. (However, the dominant eigenfunction in the λ-eigenvalue formulation of the NTE is physical *only* when $\lambda = 1$. Otherwise, for $\lambda \neq 1$ and especially for $\lambda \ll 1$ or $\lambda \gg 1$, it is an approximate, steady-state description of an inherently time-dependent physical system.)

The situation is more complex for thermal hydraulics. The neutronics description is fixed by Eqs. (2.7) to keep in line with the state-of-the-art in reactor physics simulation. Eqs. (2.7) are a valid description for most reactor applications. In contrast, depending on the physical system of interest and the desired level of thermal hydraulic fidelity, modern reactor physics codes tend to allow for greater flexibility in the thermal hydraulics description. For example, the mathematical description of the flow of single-phase water, multi-phase water, molten salt, and high-temperature gas in a PWR, BWR, MSR, and HTGR, respectively, are considerably different. While in theory a set of multiphase Navier Stokes Equations with suitable equations of state could solve all of these problems, that description is far outside the realm of possibility to be simulated over a reactor core's dimensions on today's computer hardware. Instead, reactor physics codes tend to allow for coupling with a wide range of thermal hydraulics codes solving different sets of equations. Different thermal hy-

draulics descriptions are used for different situations via *loose-coupling* with various external thermal hydraulics solvers (codes, libraries). For these reasons, one cannot write down a single set of equations describing thermal hydraulics in near complete generality, as we were able to do for neutronics with Eqs. (2.7).

Instead, we choose to introduce a general thermal hydraulics operator $\mathcal{L}$, leaving the definition of $\mathcal{L}$ vague until we specify, or are given, a concrete thermal hydraulics description. We make the assumption that the thermal hydraulics description, whatever it may be, takes the form of Eq. (2.10), with the fission heat source distribution $h(\boldsymbol{r})$ defined by Eq. (2.9). We make no assumptions as to whether the operator $\mathcal{L}$ is linear or nonlinear:

$$\mathcal{L}T(\boldsymbol{r}) = h(\boldsymbol{r}) \ , \quad \boldsymbol{r} \in V \ . \tag{2.10}$$

We note that Eq. (2.10) *does not* include the density $\rho(\boldsymbol{r})$. If we were being completely general, we would allow the thermal hydraulics solver to determine both temperature $T(\boldsymbol{r})$ and density $\rho(\boldsymbol{r})$, for use in evaluating macroscopic cross sections via something like Eq. (2.2). Instead, we assume that, given $T(\boldsymbol{r})$, we are able to evaluate $\rho(T(\boldsymbol{r}))$ and use the equivalent of Eq. (2.5) to update our macroscopic cross sections.

Since our description of thermal hydraulics in Eq. (2.10) is abstract, it may help to give a few concrete examples. We do this next:

(i) First, we consider a fissioning solid, encompassing an arbitrary domain V. The solid conducts heat (via thermal diffusion) but neither deforms (via thermal expansion) nor moves (via any macroscopic advective process). The thermal conductivity k [$\mathrm{W \cdot cm^{-1} \cdot K^{-1}}$] of this solid depends on space $\boldsymbol{r}$ but not on temperature $T(\boldsymbol{r})$ [K]. The solid is surrounded by an infinite heat sink held at a fixed temperature T_{sink} [K]. A simple diffusion equation with a Dirichlet boundary condition specified on the boundary ∂V describes the temperature profile $T(\boldsymbol{r})$ [K] in the domain V in response to a prescribed fission heat source distribution $h(\boldsymbol{r})$ [$\mathrm{W \cdot cm^{-3}}$]:

$$-\boldsymbol{\nabla} \cdot k(\boldsymbol{r}) \boldsymbol{\nabla} T(\boldsymbol{r}) = h(\boldsymbol{r}) \ , \quad \boldsymbol{r} \in V \ , \tag{2.11a}$$

$$T(\boldsymbol{r}) = T_{\text{sink}} \ , \quad \boldsymbol{r} \in \partial V \ . \tag{2.11b}$$

(In the multiphysics problems we are interested in, the fission heat source distribution $h(\boldsymbol{r})$ is not prescribed; instead, it is one of the unknowns that must be iteratively solved for in the nonlinear system.)

For the thermal hydraulics problem described by Eqs. (2.11), the operator $\mathcal{L}$ in Eq. (2.10) takes the form of a linear diffusion operator:

$$\begin{aligned} \mathcal{L}T(\boldsymbol{r}) \equiv -\boldsymbol{\nabla}\cdot k(\boldsymbol{r})\boldsymbol{\nabla}T(\boldsymbol{r}) \ , \quad \boldsymbol{r}\in V \ , \\ \text{with} \ \ T(\boldsymbol{r}) = T_{\text{sink}} \ , \quad \boldsymbol{r}\in \partial V \ , \end{aligned} \tag{2.12}$$

where, for brevity, we have included the Dirichlet boundary condition Eq. (2.11b) within the definition of $\mathcal{L}$ in Eq. (2.12). This is a convention we use throughout the thesis: thermal hydraulic operators $\mathcal{L}$ include boundary condition information.

(ii) As another example, let us consider an arbitrary volume V in space enclosed by the boundary surface ∂V. An incompressible, fissile fluid flows through V with (i) a constant density ρ_0 [$\mathrm{g}\cdot\mathrm{cm}^{-3}$], (ii) a constant specific heat c_0 [$\mathrm{J}\cdot\mathrm{g}^{-1}\cdot\mathrm{K}^{-1}$], (iii) a space and temperature dependent thermal conductivity $k(\boldsymbol{r},T(\boldsymbol{r}))$ [$\mathrm{W}\cdot\mathrm{cm}^{-1}\cdot\mathrm{K}^{-1}$], and (iv) a specified velocity profile $\boldsymbol{v}(\boldsymbol{r})$ [$\mathrm{cm}\cdot\mathrm{s}^{-1}$] satisfying mass conservation (i.e. $\boldsymbol{\nabla}\cdot\boldsymbol{v}(\boldsymbol{r})=0$). For boundary conditions, we split the domain surface ∂V into two non-overlapping surfaces ∂V_{d} and ∂V_{n}, with $\partial V_d \cup \partial V_n = \partial V$, $\partial V_d \cap \partial V_n = \varnothing$, and $\partial V_d \neq \varnothing$. On ∂V_n, we set the Neumann boundary condition $\boldsymbol{\nabla}T(\boldsymbol{r})\cdot\hat{\boldsymbol{n}}(\boldsymbol{r})=0$. On ∂V_d, we set the Dirichlet condition $T(\boldsymbol{r})=T_{\mathrm{in}}$.

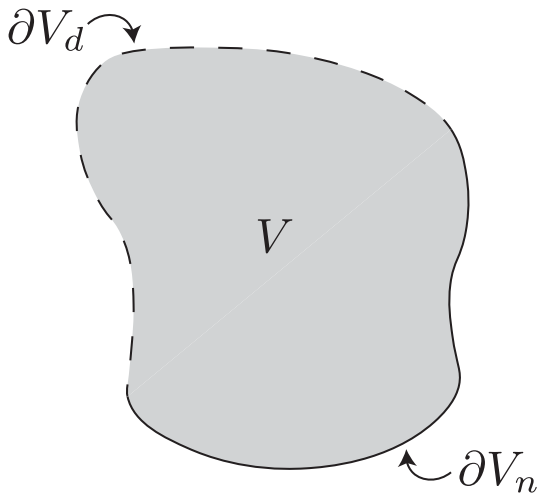


Figure 2.1: Sample Thermal Hydraulics Problem Domain.

In response to a specified fission heat source distribution $h(\boldsymbol{r})$ [$\mathrm{W}\cdot\mathrm{cm}^{-3}$], the temperature $T(\boldsymbol{r})$ [K] in V is governed by the nonlinear (owing to $k(\boldsymbol{r},T(\boldsymbol{r}))$) advection-diffusion equation:

$$-\boldsymbol{\nabla}\cdot k(\boldsymbol{r},T(\boldsymbol{r}))\boldsymbol{\nabla}T(\boldsymbol{r})+\rho_0 c_0\boldsymbol{v}(\boldsymbol{r})\cdot\boldsymbol{\nabla}T(\boldsymbol{r})=h(\boldsymbol{r})\ ,\quad \boldsymbol{r}\in V\ , \tag{2.13a}$$

$$\boldsymbol{\nabla}T(\boldsymbol{r})\cdot\hat{\boldsymbol{n}}(\boldsymbol{r})=0\ ,\quad \boldsymbol{r}\in\partial V_n\ , \tag{2.13b}$$

$$T(\boldsymbol{r})=T_{\mathrm{in}}\ ,\quad \boldsymbol{r}\in\partial V_d\ . \tag{2.13c}$$

In order to write Eqs. (2.13) in the form of Eq. (2.10), we define the operator $\mathcal{L}$ to be the *nonlinear* advection-diffusion operator:

$$\begin{aligned}\mathcal{L}T(\boldsymbol{r})\equiv&-\boldsymbol{\nabla}\cdot k(\boldsymbol{r},T(\boldsymbol{r}))\boldsymbol{\nabla}T(\boldsymbol{r})+\rho_0 c_0\boldsymbol{v}(\boldsymbol{r})\cdot\boldsymbol{\nabla}T(\boldsymbol{r})\ ,\quad \boldsymbol{r}\in V\ ,\\ &\text{with}\ \ \boldsymbol{\nabla}T(\boldsymbol{r})\cdot\hat{\boldsymbol{n}}(\boldsymbol{r})=0\ ,\quad \boldsymbol{r}\in\partial V_n\ ,\\ &\text{and}\ \ T(\boldsymbol{r})=T_{\mathrm{in}}\ ,\quad \boldsymbol{r}\in\partial V_d\ ,\end{aligned} \tag{2.14}$$

where, again, we have included boundary conditions in the definition of $\mathcal{L}$.

We note that combining either the definition of $\mathcal{L}$ given in Eq. (2.12) or Eq. (2.14) with Eqs. (2.6), (2.7), and (2.10) results in a fully specified, neutron transport – thermal hydraulics multiphysics problem (given known material properties, the dependence of density on temperature, and neutronics parameter data). NILO-CMFD and related multiphysics iterative method we discuss later in this thesis are designed to solve these (and more general) types of problems.

Chapter 3.

Transport Iterative Methods

In the previous chapter, we introduced physical and mathematical descriptions of the single-physics components that interact in the nuclear reactor core, neutron transport – thermal hydraulics multiphysics problem. In order to solve this multiphysics problem, we must simultaneously consider the coupled set of equations describing macroscopic cross sections [Eqs. (2.5)-(2.6)], neutron transport [Eqs. (2.7)], and thermal hydraulics [Eqs. (2.9)-(2.10)]. (We also require physical problem parameters, e.g., domain size, system power, etc., along with a concrete definition of the thermal hydraulics operator $\mathcal{L}$, e.g., Eqs. (2.12), (2.14), etc.)

This thesis focuses on the use of loosely-coupled iterative methods based on Coarse-Mesh Finite Difference (CMFD) to solve neutron transport – thermal hydraulics multiphysics problems. Coarse-Mesh Finite Difference (CMFD) is a diffusion-based, transport acceleration technique that is widely used in modern deterministic reactor physics codes (e.g., MPACT [55], OpenMOC [7], etc.). For single-physics transport (i.e. problems without feedback), CMFD has a strong theoretical foundation and a proven track record of successful application over the past several decades. For single-physics problems in an optimal setting (i.e. with satisfactory discretization), CMFD converges linearly with an optimal spectral radius ρ of approximately 0.22. Given reasonable convergence criteria, this translates to convergence in roughly 10-20 iterations.

Unfortunately, when CMFD is naïvely applied to multiphysics problems (i.e. problems with feedback) in a simple relaxation-based approach that we refer to as *Relaxed-CMFD* (R-CMFD), the performance of the method suffers. In the presence of feedback, R-CMFD may not achieve the optimal CMFD spectral radius ($\rho \approx 0.22$), regardless of discretization. *The presence of feedback increases R-CMFD's spectral*

radius, resulting in a degraded convergence rate. In extreme scenarios, this degradation can result in divergence.

Before we introduce R-CMFD and its associated multiphysics-driven issues (along with NILO-CMFD and related iterative approaches designed to combat R-CMFD's limitations), we first derive and discuss CMFD in a single-physics setting (i.e. as applied to a neutron transport problem without thermal hydraulic feedback). This derivation is useful for several reasons:

(i) The single-physics neutron transport equation and the iterative methods designed to solve it are much simpler to describe and understand than their multiphysics counterparts. Temporarily focusing on single-physics transport gives us a relatively uncluttered workspace to introduce concepts and notation used throughout the remainder of this thesis.

(ii) Our strategy in Ch. 4 and beyond is to modify the CMFD procedure to robustly and efficiently include multiphysics feedback. Before including feedback, it is informative to see the motivation behind CMFD and how it behaves in the single-physics setting for which it was originally designed. Later, this will allow us to (i) inspect the ways in which CMFD *"misbehaves"* in the presence of multiphysics feedback, and (ii) devise methods to address this misbehavior.

(We quickly pause for a note on terminology. Coarse-Mesh Finite Difference (CMFD) is a method designed explicitly with discretized transport problems in mind. As the name suggests, *Coarse-Mesh* Finite Difference (CMFD) employs a coarse mesh (in space and/or energy). To mimic terminology used in the literature, the application of the method towards a continuous problem is more accurately described as *Nonlinear Diffusion Acceleration* (NDA). Regardless, we use the term "CMFD" throughout this thesis for both the continuous and discretized cases. While technically incorrect, this simplifies our discussion.)

The remainder of this single-physics-focused chapter is structured as follows. First, we describe the single-physics transport problem used to introduce CMFD. This problem is a simple reframing of Eqs. (2.7), removing the dependence of macroscopic cross sections and related nuclear data on background material temperature (and density). Next, we describe the Source Iteration (SI) method for iteratively solving transport eigenvalue problems. Coarse-Mesh Finite Difference (CMFD) augments Source Iteration (SI) with a low-order, transport-corrected diffusion problem. After introducing SI, we derive CMFD's low-order diffusion problem and outline the CMFD iterative

method as a whole. Again, all discussions in this chapter are focused on continuous, single-physics neutron transport. *Feedback is momentarily ignored.*

3.1. Transport Problem (Single-Physics)

In Eqs. (3.1), we rewrite Eqs. (2.7) as a single-physics, λ-eigenvalue transport problem:

$$\begin{aligned}
&\hat{\boldsymbol{\Omega}} \cdot \boldsymbol{\nabla}\psi(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E) + \Sigma_t(\boldsymbol{r}, E)\psi(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E) \\
&\qquad = \frac{1}{4\pi}\int_0^\infty \Sigma_s(\boldsymbol{r}, E' \to E)\phi(\boldsymbol{r}, E')\,\mathrm{d}E' \\
&\qquad\qquad + \lambda\frac{\chi(\boldsymbol{r}, E)}{4\pi}\int_0^\infty \nu\Sigma_f(\boldsymbol{r}, E')\phi(\boldsymbol{r}, E')\,\mathrm{d}E' \;, \\
&\qquad\qquad\qquad \boldsymbol{r} \in V \;,\quad \hat{\boldsymbol{\Omega}} \in 4\pi \;,\quad 0 < E < \infty \;,
\end{aligned} \tag{3.1a}$$

$$P = \int_0^\infty \int_V \kappa\Sigma_f(\boldsymbol{r}', E')\phi(\boldsymbol{r}', E')\,\mathrm{d}^3r'\,\mathrm{d}E' \;. \tag{3.1b}$$

As before, we define the neutron scalar flux ϕ to be the zeroth angular moment of the angular flux ψ:

$$\phi(\boldsymbol{r}, E) \equiv \int_{4\pi} \psi(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E)\,\mathrm{d}\Omega' = \text{neutron scalar flux} \;. \tag{3.1c}$$

We also define the neutron current $\boldsymbol{J}$ (a **vector**) to be the first angular moment of the angular flux ψ:

$$\boldsymbol{J}(\boldsymbol{r}, E) \equiv \int_{4\pi} \hat{\boldsymbol{\Omega}}'\psi(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E)\,\mathrm{d}\Omega' = \text{neutron current} \;. \tag{3.1d}$$

We have omitted boundary conditions in Eqs. (3.1) for simplicity. We have also assumed isotropic scattering:

$$\Sigma_s(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}' \cdot \hat{\boldsymbol{\Omega}}, E' \to E) \implies \frac{1}{4\pi}\Sigma_s(\boldsymbol{r}, E' \to E) \;.$$

We maintain this assumption throughout the remainder of this thesis.

With the inclusion of angular flux boundary conditions and appropriate problem parameter values (geometry, nuclear data, normalization constant, etc.), Eqs. (3.1) fully specify a single-physics, λ-eigenvalue transport problem with a unique, posi-

tive eigenfunction solution ψ and corresponding scalar eigenvalue λ. Since there is no temperature T (or density ρ) dependence in Eqs. (3.1), we do not require additional thermal hydraulics equations. In the remainder of this chapter, we explore the iterative solution of the single-physics Eqs. (3.1) using Source Iteration (SI) and Coarse-Mesh Finite Difference (CMFD).

3.2. Source Iteration (SI)

We start by describing the Source Iteration (SI) iterative method for solving Eqs. (3.1). CMFD builds off of SI, as we will see later, so SI is a natural place to begin.

Source Iteration (SI) involves the repeated inversion of the transport leakage-collision operator (i.e. the operator $(\hat{\boldsymbol{\Omega}} \cdot \boldsymbol{\nabla} + \Sigma_t(\boldsymbol{r}, E))(\cdot)$ acting on the angular flux ψ on the left-hand side of Eq. (3.1a)) in response to lagged estimates of the scattering and (eigenvalue-scaled) fission sources (i.e. the right-hand side of Eq. (3.1a)). The inversion of the leakage-collision operator is often performed via a "transport sweep", which is described later. *Source Iteration (SI) solves Eqs. (3.1) through a sequence of iterated transport sweeps.* We outline SI in detail below:

Source Iteration (SI) Step 0 : Initialization

Before iterating, we require initial estimates of the scalar flux $\phi^{(0)}$ and eigenvalue $\lambda^{(0)}$. Initial estimates can be obtained through a variety of techniques (e.g., guessing, solving a diffusion problem, etc.). (We use superscripts enclosed by parentheses to differentiate between iterated quantities evaluated at different points during iteration. This should not be confused with raising a quantity to a power. For example, $\phi^{(n)}$ is not the same as ϕ^n. The former is the scalar flux ϕ estimated at the start of the $(n+1)^{\text{st}}$ iteration, while the latter is the scalar flux raised to the n^{th} power.) Once initial estimates are obtained, we set the iteration index n to 0 and proceed to SI Step 1 to start the first iteration.

SI Step 1 : Neutron Transport Sweep

We use the most recent scalar flux $\phi^{(n)}$ and eigenvalue $\lambda^{(n)}$ estimates to construct lagged scattering and (eigenvalue-scaled) fission sources. We replace the right-hand side of Eq. (3.1a) with the estimated source, creating the following fixed-source transport problem that is solved for the updated angular flux $\psi^{(n+\frac{1}{2})}$

(we ignore boundary conditions for simplicity):

$$\begin{aligned}\hat{\boldsymbol{\Omega}} \cdot \boldsymbol{\nabla}\psi^{(n+\frac{1}{2})}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E) + \Sigma_t(\boldsymbol{r}, E)\psi^{(n+\frac{1}{2})}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E) \\ = \frac{1}{4\pi}\int_0^\infty \Sigma_s(\boldsymbol{r}, E' \to E)\phi^{(n)}(\boldsymbol{r}, E')\,\mathrm{d}E' \\ + \lambda^{(n)}\frac{\chi(\boldsymbol{r}, E)}{4\pi}\int_0^\infty \nu\Sigma_f(\boldsymbol{r}, E')\phi^{(n)}(\boldsymbol{r}, E')\,\mathrm{d}E' \,.\end{aligned} \tag{3.2a}$$

While calculating $\psi^{(n+\frac{1}{2})}$ from Eq. (3.2a), we update the neutron scalar flux $\phi^{(n+\frac{1}{2})}$ and current vector $\boldsymbol{J}^{(n+\frac{1}{2})}$ estimates:

$$\phi^{(n+\frac{1}{2})}(\boldsymbol{r}, E) = \int_{4\pi} \psi^{(n+\frac{1}{2})}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E)\,\mathrm{d}\Omega' \,, \tag{3.2b}$$

$$\boldsymbol{J}^{(n+\frac{1}{2})}(\boldsymbol{r}, E) = \int_{4\pi} \hat{\boldsymbol{\Omega}}'\psi^{(n+\frac{1}{2})}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E)\,\mathrm{d}\Omega' \,. \tag{3.2c}$$

In practical computer simulations, Eqs. (3.2) are "solved" in a memory-efficient manner using a transport sweep. Owing to the structure of the leakage-collision operator in Eq. (3.2a), a transport sweep can update the scalar flux $\phi^{(n+\frac{1}{2})}$ and current $\boldsymbol{J}^{(n+\frac{1}{2})}$ estimates without ever storing the full angular flux $\psi^{(n+\frac{1}{2})}$ in computer memory. In effect, we are able to *march* through the problem domain in a particular direction, updating *downstream* angular fluxes ψ while forgetting (removing from memory) *upstream* angular fluxes. The desire to avoid storing ψ in full stems from its six-dimensional phase-space. Working with iterated quantities that live in a reduced, angularly integrated, four-dimensional phase-space is preferred, e.g., $\phi^{(n+\frac{1}{2})}$ and $\boldsymbol{J}^{(n+\frac{1}{2})}$. We also mention that transport sweeps tend to be computationally expensive, and the number of sweeps should be minimized as much as possible. This motivates transport acceleration schemes such as CMFD. Transport acceleration schemes perform more work each iteration, in addition to the required transport sweep, in order to reduce the number of iterations, and hence, the number of transport sweeps required to reach convergence.

SI Step 2 : Eigenvalue Update and Normalization

To compute the end-of-iteration eigenvalue estimate $\lambda^{(n+1)}$, we integrate the transport equation [Eq. (3.1a)] over all of phase-space (space, angle, energy) and evaluate the resulting expression with the sweep-determined scalar flux $\phi^{(n+\frac{1}{2})}$

and current $\boldsymbol{J}^{(n+\frac{1}{2})}$. Solving for the eigenvalue λ and using it as the updated estimate $\lambda^{(n+1)}$ gives:

$$\lambda^{(n+1)} = \frac{\iint \boldsymbol{J}^{(n+\frac{1}{2})} \cdot \hat{\boldsymbol{n}} \, \mathrm{d}^2 r' \mathrm{d}E' + \iint \Sigma_a \phi^{(n+\frac{1}{2})} \, \mathrm{d}^3 r' \, \mathrm{d}E'}{\iint \nu \Sigma_f \phi^{(n+\frac{1}{2})} \, \mathrm{d}^3 r' \, \mathrm{d}E'} \, , \tag{3.3}$$

where dependent variables and integration bounds have been omitted from Eq. (3.3) for simplicity. We note that the spatial integral containing the neutron current $\boldsymbol{J}^{(n+\frac{1}{2})}$ in the numerator of Eq. (3.3) is a surface integral over the problem boundary ∂V, $\int_{\partial V}(\cdot) \, \mathrm{d}^2 r'$. In SI, transport-sweep-evaluated neutron current information is only required on the problem boundaries.

In order to calculate the end-of-iteration scalar flux $\phi^{(n+1)}$, we normalize the swept scalar flux $\phi^{(n+\frac{1}{2})}$ according to Eq. (3.1b). This gives:

$$\phi^{(n+1)} = \left(\frac{P}{\iint \kappa \Sigma_f \phi^{(n+\frac{1}{2})} \, \mathrm{d}^3 r' \, \mathrm{d}E'} \right) \phi^{(n+\frac{1}{2})} \, . \tag{3.4}$$

(Dependent variables and integration bounds have again been ignored for brevity.)

Now that we have both the updated eigenvalue $\lambda^{(n+1)}$ and normalized scalar flux $\phi^{(n+1)}$, we check the convergence and maximum iteration number criteria. For example, we may check if:

$$\left| \frac{\phi^{(n+1)} - \phi^{(n)}}{\phi^{(n)}} \right|_\infty < \epsilon_\phi \, ? \tag{3.5a}$$

and

$$\left| \frac{\lambda^{(n+1)} - \lambda^{(n)}}{\lambda^{(n)}} \right| < \epsilon_\lambda \, ? \tag{3.5b}$$

are both satisfied for some user-chosen criteria $\epsilon_\phi \ll 1$ and $\epsilon_\lambda \ll 1$ to determine whether the solution estimates are suitably converged to a fixed-point. We may also wish to check whether the iteration index has reached a user-specified maximum $n_{\max}$:

$$\text{is } \; (n+1) = n_{\max} \, ? \tag{3.6}$$

If either of these criteria have been met, we exit the loop. Otherwise, we increment the iteration index n by one and move back to SI Step 1 to begin the next iteration.

The Source Iteration (SI) procedure outlined above for solving Eqs. (3.1) suffers from extremely slow convergence for many problems of practical interest. For eigenvalue problems with a large neutron mean-free-path (optical) thickness, SI may take hundreds, or even thousands of iterations to converge to an acceptable tolerance. This is due to the method's poor performance at reducing low-spatial-frequency (i.e. "flat") error modes. Since each SI iteration requires a transport sweep [Eqs. (3.2)], a notably expensive calculation, the method becomes unacceptably expensive for many problems that require an unreasonably high number of iterations to converge.

As an alternative, researchers use *transport acceleration schemes* to augment transport sweeps with *low-order acceleration equations*. Accelerated transport iterative methods, for example, Diffusion Synthetic Acceleration (DSA) and Coarse-Mesh Finite Difference (CMFD) (discussed next), have an increased cost-per-iteration as compared to Source Iteration (SI) but tend to converge in many fewer iterations. For example, for single-physics transport eigenvalue problems, CMFD tends to converge in $O(10)$ iterations.

3.3. Coarse-Mesh Finite Difference (CMFD)

To begin our discussion of Coarse-Mesh Finite Difference (CMFD), we derive the low-order, transport-corrected, diffusion eigenvalue problem that the method employs alongside standard transport sweeps [Eqs. (3.2)]. In this thesis, we often refer to this transport-corrected diffusion problem as, simply, the *CMFD problem* or the *CMFD equations*.

We start the derivation of the CMFD equations by integrating the transport equation [Eq. (3.1a)] over angle with $\int_{4\pi}(\cdot)\mathrm{d}\Omega'$. This results in the familiar *neutron balance equation*:

$$\begin{aligned}\boldsymbol{\nabla}\cdot\boldsymbol{J}(\boldsymbol{r},E)+\Sigma_t(\boldsymbol{r},E)\phi(\boldsymbol{r},E)&\\ =\int_0^\infty\Sigma_s(\boldsymbol{r},E'\to E)\phi(\boldsymbol{r},E')\,\mathrm{d}E'&\\ +\lambda\chi(\boldsymbol{r},E)\int_0^\infty\nu\Sigma_f(\boldsymbol{r},E')\phi(\boldsymbol{r},E')\,\mathrm{d}E'\ .&\end{aligned}\tag{3.7}$$

Next, we define the standard diffusion coefficient $D(\boldsymbol{r},E)$ (for a problem with isotropic

scattering):

$$D(\boldsymbol{r}, E) = \frac{1}{3\Sigma_t(\boldsymbol{r}, E)} \,, \tag{3.8}$$

and we remind ourselves of the diffusion-theory approximation to the neutron current $\boldsymbol{J}$ (Fick's law):

$$\boldsymbol{J}(\boldsymbol{r}, E) \approx -D(\boldsymbol{r}, E)\boldsymbol{\nabla}\phi(\boldsymbol{r}, E) \,. \tag{3.9}$$

To proceed, we manipulate the neutron leakage term in Eq. (3.7):

$$\begin{aligned} \boldsymbol{\nabla}\cdot\boldsymbol{J}(\boldsymbol{r}, E) &= -\boldsymbol{\nabla}\cdot D(\boldsymbol{r}, E)\boldsymbol{\nabla}\phi(\boldsymbol{r}, E) \\ &\qquad + \boldsymbol{\nabla}\cdot\big(\boldsymbol{J}(\boldsymbol{r}, E) + D(\boldsymbol{r}, E)\boldsymbol{\nabla}\phi(\boldsymbol{r}, E)\big) \qquad (3.10a) \\ &= -\boldsymbol{\nabla}\cdot D(\boldsymbol{r}, E)\boldsymbol{\nabla}\phi(\boldsymbol{r}, E) + \boldsymbol{\nabla}\cdot\hat{\boldsymbol{D}}(\boldsymbol{r}, E)\phi(\boldsymbol{r}, E) \,, \end{aligned}$$

where we have defined the *nonlinear transport correction* ***vector*** $\hat{\boldsymbol{D}}(\boldsymbol{r}, E)$ to be:

$$\hat{\boldsymbol{D}}(\boldsymbol{r}, E) \equiv \frac{\boldsymbol{J}(\boldsymbol{r}, E) + D(\boldsymbol{r}, E)\boldsymbol{\nabla}\phi(\boldsymbol{r}, E)}{\phi(\boldsymbol{r}, E)} \,. \tag{3.10b}$$

We stress that no approximations were made in arriving at Eqs. (3.10). In particular, the validity of diffusion theory on our transport problem of interest does not influence the hard fact that Eqs. (3.10) are exact! Next, we insert the reformulated (but equivalent!) expression for the transport leakage [Eqs. (3.10)] into Eq. (3.7). We arrive at the CMFD equations:

$$\begin{aligned} &-\boldsymbol{\nabla}\cdot D(\boldsymbol{r}, E)\boldsymbol{\nabla}\phi(\boldsymbol{r}, E) + \boldsymbol{\nabla}\cdot\hat{\boldsymbol{D}}(\boldsymbol{r}, E)\phi(\boldsymbol{r}, E) \\ &\qquad + \Sigma_t(\boldsymbol{r}, E)\phi(\boldsymbol{r}, E) = \int_0^\infty \Sigma_s(\boldsymbol{r}, E' \to E)\phi(\boldsymbol{r}, E')\,\mathrm{d}E' \qquad (3.11a) \\ &\qquad\qquad + \lambda\chi(\boldsymbol{r}, E)\int_0^\infty \nu\Sigma_f(\boldsymbol{r}, E')\phi(\boldsymbol{r}, E')\,\mathrm{d}E' \,, \end{aligned}$$

(The power normalization condition is unaffected by our manipulations.)

$$P = \int_0^\infty \int_V \kappa\Sigma_f(\boldsymbol{r}', E')\phi(\boldsymbol{r}', E')\,\mathrm{d}^3r'\,\mathrm{d}E' \,. \tag{3.11b}$$

No approximations have been made in forming Eqs. (3.11). The scalar flux ϕ and eigenvalue λ solutions to Eqs. (3.11) are equivalent to those of Eqs. (3.1). Unfortunately, with Eqs. (3.11), we have obtained a set of two equations with five unknowns: the scalar flux ϕ, eigenvalue λ, and the transport correction vector $\hat{\boldsymbol{D}}$. As it stands,

we cannot solve Eqs. (3.11). In Coarse-Mesh Finite Difference (CMFD), the idea is to iteratively lag the transport correction vector $\hat{\boldsymbol{D}}$ in Eqs. (3.11), using its definition [Eq. (3.10b)] and the most recent, transport-sweep determined estimates of the scalar flux ϕ and current vector $\boldsymbol{J}$. The motivation behind lagging $\hat{\boldsymbol{D}}$ [Eq. (3.10b)] is that in many situations of practical interest, Fick's law [Eq. (3.9)] is fairly accurate. When this is the case, the components of $\hat{\boldsymbol{D}}$ are small and thus lag-able. As a problem becomes more diffusive, the numerator of $\hat{\boldsymbol{D}}$ [Eq. (3.10b)] limits to the zero vector! Upon convergence (i.e. given the correct $\hat{\boldsymbol{D}}$), Eqs. (3.1) and (3.11) are consistent (i.e. they have identical scalar flux ϕ and eigenvalue solutions λ).

Next, we explicitly outline the CMFD iteration scheme. We mention that CMFD Steps 0 – 1 are identical to SI Steps 0 – 1, defined above.

Coarse-Mesh Finite Difference (CMFD) Step 0 : Initialization

This is identical to SI Step 0.

CMFD Step 1 : Neutron Transport Sweep

This is identical to SI Step 1. (See Eqs. (3.2).)

CMFD Step 2 : CMFD Solve

Given the transport sweep-determined estimates of the neutron scalar flux $\phi^{(n+\frac{1}{2})}$ and current vector $\boldsymbol{J}^{(n+\frac{1}{2})}$ from CMFD Step 1, we construct an estimate of the transport correction vector $\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}$ [Eq. (3.10b)]:

$$\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}(\boldsymbol{r}, E) = \frac{\boldsymbol{J}^{(n+\frac{1}{2})}(\boldsymbol{r}, E) + D(\boldsymbol{r}, E)\boldsymbol{\nabla}\phi^{(n+\frac{1}{2})}(\boldsymbol{r}, E)}{\phi^{(n+\frac{1}{2})}(\boldsymbol{r}, E)} \,. \tag{3.12}$$

Using this lagged $\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}$ [Eq. (3.12)], we solve the low-order, transport-corrected, diffusion eigenvalue problem described by Eqs. (3.13) for the end-of-iteration scalar flux $\phi^{(n+1)}$ and eigenvalue $\lambda^{(n+1)}$ estimates. This eigenvalue problem can be solved using a variety of techniques, e.g., Wielandt shifted power iteration [124], generalized Davidson [19], etc.:

$$\begin{aligned} -\boldsymbol{\nabla}\cdot D(\boldsymbol{r}, E)\boldsymbol{\nabla}\phi^{(n+1)}(\boldsymbol{r}, E) + \boldsymbol{\nabla}\cdot\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}(\boldsymbol{r}, E)\phi^{(n+1)}(\boldsymbol{r}, E) \\ + \Sigma_t(\boldsymbol{r}, E)\phi^{(n+1)}(\boldsymbol{r}, E) = \int_0^\infty \Sigma_s(\boldsymbol{r}, E' \to E)\phi^{(n+1)}(\boldsymbol{r}, E')\,\mathrm{d}E' \\ + \lambda^{(n+1)}\chi(\boldsymbol{r}, E)\int_0^\infty \nu\Sigma_f(\boldsymbol{r}, E')\phi^{(n+1)}(\boldsymbol{r}, E')\,\mathrm{d}E' \,, \end{aligned} \tag{3.13a}$$

$$P = \int_0^\infty \int_V \kappa\Sigma_f(\boldsymbol{r}', E')\phi^{(n+1)}(\boldsymbol{r}', E')\,\mathrm{d}^3r'\,\mathrm{d}E' \ . \tag{3.13b}$$

Finally, we check max iteration and convergence criteria, exiting the CMFD iteration if they have been met. Otherwise, we increment the iteration index n by one and proceed to CMFD Step 1 to begin the next iteration.

When applied to a suitably discretized, single-physics, transport eigenvalue problem (e.g., Eqs. (3.1)), the above CMFD algorithm performs exceptionally well, converging in $O(10)$ iterations with a low spectral radius ($\rho \approx 0.22$). Unfortunately, when the above approach is slightly modified (without great care) to solve multiphysics problems (e.g., the coupled set of Eqs. (2.5), (2.6), (2.7), (2.9), (2.10)), the method may not achieve its optimal convergence rate. That is, the method's spectral radius ρ may increase above ≈ 0.22. Under certain extreme conditions, the spectral radius may climb above one, resulting in divergence. In the remainder of this thesis, we discuss these issues and propose novel remedies to address them.

Chapter 4.

Multiphysics Iterative Methods

In the previous chapter, we ignored feedback in order to focus on single-physics transport iterative methods. Specifically, we outlined the Source Iteration (SI) and Coarse-Mesh Finite Difference (CMFD) methods applied to the neutron transport eigenvalue problem described by Eqs. (3.1). In the present chapter, we bring feedback back into the fold and cover CMFD-based iterative methods for multiphysics problems.

To begin, we introduce a few conventions and notational shortcuts to simplify our discussion. Then, we reconsider the neutron transport – thermal hydraulics multiphysics problem presented in Ch. 2. Next, starting from the multiphysics problem's neutron transport equation, we derive a transport-corrected diffusion equation that includes temperature dependent nuclear data (e.g., macroscopic cross sections, fission spectrum, etc.). Given the proper transport correction vector $\hat{\boldsymbol{D}}$, the original transport and derived (transport-corrected) diffusion equations provide accurate and consistent descriptions of neutron transport behavior in the multiphysics system.

We then outline a pair of related iterative methods for solving neutron transport – thermal hydraulics multiphysics problems: *Multiphysics CMFD* (M-CMFD) and *Relaxed CMFD* (R-CMFD). R-CMFD is the default multiphysics iteration scheme used by the Michigan Parallel Characteristics-based Transport (MPACT) reactor physics code [55]. We view R-CMFD as the industry-standard in loosely-coupled iterative methods for neutron transport problems with multiphysics feedback. We measure the performance of novel methods put forward in this thesis against R-CMFD.

Following descriptions of M-CMFD and R-CMFD, we discuss the strengths and weaknesses of each approach. At this point, we will have covered all of the necessary background material to move into the novel contributions of this thesis. As a first step

into original work, we momentarily depart from practical methods (e.g., M-CMFD and R-CMFD) to introduce two purely theoretical methods that serve a pedagogical role. We refer to these approaches as Theoretical Method 0 (TM-0) and Theoretical Method 1 (TM-1). Although TM-0 and TM-1 cannot be implemented in a computer code, they possess an alluring property, namely, they converge in a single iteration!

Using TM-0 and TM-1 as a foundation, we derive the X-CMFD method. While X-CMFD is more grounded (i.e. implementable) than the purely theoretical TM-0 and TM-1, the approach is still not entirely practical. The key feature of X-CMFD is the direct inclusion of thermal hydraulic feedback and nuclear data updates in the transport-corrected, diffusion eigenvalue problem. Each X-CMFD iteration introduces a nonlinear problem that couples the (transport-corrected) neutron diffusion eigenvalue equations, the full thermal hydraulics equations (represented by the operator $\mathcal{L}$), and any required nuclear data utilities. The essence of this idea has been proposed in various forms under several different pseudonyms in a number of recent publications. (The approach is captured by both the CMFD-Coupling and JFNK-based methods described in [110, 111]. In [91], the Nonlinear Fully Coupled Diffusion Acceleration (NFCDA) method is equivalent to what we refer to as X-CMFD. The name "X-CMFD" is also used in [91, 93] to refer to a specific solution strategy for NFCDA. In this thesis, we ignore this naming overlap and instead fully describe what we think of as X-CMFD.)

For multiphysics problems, X-CMFD consistently achieves an outer iteration spectral radius of $\rho \approx 0.22$. That is, for multiphysics problems, X-CMFD has the same iterative convergence rate that CMFD achieves when applied to single-physics transport problems (assuming nominal conditions). However, X-CMFD's ideal convergence rate comes at a potentially steep price. In order to solve the nonlinear (transport-corrected) neutron diffusion – thermal hydraulics problem introduced by X-CMFD each *outer iteration*, the method requires multiple applications of the thermal hydraulics solver and nuclear data update routines within *inner iterations*. If either of these procedures is computationally expensive, the X-CMFD approach is impractical. To deal with this situation, we propose the *Nonlinearly Implicit Low-Order CMFD* (NILO-CMFD) method, which addresses the drawbacks of M-CMFD, R-CMFD, and X-CMFD. NILO-CMFD and an incomplete variant we refer to as *NILO-A* are the primary contributions of this thesis. Detailed descriptions of both NILO-A and NILO-CMFD are presented at the conclusion of this chapter.

4.1. Terminology and Notation

In this section, we introduce a few pieces of simplifying terminology and notation that are used in the remainder of this chapter. For example, we occasionally leave out integration-bounds for the sake of readability. When integration bounds are not provided, the definitions in Eqs. (4.1) should be assumed. Unspecified bounds of integration are over the entirety of a phase-space dimension:

$$\int (\cdot)\, \mathrm{d}^3 r' \equiv \int_V (\cdot)\, \mathrm{d}^3 r' \quad , \int (\cdot)\, \mathrm{d}\Omega' \equiv \int_{4\pi} (\cdot)\, \mathrm{d}\Omega' \quad , \int (\cdot)\, \mathrm{d}E' \equiv \int_0^\infty (\cdot)\, \mathrm{d}E' \; . \tag{4.1}$$

Also, in the full neutron transport – thermal hydraulics multiphysics problem, several nuclear data parameters depend on temperature (with an implied density dependence, e.g., Eq. (2.5)). These include: Σ_t, Σ_s, χ, $\nu\Sigma_f$, and $\kappa\Sigma_f$. These nuclear data quantities act as unknowns alongside the scalar flux ϕ, eigenvalue λ, and temperature T that we must iteratively solve for in the multiphysics problem. To be succinct, we use the following shorthand to refer to the update of **all** nuclear data in response to a new temperature estimate, say, for example, $T^{(n)}$:

$$\Sigma_u^{(n)}(\boldsymbol{r}, E) \equiv \Sigma_u(\boldsymbol{r}, T^{(n)}(\boldsymbol{r}), E) \; . \tag{4.2}$$

Thus, after Eq. (4.2) is evaluated, we have new estimates of $\Sigma_t^{(n)}(\boldsymbol{r}, E)$, $\Sigma_s^{(n)}(\boldsymbol{r}, E' \to E)$, $\chi^{(n)}(\boldsymbol{r}, E)$, $\nu\Sigma_f^{(n)}(\boldsymbol{r}, E)$, and $\kappa\Sigma_f^{(n)}(\boldsymbol{r}, E)$, each evaluated at the temperature $T^{(n)}$. Σ_u is used to refer to **all** of these quantities simultaneously. At some points in the text, we refer to this collection of parameters, Σ_u, as "cross sections" and the operations performed in Eq. (4.2) by a nuclear data library as "updating the cross sections", even though χ, $\nu\Sigma_f$, and $\kappa\Sigma_f$ are technically not cross sections themselves. We ignore this technicality in favor of simplicity.

4.2. Neutron Transport – Thermal Hydraulics Problem

In this section, we reconsider the full neutron transport – thermal hydraulics multiphysics problem of interest to our work. Descriptions of multiphysics iterative methods presented in this chapter are applied to this problem. We also derive the CMFD transport-corrected diffusion equations as they apply to a transport problem with

temperature dependent nuclear data: $\Sigma_u(T(\boldsymbol{r}))$. (Again, Σ_u is shorthand for Σ_t, Σ_s, χ, $\nu\Sigma_f$, and $\kappa\Sigma_f$.) This section serves as a brief summary and re-framing of the material covered in Chs. 2 and 3.

The neutron transport – thermal hydraulics multiphysics problem is described below by the coupled set of Eqs. (4.3)-(4.4). We require a nuclear data library capable of evaluating macroscopic cross sections and related nuclear data as functions of space, temperature, and energy. We treat this nuclear data library as a black box. As a reminder, we have removed any explicit dependence of nuclear data on material density by assuming that a dependence of density ρ on temperature T is available: $\rho(T(\boldsymbol{r}))$. We have also assumed isotropic neutron scattering and ignored boundary conditions for convenience. The resulting coupled equations are:

$$\begin{aligned}\hat{\boldsymbol{\Omega}} \cdot \boldsymbol{\nabla}\psi(\boldsymbol{r},\hat{\boldsymbol{\Omega}},E) + \Sigma_t(\boldsymbol{r},T(\boldsymbol{r}),E)\psi(\boldsymbol{r},\hat{\boldsymbol{\Omega}},E) \\ = \frac{1}{4\pi}\iint \Sigma_s(\boldsymbol{r},T(\boldsymbol{r}),E'\to E)\psi(\boldsymbol{r},\hat{\boldsymbol{\Omega}}',E')\,\mathrm{d}\Omega'\,\mathrm{d}E' \\ + \lambda\frac{\chi(\boldsymbol{r},T(\boldsymbol{r}),E)}{4\pi}\iint \nu\Sigma_f(\boldsymbol{r},T(\boldsymbol{r}),E')\psi(\boldsymbol{r},\hat{\boldsymbol{\Omega}}',E')\,\mathrm{d}\Omega'\,\mathrm{d}E'\ ,\end{aligned} \tag{4.3a}$$

$$P = \iiint \kappa\Sigma_f(\boldsymbol{r}',T(\boldsymbol{r}'),E')\psi(\boldsymbol{r}',\hat{\boldsymbol{\Omega}}',E')\,\mathrm{d}^3r'\,\mathrm{d}\Omega'\,\mathrm{d}E' = \int h(\boldsymbol{r}')\mathrm{d}^3r'\ , \tag{4.3b}$$

$$h(\boldsymbol{r}) \equiv \iint \kappa\Sigma_f(\boldsymbol{r},T(\boldsymbol{r}),E')\psi(\boldsymbol{r},\hat{\boldsymbol{\Omega}}',E')\,\mathrm{d}\Omega'\,\mathrm{d}E'\ , \tag{4.4a}$$

$$\mathcal{L}T(\boldsymbol{r}) = h(\boldsymbol{r})\ . \tag{4.4b}$$

To fully define the above problem, we require a specification of the problem domain V, boundary conditions, and a concrete definition of the thermal hydraulics operator $\mathcal{L}$. In this chapter, we leave $\mathcal{L}$ unspecified. Doing this makes our description of multiphysics iterative methods more general. The iterative method descriptions presented in this chapter may be straightforwardly applied to any neutron transport – thermal hydraulics problem, as long as the thermal hydraulics of interest takes the general form of Eqs. (4.4). This is a tangible benefit of one of our research ground rules: *We assume no detailed knowledge of the thermal hydraulics solver's internals. All that we ask is that given a heat source distribution, the thermal hydraulics solver will return the corresponding temperature (and density) solution field(s).* This restriction forbids the use of tight-coupling methods (e.g., Newton's method, Jacobian-Free

Newton Krylov (JFNK) [47]), since we cannot assume access to residuals in proprietary codes. However, it allows the methods we discuss to easily accommodate the use of existing, single-physics, thermal hydraulics solvers (e.g., COBRA-TF (CTF) [87]). We refer the reader to Ch. 2 Eqs. (2.12) and (2.14) for concrete descriptions of two example $\mathcal{L}$'s.

Next, we form CMFD-based transport-corrected diffusion equations from the multiphysics transport Eqs. (4.3). The following derivation is similar to the derivation in Ch. 3 for single-physics transport problems, except, now that we are dealing with multiphysics problems, nuclear data Σ_u and related parameters (D and $\hat{\boldsymbol{D}}$) in the resulting diffusion system are temperature (T) dependent.

First, we repeat the definitions of the neutron scalar flux ϕ and current vector $\boldsymbol{J}$ as angular moments of the angular flux ψ:

$$\phi(\boldsymbol{r}, E) \equiv \int_{4\pi} \psi(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E)\, \mathrm{d}\Omega' \,, \tag{4.5a}$$

$$\boldsymbol{J}(\boldsymbol{r}, E) \equiv \int_{4\pi} \hat{\boldsymbol{\Omega}}' \psi(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E)\, \mathrm{d}\Omega' \,. \tag{4.5b}$$

Next, we integrate Eq. (4.3a) over angle $\int(\cdot)\,\mathrm{d}\Omega'$ and use the definitions in Eqs. (4.5) to write the neutron balance equation with temperature dependent nuclear data:

$$\begin{aligned} &\boldsymbol{\nabla} \cdot \boldsymbol{J}(\boldsymbol{r}, E) + \Sigma_t(\boldsymbol{r}, T(\boldsymbol{r}), E)\phi(\boldsymbol{r}, E) \\ &\quad = \int_0^\infty \Sigma_s(\boldsymbol{r}, T(\boldsymbol{r}), E' \to E)\phi(\boldsymbol{r}, E')\, \mathrm{d}E' \\ &\qquad + \lambda\chi(\boldsymbol{r}, T(\boldsymbol{r}), E) \int_0^\infty \nu\Sigma_f(\boldsymbol{r}, T(\boldsymbol{r}), E')\phi(\boldsymbol{r}, E')\, \mathrm{d}E' \,. \end{aligned} \tag{4.6}$$

We define the transport correction vector $\hat{\boldsymbol{D}}$ as:

$$\hat{\boldsymbol{D}}(\boldsymbol{r}, T(\boldsymbol{r}), E) \equiv \frac{\boldsymbol{J}(\boldsymbol{r}, E) + D(\boldsymbol{r}, T(\boldsymbol{r}), E)\boldsymbol{\nabla}\phi(\boldsymbol{r}, E)}{\phi(\boldsymbol{r}, E)} \,, \tag{4.7a}$$

For the moment, we leave the definition of D unspecified. In theory, D can take any form. That is, our derivation does not depend on the definition of D. In practice, D is chosen to well-approximate the standard diffusion coefficient. In the most general case, D and, by extension, $\hat{\boldsymbol{D}}$ may depend on space $\boldsymbol{r}$, material temperature T, and neutron energy E. Using the definition of $\hat{\boldsymbol{D}}$ in Eq. (4.7a), we manipulate the neutron balance equation [Eq. (4.6)] to produce the following transport-corrected diffusion

equation:

$$
\begin{aligned}
-\boldsymbol{\nabla}\cdot D(\boldsymbol{r},T(\boldsymbol{r}),E)\boldsymbol{\nabla}\phi(\boldsymbol{r},E)+\boldsymbol{\nabla}\cdot\hat{\boldsymbol{D}}(\boldsymbol{r},T(\boldsymbol{r}),E)\phi(\boldsymbol{r},E)& \\
+\Sigma_t(\boldsymbol{r},T(\boldsymbol{r}),E)\phi(\boldsymbol{r},E)=\int_0^\infty \Sigma_s(\boldsymbol{r},T(\boldsymbol{r}),E'\rightarrow E)\phi(\boldsymbol{r},E')\,\mathrm{d}E'& \\
+\lambda\chi(\boldsymbol{r},T(\boldsymbol{r}),E)\int_0^\infty \nu\Sigma_f(\boldsymbol{r},T(\boldsymbol{r}),E')\phi(\boldsymbol{r},E')\,\mathrm{d}E'\,,&
\end{aligned}
\tag{4.7b}
$$

satisfying the power normalization condition:

$$
P=\iint \kappa\Sigma_f(\boldsymbol{r}',T(\boldsymbol{r}'),E')\phi(\boldsymbol{r}',E')\,\mathrm{d}^3r'\,\mathrm{d}E'\,. \tag{4.7c}
$$

Sticking to convention, we define D in Eqs. (4.7) to be the standard diffusion coefficient (for an isotropic scattering problem):

$$
D(\boldsymbol{r},T(\boldsymbol{r}),E)\equiv\frac{1}{3\Sigma_t(\boldsymbol{r},T(\boldsymbol{r}),E)}\,. \tag{4.8}
$$

No approximations were made in moving from Eqs. (4.3) to Eqs. (4.7)-(4.8). These two sets of equations present mechanically different but ultimately consistent descriptions of the transport physics in the multiphysics problem [Eqs. (4.3)-(4.4)].

To summarize, in this section, we restated the neutron transport – thermal hydraulics multiphysics problem [Eqs. (4.3)-(4.4)] and derived a transport-corrected diffusion eigenvalue problem with temperature dependent nuclear data [Eqs. (4.7)-(4.8)]. No approximations were made in deriving the diffusion-like Eqs. (4.7)-(4.8) from the transport Eqs. (4.3). In this section, we have not provided an in-depth description of how nuclear data is updated with respect to new temperatures (and densities). For a concrete description of this process, we refer the reader to [46, 55] for a look at nuclear data processing in the MPACT reactor physics code.

4.3. Multiphysics CMFD (M-CMFD) & Relaxed CMFD (R-CMFD)

Next, we outline two related iterative methods for solving the neutron transport – thermal hydraulics multiphysics problem specified by Eqs. (4.3)-(4.4). Both approaches act as simple multiphysics-modifications of the single-physics CMFD method

outlined in Ch. 3. We refer to these multiphysics iterative methods as *Multiphysics CMFD* (M-CMFD) and *Relaxed CMFD* (R-CMFD).

R-CMFD is a standard multiphysics iterative method used by modern reactor physics research codes. For example, R-CMFD is the default multiphysics iterative method used by the Michigan Parallel Characteristics-based Transport (MPACT) code [55]. Later, in Chs. 6 & 7, we use R-CMFD as a benchmark against which we measure the performance of the novel NILO-A and NILO-CMFD methods, the main contributions of this thesis. The R-CMFD method is developed by slightly modifying the prescription of Multiphysics CMFD (M-CMFD). Before covering the necessary modifications required to form R-CMFD from M-CMFD, namely, the inclusion of a relaxation factor, we first outline the simpler Multiphysics CMFD (M-CMFD) method:

Multiphysics CMFD (M-CMFD) Step 0 : Initialization

Before iterating, we require initial estimates of the scalar flux $\phi^{(0)}$, eigenvalue $\lambda^{(0)}$, and nuclear data $\Sigma_u^{(0)}$. (As a reminder, in writing "nuclear data $\Sigma_u^{(0)}$", we are referring to $\Sigma_t^{(0)}$, $\Sigma_s^{(0)}$, $\chi^{(0)}$, $\nu\Sigma_f^{(0)}$, and $\kappa\Sigma_f^{(0)}$.) After determining initial estimates, we set the iteration index n to zero and proceed to the first iteration.

M-CMFD Step 1 : Neutron Transport Sweep

At the beginning of the $(n+1)^{\text{st}}$ iteration, the most recent estimates of the scalar flux $\phi^{(n)}$, eigenvalue $\lambda^{(n)}$, and nuclear data $\Sigma_u^{(n)}$ are available – either from the initialization step if $n = 0$, or from the previous iteration if $n \geq 1$. We insert these estimates into Eq. (4.3a) to obtain a fixed-source neutron transport equation [Eq. (4.9a)] for the angular flux $\psi^{(n+\frac{1}{2})}$. (Again, for simplicity, we ignore boundary conditions throughout this chapter.) By sweeping Eq. (4.9a), we update the neutron scalar flux $\phi^{(n+\frac{1}{2})}$ [Eq. (4.9b)] and current vector $\boldsymbol{J}^{(n+\frac{1}{2})}$ [Eq. (4.9c)] estimates:

$$\begin{aligned}
&\hat{\boldsymbol{\Omega}} \cdot \boldsymbol{\nabla}\psi^{(n+\frac{1}{2})}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E) + \Sigma_t^{(n)}(\boldsymbol{r}, E)\psi^{(n+\frac{1}{2})}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E) \\
&\quad = \frac{1}{4\pi}\int_0^\infty \Sigma_s^{(n)}(\boldsymbol{r}, E' \to E)\phi^{(n)}(\boldsymbol{r}, E')\,\mathrm{d}E' \\
&\qquad + \lambda^{(n)}\frac{\chi^{(n)}(\boldsymbol{r}, E)}{4\pi}\int_0^\infty \nu\Sigma_f^{(n)}(\boldsymbol{r}, E')\phi^{(n)}(\boldsymbol{r}, E')\,\mathrm{d}E' \,,
\end{aligned} \tag{4.9a}$$

$$\phi^{(n+\frac{1}{2})}(\boldsymbol{r}, E) = \int_{4\pi} \psi^{(n+\frac{1}{2})}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E)\, \mathrm{d}\Omega' \ , \tag{4.9b}$$

$$\boldsymbol{J}^{(n+\frac{1}{2})}(\boldsymbol{r}, E) = \int_{4\pi} \hat{\boldsymbol{\Omega}}' \psi^{(n+\frac{1}{2})}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E)\, \mathrm{d}\Omega' \ . \tag{4.9c}$$

We pause to clarify our use of iteration indices. We refer to the $(n+1)^{\text{st}}$ iteration as the iteration that begins with initial estimates superscripted by (n) and concludes with updated estimates superscripted by $(n+1)$. Intermediate quantities in an iteration have superscripts that lie between these two integer values, e.g., $\phi^{(n+\frac{1}{2})}$, or include an additional *inner iteration* index, e.g., $\phi^{(n,p)}$, to distinguish from *outer iteration* quantities, e.g., $\phi^{(n)}$.

As an example, at the start of the the 1^{st} iteration of M-CMFD, scalar fluxes from the initialization step $\phi^{(0)}$ are the most up-to-date. At the end of the 1^{st} iteration, updated scalar flux estimates $\phi^{(1)}$ are available. Similarly, the 2^{nd} iteration of Multiphysics CMFD begins with estimates of the scalar flux $\phi^{(1)}$, eigenvalue $\lambda^{(1)}$, and nuclear data $\Sigma_u^{(1)}$ from the end of the 1^{st} iteration and concludes with updated values $\phi^{(2)}$, $\lambda^{(2)}$, $\Sigma_u^{(2)}$. These, in turn, are fed into the 3^{rd} iteration, and so on …

M-CMFD Step 2 : Thermal Hydraulics Solve

Using the latest nuclear data $\Sigma_u^{(n)}$ and scalar flux $\phi^{(n+\frac{1}{2})}$ estimates, we update the fission heat source distribution $h^{(n+\frac{1}{2})}$:

$$h^{(n+\frac{1}{2})}(\boldsymbol{r}) = \int_0^\infty \kappa\Sigma_f^{(n)}(\boldsymbol{r}, E')\phi^{(n+\frac{1}{2})}(\boldsymbol{r}, E')\, \mathrm{d}E' \ . \tag{4.10a}$$

Then we solve the thermal hydraulics problem for the updated temperature $T^{(n+\frac{1}{2})}$:

$$\mathcal{L}T^{(n+\frac{1}{2})}(\boldsymbol{r}) = h^{(n+\frac{1}{2})}(\boldsymbol{r}) \ . \tag{4.10b}$$

(As a reminder, the inversion of the thermal hydraulics operator $\mathcal{L}$ represents the operations performed by a user-chosen, loosely-coupled, thermal hydraulics code, e.g., the COBRA-TF (CTF) subchannel code.)

M-CMFD Step 3 : Nuclear Data Evaluation

The temperature $T^{(n+\frac{1}{2})}$ is used to evaluate new nuclear data estimates $\Sigma_u^{(n+\frac{1}{2})}$:

$$\Sigma_u^{(n+\frac{1}{2})}(\boldsymbol{r}, E) = \Sigma_u(\boldsymbol{r}, T^{(n+\frac{1}{2})}(\boldsymbol{r}), E) \ . \tag{4.11}$$

(As a reminder, the notation used in Eq. (4.11) refers to the simultaneous update of $\Sigma_t^{(n+\frac{1}{2})}$, $\Sigma_s^{(n+\frac{1}{2})}$, $\chi^{(n+\frac{1}{2})}$, $\nu\Sigma_f^{(n+\frac{1}{2})}$, and $\kappa\Sigma_f^{(n+\frac{1}{2})}$.)

We then calculate the standard diffusion coefficient $D^{(n+\frac{1}{2})}$ (Eq. (4.8), assuming isotropic scattering) and the transport correction vector $\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}$ [Eq. (4.7a)]:

$$D^{(n+\frac{1}{2})}(\boldsymbol{r}, E) = \frac{1}{3\Sigma_t^{(n+\frac{1}{2})}(\boldsymbol{r}, E)} , \tag{4.12a}$$

$$\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}(\boldsymbol{r}, E) = \frac{\boldsymbol{J}^{(n+\frac{1}{2})}(\boldsymbol{r}, E) + D^{(n+\frac{1}{2})}(\boldsymbol{r}, E)\boldsymbol{\nabla}\phi^{(n+\frac{1}{2})}(\boldsymbol{r}, E)}{\phi^{(n+\frac{1}{2})}(\boldsymbol{r}, E)} . \tag{4.12b}$$

M-CMFD Step 4 : Diffusion Solve

By lagging the diffusion coefficient $D^{(n+\frac{1}{2})}$, transport correction vector $\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}$, and nuclear data $\Sigma_u^{(n+\frac{1}{2})}$ in Eqs. (4.7), we obtain a linear transport-corrected diffusion eigenvalue problem for $\phi^{(n+1)}$ and $\lambda^{(n+1)}$:

$$\begin{aligned} &- \boldsymbol{\nabla} \cdot D^{(n+\frac{1}{2})}(\boldsymbol{r}, E)\boldsymbol{\nabla}\phi^{(n+1)}(\boldsymbol{r}, E) + \boldsymbol{\nabla} \cdot \hat{\boldsymbol{D}}^{(n+\frac{1}{2})}(\boldsymbol{r}, E)\phi^{(n+1)}(\boldsymbol{r}, E) \\ &\quad + \Sigma_t^{(n+\frac{1}{2})}(\boldsymbol{r}, E)\phi^{(n+1)}(\boldsymbol{r}, E) = \int_0^\infty \Sigma_s^{(n+\frac{1}{2})}(\boldsymbol{r}, E' \to E)\phi^{(n+1)}(\boldsymbol{r}, E')\,\mathrm{d}E' \\ &\quad + \lambda^{(n+1)}\chi^{(n+\frac{1}{2})}(\boldsymbol{r}, E)\int_0^\infty \nu\Sigma_f^{(n+\frac{1}{2})}(\boldsymbol{r}, E')\phi^{(n+1)}(\boldsymbol{r}, E')\,\mathrm{d}E' , \end{aligned} \tag{4.13a}$$

$$P = \int_0^\infty \int_V \kappa\Sigma_f^{(n+\frac{1}{2})}(\boldsymbol{r}', E')\phi^{(n+1)}(\boldsymbol{r}', E')\,\mathrm{d}^3r'\,\mathrm{d}E' . \tag{4.13b}$$

These equations are sufficiently solved for the end-of-iteration scalar flux $\phi^{(n+1)}$ and eigenvalue $\lambda^{(n+1)}$.

It is still necessary to specify the end-of-iteration nuclear data estimates $\Sigma_u^{(n+1)}$. We do this by simply using the values computed in Step 3 [Eq. (4.11)]:

$$\Sigma_u^{(n+1)} = \Sigma_u^{(n+\frac{1}{2})} . \tag{4.14}$$

We then check the convergence and max iteration criteria. If either condition has been met, we are finished! Otherwise, we increase the iteration index n by one and return to Step 1 to begin the next iteration.

The Multiphysics CMFD (M-CMFD) scheme exhibits stability and convergence rate issues when applied to multiphysics problems with typical reactor feedback in-

tensities. These issues often prevent the method's practical use. Current real-world implementations often address these issues with *relaxation*, a standard "engineering approach" used in iterative methods across a wide-range of disciplines (e.g., under-relaxed smoothers for linear algebra, etc.).

One common approach to add relaxation to Multiphysics CMFD (M-CMFD) is to modify the fission heat source distribution supplied to the thermal hydraulics solver in M-CMFD Step 2 [Eqs. (4.10)]. We refer to the resulting relaxed scheme as Relaxed CMFD (R-CMFD). This method is outlined next:

R-CMFD Step 0 : Initialization

This step is identical to M-CMFD Step 0.

R-CMFD Step 1 : Neutron Transport Sweep

This step is identical to M-CMFD Step 1. (See Eqs. (4.9).)

R-CMFD Step 2 : Thermal Hydraulics Solve

As in M-CMFD Step 2, we update the fission heat source distribution $h^{(n+\frac{1}{2})}$ using the latest nuclear data $\Sigma_u^{(n)}$ and scalar flux $\phi^{(n+\frac{1}{2})}$ estimates:

$$h^{(n+\frac{1}{2})}(\boldsymbol{r}) = \int_0^\infty \kappa\Sigma_f^{(n)}(\boldsymbol{r}, E')\phi^{(n+\frac{1}{2})}(\boldsymbol{r}, E')\,\mathrm{d}E' \,. \tag{4.15a}$$

Using the *relaxation factor* α, we compute the *relaxed heat source distribution* $\tilde{h}^{(n+\frac{1}{2})}$ using a weighted average of $h^{(n+\frac{1}{2})}$ and $h^{(n-\frac{1}{2})}$, the Eq. (4.15a)-evaluated heat source distributions from the current and previous iterations, respectively:

$$\tilde{h}^{(n+\frac{1}{2})}(\boldsymbol{r}) = \alpha h^{(n+\frac{1}{2})}(\boldsymbol{r}) + (1-\alpha)h^{(n-\frac{1}{2})}(\boldsymbol{r}) \,. \tag{4.15b}$$

The relaxation factor α falls within $0 \leq \alpha \leq 1$; $\alpha = 1$ corresponds to no relaxation, $\alpha = 0$ corresponds to full relaxation. We note the distinction between the unrelaxed heat source distribution $h^{(n+\frac{1}{2})}$ [Eq. (4.15a)] and the relaxed heat source distribution $\tilde{h}^{(n+\frac{1}{2})}$ [Eq. (4.15b)]. The presence, or lack thereof, of an over-tilde distinguishes between the two. The two heat source distributions are equivalent upon convergence, or when no relaxation is applied ($\alpha = 1$).

Using the relaxed heat source distribution $\tilde{h}^{(n+\frac{1}{2})}$, we solve the thermal hy-

draulics problem for the updated temperature $T^{(n+\frac{1}{2})}$:

$$\mathcal{L}T^{(n+\frac{1}{2})}(\boldsymbol{r}) = \tilde{h}^{(n+\frac{1}{2})}(\boldsymbol{r}) \; . \tag{4.15c}$$

R-CMFD Step 3 : Nuclear Data Evaluation

This step is identical to M-CMFD Step 3. (See Eqs. (4.11)-(4.12).)

R-CMFD Step 4 : Diffusion Solve

This step is identical to M-CMFD Step 4. (See Eqs. (4.13)-(4.14).)

We note that when no relaxation is applied in Eq. (4.15b) (i.e. $\alpha = 1$ and $\tilde{h}^{(n+\frac{1}{2})} = h^{(n+\frac{1}{2})}$), the M-CMFD and R-CMFD methods are identical. R-CMFD generalizes M-CMFD. As such, we will primarily mention R-CMFD throughout the remainder of this thesis, keeping in mind that we can revert back to M-CMFD by setting the relaxation factor $\alpha = 1$. We also note that when R-CMFD is applied to a single-physics transport problem, that is, a problem with temperature-independent nuclear data, the method is equivalent to the single-physics CMFD method outlined in Ch. 3 (the same condition holds for every practical multiphysics iterative method we discuss in this chapter, the methods do not "break-down" when feedback is completely removed).

We now provide some general remarks. Relaxed CMFD (R-CMFD) is a simple modification of the single-physics neutronics acceleration scheme Coarse-Mesh Finite Difference (CMFD). Each iteration of R-CMFD involves the following nontrivial (i.e. potentially costly) evaluations:

(i) A transport sweep of Eqs. (4.9), which involves the inversion of the transport leakage-collision operator $(\hat{\boldsymbol{\Omega}} \cdot \boldsymbol{\nabla} + \Sigma_t)$.

(ii) A thermal hydraulics solve in Eqs. (4.15), which requires the inversion of the thermal hydraulics operator $\mathcal{L}$.

(As a reminder, for the sake of generality, we have left the definition of $\mathcal{L}$ unspecified. Depending on the concrete definition of $\mathcal{L}$, the cost of inverting it can vary widely. For example, inverting $\mathcal{L}$ may be as simple as solving a 1-D, axial, energy balance in a system-level thermal hydraulics code, or as involved as solving a coupled set of fluid mass, momentum, and energy conservation equations in a subchannel code. The key point here is that the cost of solving

Eq. (4.15c) may vary by orders of magnitude depending on the precise definition of $\mathcal{L}$. Thus, we cannot assume that inverting $\mathcal{L}$ is inexpensive.)

(iii) Evaluation of nuclear data Σ_u (macroscopic cross sections and related parameters) in Eq. (4.11).

(We have left the description of the nuclear data update in Eq. (4.11) unstated. In this thesis, we treat nuclear data evaluations as a black box. Like the inversion of $\mathcal{L}$, the cost associated with updating Σ_u is problem dependent.)

(iv) Solution of the transport-corrected diffusion eigenvalue problem in Eqs. (4.13).

(Eqs. (4.13) reside in an angularly integrated, four-dimensional phase space, two dimensions smaller than the six-dimensional phase space occupied by the transport problem that is swept in Eqs. (4.9). Eqs. (4.9) specify a fixed-source transport problem, while Eqs. (4.13) specify a diffusion eigenvalue problem. In general, eigenvalue neutronics problems are more costly to solve than fixed-source problems. While the reduced phase space of the diffusion system helps to limit costs, the time required to solve the diffusion eigenvalue problem in Eqs. (4.13) is not negligible.)

To reiterate, in every R-CMFD iteration we must: (i) perform a transport sweep, (ii) solve a thermal hydraulics problem, (iii) update cross sections and related nuclear data, and (iv) solve a diffusion eigenvalue problem. The cost of each of these components can vary widely from problem to problem. A major benefit of R-CMFD is that each of the above nontrivial operations (i)-(iv) is performed only once during every iteration. Thus, on a per iteration basis, R-CMFD requires the least amount of computational effort of any of the multiphysics iterative methods discussed in this chapter.

As stated in Ch. 3, when CMFD is applied to single-physics transport eigenvalue problems under nominal conditions (e.g., with suitable discretization), it achieves a spectral radius of $\rho \approx 0.22$. This is a major improvement over unaccelerated Source Iteration (SI), yielding convergence in about ten iterations for CMFD rather than hundreds or even thousands for SI, depending on the optical thickness of the problem. Unfortunately, feedback-induced nonlinearities detrimentally affect M-CMFD's performance, leading to an increased spectral radius and possibly even divergence. This happens even though the magnitude of the nuclear data nonlinearities are small

in practical reactor calculations:

$$\frac{T}{\Sigma_u}\left|\frac{\partial \Sigma_u}{\partial T}\right| \approx O(10^{-2}) \ .$$

To contend with M-CMFD's poor convergence behavior in the presence of feedback, *relaxation* is typically used in the Relaxed CMFD (R-CMFD) method (see Eqs. (4.15) with $0 \leq \alpha < 1$). The relaxation factor α affects the heat source distribution supplied to the thermal hydraulics problem in R-CMFD Step 2. $\alpha = 0$ corresponds to full relaxation, with $\alpha = 1$ indicating no relaxation. Typically, a value of $0.5 < \alpha < 0.7$ is used for reactor physics applications (e.g., the MPACT code uses $\alpha = 0.5$ as its default [80]) – although, the optimal relaxation factor varies from problem to problem, and costly parametric studies are required to determine the optimal value for any given scenario. Also, for sufficiently difficult problems, R-CMFD will diverge, regardless of the relaxation factor chosen.

To put this into perspective, even though R-CMFD has minimal evaluation of (potentially) costly components in each iteration, its larger spectral radius can lead to a requirement of many outer iterations to solve a given problem. Relaxation can help alleviate this issue, but it is not a panacea. This often results in R-CMFD being computationally expensive.

4.3.1. NOPC-CMFD

Shen's Nearly Optimally Partially Converged-CMFD (NOPC-CMFD) method [91, 94, 96] addresses M-CMFD's poor convergence properties without using an explicit relaxation factor. Rather, NOPC-CMFD *partially* converges the low-order CMFD diffusion problem, using a Fourier analysis-based prescription to determine a "nearly optimal" degree of partial convergence for a given multiphysics problem. This helps to stabilize iterations. We view NOPC-CMFD to be mathematically-based, owing to its dependence on Fourier analysis results. In this thesis, we propose an alternative, physics-based method, NILO-CMFD, which does not require the use of a Fourier analysis to optimize performance.

4.3.2. Summary

All of Chs. 1 – 3 up until this point has been a review of prior work, conducted by others in the field. From this point forward, unless otherwise stated, the work

presented is, at least in-part, the original contribution of the primary author.

In the remainder of this thesis, we explore alternatives to R-CMFD. We have already noted R-CMFD's stability issues, suggesting that there is significant room for improvement. In the remainder of this chapter, we develop approaches, both theoretical and practical, that aim to improve the stability of R-CMFD and preserve a low iterative spectral radius, even in the presence of strong feedback.

4.4. Theoretical Methods

Next, we shift away from the flawed (owing to poor stability) M-CMFD and R-CMFD iterative methods. Instead, we explore two theoretical approaches to solving the neutron transport – thermal hydraulics problem in Eqs. (4.3)-(4.4), namely, Theoretical Method 0 (TM-0) and Theoretical Method 1 (TM-1). Unlike M-CMFD and R-CMFD, TM-0 and TM-1 cannot be straightforwardly implemented in a computer code. Rather, these methods serve as a purely theoretical, mathematical starting point to which we apply a series of practical approximations in order to derive more realistic (i.e. implementable) iterative methods that seek to improve upon M-CMFD and R-CMFD.

4.4.1. Theoretical Method 0 (TM-0)

Theoretical Method 0 (TM-0) is "derived" by simply adding iteration superscripts to the neutron transport – thermal hydraulics problem [Eqs. (4.3)-(4.4)]. (Additionally, we use the shorthand described by Eq. (4.2) to refer to nuclear data evaluations.) An outline of TM-0 follows:

TM-0 Step 0 : Initialization

Initial estimates are not needed for this method. Rather, we simply set the iteration index n to 0 and proceed to the first, **and only**, iteration.

TM-0 Step 1 : Transport – Thermal Hydraulics Solve

We solve the following coupled neutron transport [Eqs. (4.16)], thermal hydraulics [Eqs. (4.17)], and nuclear data update [Eq. (4.18)] equations, in which all unknown quantities are evaluated with iteration superscript $(n+1)$, for the angular flux $\psi^{(n+1)}$, eigenvalue $\lambda^{(n+1)}$, temperature $T^{(n+1)}$, and nuclear data $\Sigma_u^{(n+1)}$:

(A gentle message of reassurance may be needed before jumping into the equations included in Theoretical Method 0 (TM-0). At first glance, the coupled Eqs. (4.16)-(4.18) may look impenetrable with their abundance of iteration superscripts. However, we reassure the reader that all iteration superscripts in these equations are identically $(n+1)$. Eqs. (4.16)-(4.18) are notationally more complex, but, in actuality, **identical** to the equations describing the original neutron transport – thermal hydraulics multiphysics problem in Eqs. (4.3)-(4.4).)

$$\begin{aligned}\hat{\boldsymbol{\Omega}} \cdot \boldsymbol{\nabla} \psi^{(n+1)}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E) + \Sigma_t^{(n+1)}(\boldsymbol{r}, E)\psi^{(n+1)}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E) \\ = \frac{1}{4\pi} \iint \Sigma_s^{(n+1)}(\boldsymbol{r}, E' \rightarrow E)\psi^{(n+1)}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E') \, \mathrm{d}\Omega' \, \mathrm{d}E' \\ + \lambda^{(n+1)} \frac{\chi^{(n+1)}(\boldsymbol{r}, E)}{4\pi} \iint \nu\Sigma_f^{(n+1)}(\boldsymbol{r}, E')\psi^{(n+1)}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E') \, \mathrm{d}\Omega' \, \mathrm{d}E' \, ,\end{aligned} \tag{4.16a}$$

$$P = \iiint \kappa\Sigma_f^{(n+1)}(\boldsymbol{r}', E')\psi^{(n+1)}(\boldsymbol{r}', \hat{\boldsymbol{\Omega}}', E') \, \mathrm{d}^3 r' \, \mathrm{d}\Omega' \, \mathrm{d}E' \, , \tag{4.16b}$$

$$h^{(n+1)}(\boldsymbol{r}) = \iint \kappa\Sigma_f^{(n+1)}(\boldsymbol{r}, E')\psi^{(n+1)}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E') \, \mathrm{d}\Omega' \, \mathrm{d}E' \, , \tag{4.17a}$$

$$\mathcal{L}T^{(n+1)}(\boldsymbol{r}) = h^{(n+1)}(\boldsymbol{r}) \, , \tag{4.17b}$$

$$\Sigma_u^{(n+1)}(\boldsymbol{r}, E) = \Sigma_u(\boldsymbol{r}, T^{(n+1)}(\boldsymbol{r}), E) \, . \tag{4.18}$$

After solving Eqs. (4.16)-(4.18), we exit the iteration loop. The solution values of Eqs. (4.16)-(4.18) are equivalent to those of the original multiphysics problem [Eqs. (4.3)-(4.4)]. TM-0 converges in a single iteration.

Again, we stress that Theoretical Method 0 (TM-0) converges in a single iteration. The spectral radius of TM-0 is zero! Unfortunately, in order to perform a single iteration of TM-0, we must solve a problem [Eqs. (4.16)-(4.18)] that is identical to the original multiphysics problem [Eqs. (4.3)-(4.4)]!

Let us define a "practical" iterative method as: a process by which we break down the solution of a difficult problem into a sequence of solutions to simpler, more solvable, problems. TM-0 does not satisfy this definition. A single iteration of TM-0 is in no way "simpler" than solving the original "difficult" neutron transport – thermal hydraulics multiphysics problem directly [Eqs. (4.3)-(4.4)]. TM-0 is not a practical

iterative method. Instead, it is a theoretical approach. While TM-0 cannot be implemented in a computer code, it serves as a mathematical base that will be modified and built upon in upcoming sections.

4.4.2. Theoretical Method 1 (TM-1)

Theoretical Method 1 (TM-1) is a simple modification to Theoretical Method 0 (TM-0). In addition to Eqs. (4.16)-(4.18), TM-1 Step 1 additionally includes the transport-corrected diffusion equation and related parameter definitions. Whereas TM-0 Step 1 involves a coupled neutron transport – thermal hydraulics problem [Eqs. (4.16)-(4.18)], TM-1 Step 1 includes the solution of a coupled neutron transport – thermal hydraulics – (transport-corrected) neutron diffusion problem. Theoretical Method 1 (TM-1) is outlined below:

TM-1 Step 0 : Initialization

This is identical to TM-0 Step 0.

TM-1 Step 1 : Transport – Thermal Hydraulics – Diffusion Solve

We solve the coupled neutron transport [Eqs. (4.19)], thermal hydraulics [Eqs. (4.20)], nuclear data update [Eq. (4.21)], and transport-corrected neutron diffusion [Eqs. (4.22)] equations for the scalar flux $\phi^{(n+1)}$, eigenvalue $\lambda^{(n+1)}$, temperature $T^{(n+1)}$, and nuclear data $\Sigma_u^{(n+1)}$ updates:

$$\begin{aligned}
&\hat{\boldsymbol{\Omega}} \cdot \boldsymbol{\nabla} \psi^{(n+1)}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E) + \Sigma_t^{(n+1)}(\boldsymbol{r}, E) \psi^{(n+1)}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}, E) \\
&\quad = \frac{1}{4\pi} \int \Sigma_s^{(n+1)}(\boldsymbol{r}, E' \to E) \phi^{(n+1)}(\boldsymbol{r}, E') \, \mathrm{d}E' \\
&\qquad + \lambda^{(n+1)} \frac{\chi^{(n+1)}(\boldsymbol{r}, E)}{4\pi} \int \nu \Sigma_f^{(n+1)}(\boldsymbol{r}, E') \phi^{(n+1)}(\boldsymbol{r}, E') \, \mathrm{d}E' \, ,
\end{aligned} \tag{4.19a}$$

$$\phi^{(n+1)}(\boldsymbol{r}, E) = \int_{4\pi} \psi^{(n+1)}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E) \, \mathrm{d}\Omega' \, , \tag{4.19b}$$

$$\boldsymbol{J}^{(n+1)}(\boldsymbol{r}, E) = \int_{4\pi} \hat{\boldsymbol{\Omega}}' \psi^{(n+1)}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E) \, \mathrm{d}\Omega' \, , \tag{4.19c}$$

$$\begin{aligned}
&\hat{\boldsymbol{D}}^{(n+1)}(\boldsymbol{r}, T^{(n+1)}(\boldsymbol{r}), E) \\
&\quad = \frac{\boldsymbol{J}^{(n+1)}(\boldsymbol{r}, E) + D(\boldsymbol{r}, T^{(n+1)}(\boldsymbol{r}), E) \boldsymbol{\nabla} \phi^{(n+1)}(\boldsymbol{r}, E)}{\phi^{(n+1)}(\boldsymbol{r}, E)} \, ,
\end{aligned} \tag{4.19d}$$

$$h^{(n+1)}(\boldsymbol{r}) = \int \kappa\Sigma_f^{(n+1)}(\boldsymbol{r}, E')\phi^{(n+1)}(\boldsymbol{r}, E')\,\mathrm{d}E' \;, \tag{4.20a}$$

$$\mathcal{L}T^{(n+1)}(\boldsymbol{r}) = h^{(n+1)}(\boldsymbol{r}) \;, \tag{4.20b}$$

$$\Sigma_u^{(n+1)}(\boldsymbol{r}, E) = \Sigma_u(\boldsymbol{r}, T^{(n+1)}(\boldsymbol{r}), E) \;, \tag{4.21}$$

$$\begin{aligned} -\boldsymbol{\nabla}\cdot D(\boldsymbol{r}, T^{(n+1)}(\boldsymbol{r}), E)\boldsymbol{\nabla}\phi^{(n+1)}(\boldsymbol{r}, E) + \boldsymbol{\nabla}\cdot\hat{\boldsymbol{D}}^{(n+1)}(\boldsymbol{r}, T^{(n+1)}(\boldsymbol{r}), E)\phi^{(n+1)}(\boldsymbol{r}, E) \\ + \Sigma_t^{(n+1)}(\boldsymbol{r}, E)\phi^{(n+1)}(\boldsymbol{r}, E) = \int \Sigma_s^{(n+1)}(\boldsymbol{r}, E' \to E)\phi^{(n+1)}(\boldsymbol{r}, E')\,\mathrm{d}E' \\ + \lambda^{(n+1)}\chi^{(n+1)}(\boldsymbol{r}, E)\int \nu\Sigma_f^{(n+1)}(\boldsymbol{r}, E')\phi^{(n+1)}(\boldsymbol{r}, E')\,\mathrm{d}E' \;, \end{aligned} \tag{4.22a}$$

$$P = \iint \kappa\Sigma_f^{(n+1)}(\boldsymbol{r}', E')\phi^{(n+1)}(\boldsymbol{r}', E')\,\mathrm{d}^3r'\,\mathrm{d}E' \;. \tag{4.22b}$$

In general, $D(\boldsymbol{r}, T^{(n+1)}(\boldsymbol{r}), E)$ may take any reasonable form. We choose to define it as the standard diffusion coefficient:

$$D^{(n+1)}(\boldsymbol{r}, E) = D(\boldsymbol{r}, T^{(n+1)}(\boldsymbol{r}), E) = \frac{1}{3\Sigma_t^{(n+1)}(\boldsymbol{r}, E)} \;. \tag{4.23}$$

After solving Eqs. (4.19)-(4.22), we exit the iteration loop. As with TM-0, no terms are lagged, and TM-1 converges in a single iteration.

Both TM-0 and TM-1 converge "instantly", their spectral radii are zero! By adding a transport-corrected diffusion equation to TM-0 Step 1 in order to form TM-1 Step 1, we have made no approximations. As with TM-0 Step 1 [Eqs. (4.16)-(4.18)], TM-1 Step 1 [Eqs. (4.19)-(4.22)] is just as difficult to evaluate as the direct solution of the original neutron transport – thermal hydraulics problem [Eqs. (4.3)-(4.4)]. Like TM-0, TM-1 is not a practical method. However, TM-1 has a convenient form from which we can derive more practical, CMFD-based, multiphysics iterative methods. Next, we motivate the manipulations we are about to perform on TM-1.

Looking at TM-1 Step 1, let us examine the connections between Eqs. (4.19)-(4.22). That is, how do the neutron transport [Eqs. (4.19)], thermal hydraulics [Eqs. (4.20)], nuclear data [Eq. (4.21)], and transport-corrected diffusion [Eq. (4.22)] components

of this coupled problem depend on one another?

For now, let us focus on Eqs. (4.22). Taken alone, Eqs. (4.22) specify a transport-corrected diffusion eigenvalue problem. This particular transport-corrected diffusion eigenvalue problem is nonstandard since, in addition to the scalar flux $\phi^{(n+1)}$ and eigenvalue $\lambda^{(n+1)}$ unknowns, the nuclear data $\Sigma_u^{(n+1)}$, diffusion coefficient $D^{(n+1)}$, and transport correction vector $\hat{\boldsymbol{D}}^{(n+1)}$ are also unknowns. Therefore, one cannot solve Eqs. (4.22) for the scalar flux $\phi^{(n+1)}$ and eigenvalue $\lambda^{(n+1)}$ updates in a vacuum. We require additional equations to specify the other unknowns: $\Sigma_u^{(n+1)}$, $D^{(n+1)}$, and $\hat{\boldsymbol{D}}^{(n+1)}$.

In Theoretical Method 1 (TM-1), the extra unknowns ($\Sigma_u^{(n+1)}$, $D^{(n+1)}$, and $\hat{\boldsymbol{D}}^{(n+1)}$) in the transport-corrected diffusion eigenvalue problem [Eqs. (4.22)] are determined by the full (i.e. unapproximated) transport Eqs. (4.19), thermal hydraulics Eqs. (4.20), and nuclear data update Eq. (4.21). The values of $\Sigma_u^{(n+1)}$, $D^{(n+1)}$, and $\hat{\boldsymbol{D}}^{(n+1)}$ determined by Eqs. (4.19)-(4.21) in turn depend, either directly or indirectly, on the scalar flux solution $\phi^{(n+1)}$ of the transport-corrected diffusion eigenvalue problem itself [Eqs. (4.22)]. **We describe terms like these, that appear in a transport-corrected diffusion problem and depend in any way on $\phi^{(n+1)}$, the scalar flux solution of the diffusion problem itself, as being *nonlinearly implicit*.** In TM-1's transport-corrected diffusion Eqs. (4.22), $\Sigma_u^{(n+1)}$, $D^{(n+1)}$, and $\hat{\boldsymbol{D}}^{(n+1)}$ are all nonlinearly implicit through their dependence on $\phi^{(n+1)}$ in Eqs. (4.19)-(4.21).

In contrast, we describe quantities appearing in a transport-corrected diffusion problem that do not depend on the scalar flux solution $\phi^{(n+1)}$ of the diffusion problem as being *explicitly evaluated*. For example, the nuclear data $\Sigma_u^{(n+\frac{1}{2})}$, diffusion coefficient $D^{(n+\frac{1}{2})}$, and transport correction vector $\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}$ in M-CMFD and R-CMFD Step 4 Eqs. (4.13) are explicitly evaluated. Explicit terms like these are often described as being *lagged*. That is, in the $(n+1)^{\text{st}}$ iteration of M-CMFD and R-CMFD, $\Sigma_u^{(n+\frac{1}{2})}$, $D^{(n+\frac{1}{2})}$, and $\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}$ are determined prior to evaluation of Step 4 and fixed throughout the solution of Eqs. (4.13). Since they are fixed during Step 4, they do not directly depend on the scalar flux solution $\phi^{(n+1)}$ of the transport-corrected diffusion problem [Eqs. (4.13)]. In this way, they are explicitly evaluated.

To add more terminology to the mix, we say that in the TM-1 transport-corrected diffusion Eqs. (4.22), $\Sigma_u^{(n+1)}$, $D^{(n+1)}$, and $\hat{\boldsymbol{D}}^{(n+1)}$ are *fully nonlinearly implicit* because they are evaluated in connection with the full (i.e. unapproximated) transport, thermal hydraulics, and nuclear data update physics as specified by Eqs. (4.19)-(4.21).

Shortly, we will see how the fully nonlinearly implicit treatment of parameters in

the transport-corrected diffusion problem may drastically increase its solution cost, as well as the cost of the multiphysics iterative method as a whole. For example, the impracticality of TM-1 originates from the fully nonlinearly implicit treatment of Σ_u, D, and $\hat{\boldsymbol{D}}$ in Eqs. (4.22). In effect, with all parameters being fully nonlinearly implicit, this means that we do not approximate the transport, thermal hydraulics, or nuclear data physics in the TM-1 transport-corrected diffusion problem.

In the X-CMFD, NILO-A, and NILO-CMFD methods, introduced shortly hereafter, the direct connections to *high-order* (i.e. unapproximated) physics in the TM-1 transport-corrected diffusion problem are "loosened" or, in the case of the connection with the transport physics, "severed". In X-CMFD, NILO-A, and NILO-CMFD, the nonlinear connection between the transport-corrected diffusion and transport equations, e.g., between Eqs. (4.19) and (4.22) in TM-1, is removed by explicitly (not implicitly) evaluating D and $\hat{\boldsymbol{D}}$ in the transport-corrected diffusion problem. In NILO-CMFD, we additionally loosen the connections between the transport-corrected diffusion problem and the thermal hydraulics and nuclear data update physics by using *low-order* (i.e. approximate) descriptions of the behavior of temperature $T^{(n+1)}$ and nuclear data $\Sigma_u^{(n+1)}$ in response to the transport-corrected diffusion scalar flux solution $\phi^{(n+1)}$. In NILO-CMFD, we describe these temperatures and nuclear data as being *approximately nonlinearly implicit* (as opposed to fully nonlinearly implicit), since they still depend on the scalar flux solution of the transport-corrected diffusion problem, but in an approximated manner.

We now refocus. From our previous discussions of M-CMFD, R-CMFD, and TM-1, we recap a few key points:

- In M-CMFD and R-CMFD, the nuclear data Σ_u, diffusion coefficient D, and transport correction vector $\hat{\boldsymbol{D}}$ appearing in the methods' transport-corrected diffusion eigenvalue problem [Eqs. (4.13)] are explicitly evaluated. They are lagged.

- While an iteration of M-CMFD and R-CMFD involves a single evaluation of each of the (potentially) expensive component solves and updates mentioned previously (i.e. a single transport sweep, inversion of the thermal hydraulics operator $\mathcal{L}$, nuclear data update, and transport-corrected diffusion eigenvalue solve), these approaches suffer from stability and convergence rate degradation when applied to multiphysics problems with realistic feedback intensities. (The underperformance of these methods is due to too many terms being treated

explicitly in the transport-corrected diffusion Eqs. (4.13). In the M-CMFD and R-CMFD transport-corrected diffusion solve, the thermal hydraulics and nuclear data physics are not treated implicitly enough.)

- In TM-1, the nuclear data, diffusion coefficient, and transport correction vector appearing in the transport-corrected diffusion eigenvalue Eqs. (4.22) are *nonlinearly implicit*, since they depend on the scalar flux solution of the transport-corrected diffusion problem itself. To be more specific, these terms are *fully nonlinearly implicit*, since they are evaluated in connection with the unapproximated transport, thermal hydraulics, and nuclear data physics as specified by Eqs. (4.19)-(4.21).

- TM-1 converges in a single iteration: the spectral radius of the method is zero. However, the cost of evaluating a single TM-1 iteration is no less difficult than directly solving the original neutron transport – thermal hydraulics multiphysics problem specified by Eqs. (4.3)-(4.4). (These comments imply that TM-1 is unrealistic because **all** of the terms in its transport-corrected diffusion problem are treated fully implicitly.)

Taking these points into consideration, the remainder of this thesis aims to explore the following question: "By reducing the strict fully implicitness of terms in the TM-1 transport-corrected diffusion equation, can we develop a CMFD-based multiphysics iterative method that approaches the stability and convergence rate of TM-1, while more closely matching the cost-per-iteration of M-CMFD and R-CMFD?" To address this question, we make a series of practical approximations, starting from the fully nonlinearly implicit transport-corrected diffusion Eqs. (4.22) of TM-1 and methodically reducing the "fully implicit"-ness of terms.

4.5. X-CMFD

The X-CMFD method is derived from Theoretical Method 1 (TM-1) by lagging the diffusion coefficient D and transport-correction vector $\hat{\boldsymbol{D}}$ in the transport-corrected diffusion eigenvalue problem [Eqs. (4.22)]. In X-CMFD, rather than being treated fully nonlinearly implicitly, the transport correction vector $\hat{\boldsymbol{D}}$ is treated explicitly. This makes it possible to de-couple the transport physics [Eqs. (4.19)] from the transport-corrected diffusion problem [Eqs. (4.22)]. In order to evaluate $\hat{\boldsymbol{D}}$ explicitly,

we need to decide on its lagged value. In X-CMFD, we do what experience suggests and perform one transport sweep each outer iteration to obtain a suitable, explicit value of $\hat{\boldsymbol{D}}$. Additionally, in order to keep the X-CMFD method similar in structure to M-CMFD and R-CMFD, we include an additional high-order thermal hydraulics solve and nuclear data update in each outer iteration. However, we note that these steps are somewhat redundant in X-CMFD, considering the inclusion of these high-order components in the method's transport-corrected diffusion solve. We outline X-CMFD below:

X-CMFD Step 0 : Initialization

This step is identical to M/R-CMFD Step 0.

X-CMFD Step 1 : Neutron Transport Sweep

This step is identical to M/R-CMFD Step 1. (See Eqs. (4.9).)

X-CMFD Step 2 : Thermal Hydraulics Solve

This step is identical to M/R-CMFD Step 2, **without relaxation**. (See Eqs. (4.10) or Eqs. (4.15), with $\alpha = 1$.)

X-CMFD Step 3 : Nuclear Data Evaluation

This step is identical to M/R-CMFD Step 3. (See Eqs. (4.11)-(4.12).)

X-CMFD Step 4 : Diffusion – Thermal Hydraulics Solve

Using the diffusion coefficient $D^{(n+\frac{1}{2})}$ and transport-correction vector $\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}$ calculated in X-CMFD Step 3, we solve the following coupled system of thermal hydraulics, nuclear data update, and transport-corrected diffusion equations for the end-of-iteration scalar flux $\phi^{(n+1)}$, eigenvalue $\lambda^{(n+1)}$, and nuclear data $\Sigma_u^{(n+1)}$:

$$h^{(n+1)}(\boldsymbol{r}) = \int \kappa\Sigma_f^{(n+1)}(\boldsymbol{r}, E')\phi^{(n+1)}(\boldsymbol{r}, E')\,\mathrm{d}E' \;, \tag{4.24a}$$

$$\mathcal{L}T^{(n+1)}(\boldsymbol{r}) = h^{(n+1)}(\boldsymbol{r}) \;, \tag{4.24b}$$

$$\Sigma_u^{(n+1)}(\boldsymbol{r}, E) = \Sigma_u(\boldsymbol{r}, T^{(n+1)}(\boldsymbol{r}), E) \;, \tag{4.25}$$

$$
\begin{aligned}
&-\boldsymbol{\nabla}\cdot D^{(n+\frac{1}{2})}\boldsymbol{\nabla}\phi^{(n+1)}(\boldsymbol{r},E)+\boldsymbol{\nabla}\cdot\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}(\boldsymbol{r},E)\phi^{(n+1)}(\boldsymbol{r},E)\\
&\quad+\Sigma_t^{(n+1)}(\boldsymbol{r},E)\phi^{(n+1)}(\boldsymbol{r},E)=\int\Sigma_s^{(n+1)}(\boldsymbol{r},E'\rightarrow E)\phi^{(n+1)}(\boldsymbol{r},E')\,\mathrm{d}E'\\
&\qquad+\lambda^{(n+1)}\frac{\chi^{(n+1)}(\boldsymbol{r},E)}{4\pi}\int\nu\Sigma_f^{(n+1)}(\boldsymbol{r},E')\phi^{(n+1)}(\boldsymbol{r},E')\,\mathrm{d}E'\ ,
\end{aligned}
\tag{4.26a}
$$

$$
P=\iint\kappa\Sigma_f^{(n+1)}(\boldsymbol{r}',E')\phi^{(n+1)}(\boldsymbol{r}',E')\,\mathrm{d}^3r'\,\mathrm{d}E'\ . \tag{4.26b}
$$

We refer to the problem specified by Eqs. (4.24)-(4.26) as the *X-CMFD nonlinear diffusion problem.*

After solving the X-CMFD nonlinear diffusion problem [Eqs. (4.24)-(4.26)], we check convergence and max iteration criteria. If they are not met, we increase the iteration index n by one and return to Step 1 to begin the next iteration.

X-CMFD Step 4 requires the solution of a coupled system of thermal hydraulics, nuclear data update, and transport-corrected diffusion equations [Eqs. (4.24)-(4.26)]. We refer to Eqs. (4.24)-(4.26) as the X-CMFD nonlinear diffusion problem. By explicitly evaluating the transport correction vector $\hat{\boldsymbol{D}}$ in X-CMFD Step 4, we remove the nonlinear connection between the transport-corrected neutron diffusion and neutron transport physics in TM-1 Step 1's Eqs. (4.19)-(4.22). Severing this link transforms the purely theoretical TM-1 into the (at least somewhat) practical X-CMFD method. (Variations of X-CMFD can be derived without including X-CMFD Steps 2 – 3. However, we include these somewhat redundant steps in order to write X-CMFD in a form similar to the standard R-CMFD method. By doing this, we simplify the implementation of X-CMFD in computer codes that already implement R-CMFD.)

Unlike TM-1, it is possible to implement X-CMFD in a real-world computer code. However, while X-CMFD is implementable, it is not necessarily efficient. In order to implement X-CMFD, it is necessary to solve the X-CMFD nonlinear diffusion problem [Eqs. (4.24)-(4.26)] every iteration. The solution of this nonlinear problem involves its own iterative procedure, which we refer to as the *inner iteration.* We contrast this with the *outer iteration*, e.g., evaluation of X-CMFD Steps 1 – 4. Next, we present two inner iteration schemes based on previous work by Walker and Shen [91, 93, 110, 111]. This will show the limitations of X-CMFD, namely, its tendency to overburden the thermal hydraulics solver and nuclear data update library as compared to M-CMFD and R-CMFD.

4.5.1. Inner Iteration

The crux of a real-world implementation of X-CMFD is the development of a suitable inner iteration scheme to solve the X-CMFD nonlinear diffusion problem [Eqs. (4.24)-(4.26)]. Since the temperatures and nuclear data in Eqs. (4.24)-(4.26) are fully nonlinearly implicit, X-CMFD inner iterations require that we consider the high-order (i.e. unapproximated) thermal hydraulics solver and nuclear data update utilities in the inner iterations.

In this subsection, we describe two inner iteration approaches for X-CMFD, namely, the "Eigenvalue Level" scheme based on work by Walker [110, 111], and the "Nonlinear Power Iteration" scheme based on work by Shen [91, 93]. Up to this point, we have exclusively used the outer iteration index n. Now that we are describing inner iterations, we must introduce new notation. We use the inner iteration index p and refer to starting estimates of the $(p+1)^{\text{st}}$ inner iteration of the $(n+1)^{\text{st}}$ outer iteration with the iteration superscript $(n+1,p)$. Similarly, concluding estimates of this same inner iteration are referred to with the iteration superscript $(n+1,p+1)$.

Eigenvalue Level (EL)

One approach to solving X-CMFD's Eqs. (4.24)-(4.26) is to iterate between thermal hydraulics solves, nuclear data updates, and diffusion eigenvalue solves. This procedure was used by Walker in [110, 111]. We call Walker's method "Eigenvalue Level" (EL) inner iteration:

Eigenvalue Level (EL) Step 0 : Initialization

Before entering the inner iteration for the $(n+1)^{\text{st}}$ X-CMFD outer iteration, we require initial inner iteration estimates of the scalar flux $\phi^{(n+1,0)}$ and nuclear data $\Sigma_u^{(n+1,0)}$. We use the most up-to-date values available for these estimates:

$$\phi^{(n+1,0)} = \phi^{(n+\frac{1}{2})} , \tag{4.27a}$$

$$\Sigma_u^{(n+1,0)} = \Sigma_u^{(n+\frac{1}{2})} . \tag{4.27b}$$

After setting initial estimates, we set the inner iteration index p to 0 and continue to EL Step 1 to begin the inner iteration procedure.

EL Step 1 : Thermal Hydraulics Solve

At the beginning of the $(p+1)^{\text{st}}$ inner iteration of the $(n+1)^{\text{st}}$ outer iteration, we use the most recent estimates of the scalar flux $\phi^{(n+1,p)}$ and nuclear data $\Sigma_u^{(n+1,p)}$ to update the inner iteration fission heat source distribution $h^{(n+1,p+1)}$:

$$h^{(n+1,p+1)}(\boldsymbol{r}) = \int \kappa\Sigma_f^{(n+1,p)}(\boldsymbol{r}, E')\phi^{(n+1,p)}(\boldsymbol{r}, E')\,\mathrm{d}E' \ . \tag{4.28a}$$

Using the heat source distribution $h^{(n+1,p+1)}$, we solve the full thermal hydraulics problem for the updated inner iteration temperature $T^{(n+1,p+1)}$:

$$\mathcal{L}T^{(n+1,p+1)}(\boldsymbol{r}) = h^{(n+1,p+1)}(\boldsymbol{r}) \ . \tag{4.28b}$$

EL Step 2 : Nuclear Data Evaluation

The temperature estimate $T^{(n+1,p+1)}$ is used by the nuclear data processing utility to evaluate new inner iteration nuclear data estimates:

$$\Sigma_u^{(n+1,p+1)}(\boldsymbol{r}, E) = \Sigma_u(\boldsymbol{r}, T^{(n+1,p+1)}(\boldsymbol{r}), E) \ . \tag{4.29}$$

EL Step 3 : Diffusion Eigenvalue Solve

By lagging nuclear data, we form a linear, transport-corrected, diffusion eigenvalue problem. We solve this problem for new inner iteration scalar flux $\phi^{(n+1,p+1)}$ and eigenvalue $\lambda^{(n+1,p+1)}$ estimates. (As a reminder, throughout the inner iterations, the diffusion coefficient $D^{(n+\frac{1}{2})}$ and transport correction vector $\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}$ are held fixed. In X-CMFD, these terms are explicitly evaluated):

$$\begin{aligned}
&-\boldsymbol{\nabla}\cdot D^{(n+\frac{1}{2})}(\boldsymbol{r}, E)\boldsymbol{\nabla}\phi^{(n+1,p+1)}(\boldsymbol{r}, E) + \boldsymbol{\nabla}\cdot\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}(\boldsymbol{r}, E)\phi^{(n+1,p+1)}(\boldsymbol{r}, E) \\
&\quad + \Sigma_t^{(n+1,p+1)}(\boldsymbol{r}, E)\phi^{(n+1,p+1)}(\boldsymbol{r}, E) \\
&\qquad = \int \Sigma_s^{(n+1,p+1)}(\boldsymbol{r}, E' \to E)\phi^{(n+1,p+1)}(\boldsymbol{r}, E')\,\mathrm{d}E' \\
&\qquad\quad + \lambda^{(n+1,p+1)}\chi^{(n+1,p+1)}(\boldsymbol{r}, E) \\
&\qquad\qquad \times \int \nu\Sigma_f^{(n+1,p+1)}(\boldsymbol{r}, E')\phi^{(n+1,p+1)}(\boldsymbol{r}, E')\,\mathrm{d}E' \ ,
\end{aligned} \tag{4.30a}$$

$$P = \iint \kappa\Sigma_f^{(n+1,p+1)}(\boldsymbol{r}', E')\phi^{(n+1,p+1)}(\boldsymbol{r}', E')\,\mathrm{d}^3r'\,\mathrm{d}E' \ . \tag{4.30b}$$

(Since Eqs. (4.30) specify a standard diffusion eigenvalue problem, any off-the-shelf method may be used to solve them.)

Next, we check inner iteration convergence and max iteration criteria. If these have been met, we set the concluding inner-iteration index for the $(n+1)^{\text{st}}$ outer iteration, $P^{(n+1)} = p + 1$ (not to be confused with the system power P), and move to the finalization step (written below). Otherwise, we increment the inner iteration index p by one and return to EL Step 1 to start the next inner iteration.

EL : Finalization

Before exiting the inner iteration completely, we set estimates of the scalar flux $\phi^{(n+1)}$, eigenvalue $\lambda^{(n+1)}$, and nuclear data $\Sigma_u^{(n+1)}$ to be used in the next outer iteration:

$$\phi^{(n+1)} = \phi^{(n+1,P^{(n+1)})} , \tag{4.31a}$$

$$\lambda^{(n+1)} = \lambda^{(n+1,P^{(n+1)})} , \tag{4.31b}$$

$$\Sigma_u^{(n+1)} = \Sigma_u^{(n+1,P^{(n+1)})} . \tag{4.31c}$$

Nonlinear Power Iteration (N-PI)

Each Eigenvalue Level (EL) inner iteration requires a diffusion eigenvalue solve [Eqs. (4.30)]. In the Nonlinear Power Iteration (N-PI) inner iteration scheme, as proposed by Shen [91, 93], a single step of power iteration is used instead.

Power iteration is a simple iterative procedure used to find the dominant (i.e. largest) eigenvalue of a system, along with its corresponding eigenvector / eigenfunction. The method can be optionally accelerated by shifting the system's eigenvalue spectrum using a Wielandt shift. For our numerical results (presented later), as well as in Shen's work [91, 93], a Wielandt shift is used. For simplicity of explanation, we do not Wielandt-shift in the following Nonlinear Power Iteration outline:

Nonlinear Power Iteration (N-PI) Step 0 : Initialization

We require initial estimates of the scalar flux $\phi^{(n+1,0)}$, eigenvalue $\lambda^{(n+1,0)}$, and nuclear data $\Sigma_u^{(n+1,0)}$ before entering the inner iteration. Initial estimates are

set at their most-recently determined values:

$$\phi^{(n+1,0)} = \phi^{(n+\frac{1}{2})} \,, \tag{4.32a}$$

$$\lambda^{(n+1,0)} = \lambda^{(n)} \,, \tag{4.32b}$$

$$\Sigma_u^{(n+1,0)} = \Sigma_u^{(n+\frac{1}{2})} \,. \tag{4.32c}$$

N-PI Step 1 : Thermal Hydraulics Solve

This step is identical to EL Step 1. (See Eqs. (4.28).)

N-PI Step 2 : Nuclear Data Evaluation

This step is identical to EL Step 2. (See Eq. (4.29).)

N-PI Step 3 : Single Power Iteration

To facilitate a compact description of this step, we define an operator notation for the transport-corrected diffusion equation with temperature-dependent nuclear data evaluated as $\Sigma_u^{(n+1,p+1)}$. We define the (transport-corrected) leakage + collision - inscatter operator $\mathcal{M}^{(n+1,p+1)}$ by:

$$\begin{aligned} &\mathcal{M}^{(n+1,p+1)} \\ &\quad \equiv -\boldsymbol{\nabla} \cdot D^{(n+\frac{1}{2})}(\boldsymbol{r}, E)\boldsymbol{\nabla}(\cdot) + \boldsymbol{\nabla} \cdot \hat{\boldsymbol{D}}^{(n+\frac{1}{2})}(\boldsymbol{r}, E)(\cdot) \\ &\qquad + \Sigma_t^{(n+1,p+1)}(\boldsymbol{r}, E)(\cdot) - \int \Sigma_s^{(n+1,p+1)}(\boldsymbol{r}, E' \to E)(\cdot)\, \mathrm{d}E' \,, \end{aligned} \tag{4.33a}$$

the fission neutron production operator $\mathcal{F}^{(n+1,p+1)}$ by:

$$\mathcal{F}^{(n+1,p+1)} \equiv \chi^{(n+1,p+1)}(\boldsymbol{r}, E) \int \nu\Sigma_f^{(n+1,p+1)}(\boldsymbol{r}, E')(\cdot)\, \mathrm{d}E' \,, \tag{4.33b}$$

and the fission heat generation operator $\mathcal{K}^{(n+1,p+1)}$ by:

$$\mathcal{K}^{(n+1,p+1)} \equiv \kappa\Sigma_f^{(n+1,p+1)}(\boldsymbol{r}, E)(\cdot) \,, \tag{4.33c}$$

We also define the following operator representing integration over the problem domain and energy spectrum:

$$\langle(\cdot)\rangle \equiv \int_0^\infty \int_V (\cdot)\, \mathrm{d}^3r'\, \mathrm{d}E' \,. \tag{4.33d}$$

Using the operators $\mathcal{M}$, $\mathcal{F}$, and $\mathcal{K}$ along with the operator $\langle(\cdot)\rangle$ defined in Eqs. (4.33), we can rewrite the linear diffusion eigenvalue problem from Eigenvalue Level (EL) Step 3 [Eqs. (4.30)] in operator notation as:

$$\mathcal{M}^{(n+1,p+1)}\phi^{(n+1,p+1)} = \lambda^{(n+1,p+1)}\mathcal{F}^{(n+1,p+1)}\phi^{(n+1,p+1)} \ , \tag{4.34a}$$

$$P = \langle \mathcal{K}^{(n+1,p+1)}\phi^{(n+1,p+1)} \rangle \ . \tag{4.34b}$$

In Step 3 of the Nonlinear Power Iteration (N-PI) method, we **do not** fully solve Eqs. (4.34). Instead, we perform a single power iteration. By lagging the right-hand-side of Eq. (4.34a), we transform the eigenvalue diffusion problem in Eqs. (4.34) into the fixed-source diffusion problem:

$$\mathcal{M}^{(n+1,p+1)}\tilde{\phi}^{(n+1,p+1)} = \lambda^{(n+1,p)}\mathcal{F}^{(n+1,p+1)}\phi^{(n+1,p)} \ . \tag{4.35}$$

After solving Eq. (4.35), we update the eigenvalue estimate with:

$$\lambda^{(n+1,p+1)} = \lambda^{(n+1,p)} \frac{\langle \mathcal{F}^{(n+1,p+1)}\phi^{(n+1,p)} \rangle}{\langle \mathcal{F}^{(n+1,p+1)}\tilde{\phi}^{(n+1,p+1)} \rangle} \ . \tag{4.36}$$

We then normalize $\tilde{\phi}^{(n+1,p+1)}$ to satisfy the power-normalization criteria in Eq. (4.34b):

$$\phi^{(n+1,p+1)} = \tilde{\phi}^{(n+1,p+1)} \frac{P}{\langle \mathcal{K}^{(n+1,p+1)}\tilde{\phi}^{(n+1,p+1)} \rangle} \ . \tag{4.37}$$

We then check inner iteration convergence and max iteration criteria. If they have been met, we set $P^{(n+1)} = p+1$, and continue to N-PI Finalization (below). Otherwise, we increase the inner iteration index p by one and return to Step 1.

N-PI : Finalization

This step is identical to EL : Finalization. (See Eqs. (4.31).)

4.5.2. Summary Remarks

We now provide some summary remarks. In X-CMFD's nonlinear diffusion problem [Eqs. (4.24)-(4.26)], the standard diffusion coefficient D and transport correction vector $\hat{\boldsymbol{D}}$ are evaluated explicitly. These terms are lagged, effectively, separating the transport physics from Theoretical Method 1 (TM-1)'s Eqs. (4.19)-(4.22).

Like TM-1, the temperatures and nuclear data in X-CMFD's nonlinear diffusion problem [Eqs. (4.24)-(4.26)] are fully nonlinearly implicit. They are evaluated in response to the diffusion scalar flux solution $\phi^{(n+1)}$ while considering the feedback of the unapproximated thermal hydraulics solver [Eqs. (4.24)] and nuclear data update utilities [Eq. (4.25)]. As a consequence, these high-order components appear in the X-CMFD inner iteration procedure.

For example, in both the Eigenvalue Level (EL) and Nonlinear Power Iteration (N-PI) inner iteration methods outlined above, a full thermal hydraulics solve (i.e. inversion of $\mathcal{L}$) is required in Step 1 [Eqs. (4.28)]. Likewise, both methods include a full nuclear data update in Step 2 [Eq. (4.29)]. Neither of these inner iteration schemes converges instantly. In general, their spectral radii are greater than zero, requiring multiple inner iterations to reach a suitable tolerance to declare Eqs. (4.24)-(4.26) "solved" for a given outer iteration. Thus, to evaluate an outer iteration of X-CMFD, multiple inversions of $\mathcal{L}$ and nuclear data updates are required. In contrast, in the M-CMFD and R-CMFD methods, each of these high-order components is evaluated a single time each outer iteration (in Eqs. (4.10) and (4.11), respectively). The question becomes: "Are the extra thermal hydraulics solves and nuclear data evaluations required each X-CMFD outer iteration worth it?" In the next few paragraphs, we address this question.

Based on problems that we have tested (some of which are presented in Chs. 6 and 7) as well as numerical experiments and theoretical analyses performed elsewhere [91, 93, 110, 111], X-CMFD has exceptional stability and a consistent outer iteration spectral radius similar to the spectral radius of CMFD applied to transport problems without feedback, $\rho \approx 0.22$. X-CMFD's lack of a relaxation factor also provides a benefit, by removing the problem-dependent relaxation parameter that a code user must supply to R-CMFD.

Unfortunately, these benefits come at a potentially significant cost imposed by X-CMFD Step 4's nonlinear diffusion problem [Eqs. (4.24)-(4.26)]. X-CMFD *does not* specify a method to solve this nonlinear problem. Many possibilities exist; for example, the Eigenvalue Level (EL) and Nonlinear Power Iteration (N-PI) approaches described above. However, regardless of the method chosen to solve Eqs. (4.24)-(4.26), it will almost certainly require multiple thermal hydraulics solves and nuclear data evaluations to address Eqs. (4.24) and (4.25), respectively. As noted earlier, the thermal hydraulics operator $\mathcal{L}$ may be expensive to invert, and the same applies to nuclear data updates. We wish to avoid the evaluation of these components when pos-

sible. So, even though X-CMFD is stable, requires no relaxation factor, and achieves a near optimal spectral radius (effectively minimizing the number of outer iterations required to solve a problem) the method may require an unfeasible number of thermal hydraulics solves and nuclear data updates during the solution of the nonlinear diffusion problem specified by Eqs. (4.24)-(4.26), which, again, must be solved every outer iteration. This indicates the possibility of an unacceptably high computational cost for each X-CMFD outer iteration. NILO-CMFD, proposed next, seeks to remove this drawback of X-CMFD, while also presenting a clear and consistent improvement over M-CMFD and R-CMFD.

To summarize: in this section, we derived the X-CMFD method from Theoretical Method 1 (TM-1) by decoupling the transport physics [Eqs. (4.19)] from the transport-corrected diffusion problem [Eqs. (4.22)]. This was done by lagging, or rather, explicitly evaluating the transport correction vector $\hat{\boldsymbol{D}}$. To derive the M-CMFD method from TM-1, we additionally explicitly evaluate the nuclear data (and by extension, temperatures) in the transport-corrected diffusion problem in M-CMFD. Effectively, this decouples the transport, nuclear data, and thermal hydraulics physics from the transport-corrected diffusion problem. (This simplification comes at a great cost to stability and efficiency, as mentioned previously.) NILO-CMFD, proposed next, acts as a middle-ground between the extremes of M-CMFD and X-CMFD.

4.6. Nonlinearly Implicit Low-Order CMFD (NILO-CMFD)

Nonlinearly Implicit Low-Order CMFD (NILO-CMFD) is designed to maintain the stability and fast convergence rate of X-CMFD, while reducing the cost associated with the implicit evaluation of temperatures and nuclear data in the nonlinear transport-corrected diffusion solve. In NILO-CMFD, just as in X-CMFD, the diffusion problem includes nonlinearly implicit temperatures and nuclear data. Therefore, both methods require the solution of a nonlinear diffusion problem through some inner iteration procedure. X-CMFD's low-order problem uses fully implicit temperatures and nuclear data, meaning that they are evaluated using their respective high-order physics components. This requries the inclusion of both the full thermal hydraulics solver [Eqs. (4.24)] and full nuclear data processing utilities [Eq. (4.25)]

in the method's nonlinear diffusion problem [Eqs. (4.24)-(4.26)]. In NILO-CMFD, we remove these potentially expensive high-order components from the nonlinear diffusion problem and replace them by approximate descriptions. We say that in NILO-CMFD's nonlinear diffusion problem, temperatures and nuclear data are *approximately nonlinearly implict*, to contrast them with X-CMFD's *fully nonlinearly implicit* temperatures and nuclear data. The hope is that the approximately nonlinearly implicit temperatures and nuclear data used by NILO-CMFD are "good enough" to replicate the outer iteration stability and convergence rate of X-CMFD. In later chapters (Chs. 6 & 7), we show that this aim can be realized.

NILO-CMFD Steps 0 – 3 are identical to M-CMFD and X-CMFD Steps 0 – 3. To start a NILO-CMFD outer iteration, we first perform a transport sweep, then solve a full thermal hydraulics problem (by inverting $\mathcal{L}$), and then evaluate new nuclear data using the full nuclear data processing library and determine the transport-correction vector $\hat{\boldsymbol{D}}$. The nonlinear diffusion problem involved in NILO-CMFD Step 4 is only slightly different from nonlinear diffusion problem in X-CMFD Step 4. Next, we outline the NILO-CMFD nonlinear diffusion problem for two variants of the method. We start by introducing an "incomplete" form of NILO-CMFD, which we refer to as "NILO-A". We then continue on to derive the full NILO-CMFD method.

4.6.1. NILO-A

In this subsection, we derive the nonlinear diffusion problem for an incomplete variant of NILO-CMFD that we call NILO-A. In NILO-A, we assume that full nuclear data evaluations are inexpensive and full thermal hydraulics solves (i.e. inversions of $\mathcal{L}$) are expensive. To derive NILO-A's nonlinear diffusion problem, we directly modify X-CMFD's nonlinear diffusion problem to remove its dependence on the full thermal hydraulics equations ($\mathcal{L}$) [Eqs. (4.24)]. We also use the fact that Steps 0 – 3 of M-CMFD (i.e. R-CMFD without relaxation, $\alpha = 1$) and NILO-A / NILO-CMFD are identical.

Let us consider the heat source distribution estimates defined in Eqs. (4.10a) and (4.24a). We repeat these equations here for convenience:

$$h^{(n+\frac{1}{2})}(\boldsymbol{r}) = \int_0^\infty \kappa\Sigma_f^{(n)}(\boldsymbol{r}, E')\phi^{(n+\frac{1}{2})}(\boldsymbol{r}, E')\,\mathrm{d}E' \,, \tag{4.38a}$$

$$h^{(n+1)}(\boldsymbol{r}) = \int_0^\infty \kappa\Sigma_f^{(n+1)}(\boldsymbol{r}, E')\phi^{(n+1)}(\boldsymbol{r}, E')\,\mathrm{d}E' \,. \tag{4.38b}$$

We also consider the $\mathcal{L}$-evaluated temperatures associated with these heat source distributions in Eqs. (4.10b) and (4.24b), again, repeated here for convenience:

$$\mathcal{L}T^{(n+\frac{1}{2})}(\boldsymbol{r}) = h^{(n+\frac{1}{2})}(\boldsymbol{r}) \ , \tag{4.39a}$$

$$\mathcal{L}T^{(n+1)}(\boldsymbol{r}) = h^{(n+1)}(\boldsymbol{r}) \ . \tag{4.39b}$$

We then write:

$$T^{(n+1)}(\boldsymbol{r}) = T^{(n+\frac{1}{2})}(\boldsymbol{r}) + \left(T^{(n+1)}(\boldsymbol{r}) - T^{(n+\frac{1}{2})}(\boldsymbol{r})\right) \ . \tag{4.40}$$

Using Eqs. (4.39), we can rewrite Eq. (4.40) as:

$$T^{(n+1)}(\boldsymbol{r}) = T^{(n+\frac{1}{2})}(\boldsymbol{r}) + \left(\mathcal{L}^{-1}h^{(n+1)}(\boldsymbol{r}) - \mathcal{L}^{-1}h^{(n+\frac{1}{2})}(\boldsymbol{r})\right) \ . \tag{4.41}$$

Eq. (4.41) is an exact expression for the temperature solution obtained from X-CMFD Step 4's nonlinear diffusion problem [Eqs. (4.24)-(4.26)]. That is, we can replace Eq. (4.24b) with Eq. (4.41) without introducing any approximations. However, Eq. (4.41) involves the inversion of the full (possibly nonlinear) thermal hydraulics operator $\mathcal{L}$. With NILO-A, we separate this potentially expensive component from the nonlinear diffusion problem. In order to do this, NILO-A replaces the full thermal hydraulics operator $\mathcal{L}$ in Eq. (4.41) by an *approximate* thermal hydraulics operator L:

$$T^{(n+1)}(\boldsymbol{r}) = T^{(n+\frac{1}{2})}(\boldsymbol{r}) + \left(L^{-1}h^{(n+1)}(\boldsymbol{r}) - L^{-1}h^{(n+\frac{1}{2})}(\boldsymbol{r})\right) \ . \tag{4.42}$$

By replacing Eq. (4.24b) with Eq. (4.42) in Eqs. (4.24)-(4.26), we arrive at NILO-A's nonlinear diffusion problem. We say that this problem includes approximately nonlinearly implicit temperatures, since we have replaced the full thermal hydraulics physics encoded by $\mathcal{L}$ with the approximate thermal hydraulics operator L. The inner iteration solution of NILO-A's nonlinear diffusion problem does not involve the inversion of the full thermal hydraulics operator $\mathcal{L}$! Instead, it requires that we invert L, which we have yet to define! As with X-CMFD, we may have to invert L multiple times during NILO-A's inner iterations. With simple modifications, the Eigenvalue Level (EL) and Nonlinear Power Iteration (N-PI) inner iteration procedures can be written to solve NILO-A's nonlinear diffusion problem. We show this in greater detail later.

(We note that, in general, the advanced temperature estimate in X-CMFD [$T^{(n+1)}$

appearing in Eqs. (4.24b) and (4.41), equivalently] and NILO-A [$T^{(n+1)}$ appearing in Eq. (4.42)] are different from one another. The NILO-A advanced temperature $T^{(n+1)}$ is an **approximation** of the X-CMFD advanced temperature $T^{(n+1)}$.)

Next, we discuss the approximate thermal hydraulics operator L. We start by noting an extreme case, when the approximate (L) and full ($\mathcal{L}$) thermal hydraulics operators are equal, $L = \mathcal{L}$. In this case, Eqs. (4.41) and (4.42) are identical. No approximations are made to X-CMFD in obtaining Eq. (4.41). Thus, if X-CMFD's Eq. (4.24b) is replaced by Eq. (4.41), the outer iteration convergence rate is unaffected. Therefore, replacing X-CMFD's Eq. (4.24b) by Eq. (4.42) to create NILO-A and using the fact that when $L = \mathcal{L}$, Eqs. (4.41) and (4.42) are equivalent, we see that X-CMFD and NILO-A have equivalent outer iteration convergence properties. Since, through experience and other supporting research, we have seen that X-CMFD is robust with a near-optimal convergence rate, then NILO-A will perform well in this situation, with respect to the number of outer iterations required for convergence.

Of course, by using $L = \mathcal{L}$ in NILO-A, we have not accomplished what we set out to do. Namely, we have not separated the full thermal hydraulics solver from the nonlinear diffusion problem. By using $L = \mathcal{L}$ in NILO-A, we achieve an optimal outer iteration convergence rate, but at the cost of overburdening the thermal hydraulics solver during inner iterations, just like with X-CMFD. Instead, we use an approximate L that is inexpensive to invert, but does a satisfactory job of representing the physics contained in the full thermal hydraulics operator $\mathcal{L}$. The question is, what constitutes a satisfactory representation of the physics encoded in $\mathcal{L}$? Our belief is that as long as L reproduces the low-spatial-frequency temperature (and density) variations predicted by $\mathcal{L}$, NILO-A will achieve the same outer iteration stability and convergence behavior of X-CMFD with a significantly lowered inner-iteration cost. In the next several chapters, we attempt to demonstrate this with several preliminary implementations of NILO-A.

Up to this point, our discussion of the full $\mathcal{L}$ and approximate L thermal hydraulics operators has been rather abstract. In order to ground ourselves, we give a brief preview of the form that these operators will take in Ch. 7's NILO-A numerical results on practical, 3-D, reactor neutron transport – thermal hydraulics multiphysics problems. In Ch. 7, we discuss the implementation and performance of NILO-A in the Michigan Parallel Characteristics-based Transport (MPACT) code [55]. In MPACT, multiple thermal hydraulics descriptions for Pressurized Water Reactor (PWR)'s are available [32]. These descriptions differ in physical accuracy and, in turn, complexity

and computational cost.

For higher-accuracy thermal hydraulics descriptions, MPACT can be loosely coupled to the COBRA-TF (CTF) subchannel code [87]. CTF solves a set of mass, momentum, and energy conservation equations for a two-fluid, three-field description of Light Water Reactor (LWR) thermal hydraulics [87]. In our MPACT-based numerical experiments of NILO-A, we use CTF as our full thermal hydraulics operator $\mathcal{L}$.

MPACT also includes several simplified Thermal Hydraulics (simTH) descriptions that rely on a one-dimensional fluid energy (enthalpy) balance and steam tables for their reactor coolant descriptions [55]. This requires the user to specify a fixed mass flow rate in each coolant "channel". Three versions of simTH are included in MPACT, each differing in its definition of what constitutes a coolant "channel" (i.e. a radial zone of fluid for which a single axial enthalpy balance applies). From finest to coarsest radial discretization, these alternate descriptions are the channel, node (quarter-assembly), and assembly versions of simTH [32]. In our MPACT numerical results, we use the assembly-wise version of simTH as the NILO-A approximate thermal hydraulics operator L. While CTF and simTH simulate the same physical process, the transfer of heat through a reactor core, their computational costs differ significantly. In some practical cases, simTH can be several orders of magnitude cheaper to evaluate than CTF. We stress that our use of CTF as $\mathcal{L}$ and simTH as L in Ch. 7 is merely a choice based on the current capabilities and structure of MPACT. In NILO-A (and NILO-CMFD), the operators $\mathcal{L}$ and L may take on a variety of forms depending on the practical problem of interest and available thermal hydraulics solvers.

4.6.2. NILO-CMFD

NILO-A is an incomplete version of NILO-CMFD. To derive the complete NILO-CMFD method, we make a few additional modifications to X-CMFD's nonlinear diffusion problem. NILO-A is meant to deal with situations in which (i) the full thermal hydraulics operator ($\mathcal{L}$) is expensive to invert, and (ii) nuclear data are expensive to evaluate. As with NILO-A, the full NILO-CMFD method reduces stress on the thermal hydraulics solver ($\mathcal{L}$) by replacing X-CMFD's Eq. (4.24b) by Eq. (4.42). In addition, NILO-CMFD removes Eq. (4.25) from X-CMFD's nonlinear diffusion problem. We start by rewriting Eq. (4.25), whose evaluation we are trying

to avoid:

$$\Sigma_u^{(n+1)}(\boldsymbol{r}, E) = \Sigma_u(\boldsymbol{r}, T^{(n+1)}(\boldsymbol{r}), E) \ . \tag{4.43}$$

By omitting dependent variables (space and energy), we rewrite Eq. (4.43) more simply as:

$$\Sigma_u^{(n+1)} = \Sigma_u(T^{(n+1)}) \ . \tag{4.44}$$

Without approximation, we rewrite Eq. (4.44) as:

$$\Sigma_u(T^{(n+1)}) = \Sigma_u(T^{(n+\frac{1}{2})} + (T^{(n+1)} - T^{(n+\frac{1}{2})})) \ , \tag{4.45}$$

which can be approximated by the first-order Taylor expansion:

$$\Sigma_u(T^{(n+1)}) \approx \Sigma_u(T^{(n+\frac{1}{2})}) + \frac{\partial \Sigma_u(T^{(n+\frac{1}{2})})}{\partial T}\left(T^{(n+1)} - T^{(n+\frac{1}{2})}\right) \ . \tag{4.46}$$

In NILO-CMFD, we use the above expansion to represent $\Sigma_u^{(n+1)}$. That is, in NILO-CMFD as well as using Eq. (4.42) to replace Eq. (4.24b), we additionally replace X-CMFD's Eq. (4.25) by:

$$\Sigma_u^{(n+1)} = \Sigma_u(T^{(n+\frac{1}{2})}) + \frac{\partial \Sigma_u(T^{(n+\frac{1}{2})})}{\partial T}\left(T^{(n+1)} - T^{(n+\frac{1}{2})}\right) \ . \tag{4.47}$$

It is still necessary to specify the derivative term in Eq. (4.47). Chances are, we will not have access to nuclear data derivatives considering our treatment of the nuclear data library as a black box. Instead, we must specify a procedure to estimate this derivative. One simple approach is to calculate a first order finite difference estimate by slightly perturbing the temperatures $T^{(n+\frac{1}{2})}$ by a small parameter $\epsilon \ll 1$ and re-evaluating nuclear data using the nuclear data library. Doing this requires one additional full evaluation of nuclear data each outer iteration:

$$\frac{\partial \Sigma_u(T^{(n+\frac{1}{2})})}{\partial T} \approx \dot{\Sigma}_u^{(n+\frac{1}{2})} \equiv \frac{\Sigma_u((1+\epsilon)T^{(n+\frac{1}{2})}) - \Sigma_u^{(n+\frac{1}{2})}}{\epsilon T^{(n+\frac{1}{2})}} \ . \tag{4.48}$$

We note that the above approximation need only be made if nuclear data derivatives with respect to temperature are not readily available. Using this approximation, we rewrite Eq. (4.47) as:

$$\Sigma_u^{(n+1)} = \Sigma_u^{(n+\frac{1}{2})} + \dot{\Sigma}_u^{(n+\frac{1}{2})}\left(T^{(n+1)} - T^{(n+\frac{1}{2})}\right) \ . \tag{4.49}$$

Eq. (4.49) is a simple linear function of $T^{(n+1)}$ and, in general, is *much* less expensive to evaluate than the unapproximated nuclear data update in Eq. (4.44).

4.7. NILO-CMFD Outlines

In this section, we consolidate equations scattered throughout previous sections into a complete outline of NILO-A and NILO-CMFD. This section serves as a reference to give readers a full picture of NILO-A and NILO-CMFD without needing to jump around through multiple equation references. This section includes the Nonlinear Power Iteration (N-PI) procedure as it applies to NILO-A and NILO-CMFD's nonlinear diffusion problems.

4.7.1. NILO-A

Below, we outline the NILO-A simplification of NILO-CMFD, including a description of an example inner iteration procedure, namely, a modified version of the Nonlinear Power Iteration (N-PI) scheme introduced previously for X-CMFD:

NILO-A Step 0 : Initialization

Before iterating, we require initial outer iteration estimates of the scalar flux $\phi^{(0)}$, eigenvalue $\lambda^{(0)}$, and nuclear data $\Sigma_u^{(0)}$. After these are determined, we set the outer iteration index n to zero and proceed to NILO-A Step 1.

NILO-A Step 1 : Neutron Transport Sweep

We begin the $(n{+}1)^{\text{st}}$ outer iteration by performing a transport sweep using the most up-to-date estimates of the scalar flux $\phi^{(n)}$, eigenvalue $\lambda^{(n)}$, and nuclear data $\Sigma_u^{(n)}$. While sweeping, we update scalar flux $\phi^{(n+\frac{1}{2})}$ and current vector $\boldsymbol{J}^{(n+\frac{1}{2})}$ estimates:

$$\begin{aligned}\hat{\boldsymbol{\Omega}}\cdot\boldsymbol{\nabla}\psi^{(n+\frac{1}{2})}(\boldsymbol{r},\hat{\boldsymbol{\Omega}},E)+\Sigma_t^{(n)}(\boldsymbol{r},E)\psi^{(n+\frac{1}{2})}(\boldsymbol{r},\hat{\boldsymbol{\Omega}},E)\\ =\frac{1}{4\pi}\int_0^\infty \Sigma_s^{(n)}(\boldsymbol{r},E'\to E)\phi^{(n)}(\boldsymbol{r},E')\,\mathrm{d}E'\\ +\lambda^{(n)}\frac{\chi^{(n)}(\boldsymbol{r},E)}{4\pi}\int_0^\infty \nu\Sigma_f^{(n)}(\boldsymbol{r},E')\phi^{(n)}(\boldsymbol{r},E')\,\mathrm{d}E'\,,\end{aligned}\tag{4.50a}$$

$$\phi^{(n+\frac{1}{2})}(\boldsymbol{r}, E) = \int_{4\pi} \psi^{(n+\frac{1}{2})}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E)\, \mathrm{d}\Omega' \ , \tag{4.50b}$$

$$\boldsymbol{J}^{(n+\frac{1}{2})}(\boldsymbol{r}, E) = \int_{4\pi} \hat{\boldsymbol{\Omega}}' \psi^{(n+\frac{1}{2})}(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}', E)\, \mathrm{d}\Omega' \ . \tag{4.50c}$$

NILO-A Step 2 : Thermal Hydraulics Solve

Using the latest nuclear data $\Sigma_u^{(n)}$ and scalar flux $\phi^{(n+\frac{1}{2})}$ estimates, we update the fission heat source distribution $h^{(n+\frac{1}{2})}$ and solve the thermal hydraulics problem for the updated temperature $T^{(n+\frac{1}{2})}$:

$$h^{(n+\frac{1}{2})}(\boldsymbol{r}) = \int_0^\infty \kappa\Sigma_f^{(n)}(\boldsymbol{r}, E')\phi^{(n+\frac{1}{2})}(\boldsymbol{r}, E')\, \mathrm{d}E' \ , \tag{4.51a}$$

$$\mathcal{L}T^{(n+\frac{1}{2})}(\boldsymbol{r}) = h^{(n+\frac{1}{2})}(\boldsymbol{r}) \ . \tag{4.51b}$$

NILO-A Step 3 : Nuclear Data Evaluation

We supply the temperature estimate $T^{(n+\frac{1}{2})}$ to the nuclear data library to update nuclear data $\Sigma_u^{(n+\frac{1}{2})}$:

$$\Sigma_u^{(n+\frac{1}{2})}(\boldsymbol{r}, E) = \Sigma_u(\boldsymbol{r}, T^{(n+\frac{1}{2})}(\boldsymbol{r}), E) \ . \tag{4.52}$$

The practical implementation of Eq. (4.52) can be expensive for the following reasons. When a new temperature T (and density ρ) are provided, the cross section package determines new fine-group cross sections for each isotope for each spatial cell, using a process that may involve the evaluation of integrals and table lookups. Then, the fine-group cross sections are collapsed onto the coarse-group structure used in the simulation. In depletion problems, the number of isotopes can increase significantly. The combination of all these factors can lead to significant computational expense.

After evaluating Eq. (4.52), we then update the estimates of the standard diffusion coefficient $D^{(n+\frac{1}{2})}$ and the transport correction vector $\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}$:

$$D^{(n+\frac{1}{2})}(\boldsymbol{r}, E) = \frac{1}{3\Sigma_t^{(n+\frac{1}{2})}(\boldsymbol{r}, E)} \ , \tag{4.53a}$$

$$\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}(\boldsymbol{r}, E) = \frac{\boldsymbol{J}^{(n+\frac{1}{2})}(\boldsymbol{r}, E) + D^{(n+\frac{1}{2})}(\boldsymbol{r}, E)\boldsymbol{\nabla}\phi^{(n+\frac{1}{2})}(\boldsymbol{r}, E)}{\phi^{(n+\frac{1}{2})}(\boldsymbol{r}, E)} \ . \tag{4.53b}$$

NILO-A Step 4 : Nonlinear Diffusion Solve

Using inner iterations, we solve the following nonlinear diffusion problem for the end-of-outer-iteration scalar flux $\phi^{(n+1)}$, eigenvalue $\lambda^{(n+1)}$, and nuclear data $\Sigma_u^{(n+1)}$:

$$h^{(n+1)}(\boldsymbol{r}) = \int \kappa\Sigma_f^{(n+1)}(\boldsymbol{r}, E')\phi^{(n+1)}(\boldsymbol{r}, E')\,\mathrm{d}E' \;, \tag{4.54a}$$

$$T^{(n+1)}(\boldsymbol{r}) = T^{(n+\frac{1}{2})}(\boldsymbol{r}) + \left(L^{-1}h^{(n+1)}(\boldsymbol{r}) - L^{-1}h^{(n+\frac{1}{2})}(\boldsymbol{r})\right) \;, \tag{4.54b}$$

$$\Sigma_u^{(n+1)}(\boldsymbol{r}, E) = \Sigma_u(\boldsymbol{r}, T^{(n+1)}(\boldsymbol{r}), E) \;, \tag{4.55}$$

$$\begin{aligned} -\boldsymbol{\nabla}\cdot D^{(n+\frac{1}{2})}\boldsymbol{\nabla}\phi^{(n+1)}(\boldsymbol{r}, E) + \boldsymbol{\nabla}\cdot\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}(\boldsymbol{r}, E)\phi^{(n+1)}(\boldsymbol{r}, E) \\ + \Sigma_t^{(n+1)}(\boldsymbol{r}, E)\phi^{(n+1)}(\boldsymbol{r}, E) = \int \Sigma_s^{(n+1)}(\boldsymbol{r}, E' \to E)\phi^{(n+1)}(\boldsymbol{r}, E')\,\mathrm{d}E' \\ + \lambda^{(n+1)}\frac{\chi^{(n+1)}(\boldsymbol{r}, E)}{4\pi}\int \nu\Sigma_f^{(n+1)}(\boldsymbol{r}, E')\phi^{(n+1)}(\boldsymbol{r}, E')\,\mathrm{d}E' \;, \end{aligned} \tag{4.56a}$$

$$P = \iint \kappa\Sigma_f^{(n+1)}(\boldsymbol{r}', E')\phi^{(n+1)}(\boldsymbol{r}', E')\,\mathrm{d}^3r'\,\mathrm{d}E' \;. \tag{4.56b}$$

There are many choices as to the inner iteration scheme used to solve Eqs. (4.54)-(4.56). Below, we show the Nonlinear Power Iteration (N-PI) scheme as applied to NILO-A's nonlinear diffusion problem. We show this specific example because it was used (albeit with a Wielandt shift) in our 1-D test code and MPACT to generate numerical results presented in upcoming chapters. We do not claim that this is the most efficient inner iteration procedure, only that it works adequately for the problems we have tested.

Nonlinear Power Iteration (N-PI) Step 0 : Initialization

Before starting the inner iterations, we choose initial estimates of the scalar flux $\phi^{(n+1,0)}$, eigenvalue $\lambda^{(n+1,0)}$, and nuclear data $\Sigma_u^{(n+1,0)}$, using their most up-to-date outer iteration values:

$$\phi^{(n+1,0)} = \phi^{(n+\frac{1}{2})} \;, \tag{4.57a}$$

$$\lambda^{(n+1,0)} = \lambda^{(n)} \;, \tag{4.57b}$$

$$\Sigma_u^{(n+1,0)} = \Sigma_u^{(n+\frac{1}{2})} \;. \tag{4.57c}$$

We then set the inner iteration index p for the $(n+1)^{\text{st}}$ outer iteration to zero and proceed to N-PI Step 1.

N-PI Step 1 : Approximate Thermal Hydraulics Solve

We calculate the inner iteration temperature estimate $T^{(n+1,p)}$ using an approximate thermal hydraulics update:

$$h^{(n+1,p+1)}(\boldsymbol{r}) = \int \kappa\Sigma_f^{(n+1,p)}(\boldsymbol{r}, E')\phi^{(n+1,p)}(\boldsymbol{r}, E')\,\mathrm{d}E' \;, \tag{4.58a}$$

$$T^{(n+1,p+1)}(\boldsymbol{r}) = T^{(n+\frac{1}{2})}(\boldsymbol{r}) + \left(L^{-1}h^{(n+1,p+1)}(\boldsymbol{r}) - L^{-1}h^{(n+\frac{1}{2})}(\boldsymbol{r})\right) \;, \tag{4.58b}$$

where $T^{(n+\frac{1}{2})} = \mathcal{L}^{-1}h^{(n+\frac{1}{2})}$ is fixed throughout the inner iterations. (In every inner iteration, the approximate thermal hydraulics operator L must be inverted to evaluate $L^{-1}h^{(n+1,p+1)}$ in Eq. (4.58b). In each **outer** iteration, a **single** additional inversion of L is also required to evaluate $L^{-1}h^{(n+\frac{1}{2})}$, after which this term can be saved and reused.)

N-PI Step 2 : Nuclear Data Evaluation

We pass $T^{(n+1,p+1)}$ to the nuclear data processing utilities to evaluate new nuclear data estimates $\Sigma_u^{(n+1,p+1)}$:

$$\Sigma_u^{(n+1,p+1)}(\boldsymbol{r}, E) = \Sigma_u(\boldsymbol{r}, T^{(n+1,p+1)}(\boldsymbol{r}), E) \;. \tag{4.59}$$

N-PI Step 3 : Single Power Iteration

For the current inner iteration, and purely for the sake of notational convenience, we define the operators $\mathcal{M}^{(n+1,p+1)}$, $\mathcal{F}^{(n+1,p+1)}$, and $\mathcal{K}^{(n+1,p+1)}$ as well as the integration operator $\langle(\cdot)\rangle$:

$$\begin{aligned}\mathcal{M}^{(n+1,p+1)} &\\ &\equiv -\boldsymbol{\nabla}\cdot D^{(n+\frac{1}{2})}(\boldsymbol{r}, E)\boldsymbol{\nabla}(\cdot) + \boldsymbol{\nabla}\cdot\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}(\boldsymbol{r}, E)(\cdot) \\ &\quad + \Sigma_t^{(n+1,p+1)}(\boldsymbol{r}, E)(\cdot) - \int \Sigma_s^{(n+1,p+1)}(\boldsymbol{r}, E'\to E)(\cdot)\,\mathrm{d}E' \;,\end{aligned} \tag{4.60a}$$

$$\mathcal{F}^{(n+1,p+1)} \equiv \chi^{(n+1,p+1)}(\boldsymbol{r}, E)\int \nu\Sigma_f^{(n+1,p+1)}(\boldsymbol{r}, E')(\cdot)\,\mathrm{d}E' \;, \tag{4.60b}$$

$$\mathcal{K}^{(n+1,p+1)} \equiv \kappa\Sigma_f^{(n+1,p+1)}(\boldsymbol{r}, E)(\cdot) \;, \tag{4.60c}$$

$$\langle(\cdot)\rangle \equiv \int_0^\infty \int_V (\cdot)\,\mathrm{d}^3r'\,\mathrm{d}E' \,. \tag{4.60d}$$

We solve the following fixed-source neutron diffusion problem, written using the above operator notation, for the un-normalized scalar flux update $\tilde{\phi}^{(n+1,p+1)}$ (the over-tilde signals that this quantity is not normalized):

$$\mathcal{M}^{(n+1,p+1)}\tilde{\phi}^{(n+1,p+1)} = \lambda^{(n+1,p)}\mathcal{F}^{(n+1,p+1)}\phi^{(n+1,p)} \,. \tag{4.61}$$

We then update the eigenvalue estimate $\lambda^{(n+1,p+1)}$ with:

$$\lambda^{(n+1,p+1)} = \lambda^{(n+1,p)}\frac{\langle\mathcal{F}^{(n+1,p+1)}\phi^{(n+1,p)}\rangle}{\langle\mathcal{F}^{(n+1,p+1)}\tilde{\phi}^{(n+1,p+1)}\rangle} \,. \tag{4.62}$$

We then normalize $\tilde{\phi}^{(n+1,p+1)}$ to the power normalization criteria to get $\phi^{(n+1,p+1)}$:

$$\phi^{(n+1,p+1)} = \tilde{\phi}^{(n+1,p+1)}\frac{P}{\langle\mathcal{K}^{(n+1,p+1)}\tilde{\phi}^{(n+1,p+1)}\rangle} \,. \tag{4.63}$$

We then check inner iteration convergence and max iteration criteria. If they are met, we set the concluding inner iteration index to $P^{(n+1)} = p+1$ and continue to the N-PI Finalization (below). Otherwise, we increase the inner iteration index p by one and return to N-PI Step 1.

N-PI : Finalization

Before exiting the inner iteration completely, we set estimates of the scalar flux $\phi^{(n+1)}$, eigenvalue $\lambda^{(n+1)}$, and nuclear data $\Sigma_u^{(n+1)}$ to be used in the next outer iteration. In effect, we are setting the "solution" values of Eqs. (4.54)-(4.56) for the current outer iteration:

$$\phi^{(n+1)} = \phi^{(n+1,P^{(n+1)})} \,, \tag{4.64a}$$

$$\lambda^{(n+1)} = \lambda^{(n+1,P^{(n+1)})} \,, \tag{4.64b}$$

$$\Sigma_u^{(n+1)} = \Sigma_u^{(n+1,P^{(n+1)})} \,. \tag{4.64c}$$

After the inner iterations are complete, we check outer iteration convergence and max iteration criteria. If they are not met, we increase the outer iteration index n by one and return to NILO-A Step 1 to begin the next outer iteration. If they have been met, we are done!

4.7.2. NILO-CMFD

In the full NILO-CMFD method, the only piece of the above NILO-A outline that is modified is Eq. (4.55), which is replaced by Eq. (4.49). This leads to a modification of the inner iteration's N-PI Step 2. In NILO-CMFD, the call to the nuclear data library [Eq. (4.59)] is replaced by an approximation:

N-PI Step 2 : Approximate Nuclear Data Evaluation

Using a first-order Taylor expansion, we replace Eq. (4.59) by:

$$\Sigma_u^{(n+1,p+1)} = \Sigma_u^{(n+\frac{1}{2})} + \dot{\Sigma}_u^{(n+\frac{1}{2})}\left(T^{(n+1,p+1)} - T^{(n+\frac{1}{2})}\right) . \tag{4.65}$$

An approximate cross section derivative $\dot{\Sigma}_u^{(n+\frac{1}{2})}$ could be obtained by finite difference (e.g., similar to Eq. (4.48)).

4.8. Discussion

In this chapter, we introduced several multiphysics iterative methods to solve the neutron transport – thermal hydraulics multiphysics problem described by Eqs. (4.3)-(4.4). Of these approaches, Theoretical Method 0 (TM-0) and Theoretical Method 1 (TM-1) are purely theoretical. Although TM-0 and TM-1 converge in a single outer iteration, neither method is implementable in a computer code. Instead, these methods take on a pedagogical role. By observing the structure of TM-1, we notice that the nuclear data $\Sigma_u^{(n+1)}$, diffusion coefficient $D^{(n+1)}$, and transport correction vector $\hat{\boldsymbol{D}}^{(n+1)}$ included in its CMFD-based, transport-corrected, diffusion equations [Eqs. (4.22)] depend on the scalar flux solution $\phi^{(n+1)}$ of that same diffusion equation. We describe these terms as being nonlinearly implicit. TM-1's nonlinearly implicit terms require the inclusion of transport [Eqs. (4.19)], thermal hydraulics [Eqs. (4.20)], and nuclear data [Eq. (4.21)] physics in the transport-corrected neutron diffusion equations [Eqs. (4.22)]. By using nonlinearly implicit terms in the transport-corrected diffusion equations [Eqs. (4.22)], we arrive at a nonlinear diffusion problem [Eqs. (4.19)-(4.22)]. The resulting nonlinear diffusion problem in TM-1 is too difficult to solve for the TM-1 method to be of practical use.

The X-CMFD method can be understood as a modification to TM-1. By including a transport sweep [Eqs. (4.9)] in the outer iteration procedure, we are able to separate the transport physics from the TM-1 nonlinear diffusion problem. In X-CMFD, rather

than being nonlinearly implicit, the diffusion problem's diffusion coefficient $D^{(n+\frac{1}{2})}$ and transport correction vector $\hat{\boldsymbol{D}}^{(n+\frac{1}{2})}$ are evaluated explicitly, they are lagged using transport-sweep-determined parameter estimates.

Although X-CMFD is implementable, it is not necessarily efficient. To solve X-CMFD's nonlinear diffusion problem [Eqs. (4.24)-(4.26)], we must perform inner iterations. In this chapter, we included two inner iteration descriptions based on previous work by Walker [110, 111] and Shen [91, 93]. We refer to these schemes as Eigenvalue Level (EL) and Nonlinear Power Iteration (N-PI) inner iteration. Both procedures require multiple thermal hydraulics and nuclear data updates each outer iteration. This can lead to a massive expense per X-CMFD outer iteration if either of these updates is expensive and a suitably large number of inner iterations are required to "solve" the X-CMFD nonlinear diffusion problem.

To combat the burden placed on the thermal hydraulics solver and nuclear data update utilities by X-CMFD, we introduced the NILO-A and NILO-CMFD methods and the idea of approximately implicit temperatures and nuclear data. In NILO-CMFD, rather than coupling the transport-corrected diffusion equations directly to the full thermal hydraulics and nuclear data update physics, approximate versions of these physics are used. We propose two variants: the incomplete NILO-A method and the full NILO-CMFD method. Both versions use an approximate thermal hydraulics operator L to represent the full thermal hydraulics operator $\mathcal{L}$ in the NILO nonlinear diffusion solve. The full NILO-CMFD method additionally approximates nuclear data updates in the nonlinear diffusion solve. In the next few chapters, we examine the performance of NILO-A and NILO-CMFD as compared to M-CMFD, R-CMFD, and X-CMFD. First, we look at a 1-D model problem in Chs. 5 and 6. Then, in Ch. 7, we discuss the implementation of NILO-A in the 3-D MPACT reactor physics research code. In MPACT, we use the subchannel thermal hydraulics code COBRA-TF (CTF) for $\mathcal{L}$ and a simplified thermal hydraulics (simTH) operator based on axial energy balances and 1-D cylindrical geometry heat conduction solves for the approximate operator L.

4.9. NILO-CMFD Modifications to Include Density

Throughout most of this thesis, we ignore density by assuming a direct dependence of density on temperature is available: $\rho(T(\boldsymbol{r}))$. This assumption is valid in Pressurized Water Reactors (PWR)'s. In this section, we briefly discuss how the NILO-CMFD procedure must be modified if such a dependence is unavailable (e.g., in a Boiling Water Reactor (BWR) under conditions of sub-cooled boiling). To do this, we first modify our description of a thermal hydraulics solve. Instead of writing:

$$\mathcal{L}T(\boldsymbol{r}) = h(\boldsymbol{r}) \ , \tag{4.66}$$

which was used throughout this chapter, we instead write:

$$\mathcal{L}_* \begin{pmatrix} T(\boldsymbol{r}) \\ \rho(\boldsymbol{r}) \end{pmatrix} = h(\boldsymbol{r}) \ , \tag{4.67}$$

where $\mathcal{L}_*$ represents a high-order thermal hydraulics solver that takes in a fission heat source distribution $h(\boldsymbol{r})$ and returns updated temperature $T(\boldsymbol{r})$ and density $\rho(\boldsymbol{r})$ fields.

Using our new thermal hydraulics operator $\mathcal{L}_*$, we rewrite the X-CMFD nonlinear diffusion problem's temperature update equation [Eq. (4.24b)] to include density:

$$\mathcal{L}_* \begin{pmatrix} T^{(n+1)}(\boldsymbol{r}) \\ \rho^{(n+1)}(\boldsymbol{r}) \end{pmatrix} = h^{(n+1)}(\boldsymbol{r}) \ . \tag{4.68}$$

Our derivation of the NILO-CMFD approximate temperature and density update equation follows closely to our previous derivation, which considered only temperature. First, we write:

$$\begin{pmatrix} T^{(n+1)}(\boldsymbol{r}) \\ \rho^{(n+1)}(\boldsymbol{r}) \end{pmatrix} = \begin{pmatrix} T^{(n+\frac{1}{2})}(\boldsymbol{r}) \\ \rho^{(n+\frac{1}{2})}(\boldsymbol{r}) \end{pmatrix} + \left(\begin{pmatrix} T^{(n+1)}(\boldsymbol{r}) \\ \rho^{(n+1)}(\boldsymbol{r}) \end{pmatrix} - \begin{pmatrix} T^{(n+\frac{1}{2})}(\boldsymbol{r}) \\ \rho^{(n+\frac{1}{2})}(\boldsymbol{r}) \end{pmatrix} \right) \ . \tag{4.69}$$

Equivalently, we write:

$$\begin{pmatrix} T^{(n+1)}(\boldsymbol{r}) \\ \rho^{(n+1)}(\boldsymbol{r}) \end{pmatrix} = \begin{pmatrix} T^{(n+\frac{1}{2})}(\boldsymbol{r}) \\ \rho^{(n+\frac{1}{2})}(\boldsymbol{r}) \end{pmatrix} + \left(\mathcal{L}_*^{-1} h^{(n+1)} - \mathcal{L}_*^{-1} h^{(n+\frac{1}{2})} \right) \ . \tag{4.70}$$

The advanced temperatures $T^{(n+1)}$ and densities $\rho^{(n+1)}$ evaluated by Eq. (4.70) are equivalent to those in X-CMFD's Eq. (4.68). We then introduce an approximate thermal hydraulics operator L_* that when provided a heat source distribution returns temperatures and densities that approximate those obtained from $\mathcal{L}_*$:

$$L_* \begin{pmatrix} T(\boldsymbol{r}) \\ \rho(\boldsymbol{r}) \end{pmatrix} = h(\boldsymbol{r}) \; . \tag{4.71}$$

Inserting the approximate thermal hydraulics operator L_* for the full thermal hydraulics operator $\mathcal{L}_*$ in Eq. (4.70), we arrive at the NILO-CMFD approximate thermal hydraulics update equation, including both temperature and density:

$$\begin{pmatrix} T^{(n+1)}(\boldsymbol{r}) \\ \rho^{(n+1)}(\boldsymbol{r}) \end{pmatrix} = \begin{pmatrix} T^{(n+\frac{1}{2})}(\boldsymbol{r}) \\ \rho^{(n+\frac{1}{2})}(\boldsymbol{r}) \end{pmatrix} + \left(L_*^{-1} h^{(n+1)} - L_*^{-1} h^{(n+\frac{1}{2})} \right) \; . \tag{4.72}$$

In both NILO-A and NILO-CMFD, we replace X-CMFD's Eq. (4.68) with Eq. (4.72). Making only this replacement, we arrive at NILO-A. To obtain the full NILO-CMFD method, we must also approximate nuclear data updates in the nonlinear diffusion problem.

We write the X-CMFD advanced nuclear data $\Sigma_u^{(n+1)}$ as a function of both the advanced temperature $T^{(n+1)}$ and density $\rho^{(n+1)}$ (for simplicity, we ignore space and energy dependence):

$$\Sigma_u^{(n+1)} = \Sigma_u(T^{(n+1)}, \rho^{(n+1)}) \; . \tag{4.73}$$

Equivalently, we write:

$$\Sigma_u^{(n+1)} = \Sigma_u(T^{(n+\frac{1}{2})} + (T^{(n+1)} - T^{(n+\frac{1}{2})}), \rho^{(n+\frac{1}{2})} + (\rho^{(n+1)} - \rho^{(n+\frac{1}{2})}) \; . \tag{4.74}$$

We approximate the above equation using a first-order multivariate Taylor expansion:

$$\begin{aligned} \Sigma_u^{(n+1)} \approx \; & \Sigma_u^{(n+\frac{1}{2})} \\ & + \frac{\partial \Sigma_u(T^{(n+\frac{1}{2})}, \rho^{(n+\frac{1}{2})})}{\partial T} \left(T^{(n+1)} - T^{(n+\frac{1}{2})} \right) \\ & + \frac{\partial \Sigma_u(T^{(n+\frac{1}{2})}, \rho^{(n+\frac{1}{2})})}{\partial \rho} \left(\rho^{(n+1)} - \rho^{(n+\frac{1}{2})} \right) \; . \end{aligned} \tag{4.75}$$

The partial derivative terms in Eq. (4.75) may not be readily available. We can

approximate these terms using finite difference:

$$\frac{\partial \Sigma_u(T^{(n+\frac{1}{2})}, \rho^{(n+\frac{1}{2})})}{\partial T} \approx \dot{\Sigma}_{u,T}^{(n+\frac{1}{2})} = \frac{\Sigma_u((1+\epsilon)T^{(n+\frac{1}{2})}, \rho^{(n+\frac{1}{2})}) - \Sigma_u^{(n+\frac{1}{2})}}{\epsilon T^{(n+\frac{1}{2})}} , \tag{4.76a}$$

$$\frac{\partial \Sigma_u(T^{(n+\frac{1}{2})}, \rho^{(n+\frac{1}{2})})}{\partial \rho} \approx \dot{\Sigma}_{u,\rho}^{(n+\frac{1}{2})} = \frac{\Sigma_u(T^{(n+\frac{1}{2})}, (1+\epsilon)\rho^{(n+\frac{1}{2})}) - \Sigma_u^{(n+\frac{1}{2})}}{\epsilon \rho^{(n+\frac{1}{2})}} . \tag{4.76b}$$

Inserting these partial derivative approximations into Eq. (4.75), we obtain the NILO-CMFD approximation to the X-CMFD nuclear data update Eq. (4.73):

$$\Sigma_u^{(n+1)} = \Sigma_u^{(n+\frac{1}{2})} + \dot{\Sigma}_{u,T}^{(n+\frac{1}{2})}\left(T^{(n+1)} - T^{(n+\frac{1}{2})}\right) + \dot{\Sigma}_{u,\rho}^{(n+\frac{1}{2})}\left(\rho^{(n+1)} - \rho^{(n+\frac{1}{2})}\right) . \tag{4.77}$$

Eq. (4.77) expresses $\Sigma_u^{(n+1)}$ as a simple linear function of $T^{(n+1)}$ and $\rho^{(n+1)}$. This equation is *much* less expensive to evaluate than the exact nuclear data update [Eq. (4.73)].

With Eqs. (4.72) and (4.77) approximating X-CMFD's Eqs. (4.68) and (4.73), respectively, we have the NILO-CMFD approximate update equations for temperature and nuclear data with the explicit inclusion of both material temperature and density.

Chapter 5.

Continuous 1-D Model

In Ch. 4, the discussion of the Multiphysics CMFD (M-CMFD), Relaxed CMFD (R-CMFD), X-CMFD, NILO-A, and NILO-CMFD multiphysics iterative methods is quite general. This discussion outlines these methods for the continuous, 3-D neutron transport – thermal hydraulics multiphysics problem introduced in Ch. 2. The use of continuous equations in Ch. 4 simplifies the description and derivation of the methods discussed. By working with continuous equations, we are free from discretization-related complexity, for example, from a deluge of discretization indices. The use of continuous equations in Ch. 4 also helps to generalize its description of iterative methods. In theory, the method outlines in Ch. 4 can be used as a rough blueprint for implementation in any loose-coupling-enabled, CMFD-based reactor physics code designed to solve the general class of neutron transport – thermal hydraulics problems defined in Ch. 2. However, we stress that for this purpose, the Ch. 4 blueprints are incomplete; they cannot be directly implemented in a computer code without taking into account the discretization schemes used by that code!

Reactor physics codes for solving multiphysics problems work with discretized, algebraic forms of the continuous differential and intergro-differential equations of primary focus to Chs. 2 & 4. In order to implement M-CMFD, R-CMFD, X-CMFD, NILO-A, and NILO-CMFD in a given code, one must transform the continuous descriptions of these methods provided in Ch. 4 into the discretized versions employed by said code.

The goal of this and the next chapter (Chs. 5 & 6) is to provide an example of how this continuous to discrete translation is performed for a 1-D neutron transport – thermal hydraulics model problem. The work performed in Chs. 5 & 6 also serves as an initial proof-of-concept of NILO-A and NILO-CMFD. This 1-D work motivated our

decision to implement NILO-A in the large-scale, 3-D reactor physics code MPACT [55]. Implementation of the full NILO-CMFD method in MPACT is ongoing. In Ch. 7, we detail the performance of NILO-A in MPACT as compared to more standard multiphysics iterative methods (e.g., R-CMFD). Owing to MPACT's complexity, a discretized description of NILO-A in MPACT would take considerable space and effort. Instead, we choose to work with simpler, 1-D equations in Chs. 5 & 6 to bridge the gap between the continuous method outlines in Ch. 4 and the numerical results of our discretized implementation in MPACT in Ch. 7.

In this short chapter (Ch. 5), we derive the continuous form of the 1-D model directly from the 3-D equations of Ch. 2. We believe that the 1-D model contains enough essential features of the 3-D physical system described in Ch. 2 that (i) extensions of the methods to higher fidelity simulations using more complex discretization schemes is straightforward, and (ii) the performance seen for the 1-D model will be indicative of the performance observed in realistic 3-D problems (i.e. MPACT simulations). In Ch. 6, we discretize the 1-D model, apply the multiphysics iterative methods described Ch. 4 to the discrete 1-D model, and compare their relative performance.

5.1. Problem Geometry

For our 1-D model, the system of interest is a Pressurized Water Reactor (PWR) fuel pin unit cell, denoted by V. A 3-D view of V is depicted in Fig. (5.1a), with a radial slice R in Fig. (5.1b). (These graphics are not drawn to scale.)

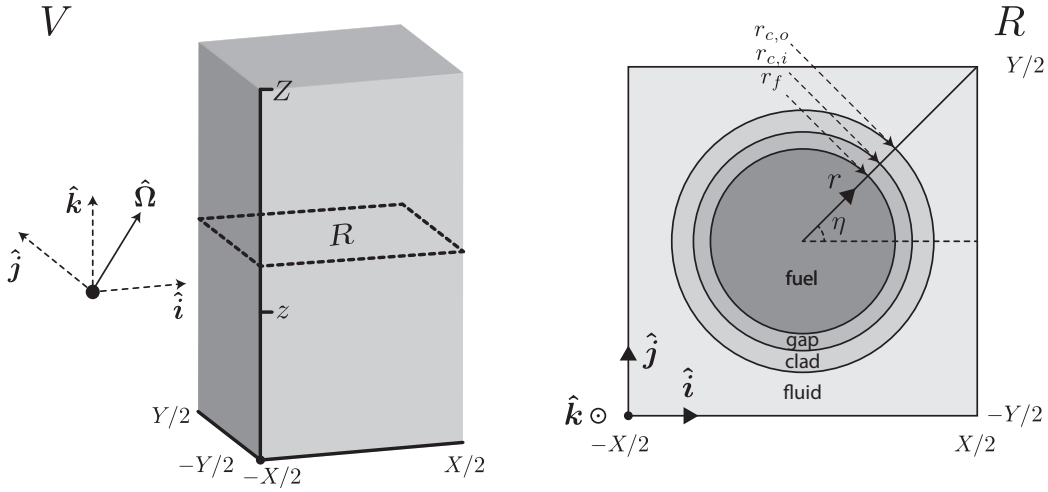


Figure 5.1: 1-D Model Geometry.

Mutually perpendicular directional unit vectors $\hat{\mathbf{i}}$, $\hat{\mathbf{j}}$, and $\hat{\mathbf{k}}$ are used. A Cartesian coordinate system is defined with coordinate scalars x, y, and z specifying location: $\mathbf{r} = x\,\hat{\mathbf{i}} + y\,\hat{\mathbf{j}} + z\,\hat{\mathbf{k}}$. The origin of this coordinate system is located at the center of the bottom face of V in Fig. (5.1a). V extends over $-X/2 < x < X/2$, $-Y/2 < y < Y/2$, $0 < z < Z$. (In this work, we have assumed $X = Y$, pin-cells are square.) A cylindrical coordinate system with radial, polar, and axial coordinates r, η, and z is used interchangeably to specify spatial location, with: $\mathbf{r} = r(\cos\eta)\,\hat{\mathbf{i}} + r(\sin\eta)\,\hat{\mathbf{j}} + z\,\hat{\mathbf{k}}$. The cylindrical coordinate system is only used in the fuel pin encompassing $0 \leq r < r_{c,o}$, $0 \leq \eta < 2\pi$, $0 < z < Z$. The origin of the Cartesian and cylindrical coordinate systems are identical.

A radial plane R in Fig. (5.1b) shows that V consists of non-overlapping fuel (V_{fuel}), gap (V_{gap}), clad (V_{clad}), and fluid (V_{fluid}) subregions. Specifically:

$$V = V_{\text{fuel}} \cup V_{\text{gap}} \cup V_{\text{clad}} \cup V_{\text{fluid}} \,, \tag{5.1a}$$

$$V_{\text{fuel}} \equiv \{(r,\eta,z) \mid 0 \leq r < r_f,\, 0 \leq \eta < 2\pi,\, 0 < z < Z\} \,, \tag{5.1b}$$

$$V_{\text{gap}} \equiv \{(r,\eta,z) \mid r_f < r < r_{c,i},\, 0 \leq \eta < 2\pi,\, 0 < z < Z\} \,, \tag{5.1c}$$

$$V_{\text{clad}} \equiv \{(r,\eta,z) \mid r_{c,i} < r < r_{c,o},\, 0 \leq \eta < 2\pi,\, 0 < z < Z\} \,. \tag{5.1d}$$

The outer boundary of V is ∂V. The top and bottom boundaries, whose surface normal vectors are $\pm\hat{\mathbf{k}}$, are ∂V_{top} and ∂V_{bot}. The four side boundaries, whose surface normal vectors are $\pm\hat{\mathbf{i}}$ or $\pm\hat{\mathbf{j}}$, are referred to collectively as ∂V_{side}. Direction vectors used for neutron direction of flight are specified with polar angle $\theta = \cos^{-1}\mu$ taken with respect to $\hat{\mathbf{k}}$: $\hat{\boldsymbol{\Omega}} = \sqrt{1-\mu^2}(\cos\omega\,\hat{\mathbf{i}} + \sin\omega\,\hat{\mathbf{j}}) + \mu\hat{\mathbf{k}}$. The azimuthal angle ω of the neutron direction-of-flight circles around the z-axis.

5.2. Neutron Transport

The 3-D, λ-eigenvalue neutron transport problem describing the distribution of neutrons in the fuel-pin V is given below in Eqs. (5.2). Except for a few notational differences, these equations are identical to those given in Ch. 2. In Eqs. (5.2), we use Ψ to refer to the six-dimensional, space-, angle-, and energy-dependent angular flux. In our 1-D model, we use ψ to refer to a radial-, azimuthal-, and energy- integrated collapse of Ψ. (ψ is defined in terms of Ψ later.) Also, in a departure from Ch. 2, we momentarily include both the background material temperature $T(\boldsymbol{r})$ and the

density $\rho(\boldsymbol{r})$ dependence of nuclear data in Eqs. (5.2). (Later, we remove the explicit dependence on material density $\rho(\boldsymbol{r})$.)

The 3-D neutron transport equation for the six-dimensional angular flux $\Psi(\mathbf{r}, \hat{\boldsymbol{\Omega}}, E)$ is given by Eq. (5.2a). A vacuum boundary condition is specified on $\partial V_{\text{vac}} \equiv \partial V_{\text{bot}} \cup \partial V_{\text{top}}$ in Eq. (5.2b). A reflective boundary condition is specified on $\partial V_{\text{refl}} \equiv \partial V_{\text{side}}$ in Eq. (5.2c). A system thermal power P normalization condition on the eigenfunction Ψ is imposed by Eq. (5.2d), with κ defined as the local thermal energy deposition (from the slowing-down of fission daughter isotopes) per fission reaction:

$$\begin{aligned}
&\hat{\boldsymbol{\Omega}} \cdot \boldsymbol{\nabla} \Psi(\mathbf{r}, \hat{\boldsymbol{\Omega}}, E) + \Sigma_t(\mathbf{r}, T(\boldsymbol{r}), \rho(\boldsymbol{r}), E) \Psi(\mathbf{r}, \hat{\boldsymbol{\Omega}}, E) \\
&\quad = \int_0^\infty \int_{4\pi} \Sigma_s(\mathbf{r}, T(\boldsymbol{r}), \rho(\boldsymbol{r}), \hat{\boldsymbol{\Omega}}' \cdot \hat{\boldsymbol{\Omega}}, E' \to E) \Psi(\mathbf{r}, \hat{\boldsymbol{\Omega}}', E') \, \mathrm{d}\Omega' \, \mathrm{d}E' \\
&\qquad + \lambda \Bigg(\frac{\chi(\mathbf{r}, T(\boldsymbol{r}), \rho(\boldsymbol{r}), E)}{4\pi} \\
&\qquad\qquad \times \int_0^\infty \int_{4\pi} \nu \Sigma_f(\mathbf{r}, T(\boldsymbol{r}), \rho(\boldsymbol{r}), E') \Psi(\mathbf{r}, \hat{\boldsymbol{\Omega}}', E') \, \mathrm{d}\Omega' \, \mathrm{d}E' \Bigg) \,, \\
&\qquad\qquad\qquad \mathbf{r} \in V \,, \quad \hat{\boldsymbol{\Omega}} \in 4\pi \,, \quad 0 < E < \infty \,,
\end{aligned} \tag{5.2a}$$

$$\begin{aligned}
&\Psi(\mathbf{r}, \hat{\boldsymbol{\Omega}}, E) = 0 \,, \\
&\qquad \mathbf{r} \in \partial V_{\text{vac}} \,, \quad \hat{\boldsymbol{\Omega}} \cdot \hat{\mathbf{n}} < 0 \,, \quad 0 < E < \infty \,,
\end{aligned} \tag{5.2b}$$

$$\begin{aligned}
&\Psi(\mathbf{r}, \hat{\boldsymbol{\Omega}}, E) = \Psi(\mathbf{r}, \hat{\boldsymbol{\Omega}}_r(\hat{\boldsymbol{\Omega}}), E) \,, \\
&\qquad \mathbf{r} \in \partial V_{\text{refl}} \,, \quad \hat{\boldsymbol{\Omega}} \cdot \hat{\mathbf{n}} < 0 \,, \quad 0 < E < \infty \,,
\end{aligned} \tag{5.2c}$$

$$P = \int_0^\infty \int_{4\pi} \int_V \kappa \Sigma_f(\mathbf{r}', T(\boldsymbol{r}'), \rho(\boldsymbol{r}'), E') \Psi(\mathbf{r}', \hat{\boldsymbol{\Omega}}', E') \, \mathrm{d}r' \, \mathrm{d}\Omega' \, \mathrm{d}E' \,. \tag{5.2d}$$

In Eq. (5.2c), the reflective outgoing direction vector $\hat{\boldsymbol{\Omega}}_r$ is defined in terms of the incoming direction vector $\hat{\boldsymbol{\Omega}}$ by:

$$\hat{\boldsymbol{\Omega}}_r(\hat{\boldsymbol{\Omega}}) = \hat{\boldsymbol{\Omega}} - 2(\hat{\boldsymbol{\Omega}} \cdot \hat{\mathbf{n}}) \hat{\mathbf{n}} \,, \quad \hat{\boldsymbol{\Omega}} \cdot \hat{\mathbf{n}} < 0 \,. \tag{5.3}$$

To construct the simplified transport equations for the 1-D model, we apply approximations to Eqs. (5.2) and its solution. First, to simplify the scattering kernel,

we assume isotropic scattering:

$$\begin{aligned}\Sigma_s(\mathbf{r}, T(\boldsymbol{r}), \rho(\boldsymbol{r}), \hat{\boldsymbol{\Omega}}' \cdot \hat{\boldsymbol{\Omega}}, E' \to E) \\ = \frac{1}{4\pi}\Sigma_s(\mathbf{r}, T(\boldsymbol{r}), \rho(\boldsymbol{r}), E' \to E) \ .\end{aligned} \tag{5.4}$$

Next, we define the multiple integral operator $\mathcal{I}$ by:

$$\mathcal{I}(\cdot) \equiv \int_0^\infty \int_0^{2\pi} \int_{-Y/2}^{Y/2} \int_{-X/2}^{X/2} (\cdot)\, \mathrm{d}x\, \mathrm{d}y\, \mathrm{d}\omega\, \mathrm{d}E \ , \tag{5.5}$$

and we assume that the neutron angular flux is *separable*:

$$\Psi(\mathbf{r}, \hat{\boldsymbol{\Omega}}, E) = f(x, y, z, \omega, E)\psi(z, \mu) \ , \tag{5.6}$$

where $f(x, y, z, \omega, E)$ is a known "shape" function, normalized by $\mathcal{I}f = 1$. Since $\mathcal{I}\Psi = \psi$, then $\psi(z, \mu)$ represents the radially, azimuthally, and energy-integrated angular flux.

Operating on Eqs. (5.2a) and (5.2b) by $\mathcal{I}$ and introducing Eq. (5.6) into the resulting equations, we easily obtain:

$$\begin{gathered}\mu\frac{\partial}{\partial z}\psi(z, \mu) + \Sigma_t(z)\psi(z, \mu) = \frac{1}{2}\Big(\Sigma_s(z) + \lambda\nu\Sigma_f(z)\Big)\phi(z) \ , \\ 0 < z < Z \ , \quad -1 \le \mu \le 1 \ ,\end{gathered} \tag{5.7a}$$

$$\phi(z) \equiv \int_{-1}^{1} \psi(z, \mu')\, \mathrm{d}\mu' = \text{scalar flux} \ , \tag{5.7b}$$

$$J(z) \equiv \int_{-1}^{1} \mu'\psi(z, \mu')\, \mathrm{d}\mu' = \text{current (a scalar!)} \ , \tag{5.7c}$$

$$\psi(0, \mu) = 0 \ , \quad 0 < \mu \le 1 \ , \tag{5.7d}$$

$$\psi(Z, \mu) = 0 \ , \quad -1 \le \mu < 0 \ , \tag{5.7e}$$

$$P = \int_{-1}^{1} \int_0^Z \kappa\Sigma_f(z')\psi(z', \mu')\, \mathrm{d}z'\, \mathrm{d}\mu' \ , \tag{5.7f}$$

where the 1-D nuclear data (cross sections, etc.) in Eqs. (5.7a) and (5.7f) are defined

by:

$$\Sigma_u(z) \equiv \frac{\mathcal{I}\big[\Sigma_u(\mathbf{r}, T(\boldsymbol{r}), \rho(\boldsymbol{r}), E) f(\mathbf{r}, \omega, E)\big]}{\mathcal{I}\big[f(\mathbf{r}, \omega, E)\big]} \ . \tag{5.8}$$

We use Σ_u to represent the collection of nuclear data Σ_t, Σ_s, χ, $\nu\Sigma_f$, and $\kappa\Sigma_f$. We note that the 1-D nuclear data $\Sigma_u(z)$ in Eq. (5.8) depends on the five-dimensional neutron shape function $f(\boldsymbol{r}, \omega, E)$ along with the 3-D background material temperature $T(\boldsymbol{r})$ and density $\rho(\boldsymbol{r})$ solutions. Later, we must choose a suitable, approximate representation of $\Sigma_u(z)$ so as not to intertwine our 1-D model with 3-D solution quantities.

Note: the neutron current $J(z)$ (a scalar) defined in Eq. (5.7c) does not play a direct role in the continuous 1-D model derived in this chapter (Ch. 5). However, it does play a role in the discrete 1-D model discussed in Ch. 6; thus, we include it here.

Eqs. (5.7) constitute our model's 1-D, monoenergetic, isotropic scattering neutron transport equations. If the assumptions stated in Eqs. (5.4) and (5.6) are valid and Eq. (5.8) is accurately evaluated, then the model Eqs. (5.7) are exact. (Generally, Eqs. (5.7) are approximate.)

5.3. Thermal Hydraulics

The thermal hydraulics equations for the 1-D model are based on a Conjugate Heat Transfer (CHT) problem of a similar form to the simplified Thermal Hydraulic (simTH) model available in MPACT [32, 55]. (As a preview, we mention that in Ch. 7's implementation of NILO-A in MPACT, we use MPACT's simTH operator as NILO-A's low-order thermal hydraulics operator L to approximate the high-order thermal hydraulics operator $\mathcal{L}$, which is described by the COBRA-TF (CTF) subchannel code.) Our 1-D model's thermal hydraulics problem is divided into separate solid and fluid subproblems defined over the non-overlapping subregions $V_{\text{solid}} \equiv V_{\text{fuel}} \cup V_{\text{gap}} \cup V_{\text{clad}}$ and V_{fluid}. Internal boundary conditions stitch the solid and fluid domains together.

We define the *axial heat generation rate* $q'(z) =$ the radially-integrated rate of local fission heat release in the pin-cell (deposited by heavy fission daughter products) per unit axial thickness:

$$q'(z) = h(z) = \kappa\Sigma_f(z)\phi(z) \ , \tag{5.9}$$

where we have defined the fission heat source distribution h in the 1-D model to be equivalent to the axial heat generation rate q'. $q'(z) = h(z)$ is the principal link between the neutronics and thermal hydraulics equations in the 1-D model. (We note that *reaction rates* of the form $\Sigma_u(z)\phi(z)$ in the 1-D model are obtained via radial integration, following the application of Eq. (5.5) in deriving Eqs. (5.7).)

Next, we consider the solid subproblem defined over V_{solid}. In describing this problem, we use the cylindrical coordinate system defined previously, but the polar coordinate η is ignored due to (assumed) symmetry. The main link between the neutronic solution and the solid subproblem's temperature distribution is the *volumetric heat generation rate* $q'''(r,z)$ = the rate of local fission heat release per unit volume. $q'''(r,z)$ is nonzero only within the fuel V_{fuel}, as the macroscopic fission cross section is zero elsewhere, and is defined in terms of the axial heat generation rate $q'(z)$ by Eq. (5.10). In the process of forming Eqs. (5.7) through the operation of Eq. (5.5), the radial and azimuthal angle dependence of solution quantities are lost, and a profile must be assumed in constructing $q'''(r,z)$. A flat profile is assumed:

$$q'''(r,z) = \frac{1}{\pi r_f^2} q'(z) \ , \quad 0 \leq r < r_f \ . \tag{5.10}$$

Through gap conductance and convective heat transfer relationships, the solid subproblem involves the *radial heat flux* $q''(r,z)$ = the directed rate at which heat flows per unit area with respect to a surface with outward normal oriented in the positive radial direction. Using Fick's law, $q''(r,z)$ is defined by:

$$q''(r,z) = -k(r,z)\frac{\partial}{\partial r} T(r,z) \ , \quad 0 \leq r \leq r_f \ , \quad r_{c,i} \leq r \leq r_{c,o} \ , \tag{5.11}$$

where $k(r,z)$ is the *thermal conductivity*. The definitions in Eqs. (5.9), (5.10), and (5.11) are valid for $0 < z < Z$.

Within the solid, the fuel $T_f(r,z)$ and cladding $T_c(r,z)$ temperature profiles are governed by the heat conduction equations stated below in Eqs. (5.12a) and (5.12b), which hold for each axial value $0 < z < Z$. Constant fuel k_f and clad k_c thermal conductivities are assumed. Axial diffusion is ignored, removing coupling between radial planes (i.e. planes with normal $\pm\hat{\boldsymbol{k}}$). Due to symmetry at $r = 0$, a zero flux Neumann boundary condition is imposed by Eq. (5.12c) at the fuel centerline. Eq. (5.12d) connects the fuel and cladding by a *gap conductance* relation with constant gap conductance h_g. A convective boundary condition is used at the outer surface

of the cladding in Eq. (5.12e) with a constant *convective heat transfer coefficient* h_b. This boundary condition relates the outer cladding surface temperature $T_c(r_{c,o}, z)$ to the *bulk average fluid temperature* $T_b(z)$, which describes an average fluid temperature at each axial height z. (The bulk average fluid temperature $T_b(z)$ is governed by an advection-diffusion equation, described shortly.) The *gap radius* r_g in Eq. (5.12d) is defined by Eq. (5.13):

$$-\frac{1}{r}\frac{\partial}{\partial r}k_f r\frac{\partial}{\partial r}T_f(r,z) = q'''(r,z)\ , \quad 0 \le r < r_f\ , \tag{5.12a}$$

$$-\frac{1}{r}\frac{\partial}{\partial r}k_c r\frac{\partial}{\partial r}T_c(r,z) = 0\ , \quad r_{c,i} < r < r_{c,o}\ , \tag{5.12b}$$

$$\frac{\partial}{\partial r}T_f(0,z) = 0\ , \tag{5.12c}$$

$$q''(r_g,z) = h_g(T_f(r_f,z) - T_c(r_{c,i},z))\ , \tag{5.12d}$$

$$q''(r_{c,o},z) = h_b(T_c(r_{c,o},z) - T_b(z))\ , \tag{5.12e}$$

$$r_g \equiv \frac{1}{2}(r_f + r_{c,i})\ . \tag{5.13}$$

Since axial diffusion is neglected in Eqs. (5.12), the radial heat flux $q''(r,z)$ can also be defined in terms of axial heat generation rate $q'(z)$ through:

$$q''(r,z) = \frac{1}{2\pi r}q'(z)\ , \quad r = r_f\ , \quad r_{c,i} \le r \le r_{c,o}\ . \tag{5.14}$$

This definition follows from the realization that since axial diffusion is ignored in the solid subregion, all heat generated in a radial slice must be removed through outward radial diffusion. This understanding is important in developing the equations describing the behavior of the fluid.

We approximate the radial heat flux $q''(r_g)$ in the gap conductance relationship [Eq. (5.12d)] using:

$$q''(r_g,z) = \frac{1}{2\pi r_g}q'(z)\ , \tag{5.15}$$

with r_g given by Eq. (5.13).

The solution of Eqs. (5.12) gives the detailed axial and radial temperature profile in V_{solid} (besides the gap). Shortly, we define a simple approximation to the weighted nuclear data $\Sigma_u(z)$ defined in Eq. (5.8). In this approximation, the influence of the solid subproblem on the 1-D nuclear data $\Sigma_u(z)$ involves only a single, representative solid temperature value at each axial coordinate z – the *fuel Doppler temperature*

$T_d(z)$, defined by:

$$T_d(z) \equiv (1-\zeta)T_f(0,z) + \zeta T_f(r_f,z) \ , \tag{5.16a}$$

$$\zeta \equiv 0.7 \ . \tag{5.16b}$$

The Doppler temperature is a weighted average of the solid temperatures at the centerline and outer surface of the fuel region. Through simple manipulations of Eqs. (5.12), and using the flat volumetric heat generation rate profile assumed in Eq. (5.10), $T_d(z)$ satisfies the following algebraic relationship:

$$T_d(z) = T_b(z) + \beta q'(z) \ , \tag{5.17}$$

where β is defined in terms of the *thermal resistance* parameters by:

$$\beta \equiv R_{\text{th},b\to f} + (1-\zeta)R_{\text{th},f} \ , \tag{5.18a}$$

$$R_{\text{th},b\to f} \equiv R_{\text{th},b} + R_{\text{th},c} + R_{\text{th},g} \ , \tag{5.18b}$$

$$R_{\text{th},b} \equiv \frac{1}{2\pi r_{c,o} h_b} \ , \tag{5.18c}$$

$$R_{\text{th},c} \equiv \frac{1}{2\pi k_c} \ln\left(\frac{r_{c,o}}{r_{c,i}}\right) \ , \tag{5.18d}$$

$$R_{\text{th},g} \equiv \frac{1}{2\pi r_g h_g} \ , \tag{5.18e}$$

$$R_{\text{th},f} \equiv \frac{1}{4\pi k_f} \ . \tag{5.18f}$$

(The appearance of the bulk average fluid temperature T_b in Eq. (5.17) originates from the inclusion of T_b in the convective heat transfer boundary condition in Eq. (5.12e). We define the equations governing T_b next.)

Next, we consider the fluid subproblem in the 1-D multiphysics model. Our 1-D fluid description is unconventional and requires motivation. Instead of using a standard 1-D advection equation to represent the bulk average coolant temperature T_b, we use an advection-diffusion equation. We give the governing equation for the 1-D model's bulk average fluid temperature T_b below:

$$\mathcal{L}T_b(z) \equiv -\frac{\mathrm{d}}{\mathrm{d}z} D_t \frac{\mathrm{d}}{\mathrm{d}z} T_b(z) + \dot{m} c_p \frac{\mathrm{d}}{\mathrm{d}z} T_b(z) = h(z) \ , \tag{5.19a}$$

with constant *turbulent diffusion coefficient* D_t, *mass flow rate* $\dot{m}$, and *isobaric heat*

capacity c_p. The turbulent diffusion term in Eq. (5.19a) is physically motivated as an approximation to the energy mixing arising from turbulent eddies in buoyancy-driven / natural convection flow [70, 128]. In operational Pressurized Water Reactors (PWR)'s with high coolant flow rate, this term can be ignored. For example, this term is not considered in MPACT's simplified Thermal Hydraulics (simTH) formulation [55].

Nonetheless, we include the turbulent diffusion term in Eq. (5.19a) to introduce extra physics / complexity into the 1-D model's high-order thermal hydraulics operator $\mathcal{L}$. A key facet of both the NILO-A and NILO-CMFD iterative methods is the use of an approximate thermal hydraulics operator L in the methods' respective nonlinear diffusion solves. By including the turbulent diffusion term in $\mathcal{L}$ [Eq. (5.19a)], we provide a natural means of forming an approximate thermal hydraulics operator L by removing the diffusion term and obtaining a simpler pure-advection equation. In this way, our 1-D model is able to test the NILO approach when we neglect certain thermal hydraulics physics in the low-order nonlinear diffusion problem. This will be further tested in Ch. 7's discussion of 3-D numerical results, with $\mathcal{L}$ set as the 3-D COBRA-TF (CTF) subchannel code and L as MPACT's internal, simplified Thermal Hydraulics (simTH) operator. In these 3-D simulations, CTF includes additional physics (e.g., mass and momentum conservation) that are neglected in simTH.

In the 1-D model, we specify the inlet fluid temperature T_{inlet} as the channel entry boundary condition:

$$T(0) = T_{\text{inlet}} \ . \tag{5.19b}$$

The outlet boundary condition is more complicated. By (i) integrating Eq. (5.19a) over the spatial domain, (ii) using Eq. (5.19b), and (iii) noting the fixed total system power P in the 1-D model, we obtain the following exact boundary relationship:

$$-D_t\left(\frac{\mathrm{d}T_b}{\mathrm{d}z}(Z) - \frac{\mathrm{d}T_b}{\mathrm{d}z}(0)\right) + \dot{m}c_p\big(T_b(Z) - T_{\text{inlet}}\big) = P \ , \tag{5.20a}$$

or

$$\dot{m}c_p T_b(Z) - D_t\frac{\mathrm{d}T_b}{\mathrm{d}z}(Z) = P + \dot{m}c_p T_{\text{inlet}} - D_t\frac{\mathrm{d}T_b}{\mathrm{d}z}(0) \ . \tag{5.20b}$$

Eq. (5.20b) involves only boundary values of $T_b(z)$ and $\frac{\mathrm{d}T_b}{\mathrm{d}z}$. Nonetheless, Eq. (5.20b) is not an independent boundary condition! (Eq. (5.20b) is automatically satisfied by any solution of Eqs. (5.19a) and (5.19b).) To uniquely specify $T_b(z)$, it is necessary to specify a boundary condition that is independent of Eq. (5.20b).

To do this, we invoke some approximations that are specific to the problem under consideration. Specifically, assuming that D_t is small, Eq. (5.19a) becomes:

$$\dot{m}c_p\frac{\mathrm{d}T_b}{\mathrm{d}z}(z) \approx h(z) = \kappa\Sigma_f(z)\phi(z) \ , \tag{5.21}$$

and since $\phi(z)$ is (roughly) cosine-shaped, with $\phi(Z) \approx 0$, Eq. (5.21) gives:

$$\frac{\mathrm{d}T_b}{\mathrm{d}z}(Z) \approx 0 \ . \tag{5.22a}$$

Eq. (5.22a) is the "outlet" boundary condition that we impose for our 1-D model. This boundary condition is nonstandard, but if we insert it into the left side of Eq. (5.20b), we obtain:

$$\dot{m}c_pT_b(Z) \approx [P + \dot{m}c_pT_{\text{inlet}}] - D_t\frac{\mathrm{d}T_b}{\mathrm{d}z}(0) \ , \tag{5.22b}$$

which is mathematically equivalent to Eq. (5.22a).

The bracketed term on the right side of Eq. (5.22b) is the value of $\dot{m}c_pT_b(Z)$ that occurs when D_t is set equal to 0 in Eqs. (5.19a) and (5.19b). Since D_t is small, the boundary condition Eq. (5.22b) [which, to repeat, is equivalent to Eq. (5.22a)] produces a value of $T_b(Z)$ that slightly differs from the pure advection ($D_t = 0$) result.

In summary, we use the following advection-diffusion problem to describe the bulk-average fluid temperature T_b in our 1-D model:

$$\mathcal{L}T_b(z) = -\frac{\mathrm{d}}{\mathrm{d}z}D_t\frac{\mathrm{d}}{\mathrm{d}z}T_b(z) + \dot{m}c_p\frac{\mathrm{d}}{\mathrm{d}z}T_b(z) = h(z) \ , \tag{5.23a}$$

$$T_b(0) = T_{\text{inlet}} \ , \tag{5.23b}$$

$$\frac{\mathrm{d}T_b}{\mathrm{d}z}(Z) \approx 0 \ . \tag{5.23c}$$

No thermal hydraulics operator is needed to describe the Doppler temperature T_d, thanks to the simplicity of the algebraic relationship in Eq. (5.17).

In realistic problems, turbulent flow tends to flatten the radial fluid temperature profile. Therefore, describing the fluid by a single bulk average temperature T_b at each axial location is acceptable. The heat source distribution term in Eqs. (5.23) is simply the axial heat generation rate $q'(z)$, due to the neglect of axial diffusion in the solid problem. This dependence follows from the same reasoning used to construct Eq. (5.14). An important consequence of Eqs. (5.23) is that given $q'(z)$

from a neutronics solution, the bulk average fluid temperature profile is immediately determined. That is, while the solid subproblem's Doppler temperature T_d depends on the fluid subproblem through the appearance of the bulk average fluid temperature T_b in Eq. (5.17), the fluid subproblem does not have a similar interdependence.

We have now introduced two temperature profiles describing the thermal hydraulics solution in the 1-D model. The Doppler temperature $T_d(z)$ [Eq. (5.17)] and the bulk average fluid temperature $T_b(z)$ [Eqs. (5.23)] are averages of solid and fluid temperature profiles, respectively. A description of both $T_d(z)$ and $T_b(z)$ at each axial location is required because the radial integration within the definition of transport nuclear data values in Eq. (5.8) extends over *both* the solid and fluid subdomains. Moreover, in realistic problems, $T_d(z)$ and $T_b(z)$ are not tightly coupled. The presence of two temperature values at each z location contrasts to un-spatially-homogenized 3-D problems, in which a single material temperature (and density) value exists at each location in space $\boldsymbol{r}$.

5.4. Nuclear Data (Cross Sections)

Macroscopic cross sections, which describe neutron interaction probabilities, depend on local temperature and background material density, and are calculated using microscopic cross sections and background isotope number densities. Temperature modifies the microscopic cross sections through the Doppler effect, and background material density directly modifies isotope number density. The precise evaluation of Eq. (5.8) would require, for each radial slice of the pin-cell (e.g., Fig. (5.1b)), detailed solution information from the full 3-D, energy dependent, coupled neutron transport – thermal hydraulics multiphysics problem. Since we seek a simpler model problem in a reduced phase space (i.e. 1-D, monoenergetic, and azimuthally integrated transport), this solution information is unavailable. Instead, as an approximation to Eq. (5.8), we use Pressurized Water Reactor (PWR) assembly-averaged nuclear data following the representation and parameter values laid out in [27]. This reference, which proposes several multiphysics benchmarks for neutron nodal diffusion calculations with thermal hydraulics feedback, describes PWR assembly-averaged nuclear data by a first-order, multivariate Taylor expansion with respect to bulk average fluid temperature T_b, bulk average fluid density ρ_b, and the square root of the Doppler temperature T_d. (A similar nuclear data representation is used in the Purdue Advanced Reactor Core Simulator (PARCS) code [40].) The bulk average fluid temperature

T_b and Doppler temperature T_d are described previously and follow Eqs. (5.23) and (5.17), respectively. The nuclear data dependence on the bulk average fluid density ρ_b can be removed by assuming a simple linear dependence between ρ_b and the bulk average fluid temperature T_b:

$$\rho_b(z) = \rho_0 + \rho_1(T_b(z) - T_{b,0}) \ , \tag{5.24}$$

where ρ_0 and ρ_1 are constants obtained from thermodynamic tables at the appropriate reactor core pressure. The nuclear data representation in [27], which provides two-group data, is further simplified by (i) using an infinite-medium group collapse to produce one-group quantities, (ii) assuming isotropic scattering in conversion from transport Σ_{tr} to total Σ_t macroscopic cross sections, and (iii) ignoring dependence on soluble boron by keeping its concentration constant at its centered value as defined in [27]. Following these modifications, the nuclear data formulation for the 1-D model is given by:

$$\begin{aligned} \Sigma_u(z) &\equiv \Sigma_u\big(T_d(z), T_b(z)\big) \\ &= \Sigma_{u,0} \\ &\quad + \Sigma_{u,1,d}\left(\sqrt{T_d(z)} - \sqrt{T_{d,0}}\right) \\ &\quad + \Sigma_{u,1,b}\left(T_b(z) - T_{b,0}\right) . \end{aligned} \tag{5.25}$$

In Eq. (5.25), the terms $\Sigma_{u,0}$, $\Sigma_{u,1,d}$, $\Sigma_{u,1,b}$, $T_{d,0}$, and $T_{b,0}$ are problem parameters whose values are specified in the 1-D model numerical results section of Ch. 6. (As before, we use Σ_u to represent the collection of nuclear data Σ_t, Σ_s, $\nu\Sigma_f$, $\kappa\Sigma_f$. Temperature dependence of the fission spectrum χ is not necessary, since the 1-D model's transport is one-group.)

The bulk average fluid density ρ_b dependence from [27] has been absorbed into the $\Sigma_{u,0}$ and $\Sigma_{u,1,b}$ values, using the linear relationship assumed in Eq. (5.24). This is an example of removing the explicit dependence of nuclear data on background material density in favor of an implied density dependence through temperature. This is possible because we have an explicit representation of the density as a function of temperature given by Eq. (5.24). As mentioned in Ch. 2, this is not always possible, but when it is able to be done, it simplifies the notation.

We emphasize that the nuclear data representation in Eq. (5.25) is dependent on *two* temperature values at each axial location, owing to the radial integration used

to reduce the original 3-D problem to 1-D. As shown in Chs. 2 & 4, in a full 3-D problem, only a single temperature value exists at each spatial location – as well as a possible density value if the density dependence on temperature is not directly available. The price that we pay for eliminating x, y, E, and ω in forming our 1-D model is to introduce a "second" temperature!

5.5. 1-D Model : Reorganization and Summary

Here, we collect together and algebraically simplify the equations that fully define the 1-D neutron transport – thermal hydraulics model.

We first modify the nuclear data representation in Eq. (5.25) by removing the dependence on the Doppler temperature T_d. To accomplish this, we simultaneously introduce Eqs. (5.9) and (5.17) into Eq. (5.25) and redefine the bulk average fluid temperature and its "centered" value as:

$$T(z) \equiv T_b(z) \ , \tag{5.26a}$$

$$T_0 \equiv T_{b,0} \ . \tag{5.26b}$$

We then introduce the *radially integrated fission heat generation rate* h defined by:

$$h(z) \equiv q'(z) = \kappa \Sigma_f(z) \phi(z) \ , \tag{5.27}$$

which is equivalent to the axial heat generation rate $q'(z)$ defined in Eq. (5.9). This allows us to remove the Doppler temperature T_d from the 1-D model and leads to nuclear data $\Sigma_u(z)$ dependent on (i) a single temperature $T(z)$, representing the bulk average fluid temperature (formerly, T_b), and (ii) the radially integrated fission heat generation rate $h(z)$. We obtain:

$$\begin{aligned} \Sigma_u(z) &= \Sigma_u\Big(T(z), h(z)\Big) \\ &= \Sigma_{u,0} \\ &\quad + \Sigma_{u,1,d} \left(\sqrt{T(z) + \beta h(z)} - \sqrt{T_{d,0}} \right) \\ &\quad + \Sigma_{u,1,b} \Big(T(z) - T_0 \Big) \ , \end{aligned} \tag{5.28}$$

where β [Eq. (5.18)], $\Sigma_{u,0}$, $\Sigma_{u,1,d}$, $T_{d,0}$, $\Sigma_{u,1,b}$, and T_0 are problem parameters given in

Ch. 6.

In a standard 3-D problem, nuclear data only depend on the local value of the temperature (and perhaps also density, if a direct dependence of density on temperature is unavailable and the explicit density dependence cannot be removed). In our radially-homogenized 1-D model, the nuclear data depend on the local values of both $T(z)$ and $h(z)$. The direct dependence of the nuclear data on heat generation $h(z)$ is nonstandard, and is due to approximations that occur in deriving the 1-D model.

The remaining equations describing neutron transport and thermal hydraulics for the model are mostly unchanged, with only minor differences introduced by the removal of the Doppler temperature from the problem, and the notational switches from $T_b(z)$ to $T(z)$ for the bulk average fluid temperature and $q'(z)$ to $h(z)$ for the fission heat generation rate.

The neutron angular flux is governed by the 1-D, monoenergetic, isotropic scattering neutron transport eigenvalue problem described in Eqs. (5.7), repeated here for convenience:

$$\mu\frac{\partial}{\partial z}\psi(z,\mu)+\Sigma_t(z)\psi(z,\mu)=\frac{1}{2}\left(\Sigma_s(z)+\lambda\nu\Sigma_f(z)\right)\phi(z)\ , \qquad 0<z<Z\ ,\ \ -1\le\mu\le 1\ , \tag{5.29a}$$

$$\phi(z)\equiv\int_{-1}^{1}\psi(z,\mu')\,\mathrm{d}\mu'=\text{scalar flux}\ , \tag{5.29b}$$

$$J(z)\equiv\int_{-1}^{1}\mu'\psi(z,\mu')\,\mathrm{d}\mu'=\text{current}\ , \tag{5.29c}$$

$$\psi(0,\mu)=0\ ,\ \ 0<\mu\le 1\ , \tag{5.29d}$$

$$\psi(Z,\mu)=0\ ,\ \ -1\le\mu<0\ , \tag{5.29e}$$

$$P=\int_{-1}^{1}\int_{0}^{Z}\kappa\Sigma_f(z')\psi(z',\mu')\,\mathrm{d}z'\,\mathrm{d}\mu'\ . \tag{5.29f}$$

Finally, the bulk average fluid temperature (formerly T_b, now T) is governed by the advection-diffusion problem expressed by Eqs. (5.23). For the 1-D model, the thermal hydraulics operator $\mathcal{L}$ is given by the drift-diffusion operator as written in Eq. (5.30a):

$$\mathcal{L}T(z) \equiv \Big(-\frac{\mathrm{d}}{\mathrm{d}z} D_t \frac{\mathrm{d}}{\mathrm{d}z} + \dot{m} c_p \frac{\mathrm{d}}{\mathrm{d}z} \Big) T(z) = h(z) \ , \tag{5.30a}$$

$$T(0) = T_{inlet} \ , \tag{5.30b}$$

$$\frac{\mathrm{d}T}{\mathrm{d}z}(Z) \approx 0 \ . \tag{5.30c}$$

Eqs. (5.27), (5.28), (5.29), and (5.30) constitute the continuous equations of our 1-D neutron transport – thermal hydraulics model. In the next chapter, we discretize the continuous form of this 1-D model so that the problem may be approximately solved on a computer. We then outline the Multiphysics CMFD (M-CMFD), Relaxed CMFD (R-CMFD), X-CMFD, NILO-A, and NILO-CMFD multiphysics iterative methods as they apply to the discrete 1-D model. Finally, we introduce problem parameters (e.g., $\dot{m}$, c_p, β, etc.) and nuclear data parameters (e.g., $\Sigma_{u,0}$, $\Sigma_{u,1,d}$, $\Sigma_{u,1,b}$, $T_{d,0}$, T_0, etc.), either taken directly or derived from values in [27]. Using these parameter values, we test each of the multiphysics iterative methods and compare their relative performance.

Chapter 6.

Discrete 1-D Model

6.1. Introduction

In the previous chapter (Ch. 5), we derived a continuous (undiscretized), 1-D, multiphysics model of neutron transport – thermal hydraulics coupling in a Pressurized Water Reactor (PWR) fuel pin unit cell. (We refer to this as the "continuous 1-D model".) In the present chapter (Ch. 6), we (i) summarize the continuous 1-D model, (ii) derive a "discrete" form of the model that is suitable for computer simulation, (iii) outline the Multiphysics CMFD (M-CMFD) (i.e. R-CMFD without relaxation, $\alpha = 1$), Relaxed CMFD (R-CMFD), X-CMFD, NILO-A, and NILO-CMFD methods as they apply to the discrete 1-D model, (iv) introduce physical- and discretization-parameters to fully specify a series of discrete model problems, and (v) compare the relative performance of M-CMFD, R-CMFD, X-CMFD, NILO-A, and NILO-CMFD on these discrete model problems. Our goal in this chapter is to summarize our initial investigations of the performance of NILO-A and NILO-CMFD. The results of this investigation led us to pursue an implementation of NILO-A in the Michigan Parallel Characteristics-based Transport (MPACT) code, a topic covered in the next chapter (Ch. 7). (For reasons outlined in Ch. 7, we have yet to implement the full NILO-CMFD method in MPACT. Therefore, Ch. 7 will only discuss the incomplete NILO-A method.)

6.2. Continuous 1-D Model

Before introducing discretization, we restate the continuous 1-D model as summarized at the conclusion of Ch. 5. In the continuous 1-D model, the neutron angular flux

distribution ψ, a function of axial position z and $\hat{\boldsymbol{k}}$-oriented angular cosine μ, is governed by the 1-D (slab geometry), isotropic scattering, monoenergetic, λ-eigenvalue transport equation:

$$\begin{aligned} \mu\frac{\partial}{\partial z}\psi(z,\mu) + \Sigma_t(z)\psi(z,\mu) &= \frac{1}{2}\big(\Sigma_s(z) + \lambda\nu\Sigma_f(z)\big)\phi(z) \ , \\ 0 < z < Z \ , \ \ -1 \leq \mu \leq 1 \ ; \end{aligned} \tag{6.1a}$$

with vacuum boundary conditions imposed on domain boundaries:

$$\psi(0,\mu) = 0 \ , \ \ 0 < \mu \leq 1 \ , \tag{6.1b}$$

$$\psi(Z,\mu) = 0 \ , \ \ -1 \leq \mu < 0 \ , \tag{6.1c}$$

and total system power (P) normalization used to enforce a unique dominant eigenfunction solution:

$$P = \int_0^Z \kappa\Sigma_f(z')\phi(z') \, \mathrm{d}z' \ . \tag{6.1d}$$

The neutron scalar flux ϕ and neutron current J (both scalars) are defined by:

$$\phi(z) \equiv \int_{-1}^{1} \psi(z,\mu') \, \mathrm{d}\mu' \ , \tag{6.1e}$$

$$J(z) \equiv \int_{-1}^{1} \mu'\psi(z,\mu') \, \mathrm{d}\mu' \ . \tag{6.1f}$$

In Eqs. (6.1), nuclear data (Σ_t, Σ_s, $\nu\Sigma_f$, and $\kappa\Sigma_f$, collectively referred to with the shorthand Σ_u) depend on the axial coordinate z through the radially-integrated fission heat source distribution $h(z)$ and bulk average fluid temperature $T(z)$, by way of the relationship:

$$\begin{aligned} \Sigma_u(z) &\equiv \Sigma_u\big(T(z), h(z)\big) \\ &= \Sigma_{u,0} \\ &\quad + \Sigma_{u,1,d}\left(\sqrt{T(z) + \beta h(z)} - \sqrt{T_{d,0}}\right) \\ &\quad + \Sigma_{u,1,b}\Big(T(z) - T_{b,0}\Big) \ . \end{aligned} \tag{6.2}$$

The radially-integrated fission heat source distribution $h(z)$ is defined by:

$$h(z) \equiv \kappa\Sigma_f(z)\phi(z) \ , \tag{6.3a}$$

and acts as a forcing term in the bulk average fluid temperature $T(z)$ equation:

$$\mathcal{L}T(z) = h(z) \ . \tag{6.3b}$$

Here, the *high-order thermal hydraulics operator* $\mathcal{L}$ is an advection-diffusion operator:

$$\mathcal{L} \equiv -\frac{\mathrm{d}}{\mathrm{d}z} D_t \frac{\mathrm{d}}{\mathrm{d}z} + \dot{m} c_p \frac{\mathrm{d}}{\mathrm{d}z} \ , \tag{6.3c}$$

with specified inlet temperature:

$$T(0) = T_{\text{in}} \ , \tag{6.3d}$$

and an outlet Neumann boundary condition:

$$\frac{\mathrm{d}T}{\mathrm{d}z}(Z) \approx 0 \ . \tag{6.3e}$$

The nonlinearly-coupled Eqs. (6.1)-(6.3) form the continuous 1-D model. To fully specify the model, we must prescribe the following constants: fuel pin height Z, total system thermal power P, bulk fluid turbulent diffusion coefficient D_t, bulk fluid advection coefficient $\dot{m}c_p$, and bulk fluid inlet temperature T_{in}. We also need the following nuclear data parameters for Eq. (6.2): $\Sigma_{u,0}$, $\Sigma_{u,1,d}$, $\Sigma_{u,1,b}$, $T_{d,0}$, and $T_{b,0}$ for each of $\Sigma_u = (\Sigma_t, \Sigma_s, \nu\Sigma_f, \kappa\Sigma_f)$. Later, we provide numerical values for each of these parameters.

6.3. Discrete 1-D Model

Next, we discretize each component of the continuous 1-D model [Eqs. (6.1)-(6.3)]. We start by discretizing the transport Eqs. (6.1) using the *Discrete-Ordinates* (S_N) method in angle and *Diamond Difference* (DD) in space. Then, we derive a discrete, transport-corrected diffusion equation. This is required by the CMFD-based Step 4 in each of the multiphysics iterative methods we discuss. We then present the discretized thermal hydraulics equations. We include a discretization of both the advection-diffusion problem in Eqs. (6.3) ($\mathcal{L}$) and a simpler, pure-advection discretization to be used as NILO-A and NILO-CMFD's low-order thermal hydraulics operator (L). Finally, we describe the nuclear data representation used in the discrete 1-D model.

6.3.1. Neutron Transport

Here, we discretize the transport component [Eqs. (6.1)] of the continuous 1-D model. We begin by discretizing the angular cosine μ using the *Discrete Ordinates* (S_N) approximation. Rather than utilizing the continuous (infinite) set of angular cosines μ spanning $-1 \leq \mu \leq 1$, the S_N approximation employs a discrete (finite) set of *ordinates* μ_m, each with an associated *weight* w_m. We use the *angular index* $m = 1, 2, \ldots, M$ (i.e. $1 \leq m \leq M$) to distinguish between different ordinate/weight pairs (μ_m, w_m) that form this *angular quadrature set*. The neutron angular flux along a given ordinate μ_m is abbreviated:

$$\psi_m(z) \equiv \psi(z, \mu_m) \ . \tag{6.4}$$

Using this shorthand, we rewrite the continuous transport equation [Eq. (6.1a)] as a set of M discrete-in-angle (but still continuous-in-space) equations:

$$\begin{gathered} \mu_m \frac{\partial}{\partial z}\psi_m(z) + \Sigma_t(z)\psi_m(z) = \frac{1}{2}\big(\Sigma_s(z) + \lambda\nu\Sigma_f(z)\big)\phi(z) \ , \\ 1 \leq m \leq M \ , \ \ 0 < z < Z \ , \end{gathered} \tag{6.5a}$$

and we approximate the scalar flux ϕ and current J by numerical quadrature:

$$\phi(z) = \sum_{m'=1}^{M} \psi_{m'}(z) w_{m'} \ , \tag{6.5b}$$

$$J(z) = \sum_{m'=1}^{M} \mu_{m'} \psi_{m'}(z) w_{m'} \ . \tag{6.5c}$$

In order to discretize Eqs. (6.5) in space, we must define a spatial mesh. Our spatial mesh is illustrated in Fig. (6.1), where we use the monotonically increasing cell-edge coordinates:

$$0 = z_{1/2} < z_{3/2} < \cdots < z_{i-1/2} < z_{i+1/2} < \cdots < z_{I-1/2} < z_{I+1/2} = Z \ ,$$

to delimit a total of I spatial cells indexed using $i = 1, 2, \ldots, I$ (i.e. $1 \leq i \leq I$), with cell i encompassing:

$$z_{i-1/2} < z < z_{i+1/2} \ ,$$

for a cell width Δz_i of:

$$\Delta z_i \equiv z_{i+1/2} - z_{i-1/2} \ . \tag{6.6}$$

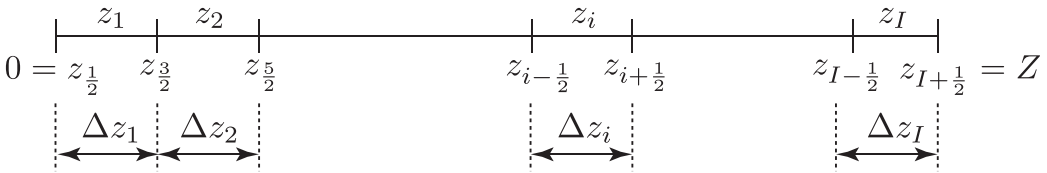


Figure 6.1: Discrete 1-D Model Spatial Mesh.

We define the cell-edge angular flux $\psi_{m,i+1/2}$, cell-average angular flux $\psi_{m,i}$, and cell-average scalar flux ϕ_i:

$$\psi_{m,i+1/2} \equiv \psi_m(z_{i+1/2}) \ , \tag{6.7a}$$

$$\psi_{m,i} \equiv \frac{1}{\Delta z_i} \int_{z_{i-1/2}}^{z_{i+1/2}} \psi_m(z') \, \mathrm{d}z' \ , \tag{6.7b}$$

$$\phi_i \equiv \frac{1}{\Delta z_i} \int_{z_{i-1/2}}^{z_{i+1/2}} \phi(z') \, \mathrm{d}z' \ , \tag{6.7c}$$

and we approximate nuclear data Σ_u as spatially constant within a given cell i:

$$\Sigma_u(z) = \Sigma_{u,i} \ , \quad z_{i-1/2} < z < z_{i+1/2} \ . \tag{6.8}$$

Then, by operating on Eq. (6.5a) with:

$$\frac{1}{\Delta z_i} \int_{z_{i-1/2}}^{z_{i+1/2}} (\cdot) \, \mathrm{d}z' \ ,$$

over each cell $1 \leq i \leq I$ in the spatial mesh, we obtain $I \times M$ particle balance equations:

$$\begin{gathered} \frac{\mu_m}{\Delta z_i} \big(\psi_{m,i+1/2} - \psi_{m,i-1/2} \big) + \Sigma_{t,i} \psi_{m,i} = \frac{1}{2} \big(\Sigma_{s,i} + \lambda \nu \Sigma_{f,i} \big) \phi_i \ , \\ 1 \leq m \leq M \ , \ \ 1 \leq i \leq I \ . \end{gathered} \tag{6.9a}$$

For the leftmost ($i = 1$) and rightmost ($i = I$) cells, the boundary incoming angular

fluxes are set by the vacuum boundary conditions imposed by Eqs. (6.1b)-(6.1c):

$$\psi_{m,1/2} = 0 \ , \quad \mu_m > 0 \ , \tag{6.9b}$$

$$\psi_{m,I+1/2} = 0 \ , \quad \mu_m < 0 \ . \tag{6.9c}$$

The power normalization condition in Eq. (6.1d) is discretized using the definitions in Eqs. (6.6), (6.7c), and (6.8):

$$P = \sum_{i'=1}^{I} \kappa \Sigma_{f,i'} \phi_{i'} \Delta z_{i'} \ . \tag{6.9d}$$

By combining Eqs. (6.5b), (6.7b), and (6.7c), we get the following expression for the cell-average neutron scalar flux ϕ_i:

$$\phi_i = \sum_{m'=1}^{M} \psi_{m',i} w_{m'} \ . \tag{6.9e}$$

We also define the cell-edge neutron current $J_{i+1/2}$:

$$J_{i+1/2} = \sum_{m'=1}^{M} \mu_{m'} \psi_{m',i+1/2} w_{m'} \ . \tag{6.9f}$$

($J_{i+1/2}$ appears in the next subsection when deriving discrete transport-corrected diffusion equations.)

Eqs. (6.9a)-(6.9f) describe a system with more unknowns than equations (i.e. the linear system is undetermined). In order to "close" this system, we impose additional *auxiliary equations*. We use the classic *Diamond Difference* (DD) closure for this purpose, which approximates the cell-average angular flux $\psi_{m,i}$ as the average of the cell-edge angular fluxes $\psi_{m,i\pm1/2}$:

$$\begin{aligned} \psi_{m,i} = \frac{1}{2}\left(\psi_{m,i-1/2} + \psi_{m,i+1/2}\right) \ , \\ 1 \leq m \leq M \ , \quad 1 \leq i \leq I \ . \end{aligned} \tag{6.9g}$$

With the linear transport system closed by Eq. (6.9g), Eqs. (6.9) represent the discrete form of the continuous transport Eqs. (6.1) used in the discrete 1-D model.

6.3.2. Transport-Corrected Diffusion

Starting with the discretized transport Eqs. (6.9), we next derive a discrete transport-corrected diffusion problem. We refer the reader to [57] for a more in-depth derivation that includes the complications that arise when using a "coarse" diffusion mesh (as measured against the "fine" transport mesh), an implementation detail typical of practical CMFD implementations. In this chapter, for simplicity, we discretize the transport and (transport-corrected) diffusion problems on the same spatial grid [Fig. (6.1)].

To begin, we take the zero'th discrete angular moment of the transport Eq. (6.9a) with:

$$\sum_{m'=1}^{M} (\cdot) w_{m'} \ .$$

We also enforce that the weights w_m in our quadrature set sum to two:

$$\sum_{m'=1}^{M} w_{m'} = 2 \ ,$$

and we use the definitions of the cell-average scalar flux ϕ_i [Eq. (6.9e)] and cell-edge current $J_{i+1/2}$ [Eq. (6.9f)] to obtain the familiar, angularly-integrated, discrete neutron balance equation:

$$\frac{1}{\Delta z_i}(J_{i+1/2} - J_{i-1/2}) + \Sigma_{a,i}\phi_i = \lambda\, \nu\Sigma_{f,i}\phi_i \ ,$$
$$1 \leq i \leq I \ . \tag{6.10a}$$

Next, without approximation, we manipulate the expressions for the interior cell-edge currents $J_{i+1/2}$, $1 \leq i \leq I-1$ to obtain:

$$\begin{aligned} J_{i+1/2} &= -\frac{D_{i+1/2}}{\Delta z_{i+1/2}}(\phi_{i+1} - \phi_i) + \left(J_{i+1/2} + \frac{D_{i+1/2}}{\Delta z_{i+1/2}}(\phi_{i+1} - \phi_i) \right) \\ &= -\frac{D_{i+1/2}}{\Delta z_{i+1/2}}(\phi_{i+1} - \phi_i) + \left(\frac{J_{i+1/2} + \frac{D_{i+1/2}}{\Delta z_{i+1/2}}(\phi_{i+1} - \phi_i)}{\phi_{i+1} + \phi_i} \right) (\phi_{i+1} + \phi_i) \\ &= -\frac{D_{i+1/2}}{\Delta z_{i+1/2}}(\phi_{i+1} - \phi_i) + \hat{D}_{i+1/2}(\phi_{i+1} + \phi_i) \ , \end{aligned} \tag{6.10b}$$

where:

$$\hat{D}_{i+1/2} \equiv \frac{J_{i+1/2} + \frac{D_{i+1/2}}{\Delta z_{i+1/2}}(\phi_{i+1} - \phi_i)}{\phi_{i+1} + \phi_i} \, .$$
$$= \textit{transport correction factor} \, . \tag{6.10c}$$

In these equations, we have introduced the *cell-edge total macroscopic cross section*:

$$\Sigma_{t,i+1/2} = \frac{\Sigma_{t,i}\Delta z_i + \Sigma_{t,i+1}\Delta z_{i+1}}{\Delta z_i + \Delta z_{i+1}} \, , \tag{6.10d}$$

the *cell-edge diffusion coefficient*:

$$D_{i+1/2} = \frac{1}{3\Sigma_{t,i+1/2}} \, , \tag{6.10e}$$

and the *cell-edge width*:

$$\Delta z_{i+1/2} = \frac{1}{2}(\Delta z_i + \Delta z_{i+1}) \, . \tag{6.10f}$$

Eq. (6.10b) is exact. We note that if $\hat{D}_{i+1/2}$ is set to 0 in this equation, the equation reduces to a discrete version of Fick's Law, which is the basis of diffusion theory. Therefore, the dimensionless quantity $\hat{D}_{i+1/2}$ is the transport correction to diffusion theory. For most problems and most spatial cells, $\hat{D}_{i+1/2}$ is small, and in CMFD and related methods described in this thesis, this term is lagged.

Eq. (6.10b) expresses the interior cell-edge currents in terms of the neighboring cell-average scalar fluxes. It remains to do this for the cell edges on the outer boundaries.

At the left boundary, we use the vacuum boundary condition of the angular flux to write:

$$\begin{aligned} J_{1/2} &= \sum_{m'=1}^{M} \mu_{m'} \psi_{m',1/2} w_{m'} \\ &= \sum_{\mu_{m'}<0} \mu_{m'} \psi_{m',1/2} w_{m'} \\ &= \left(\frac{\sum_{\mu_{m'}<0} \mu_{m'} \psi_{m',1/2} w_{m'}}{\sum_{m'=1}^{M} \psi_{m',1} w_{m'}} \right) \phi_1 \\ &= -B_{1/2}\, \phi_1 \, , \end{aligned} \tag{6.11a}$$

where

$$B_{1/2} = -\frac{\sum_{\mu_{m'}<0} \mu_{m'} \psi_{m',1/2} w_{m'}}{\sum_{m'=1}^{M} \psi_{m',1} w_{m'}}$$
$$= \frac{\sum_{\mu_{m'}<0} |\mu_{m'}| \psi_{m',1/2} w_{m'}}{\sum_{m'=1}^{M} \psi_{m',1} w_{m'}}$$
$$= -\frac{J_{1/2}}{\phi_1} \, . \tag{6.11b}$$

(The minus sign is included so that $B_{1/2} > 0$.) Eq. (6.11a) relates the cell-edge current on the left boundary to the cell-average scalar flux in the leftmost spatial cell. If the angular flux has a weak spatial variation across the leftmost cell and is nearly isotropic for outgoing angles, then

$$B_{1/2} \approx \frac{\sum_{\mu_{m'}<0} |\mu_{m'}| \psi_{m',1/2} w_{m'}}{\sum_{\mu_{m'}<0} \psi_{m',1} w_{m'}}$$
$$\approx \frac{\sum_{\mu_{m'}<0} |\mu_{m'}| w_{m'}}{\sum_{\mu_{m'}<0} w_{m'}}$$
$$\approx \frac{1}{2} \, . \tag{6.11c}$$

Therefore, unlike the interior transport correction factors $\hat{D}_{i+1/2}$, which are generally small, the *boundary transport correction factor* $B_{1/2}$ is not small, but it is *stable* – it depends weakly on the angular flux. (Hence, it too can be lagged.)

Similarly, the relation between the cell-edge current on the right boundary and the cell-averaged scalar flux in the rightmost spatial cell is:

$$J_{I+1/2} = B_{I+1/2} \, \phi_I \, , \tag{6.11d}$$

where

$$B_{I+1/2} = \frac{J_{I+1/2}}{\phi_I} \, . \tag{6.11e}$$

Eqs. (6.10b), (6.11a), and (6.11d) express the cell-edge currents $J_{i+1/2}$ in terms of the adjoining cell-average scalar fluxes. When these equations are introduced into the balance equations (6.10a), one obtains a discrete, tridiagonal, diffusion problem for the cell-averaged scalar fluxes. (Of course, this discrete problem contains the

transport correction factors $\hat{D}_{i+1/2}$, $B_{1/2}$, and $B_{I+1/2}$; all of these quantities will be lagged.)

Finally, the power normalization condition in Eq. (6.9d) is unchanged:

$$P = \sum_{i'=1}^{I} \kappa \Sigma_{f,i'} \phi_{i'} \Delta z_{i'} \ . \tag{6.11f}$$

Eqs. (6.10)-(6.11) represent the discrete, transport-corrected diffusion problem used in this chapter. The scalar flux ϕ_i and eigenvalue λ solutions of Eqs. (6.10)-(6.11) are identical to those of the discrete transport Eqs. (6.9), since no approximates were made when forming the neutron balance equation or when defining the cell-edge neutron currents. Rather, we have merely manipulated the discrete transport problem and introduced new parameter definitions. Unfortunately, taken alone, Eqs. (6.10)-(6.11) are not a closed system; there are more unknowns than equations. In order to close Eqs. (6.10)-(6.11), CMFD-based methods lag the edge-based transport-correction coefficients $\hat{D}_{i+1/2}$, $B_{1/2}$, and $B_{I+1/2}$ based on transport sweep-determined estimates of the angular flux, scalar flux, and current. By lagging these parameters, Eqs. (6.10)-(6.11) are transformed into a set of diffusion-like (i.e. angularly integrated) equations that are more easily solved than the discrete transport equations [Eqs. (6.9)] themselves .

6.3.3. Thermal Hydraulics

Thermal hydraulics in the continuous 1-D model is described by:

$$\mathcal{L}T(z) = h(z) \ , \tag{6.12}$$

where $\mathcal{L}$ is the underlying advection-diffusion differential operator [Eq. (6.3c)] with Dirichlet inlet [Eq. (6.3d)] and Neumann outlet [Eq. (6.3e)] boundary conditions, and $h(z)$ is the radially-integrated fission heat source distribution [Eq. (6.3a)]. In the continuous 1-D model, $\mathcal{L}$ governs the bulk average fluid temperature distribution $T(z)$ in response to the fission heat source distribution $h(z)$.

In the discrete 1-D model, we replace the continuous Eq. (6.12) by the discrete:

$$\bar{\mathcal{L}}T_i = h_i \ , \tag{6.13}$$

where we use an overbar to distinguish between the discrete thermal hydraulics operator $\bar{\mathcal{L}}$ in Eq. (6.13) and the continuous thermal hydraulics operator $\mathcal{L}$ in Eq. (6.12). Also, in Eq. (6.13), T_i represents cell-average fluid temperatures T_i, $1 \leq i \leq I$ and h_i represents the cell-average fission heat sources h_i, $1 \leq i \leq I$.

The discrete model's $\bar{\mathcal{L}}$ is a standard finite-volume discretization of the 1-D advection-diffusion equation, with a second-order diffusion discretization, and a first-order upwinded advection term. For the boundary conditions, we use a ghost-cell approach [60]. The inversion of $\bar{\mathcal{L}}$ required to solve Eq. (6.13) for the cell-average temperature solution T_i in response to a prescribed cell-average heat source h_i, given by:

$$h_i = \kappa \Sigma_{f,i} \phi_i \ , \tag{6.14}$$

requires the solution of the tri-diagonal system of equations:

$$\begin{aligned} -D_t\Big(\frac{T_{i+1}-T_i}{\Delta z_{i+1/2}} - \frac{T_i - T_{i-1}}{\Delta z_{i-1/2}}\Big) + \dot{m}c_p(T_i - T_{i-1}) &= h_i \Delta z_i \ , \\ 1 \leq i \leq I \ . \end{aligned} \tag{6.15a}$$

Here, the cell-edge widths $\Delta z_{i+1/2}$ are defined for both interior and boundary-adjacent cell-edges:

$$\Delta z_{i+1/2} \equiv \frac{1}{2}\big(\Delta z_i + \Delta z_{i+1}\big) \ , \quad 0 \leq i \leq I \ , \tag{6.15b}$$

the bottom- and top-boundary ghost-cell widths Δz_0 and Δz_{I+1} are defined as:

$$\Delta z_0 \equiv \Delta z_1 \ , \tag{6.15c}$$

$$\Delta z_{I+1} \equiv \Delta z_I \ , \tag{6.15d}$$

the incoming Dirichlet boundary condition is specified by:

$$\frac{1}{2}\big(T_0 + T_1\big) = T_{\text{in}} \ , \tag{6.15e}$$

and the outgoing Neumann boundary condition is enforced by:

$$T_I = T_{I+1} \ . \tag{6.15f}$$

In addition to the cell-average temperature solution T_i for each physical cell $1 \leq i \leq I$ in the problem, the solution of Eqs. (6.15) includes the ghost-cell temperatures

T_0 and T_{I+1}. These ghost-cell temperatures are not physically meaningful and can be discarded; they are a mathematical idiosyncrasy of our chosen boundary condition discretizations.

Advection

The NILO-A and NILO-CMFD methods make use of an approximate, low-order thermal hydraulics operator L to capture low-spatial-frequency feedback response. In the 1-D model, the high-order thermal hydraulics operator $\mathcal{L}$ is an advection-diffusion operator; Eqs. (6.3) and Eqs. (6.13)-(6.15) describe the continuous $\mathcal{L}$ and discrete $\bar{\mathcal{L}}$ models, respectively. In this chapter, we use a pure-advection operator for NILO-A and NILO-CMFD's low-order thermal hydraulics. For the continuous 1-D model, we write:

$$LT(z) = h(z) \ , \tag{6.16a}$$

$$L \equiv \dot{m} c_p \frac{\mathrm{d}}{\mathrm{d}z} \ , \tag{6.16b}$$

$$T(0) = T_{\text{in}} \ . \tag{6.16c}$$

We use an over-bar to distinguish between the continuous L and discrete $\bar{L}$ model low-order thermal hydraulics operators, just as we did with the continuous $\mathcal{L}$ and discrete $\bar{\mathcal{L}}$ high-order thermal hydraulics operators. We write the discretized form of Eqs. (6.16) as:

$$\bar{L} T_i = h_i \ , \tag{6.17a}$$

where the solution of the cell-average temperature T_i in response to the cell-average fission heat source h_i involves the marching solution of the following discretized advection equation, with $T_{i+1/2}$ representing cell-edge temperatures:

$$T_{i+1/2} = T_{i-1/2} + \frac{1}{\dot{m} c_p} h_i \Delta z_i \ , \quad 1 \leq i \leq I \ , \tag{6.17b}$$

$$T_i = \frac{1}{2}\left(T_{i-1/2} + T_{i+1/2}\right) \ , \quad 1 \leq i \leq I \ , \tag{6.17c}$$

$$T_{1/2} = T_{\text{in}} \ . \tag{6.17d}$$

(We note that if the turbulent diffusion coefficient D_t is set to zero in the discretized advection-diffusion Eqs. (6.15), these equations do not simplify to our chosen advection discretization in Eqs. (6.17). In our numerical results, this discretization

"inconsistency" does not detrimentally affect NILO-A or NILO-CMFD performance. This suggests a desirable flexibility in the choice of a suitable NILO low-order thermal hydraulics operator $\bar{L}$.)

6.3.4. Nuclear Data

Since our transport discretization assumes that nuclear data is constant within a cell [Eq. (6.8)], we require a representative bulk average fluid temperature and fission heat source in each cell, $1 \leq i \leq I$, in order to evaluate nuclear data via the continuous model's Eq. (6.2). To do this, we simply use the cell-averaged T_i and h_i. Therefore, in the discrete 1-D model, nuclear data is evaluated using:

$$\begin{aligned}\Sigma_{u,i} = \Sigma_{u,0} \\ &+ \Sigma_{u,1,d}\left(\sqrt{T_i + \beta h_i} - \sqrt{T_{d,0}}\right) \\ &+ \Sigma_{u,1,b}\left(T_i - T_{b,0}\right) .\end{aligned} \tag{6.18}$$

6.4. Multiphysics Iterative Methods

Next, we formulate the Multiphysics CMFD (M-CMFD), Relaxed CMFD (R-CMFD), X-CMFD, NILO-A, and NILO-CMFD multiphysics iterative methods for the discrete 1-D model. While the forthcoming outlines may be reminiscent of the more general, continuous descriptions in Ch. 4, they are written here specifically as applied to the discrete 1-D model. Thus, the outlines in this chapter can be easily translated into a computer code to generate the numerical results presented later.

6.4.1. Multiphysics CMFD (M-CMFD) & Relaxed CMFD (R-CMFD)

We begin by outlining the Relaxed CMFD (R-CMFD) method as applied to the discrete 1-D model. Since R-CMFD generalizes Multiphysics CMFD (M-CMFD), the R-CMFD outline below can be used to describe M-CMFD simply by removing relaxation, which is accomplished by setting the relaxation factor $\alpha = 1$:

M/R-CMFD Step 0 : Initialization

Before iterating, we require initial estimates of the scalar flux $\phi_i^{(0)}$, eigenvalue $\lambda^{(0)}$, and nuclear data $\Sigma_{u,i}^{(0)}$. After these are determined, we set the outer iteration index $n = 0$ and proceed to the first outer iteration.

M/R-CMFD Step 1 : Neutron Transport Sweep

By lagging the scalar flux $\phi_i^{(n)}$, eigenvalue $\lambda^{(n)}$, and nuclear data $\Sigma_{u,i}^{(n)}$ in the discrete transport Eqs. (6.9), we form the fixed-source transport problem written below, which we solve by "*sweeping*":

$$\frac{\mu_m}{\Delta z_i}\big(\psi_{m,i+1/2}^{(n+\frac{1}{2})} - \psi_{m,i-1/2}^{(n+\frac{1}{2})}\big) + \Sigma_{t,i}^{(n)}\psi_{m,i}^{(n+\frac{1}{2})} = \frac{1}{2}\big(\Sigma_{s,i}^{(n)} + \lambda^{(n)}\nu\Sigma_{f,i}^{(n)}\big)\phi_i^{(n)} \ , \tag{6.19a}$$

$$\psi_{m,i}^{(n+\frac{1}{2})} = \frac{1}{2}\big(\psi_{m,i-1/2}^{(n+\frac{1}{2})} + \psi_{m,i+1/2}^{(n+\frac{1}{2})}\big) \ , \quad 1 \le m \le M \ , \ \ 1 \le i \le I \ , \tag{6.19b}$$

$$\psi_{m,1/2}^{(n+\frac{1}{2})} = 0 \ , \ \ \mu_m > 0 \ , \tag{6.19c}$$

$$\psi_{m,I+1/2}^{(n+\frac{1}{2})} = 0 \ , \ \ \mu_m < 0 \ . \tag{6.19d}$$

While sweeping Eqs. (6.19), we update cell-average scalar flux $\phi_i^{(n+\frac{1}{2})}$ and cell-edge current $J_{i+1/2}^{(n+\frac{1}{2})}$ estimates:

$$\phi_i^{(n+\frac{1}{2})} = \sum_{m'=1}^{M} \psi_{m',i}^{(n+\frac{1}{2})} w_{m'} \ , \tag{6.20a}$$

$$J_{i+1/2}^{(n+\frac{1}{2})} = \sum_{m'=1}^{M} \mu_{m'} \psi_{m',i+1/2}^{(n+\frac{1}{2})} w_{m'} \ . \tag{6.20b}$$

M/R-CMFD Step 2 : Thermal Hydraulics Solve

Using the latest nuclear data $\Sigma_{u,i}^{(n)}$ and cell-average scalar flux $\phi_i^{(n+\frac{1}{2})}$ estimates, we update the cell-average fission heat source $h_i^{(n+\frac{1}{2})}$:

$$h_i^{(n+\frac{1}{2})} = \kappa\Sigma_{f,i}^{(n)}\phi_i^{(n+\frac{1}{2})} \ . \tag{6.21a}$$

Using the relaxation factor α, we compute the relaxed heat source $\tilde{h}_i^{(n+\frac{1}{2})}$ using $h_i^{(n+\frac{1}{2})}$ and $h_i^{(n-\frac{1}{2})}$, the Eq. (6.21a)-evaluated cell-average heat source from the current and previous outer iterations, respectively:

$$\tilde{h}_i^{(n+\frac{1}{2})} = \alpha h_i^{(n+\frac{1}{2})} + (1-\alpha) h_i^{(n-\frac{1}{2})} \ . \tag{6.21b}$$

(In Multiphysics CMFD (M-CMFD), $\alpha = 1$ and $\tilde{h}_i^{(n+\frac{1}{2})} = h_i^{(n+\frac{1}{2})}$.)

Using the relaxed heat source $\tilde{h}_i^{(n+\frac{1}{2})}$, we solve the discrete advection-diffusion thermal hydraulics problem [Eqs. (6.15)] for the updated bulk average fluid temperature $T_i^{(n+\frac{1}{2})}$:

$$\bar{\mathcal{L}} T_i^{(n+\frac{1}{2})} = \tilde{h}_i^{(n+\frac{1}{2})} \ . \tag{6.21c}$$

M/R-CMFD Step 3 : Nuclear Data Evaluation

We use the relaxed heat source $\tilde{h}_i^{(n+\frac{1}{2})}$ and the Eq. (6.21c)-determined fluid temperature $T_i^{(n+\frac{1}{2})}$ to update the nuclear data:

$$\begin{aligned} \Sigma_{u,i}^{(n+\frac{1}{2})} = \Sigma_{u,0} &+ \Sigma_{u,1,d}\big(\sqrt{T_i^{(n+\frac{1}{2})} + \beta \tilde{h}_i^{(n+\frac{1}{2})}} - \sqrt{T_{d,0}}\big) \\ &+ \Sigma_{u,1,b}\big(T_i^{(n+\frac{1}{2})} - T_{b,0}\big) \ . \end{aligned} \tag{6.22}$$

We then update the interior cell-edge total macroscopic cross sections $\Sigma_{t,i+1/2}^{(n+\frac{1}{2})}$:

$$\Sigma_{t,i+1/2}^{(n+\frac{1}{2})} = \frac{\Sigma_{t,i}^{(n+\frac{1}{2})} \Delta z_i + \Sigma_{t,i+1}^{(n+\frac{1}{2})} \Delta z_{i+1}}{\Delta z_i + \Delta z_{i+1}} \ , \tag{6.23a}$$

and the cell-edge diffusion coefficients $D_{i+1/2}^{(n+\frac{1}{2})}$:

$$D_{i+1/2}^{(n+\frac{1}{2})} = \frac{1}{3\Sigma_{t,i+1/2}^{(n+\frac{1}{2})}} \ . \tag{6.23b}$$

We then use sweep-determined cell-average scalar fluxes $\phi_i^{(n+\frac{1}{2})}$ and cell-edge currents $J_{i+1/2}^{(n+\frac{1}{2})}$ to update the interior cell-edge transport correction factors $\hat{D}_{i+1/2}^{(n+\frac{1}{2})}$:

$$\hat{D}_{i+1/2}^{(n+\frac{1}{2})} = \frac{J_{i+1/2}^{(n+\frac{1}{2})} + \frac{D_{i+1/2}^{(n+\frac{1}{2})}}{\Delta z_{i+1/2}} (\phi_{i+1}^{(n+\frac{1}{2})} - \phi_i^{(n+\frac{1}{2})})}{\phi_i^{(n+\frac{1}{2})} + \phi_{i+1}^{(n+\frac{1}{2})}} \ . \tag{6.23c}$$

We also update boundary correction factors $B_{1/2}^{(n+\frac{1}{2})}$ and $B_{I+1/2}^{(n+\frac{1}{2})}$ with:

$$B_{1/2}^{(n+\frac{1}{2})} = -\frac{J_{1/2}^{(n+\frac{1}{2})}}{\phi_1^{(n+\frac{1}{2})}} \ , \tag{6.23d}$$

$$B_{I+1/2}^{(n+\frac{1}{2})} = \frac{J_{I+1/2}^{(n+\frac{1}{2})}}{\phi_I^{(n+\frac{1}{2})}} \ . \tag{6.23e}$$

M/R-CMFD Step 4 : Diffusion Solve

By lagging the cell-edge diffusion coefficients $D_{i+1/2}^{(n+\frac{1}{2})}$, cell-edge transport correction factors $\hat{D}_{i+1/2}^{(n+\frac{1}{2})}$, boundary correction factors $B_{1/2}^{(n+\frac{1}{2})}$ and $B_{I+1/2}^{(n+\frac{1}{2})}$, and nuclear data $\Sigma_{u,i}^{(n+\frac{1}{2})}$ in Eqs. (6.10)-(6.11), we obtain a linear transport-corrected diffusion problem that can be solved for the end-of-outer-iteration scalar flux $\phi_i^{(n+1)}$ and eigenvalue $\lambda^{(n+1)}$ estimates:

$$\begin{aligned} \frac{1}{\Delta z_i}\left(J_{i+1/2}^{(n+1)} - J_{i-1/2}^{(n+1)}\right) + \Sigma_{a,i}^{(n+\frac{1}{2})}\phi_i^{(n+1)} \\ = \lambda^{(n+1)}\nu\Sigma_{f,i}^{(n+\frac{1}{2})}\phi_i^{(n+1)} \ , \\ 1 \leq i \leq I \ , \end{aligned} \tag{6.24a}$$

$$\begin{aligned} J_{i+1/2}^{(n+1)} = -\frac{D_{i+1/2}^{(n+\frac{1}{2})}}{\Delta z_{i+1/2}}\left(\phi_{i+1}^{(n+1)} - \phi_i^{(n+1)}\right) \\ + \hat{D}_{i+1/2}^{(n+\frac{1}{2})}\left(\phi_i^{(n+1)} + \phi_{i+1}^{(n+1)}\right) \ , \\ 1 \leq i \leq I-1 \ , \end{aligned} \tag{6.24b}$$

$$J_{1/2}^{(n+1)} = -B_{1/2}^{(n+\frac{1}{2})}\phi_1^{(n+1)} \ , \tag{6.24c}$$

$$J_{I+1/2}^{(n+1)} = B_{I+1/2}^{(n+\frac{1}{2})}\phi_I^{(n+1)} \ , \tag{6.24d}$$

$$P = \sum_{i'=1}^{I} \kappa\Sigma_{f,i'}^{(n+\frac{1}{2})}\phi_{i'}^{(n+1)}\Delta z_{i'} \ . \tag{6.24e}$$

We still must update the end-of-outer-iteration nuclear data estimates $\Sigma_{u,i}^{(n+1)}$. To do this, we simply use the values obtained from M/R-CMFD Step 3, Eq.

(6.22):

$$\Sigma_{u,i}^{(n+1)} = \Sigma_{u,i}^{(n+\frac{1}{2})} \ . \tag{6.25}$$

We then check convergence and max iteration criteria. If either condition has been met, we are finished! Otherwise, we increase the outer iteration index n by one and return to M/R-CMFD Step 1 to begin the next outer iteration.

As mentioned earlier, Relaxed CMFD (R-CMFD) can be reduced to Multiphysics CMFD (M-CMFD) by fixing the relaxation factor $\alpha = 1$ in M/R-CMFD Step 2, Eq. (6.21b), thereby removing relaxation and setting $\tilde{h}_i^{(n+\frac{1}{2})} = h_i^{(n+\frac{1}{2})}$.

Wielandt Shifted Power Iteration (WS-PI)

In order to perform M/R-CMFD Step 4, we must solve the linear diffusion eigenvalue problem in Eqs. (6.24) for the end-of-outer-iteration scalar flux vector $\phi_i^{(n+1)}$ and scalar eigenvalue $\lambda^{(n+1)}$. There are many viable approaches to solving this standard problem. Next, we outline one common approach, Wielandt Shifted Power Iteration (WS-PI). Our outline of WS-PI is in no way novel; it follows closely to the method description as outlined in the literature [124]. We include a detailed description of WS-PI for pedagogical reasons, to ease into the more-complex, nonstandard, WS-PI-based iterative method that we use to solve the X-CMFD, NILO-A, and NILO-CMFD nonlinear diffusion eigenvalue problems. The upcoming WS-PI outline also serves as a stepping stone to introduce the use of the Wielandt shift to accelerate unshifted Power Iteration (PI), a topic neglected in Ch. 4 for simplicity.

Next, we describe WS-PI in its more-or-less standard form as it would be applied to solve Eqs. (6.24) in the $(n+1)^{\text{st}}$ outer iteration of M-CMFD or R-CMFD:

WS-PI Step 0 : Initialization

Before starting the inner iterations in M-CMFD / R-CMFD's $(n+1)^{\text{st}}$ outer iteration, we define a few operators (matrices) for convenience. We define the lagged (owing to $D_{i+1/2}^{(n+\frac{1}{2})}$, $\hat{D}_{i+1/2}^{(n+\frac{1}{2})}$, and $\Sigma_{u,i}^{(n+\frac{1}{2})}$), discrete, (transport-corrected) leakage + absorption diffusion operator $M^{(n+\frac{1}{2})}$ by:

$$M^{(n+\frac{1}{2})}\phi_i \equiv \frac{1}{\Delta z_i}\Big[-\frac{D^{(n+\frac{1}{2})}_{i+1/2}}{\Delta z_{i+1/2}}(\phi_{i+1}-\phi_i) + \hat{D}^{(n+\frac{1}{2})}_{i+1/2}\big(\phi_i+\phi_{i+1}\big) + \frac{D^{(n+\frac{1}{2})}_{i-1/2}}{\Delta z_{i-1/2}}\big(\phi_i-\phi_{i-1}\big) - \hat{D}^{(n+\frac{1}{2})}_{i-1/2}\big(\phi_{i-1}+\phi_i\big)\Big] + \Sigma^{(n+\frac{1}{2})}_{a,i}\phi_i \ , \tag{6.26a}$$

where, for simplicity, we have only defined $M^{(n+\frac{1}{2})}$ for non-boundary-adjacent cells. Eq. (6.26a) is obtained by eliminating $J^{(n+1)}_{i\pm 1/2}$ between Eqs. (6.24a) and (6.24b). We also define the lagged fission neutron production operator $F^{(n+\frac{1}{2})}$ by:

$$F^{(n+\frac{1}{2})}\phi_i \equiv \nu\Sigma^{(n+\frac{1}{2})}_{f,i}\phi_i \ , \tag{6.26b}$$

and the lagged fission heat generation operator $K^{(n+\frac{1}{2})}$ by:

$$K^{(n+\frac{1}{2})}\phi_i \equiv \kappa\Sigma^{(n+\frac{1}{2})}_{f,i}\phi_i \ . \tag{6.26c}$$

In addition, we define the following operator, representing spatial integration over the problem domain:

$$\langle(\cdot)\rangle \equiv \sum_{i'=1}^{I}(\cdot)\Delta z_{i'} \ . \tag{6.27}$$

Using the operator notation in Eqs. (6.26)-(6.27), we can rewrite the linear diffusion eigenvalue problem in Eqs. (6.24) as:

$$M^{(n+\frac{1}{2})}\phi^{(n+1)} = \lambda^{(n+1)}F^{(n+\frac{1}{2})}\phi^{(n+1)} \ , \tag{6.28a}$$

$$P = \langle K^{(n+\frac{1}{2})}\phi^{(n+1)}\rangle \ . \tag{6.28b}$$

Next, we define initial inner iteration estimates for the scalar flux $\phi^{(n+1,0)}_i$ (eigenfunction) and eigenvalue $\lambda^{(n+1,0)}$ using their most recent outer-iteration estimates:

$$\phi^{(n+1,0)}_i = \phi^{(n+\frac{1}{2})}_i \ , \tag{6.29a}$$

$$\lambda^{(n+1,0)} = \lambda^{(n)} \ . \tag{6.29b}$$

With these estimates set, we set the inner iteration index p to zero, and move to WS-PI Step 1.

WS-PI Step 1 : Fixed-Source Diffusion Solve

We solve the following fixed-source neutron diffusion problem for the **unnormalized** updated eigenfunction estimate $\tilde{\phi}_i^{(n+1,p+1)}$, where $\lambda_s^{(n+1,p)}$ is an iteration-dependent Wielandt shift:

$$\begin{aligned}(M^{(n+\frac{1}{2})} - \lambda_s^{(n+1,p)} F^{(n+\frac{1}{2})})\tilde{\phi}^{(n+1,p+1)} \\ = (\lambda^{(n+1,p)} - \lambda_s^{(n+1,p)})F^{(n+\frac{1}{2})}\phi^{(n+1,p)} \ .\end{aligned} \tag{6.30}$$

In practice, the Wielandt shift λ_s is iterated upon, but it must be chosen carefully. To ensure convergence of WS-PI, the magnitude of the Wielandt shift must be no larger than that of the dominant eigenvalue:

$$|\lambda_s^{(n+1,p)}| < |\lambda^{(n+1)}| \ . \tag{6.31}$$

In our numerical results (discussed later) we use the following expression to iteratively update the Wielandt shift at the conclusion of the $(p+1)^{\text{st}}$ inner iteration:

$$\lambda_s^{(n+1,p+1)} = \max(r\lambda^{(n+1,p+1)} - c_1|\lambda^{(n+1,p+1)} - \lambda^{(n+1,p)}|, \lambda_m) \ , \tag{6.32a}$$

with r, c_1, and λ_m as user-defined constants. (Eq. (6.32a) is similar to the adaptive Wielandt shift expression used in MPACT [80]. In our numerical tests, we use $r = 0.98$, $c_1 = 10$, $\lambda_m = 0$.) Prior to the first inner iteration, we initialize the shift with:

$$\lambda_s^{(n+1,0)} = 0 \ . \tag{6.32b}$$

WS-PI Step 2 : Eigenvalue Update & Renormalization

We then update the eigenvalue estimate with:

$$\lambda^{(n+1,p+1)} = \lambda_s^{(n+1,p)} + (\lambda^{(n+1,p)} - \lambda_s^{(n+1,p)})\frac{\langle F^{(n+\frac{1}{2})}\phi^{(n+1,p)}\rangle}{\langle F^{(n+\frac{1}{2})}\tilde{\phi}^{(n+1,p+1)}\rangle} \ , \tag{6.33}$$

we renormalize $\tilde{\phi}_i^{(n+1,p+1)}$ to satisfy Eq. (6.28b):

$$\phi^{(n+1,p+1)} = \frac{P}{\langle K^{(n+\frac{1}{2})}\tilde{\phi}^{(n+1,p+1)}\rangle}\tilde{\phi}^{(n+1,p+1)} \,, \tag{6.34}$$

and we update the Wielandt shift using Eqs. (6.32).

Next, we check inner iteration convergence and max iteration criteria. If either is met, we set the concluding inner iteration index $P^{(n+1)} = (p+1)$ and continue to WS-PI : Finalization. Otherwise, we increment the inner iteration index p by one and return to WS-PI Step 1.

WS-PI : Finalization

Before exiting the inner iteration completely, we update estimates to be used in the next outer iteration:

$$\phi_i^{(n+1)} = \phi_i^{(n+1,P^{(n+1)})} \,, \tag{6.35a}$$

$$\lambda^{(n+1)} = \lambda^{(n+1,P^{(n+1)})} \,. \tag{6.35b}$$

This concludes our discussion of Wielandt Shifted Power Iteration (WS-PI).

6.4.2. X-CMFD, NILO-A, & NILO-CMFD

Next, we detail X-CMFD, NILO-A, and NILO-CMFD applied to the discrete 1-D model. Steps 0 – 3 of M-CMFD, R-CMFD, X-CMFD, NILO-A, and NILO-CMFD are identical. For these steps, we refer back to the M/R-CMFD outline presented above. We begin by outlining X-CMFD. Afterwards, we show how NILO-A and NILO-CMFD can be obtained through a few modifications of the X-CMFD procedure.

X-CMFD Step 0 : Initialization

This is identical to M/R-CMFD Step 0.

X-CMFD Step 1 : Neutron Transport Sweep

This is identical to M/R-CMFD Step 1. (See Eqs. (6.19)-(6.20).)

X-CMFD Step 2 : Thermal Hydraulics Solve

This is identical to M/R-CMFD Step 2, **without relaxation**. (See Eqs. (6.21), with $\alpha = 1$.)

X-CMFD Step 3 : Nuclear Data Evaluation

This is identical to M/R-CMFD Step 3. (See Eqs. (6.22)-(6.23).)

X-CMFD Step 4 : Nonlinear Diffusion Solve

We solve the following nonlinear diffusion eigenvalue problem for the end-of-outer-iteration scalar flux $\phi_i^{(n+1)}$, eigenvalue $\lambda^{(n+1)}$, and nuclear data $\Sigma_{u,i}^{(n+1)}$ estimates. We use explicitly evaluated (i.e. lagged) diffusion coefficients $D_{i+1/2}^{(n+\frac{1}{2})}$, transport correction factors $\hat{D}_{i+1/2}^{(n+\frac{1}{2})}$, and boundary correction factors $B_{1/2}^{(n+\frac{1}{2})}$, $B_{I+1/2}^{(n+\frac{1}{2})}$. All other quantities are *implicitly* evaluated through the discrete 1-D model's advection-diffusion thermal hydraulics ($\bar{\mathcal{L}}$) [Eqs. (6.13)-(6.15)] and nuclear data representation [Eq. (6.18)]:

$$\begin{aligned}\frac{1}{\Delta z_i}\big(J_{i+1/2}^{(n+1)} - J_{i-1/2}^{(n+1)}\big) + \Sigma_{a,i}^{(n+1)}\phi_i^{(n+1)} \\ = \lambda^{(n+1)}\nu\Sigma_{f,i}^{(n+1)}\phi_i^{(n+1)} \ ,\end{aligned} \tag{6.36a}$$

$$\begin{aligned}J_{i+1/2}^{(n+1)} = -\frac{D_{i+1/2}^{(n+\frac{1}{2})}}{\Delta z_{i+1/2}}\big(\phi_{i+1}^{(n+1)} - \phi_i^{(n+1)}\big) \\ + \hat{D}_{i+1/2}^{(n+\frac{1}{2})}\big(\phi_i^{(n+1)} + \phi_{i+1}^{(n+1)}\big) \ ,\end{aligned} \tag{6.36b}$$

$$J_{1/2}^{(n+1)} = -B_{1/2}^{(n+\frac{1}{2})}\phi_1^{(n+1)} \ , \tag{6.36c}$$

$$J_{I+1/2}^{(n+1)} = B_{I+1/2}^{(n+\frac{1}{2})}\phi_I^{(n+1)} \ , \tag{6.36d}$$

$$P = \sum_{i'=1}^{I} \kappa\Sigma_{f,i'}^{(n+1)}\phi_{i'}^{(n+1)}\Delta z_{i'} \ , \tag{6.36e}$$

$$h_i^{(n+1)} = \kappa\Sigma_{f,i}^{(n+1)}\phi_i^{(n+1)} \ , \tag{6.36f}$$

$$\bar{\mathcal{L}}T_i^{(n+1)} = h_i^{(n+1)} \ , \tag{6.36g}$$

$$\begin{aligned}\Sigma_{u,i}^{(n+1)} = \Sigma_{u,0} + \Sigma_{u,1,d}\big(\sqrt{T_i^{(n+1)} + \beta h_i^{(n+1)}} - \sqrt{T_{d,0}}\big) \\ + \Sigma_{u,1,b}\big(T_i^{(n+1)} - T_{b,0}\big) \ .\end{aligned} \tag{6.36h}$$

After Eqs. (6.36) are solved (shortly, we outline a WS-PI-based iterative method to accomplish this), max iteration and convergence criteria are checked.

If either condition has been met, we are finished! Otherwise, we increase the outer iteration index n by one and return to X-CMFD Step 1.

The above X-CMFD outline is transformed into NILO-A by modifying Eq. (6.36g), repeated below for convenience. (No other equations are modified in moving from X-CMFD to NILO-A!):

$$\bar{\mathcal{L}} T_i^{(n+1)} = h_i^{(n+1)} \; .$$

X-CMFD's Eq. (6.36g) uses the high-order operator $\bar{\mathcal{L}}$ to update implicit temperatures $T_i^{(n+1)}$ in the nonlinear diffusion eigenvalue solve. In NILO-A, the $\bar{\mathcal{L}}$ inversion is avoided by instead using the approximate thermal hydraulics operator $\bar{L}$. Below, we write the NILO-A approximation to the X-CMFD implicit temperature update:

$$\begin{aligned} T_i^{(n+1)} &= T_i^{(n+\frac{1}{2})} + (T_i^{(n+1)} - T_i^{(n+\frac{1}{2})}) \\ &= T_i^{(n+\frac{1}{2})} + (\bar{\mathcal{L}}^{-1} h_i^{(n+1)} - \bar{\mathcal{L}}^{-1} h_i^{(n+\frac{1}{2})}) \\ &\approx T_i^{(n+\frac{1}{2})} + (\bar{L}^{-1} h_i^{(n+1)} - \bar{L}^{-1} h_i^{(n+\frac{1}{2})}) \; . \end{aligned}$$

This yields NILO-A's approximation to X-CMFD's Eq. (6.36g):

$$T_i^{(n+1)} = T_i^{(n+\frac{1}{2})} + (\bar{L}^{-1} h_i^{(n+1)} - \bar{L}^{-1} h_i^{(n+\frac{1}{2})}) \; . \tag{6.37}$$

By replacing X-CMFD's Eq. (6.36g) with Eq. (6.37), we obtain NILO-A. No other modifications are necessary.

NILO-A and NILO-CMFD use the same approximate thermal hydraulics update [Eq. (6.37)] to replace X-CMFD's Eq. (6.36g). Additionally, NILO-CMFD approximates the exact nuclear data update [Eq. (6.36h)] used by both X-CMFD and NILO-A, with an approximate update based on a first-order Taylor expansion, as follows:

$$\begin{aligned} \Sigma_{u,i}^{(n+1)} = \Sigma_{u,i}^{(n+\frac{1}{2})} &+ \dot{\Sigma}_{u,h,i}^{(n+\frac{1}{2})} (h_i^{(n+1)} - h_i^{(n+\frac{1}{2})}) \\ &+ \dot{\Sigma}_{u,T,i}^{(n+\frac{1}{2})} (T_i^{(n+1)} - T_i^{(n+\frac{1}{2})}) \; . \end{aligned} \tag{6.38}$$

Here, $\dot{\Sigma}_{u,h,i}^{(n+\frac{1}{2})}$ and $\dot{\Sigma}_{u,T,i}^{(n+\frac{1}{2})}$ are nuclear data partial derivative estimates with respect to the fission heat source h_i and bulk average fluid temperature T_i, respectively. These derivatives can be estimated using a finite-difference approximation, for example:

$$\dot{\Sigma}_{u,h,i}^{(n+\frac{1}{2})} = \frac{\Sigma_u(T_i^{(n+\frac{1}{2})}, (1+\epsilon)h_i^{(n+\frac{1}{2})}) - \Sigma_{u,i}^{(n+\frac{1}{2})}}{\epsilon h_i^{(n+\frac{1}{2})}} \,, \tag{6.39a}$$

$$\dot{\Sigma}_{u,T,i}^{(n+\frac{1}{2})} = \frac{\Sigma_u((1+\epsilon)T_i^{(n+\frac{1}{2})}, h_i^{(n+\frac{1}{2})}) - \Sigma_{u,i}^{(n+\frac{1}{2})}}{\epsilon T_i^{(n+\frac{1}{2})}} \,, \tag{6.39b}$$

where ϵ is a small ($\epsilon \ll 1$) scalar parameter. (In our numerical results, we used Eqs. (6.39) to estimate $\dot{\Sigma}_{u,h,i}^{(n+\frac{1}{2})}$ and $\dot{\Sigma}_{u,T,i}^{(n+\frac{1}{2})}$.)

Next, we summarize the equations describing each method. X-CMFD, NILO-A, and NILO-CMFD Steps 0 – 3 are identical to M-CMFD / R-CMFD Steps 0 – 3, without relaxation ($\alpha = 1$). X-CMFD, NILO-A, and NILO-CMFD differ in their definition of the Step 4 nonlinear diffusion eigenvalue problem. In X-CMFD Step 4, we solve the coupled set of Eqs. (6.36). In NILO-A Step 4, we solve the same set of Eqs. (6.36) as X-CMFD, but with the exact temperature update Eq. (6.36g) replaced by the approximate temperature update Eq. (6.37). In NILO-CMFD Step 4, we use X-CMFD's Eqs. (6.36), but replace both the exact temperature [Eq. (6.36g)] and nuclear data [Eq. (6.36h)] updates with approximate versions described by Eqs. (6.37) and (6.38), respectively. Additionally, NILO-CMFD requires estimates of the nuclear data partial derivatives $\dot{\Sigma}_{u,h,i}^{(n+\frac{1}{2})}$ and $\dot{\Sigma}_{u,T,i}^{(n+\frac{1}{2})}$ [e.g., using Eqs. (6.39)].

Nonlinear Wielandt Shifted Power Iteration (N-WS-PI)

In order to solve X-CMFD, NILO-A, and NILO-CMFD's nonlinear diffusion problem in Step 4, *inner iterations* are required. Below, we describe a Wielandt Shifted Power Iteration (WS-PI)-based inner iteration scheme for this purpose. We refer to this procedure as Nonlinear WS-PI (N-WS-PI). N-WS-PI is not the only approach available to solve a nonlinear diffusion eigenvalue problem, but it is the method we used to obtain our numerical results. We write the N-WS-PI outline below as applied to X-CMFD, NILO-A, and NILO-CMFD, making note in **bold** when the choice of equations depends on the outer iteration multiphysics iterative method being used:

N-WS-PI Step 0 : Initialization

Before starting inner iteration, we require initial estimates of the scalar flux $\phi_i^{(n+1,0)}$ (eigenfunction), eigenvalue $\lambda^{(n+1,0)}$, nuclear data $\Sigma_{u,i}^{(n+1,0)}$, and Wielandt shift $\lambda_s^{(n+1,0)}$, for example:

$$\phi_i^{(n+1,0)} = \phi_i^{(n+\frac{1}{2})} \ , \tag{6.40a}$$

$$\lambda^{(n+1,0)} = \lambda^{(n)} \ , \tag{6.40b}$$

$$\Sigma_{u,i}^{(n+1,0)} = \Sigma_{u,i}^{(n+\frac{1}{2})} \ , \tag{6.40c}$$

$$\lambda_s^{(n+1,0)} = 0 \ . \tag{6.40d}$$

We then set the inner iteration index p to zero, and proceed to N-WS-PI Step 1 to begin the first inner iteration.

N-WS-PI Step 1 : Thermal Hydraulics Update

We use the most recent estimates of the nuclear data $\Sigma_{u,i}^{(n+1,p)}$ and scalar fluxes $\phi_i^{(n+1,p)}$ to update the heat source $h_i^{(n+1,p+1)}$:

$$h_i^{(n+1,p+1)} = \kappa \Sigma_{f,i}^{(n+1,p)} \phi_i^{(n+1,p)} \ . \tag{6.41}$$

If using X-CMFD, we update temperatures $T_i^{(n+1,p+1)}$ with:

$$T_i^{(n+1,p+1)} = \bar{\mathcal{L}}^{-1} h_i^{(n+1,p+1)} \ . \tag{6.42}$$

If using NILO-A or NILO-CMFD, we update temperatures $T_i^{(n+1,p+1)}$ with:

$$T_i^{(n+1,p+1)} = T_i^{(n+\frac{1}{2})} + \left(\bar{L}^{-1} h_i^{(n+1,p+1)} - \bar{L}^{-1} h_i^{(n+\frac{1}{2})}\right) . \tag{6.43}$$

We emphasize that only one of Eqs. (6.42) and (6.43) is evaluated. The choice depends on the multiphysics iterative method being used in the surrounding outer iteration. For X-CMFD, we use Eq. (6.42). For NILO-A and NILO-CMFD, we use Eq. (6.43).

N-WS-PI Step 2 : Nuclear Data / Operator Update

If using X-CMFD or NILO-A, we update nuclear data $\Sigma_{u,i}^{(n+1,p+1)}$ with:

$$\begin{aligned} \Sigma_{u,i}^{(n+1,p+1)} = \Sigma_{u,0} &+ \Sigma_{u,1,d}\left(\sqrt{T_i^{(n+1,p+1)} + \beta h_i^{(n+1,p+1)}} - \sqrt{T_{d,0}}\right) \\ &+ \Sigma_{u,1,b}\left(T_i^{(n+1,p+1)} - T_{b,0}\right) . \end{aligned} \tag{6.44}$$

If using NILO-CMFD, we update nuclear data $\Sigma_{u,i}^{(n+1,p+1)}$ with:

$$\begin{aligned}\Sigma_{u,i}^{(n+1,p+1)} = \Sigma_{u,i}^{(n+\frac{1}{2})} &+ \dot{\Sigma}_{u,h,i}^{(n+\frac{1}{2})}(h_i^{(n+1,p+1)} - h_i^{(n+\frac{1}{2})}) \\ &+ \dot{\Sigma}_{u,T,i}^{(n+\frac{1}{2})}(T_i^{(n+1,p+1)} - T_i^{(n+\frac{1}{2})}) \ .\end{aligned} \tag{6.45}$$

Using $\Sigma_{u,i}^{(n+1,p+1)}$, we update the following operators. First, we update the (transport-corrected) diffusion leakage + absorption operator $M^{(n+1,p+1)}$:

$$\begin{aligned}M^{(n+1,p+1)}\phi_i \equiv \frac{1}{\Delta z_i}\Big[&- \frac{D_{i+1/2}^{(n+\frac{1}{2})}}{\Delta z_{i+1/2}}(\phi_{i+1} - \phi_i) + \hat{D}_{i+1/2}^{(n+\frac{1}{2})}\big(\phi_i + \phi_{i+1}\big) \\ &+ \frac{D_{i-1/2}^{(n+\frac{1}{2})}}{\Delta z_{i-1/2}}\big(\phi_i - \phi_{i-1}\big) - \hat{D}_{i-1/2}^{(n+\frac{1}{2})}\big(\phi_{i-1} + \phi_i\big)\Big] \\ &+ \Sigma_{a,i}^{(n+1,p+1)}\phi_i \ .\end{aligned} \tag{6.46a}$$

(As in the previous WS-PI outline, we only show the definition of $M^{(n+1,p+1)}$ for non-boundary-adjacent cells, for simplicity.) We also update the fission neutron production operator $F^{(n+1,p+1)}$:

$$F^{(n+1,p+1)}\phi_i \equiv \nu\Sigma_{f,i}^{(n+1,p+1)}\phi_i \ , \tag{6.46b}$$

and the fission heat generation operator $K^{(n+1,p+1)}$:

$$K^{(n+1,p+1)}\phi_i \equiv \kappa\Sigma_{f,i}^{(n+1,p+1)}\phi_i \ . \tag{6.46c}$$

We also define the following operator to represent integration over the problem domain:

$$\langle(\cdot)\rangle \equiv \sum_{i'=1}^{I}(\cdot)\Delta z_{i'} \ . \tag{6.47}$$

N-WS-PI Step 3 : Fixed Source Diffusion Solve

We update the (unnormalized) scalar flux estimate $\tilde{\phi}_i^{(n+1,p+1)}$ by solving the Wielandt-shifted, fixed-source diffusion problem:

$$\begin{aligned}(M^{(n+1,p+1)} - \lambda_s^{(n+1,p)}F^{(n+1,p+1)})\tilde{\phi}^{(n+1,p+1)} \\ = (\lambda^{(n+1,p)} - \lambda_s^{(n+1,p)})F^{(n+1,p+1)}\phi^{(n+1,p)} \ .\end{aligned} \tag{6.48}$$

N-WS-PI Step 4 : Eigenvalue Update & Renormalization

We update the eigenvalue estimate $\lambda^{(n+1,p+1)}$ with:

$$\lambda^{(n+1,p+1)} = \lambda_s^{(n+1,p)} + (\lambda^{(n+1,p)} - \lambda_s^{(n+1,p)}) \frac{\langle F^{(n+1,p+1)} \phi^{(n+1,p)} \rangle}{\langle F^{(n+1,p+1)} \tilde{\phi}^{(n+1,p+1)} \rangle} \ . \tag{6.49}$$

We update the Wielandt shift with:

$$\lambda_s^{(n+1,p+1)} = \max(r\lambda^{(n+1,p+1)} - c_1 |\lambda^{(n+1,p+1)} - \lambda^{(n+1,p)}|, \lambda_m) \ , \tag{6.50}$$

where r, c_1, and λ_m are user-defined constants. We then renormalize the end-of-inner-iteration scalar flux $\phi_i^{(n+1,p+1)}$:

$$\phi^{(n+1,p+1)} = \frac{P}{\langle K^{(n+1,p+1)} \tilde{\phi}^{(n+1,p+1)} \rangle} \tilde{\phi}^{(n+1,p+1)} \ . \tag{6.51}$$

Then, we check inner iteration convergence and max iteration criteria. If they are met, we set the concluding inner iteration index $P^{(n+1)} = (p+1)$ and continue to N-WS-PI : Finalization. Otherwise, we increment the inner iteration index p by one and return to N-WS-PI Step 1.

N-WS-PI : Finalization

Before exiting the inner iteration completely, we set estimates to be used in the next X-CMFD, NILO-A, or NILO-CMFD outer iteration:

$$\phi_i^{(n+1)} = \phi_i^{(n+1,P^{(n+1)})} \ , \tag{6.52a}$$

$$\lambda^{(n+1)} = \lambda^{(n+1,P^{(n+1)})} \ , \tag{6.52b}$$

$$\Sigma_{u,i}^{(n+1)} = \Sigma_{u,i}^{(n+1,P^{(n+1)})} \ . \tag{6.52c}$$

This concludes our discussion of N-WS-PI, the iterative method used in our test code to solve the nonlinear diffusion eigenvalue problem required by Step 4 in every X-CMFD, NILO-A, and NILO-CMFD outer iteration.

6.5. Discrete Model Problem Parameters

Before introducing M-CMFD, R-CMFD, X-CMFD, NILO-A, and NILO-CMFD discrete 1-D model numerical results, we prescribe the parameters necessary to fully specify a set of test problems. We obtained these parameters from [27], which includes problem specifications for a series of 3-D, multigroup, Light Water Reactor (LWR) core multiphysics benchmark problems. The parameters we used were obtained from straightforward, logical simplifications of the data in [27] (e.g., using an infinite-medium spectrum-weighted group collapse, etc.).

In the forthcoming numerical tests, we used 8 unique pin-cell types, each assigned their own nuclear data parameters ($\Sigma_{u,0}$, $\Sigma_{u,1,d}$, and $\Sigma_{u,1,b}$) used in the nuclear data representation specified by Eq. (6.18). We derived these parameters from assembly-homogenized information in [27]. When referring to a specific pin-cell type in the discrete model, we use the unique identifiers (i.d.'s) 4–11. These i.d.'s correspond to assembly i.d.'s from [27]. Since the discrete model is defined over a pin-cell, one can think of each i.d. as referring to a representative pin-cell from that i.d.'s corresponding assembly in [27]. Table (6.1) provides descriptions of the fuel-rod uranium-235 enrichment (in weight-percent (w/o)) and the number of Burnable Absorber (BA) rods in each assembly from [27].

Table 6.1: Assembly Compositions.

i.d.	composition
4	2.1 w/o (U-235)
5	2.6 w/o
6	3.1 w/o
7	2.6 w/o, 12 BA
8	2.6 w/o 16 BA
9	2.6 w/o 20 BA
10	3.1 w/o 12 BA
11	3.1 w/o 16 BA

In Table (6.2), we list the $\Sigma_{u,0}$, $\Sigma_{u,1,d}$, and $\Sigma_{u,1,b}$ values used in the discrete 1-D model nuclear data representation [Eq. (6.18)] for Σ_t, Σ_s, $\nu\Sigma_f$, $\kappa\Sigma_f$. This data is listed for each pin-cell i.d. 4–11.

Table 6.2: Nuclear Data Parameters.

i.d.	4	5	6	7	8	9	10	11
$\Sigma_{t,0}$	3.483e-01	3.355e-01	3.253e-01	3.195e-01	3.147e-01	3.102e-01	3.112e-01	3.070e-01
$\Sigma_{t,1,d}$	-5.417e-05	-4.812e-05	-4.386e-05	-4.499e-05	-4.415e-05	-4.332e-05	-4.167e-05	-4.109e-05
$\Sigma_{t,1,b}$	-7.418e-04	-6.949e-04	-6.579e-04	-6.221e-04	-6.000e-04	-5.789e-04	-5.925e-04	-5.726e-04
$\Sigma_{s,0}$	3.272e-01	3.138e-01	3.031e-01	2.977e-01	2.929e-01	2.884e-01	2.891e-01	2.849e-01
$\Sigma_{s,1,d}$	-7.337e-05	-6.878e-05	-6.580e-05	-6.709e-05	-6.663e-05	-6.620e-05	-6.489e-05	-6.464e-05
$\Sigma_{s,1,b}$	-7.201e-04	-6.731e-04	-6.361e-04	-6.016e-04	-5.798e-04	-5.590e-04	-5.718e-04	-5.521e-04
$\nu\Sigma_{f,0}$	2.222e-02	2.435e-02	2.607e-02	2.200e-02	2.127e-02	2.056e-02	2.368e-02	2.293e-02
$\nu\Sigma_{f,1,d}$	-1.180e-05	-1.124e-05	-1.044e-05	-9.488e-06	-8.956e-06	-8.437e-06	-8.794e-06	-8.299e-06
$\nu\Sigma_{f,1,b}$	-2.261e-05	-2.456e-05	-2.604e-05	-2.293e-05	-2.236e-05	-2.178e-05	-2.439e-05	-2.382e-05
$\kappa\Sigma_{f,0}$	2.882e-13	3.161e-13	3.387e-13	2.853e-13	2.756e-13	2.663e-13	3.072e-13	2.973e-13
$\kappa\Sigma_{f,1,d}$	-1.560e-16	-1.486e-16	-1.380e-16	-1.250e-16	-1.186e-16	-1.118e-16	-1.158e-16	-1.100e-16
$\kappa\Sigma_{f,1,b}$	-2.938e-16	-3.193e-16	-3.387e-16	-2.975e-16	-2.904e-16	-2.828e-16	-3.166e-16	-3.094e-16

The centered Doppler temperature $T_{d,0}$ and centered bulk average fluid temperature $T_{b,0}$ values in Eq. (6.18) are constant among all i.d.'s. Their values are listed in Table (6.3).

Table 6.3: Constant Nuclear Data Parameters.

parameter	value
$T_{d,0}$	618.3 [°C]
$T_{b,0}$	306.6 [°C]

Additional parameters are given in Table (6.4).

Table 6.4: Additional Model Parameters.

parameter	value
Z	367.3 [cm]
P	66951.4 [W]
$\dot{m}$	0.311 [$\mathrm{kg \cdot s^{-1}}$]
T_{in}	286 [°C]
β	1.296 [$\mathrm{cm \cdot K \cdot W^{-1}}$]
κ	$3.206 \cdot 10^{-11}$ [$\mathrm{J \cdot fiss.^{-1}}$]
c_p	5645.84 [$\mathrm{J \cdot kg^{-1} \cdot K^{-1}}$]

With Tables (6.2)-(6.4) defined, all that remains to fully specify a discrete model problem is a chosen pin-cell i.d. (4–11), an angular quadrature set (μ_m, w_m), $1 \leq m \leq M$), a spatial mesh [Fig. (6.1)], and a value for the fluid turbulent diffusion coefficient D_t [Eqs. (6.15)]. We cover these parameters when discussing numerical results in the next section.

6.6. Numerical Results

We have developed a neutron transport – thermal hydraulics multiphysics test code to compare the relative performance of M-CMFD, R-CMFD, X-CMFD, NILO-A, and NILO-CMFD on the discrete 1-D model. A derivation of the discrete 1-D model, outlines of each iterative method (including descriptions of the inner iteration schemes we have implemented in our test code), and tables of problem parameter values are given in earlier sections.

We split our numerical results into three subsections. First, we explore the solutions of a single-physics version of the discrete 1-D model. In this version of the model, feedback is turned off by setting all $\Sigma_{u,1,d}$ and $\Sigma_{u,1,b}$ terms to zero. Next, we reintroduce feedback into the discrete 1-D model and see how the model solutions change as compared to the single-physics cases. The multiphysics solutions obtained from the 1-D model in Ch. 6 qualitatively match with 3-D reactor core results simulated with MPACT in Ch. 7. This instills some confidence that the 1-D model is capturing qualitatively realistic reactor multiphysics behavior. Finally, we explore the iterative performance of M-CMFD, R-CMFD, X-CMFD, NILO-A, and NILO-CMFD on the multiphysics version of the discrete 1-D model.

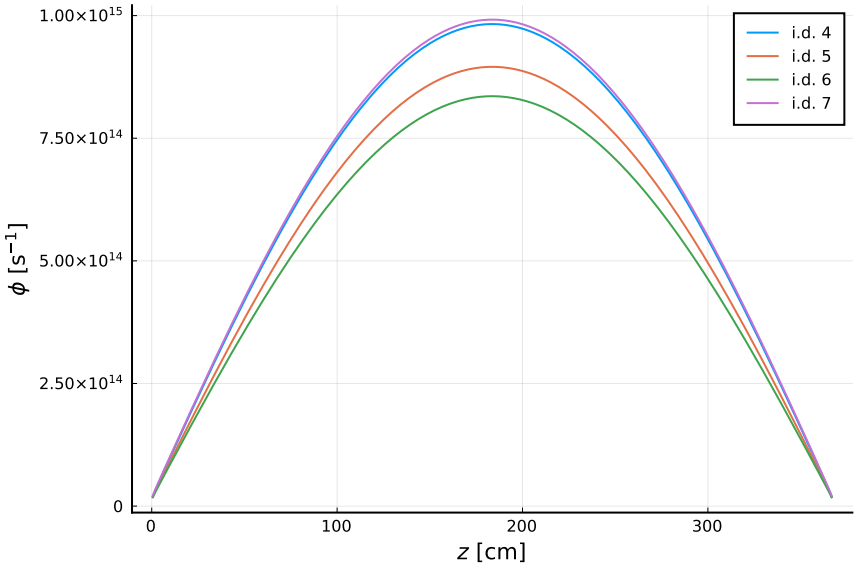
6.6.1. Single-Physics Model Solutions

Next, we remove feedback from the discrete 1-D model by setting the coefficients $\Sigma_{u,1,d}$ and $\Sigma_{u,1,b}$ in Table (6.2) to zero. All other model equations and coefficients are untouched. We refer to this modified version of the discrete 1-D model as the "single-physics model".

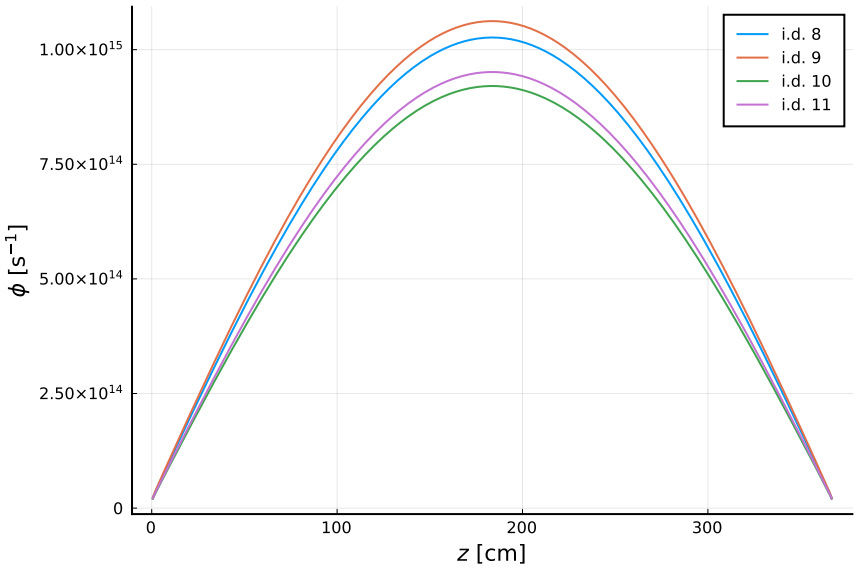
Figs. (6.2a) and (6.2b) plot single-physics model scalar flux $\phi(z)$ solution curves for pin-cell i.d.'s 4–7 and 8-11, respectively. (We note that the units on the scalar flux [s^{-1}] in Figs. (6.2) are nonstandard, owing to simplifications made when deriving the 1-D model in Ch. 5.)

In Figs. (6.2), all scalar flux solutions have a similar, cosine-like shape and are symmetric about the middle-height of the pin-cell ($z = 183.65$ [cm]). Since $\Sigma_{u,1,d}$ and $\Sigma_{u,1,b}$ are set to zero in Eq. (6.18) in the single-physics model, nuclear data Σ_u is spatially homogeneous at $\Sigma_{u,0}$ over the problem domain. That is, the scalar flux solutions in Figs. (6.2) are those of bare, homogeneous slabs. The noticeable differences in scalar flux solution magnitudes is due to the eigenfunction normalization condition in Eq. (6.9d), coupled with the fact that each pin-cell i.d. has a unique value

of $\kappa\Sigma_{f,0}$.



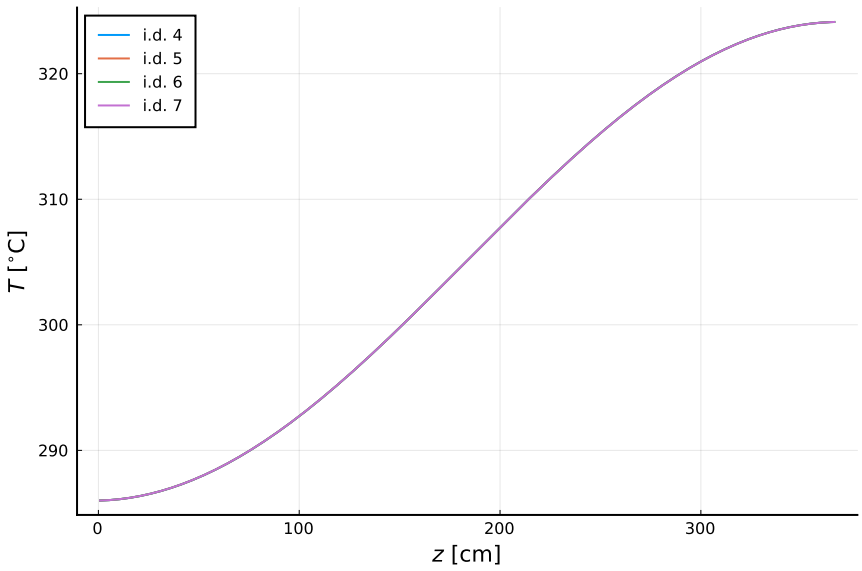
(a) i.d. = 4–7



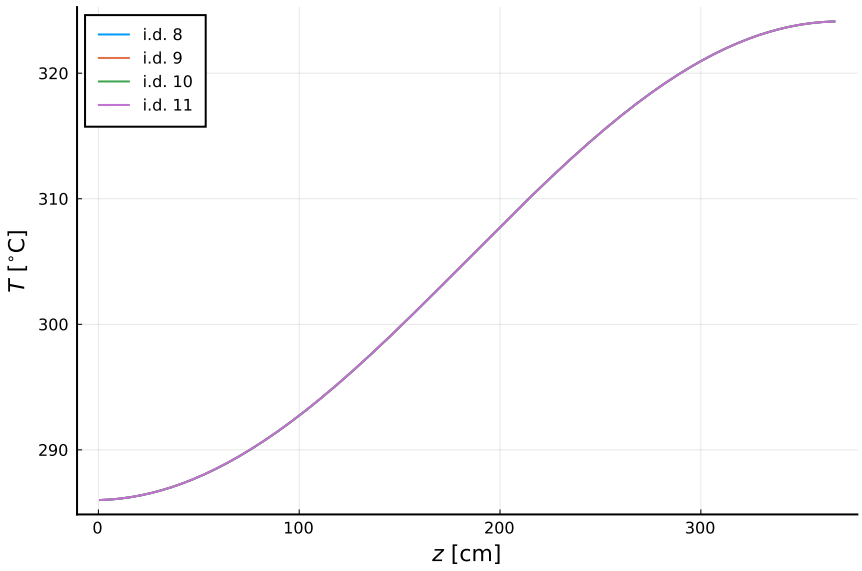
(b) i.d. = 8–11

Figure 6.2: Single-Physics Model Scalar Flux Solutions.

Figs. (6.3a) and (6.3b) show the bulk average fluid temperature $T(z)$ solutions in the single-physics model for pin-cell i.d.'s 4–7 and 8–11, respectively. The curves in Figs. (6.3) are visually indistinguishable from one another. (This will not be the case when multiphysics feedback is reintroduced into the model.) In each case, the fluid temperature monotonically increases as one moves up the core.



(a) i.d. = 4–7

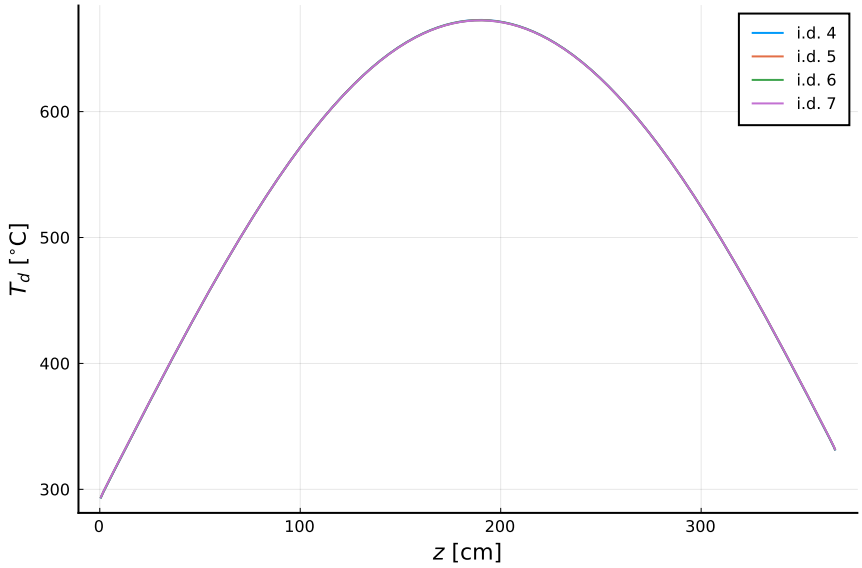


(b) i.d. = 8–11

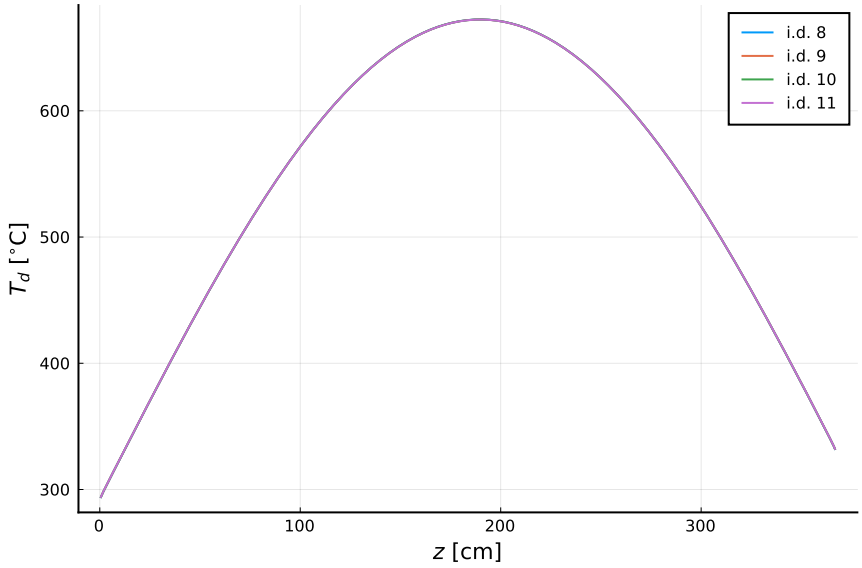
Figure 6.3: Single-Physics Model Fluid Temperature Solutions.

In Figs. (6.4a) and (6.4b), we show the single-physics model Doppler temperature $T_d(z) \equiv T(z) + \beta h(z)$ solutions for pin-cell i.d.'s 4–7 and 8-11, respectively. As with the fluid temperature $T(z)$ solutions in Figs. (6.3), the Doppler temperature $T_d(z)$ curves in Figs. (6.4) are visually indistinguishable between different pin-cell i.d.'s. (Again, this will not be the case in the multiphysics 1-D model.) The Doppler temperature solution shape follows closely to the scalar flux solution shape in Figs.

(6.2), owing to the tight coupling from scalar flux to fission heat source. However, unlike the scalar flux solutions in Figs. (6.2), Doppler temperature curves are similar in magnitude due to identical thermal power normalization. Additionally, Doppler temperature solutions are asymmetric due to the fluid temperature increase when moving upwards through the pin-cell [Figs. (6.3)].



(a) i.d. = 4–7



(b) i.d. = 8–11

Figure 6.4: Single-Physics Model Doppler Temperature Solutions.

Table (6.5) shows single-physics model λ-eigenvalue solutions for each pin-cell i.d.

Table 6.5: Single-Physics Model Eigenvalue Solutions.

i.d.	4	5	6	7	8	9	10	11
λ	0.951	0.892	0.852	0.992	1.027	1.062	0.939	0.970

All solutions in Figs. (6.2)-(6.4) and Table (6.5) were obtained from simulations run with an S_{16} Gauss-Legendre quadrature set, 500 equal width spatial cells, and a fluid turbulent diffusion coefficient $D_t = 10^2$. Additionally, each simulation was deemed converged when the infinity norm of the relative error between respective outer-iteration scalar flux and eigenvalue estimates (i.e. the "relative delta") dropped below $1 \cdot 10^{-8}$. Each simulation was run with M-CMFD (i.e. R-CMFD without relaxation; $\alpha = 1$). In Table (6.6), we show the numerical estimate of the spectral radius ρ and the iterations to convergence N for each single-physics model case. Since M-CMFD is equivalent to standard CMFD when applied to a single-phy

Table 6.6: Single-Physics Model Spectral Radii (ρ) and Iterations (N).

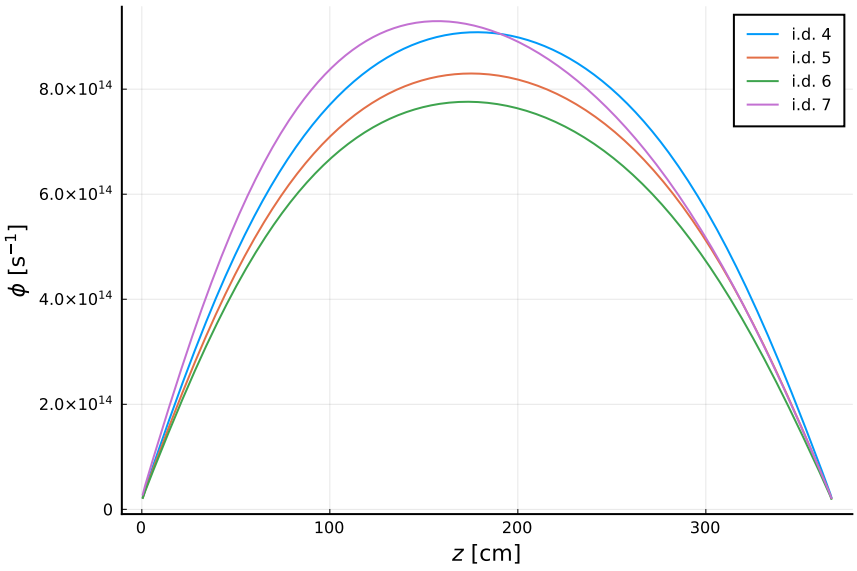
i.d. \ ρ (N)	M-CMFD
4	0.184 (12)
5	0.188 (13)
6	0.190 (13)
7	0.190 (13)
8	0.191 (13)
9	0.191 (13)
10	0.191 (13)
11	0.192 (13)

Next, we reintroduce feedback into the 1-D model and observe how the solution fields change from those of the single-physics model.

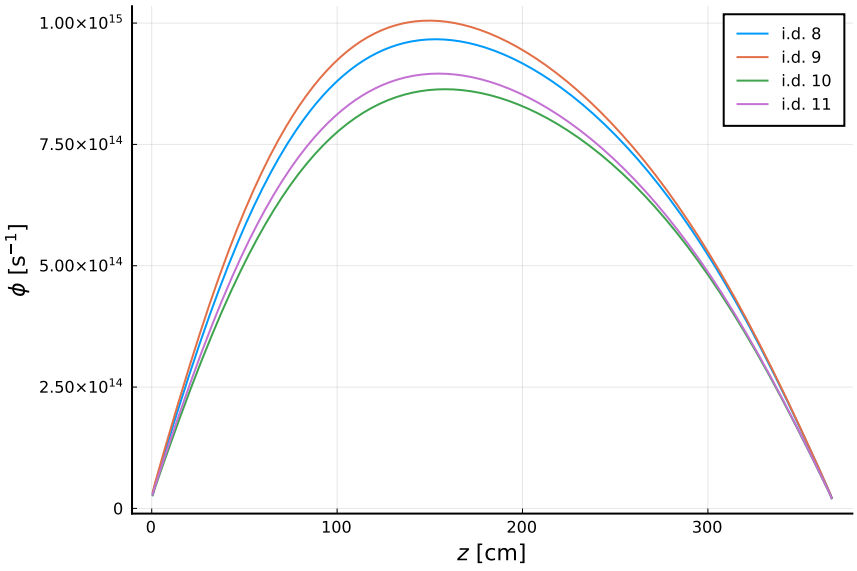
6.6.2. (Multiphysics) Discrete Model Solutions

By reintroducing the nonzero (but relatively small, $O(10^{-6})$–$O(10^{-4})$) $\Sigma_{u,1,d}$ and $\Sigma_{u,1,b}$ coefficients in Table (6.2), we move from the single-physics model back to the *multiphysics* "discrete 1-D model". In Figs. (6.5)-(6.7), we plot the discrete model scalar flux $\phi(z)$ [Fig. (6.5)], bulk average fluid temperature $T(z)$ [Fig. (6.6)], and fuel Doppler temperature $T_d(z) \equiv T(z) + \beta h(z)$ [Fig. (6.7)] solutions for each pin-cell i.d. 4–11 in the discrete 1-D model. Figs. (6.5)-(6.7) solutions were obtained using an S_{16} Gauss-Legendre angular quadrature set, a spatial mesh with 500 equal-width

cells, and a fluid turbulent diffusion coefficient $D_t = 10^2$. (For reference, using the values in Table (6.4), the advection coefficient $\dot{m}c_p$ in the fluid advection-diffusion Eqs. (6.15) is ≈ 1756. In these problems, the fluid temperature T is primarily driven by advection.) All other model parameters are set according to Tables (6.2)-(6.4).

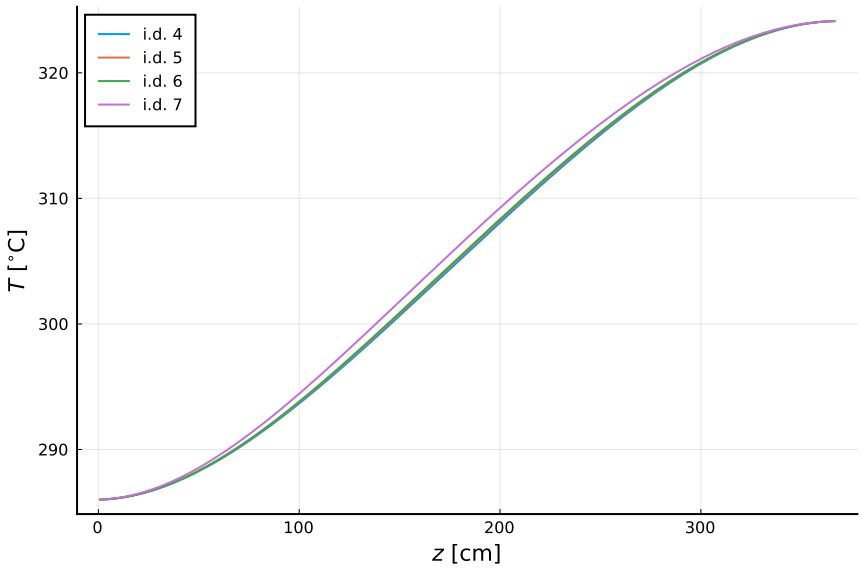


(a) i.d. = 4–7

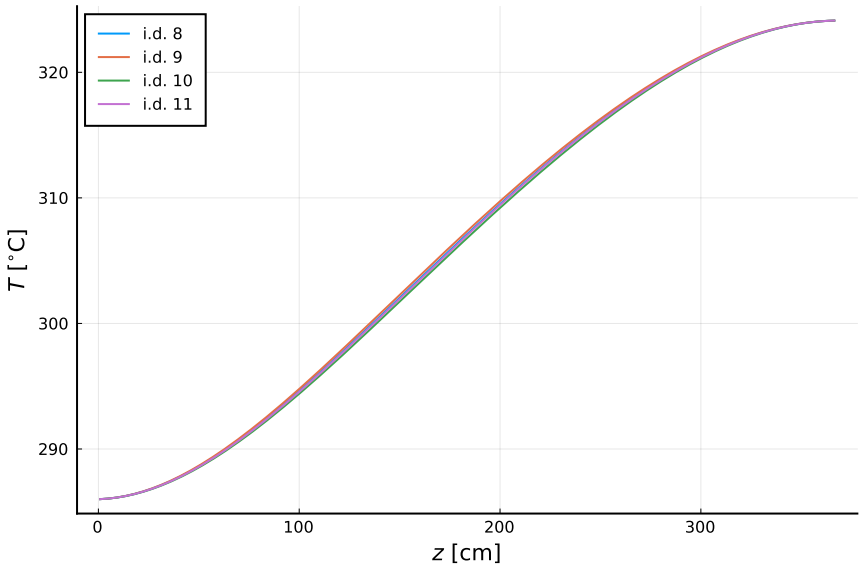


(b) i.d. = 8–11

Figure 6.5: Discrete 1-D Model Scalar Flux Solutions.

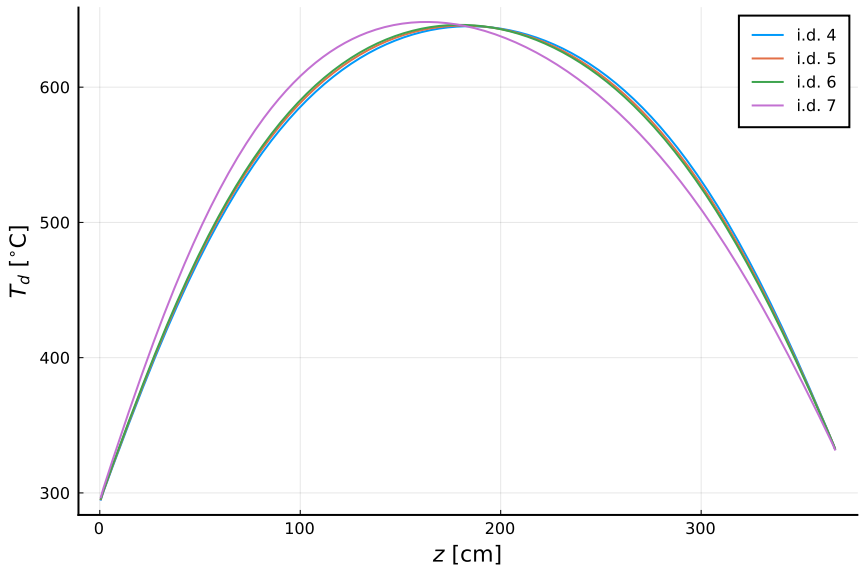


(a) i.d. = 4–7

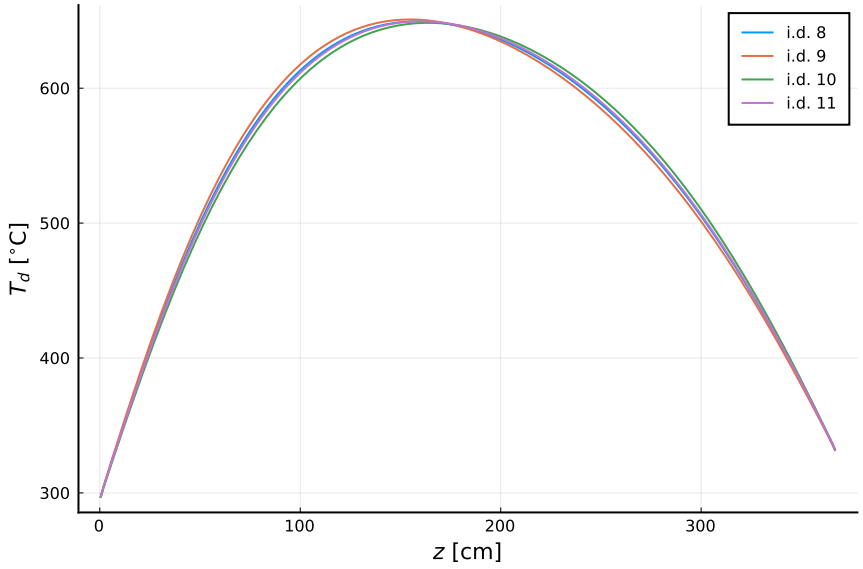


(b) i.d. = 8–11

Figure 6.6: Discrete 1-D Model Fluid Temperature Solutions.



(a) i.d. = 4–7



(b) i.d. = 8–11

Figure 6.7: Discrete 1-D Model Doppler Temperature Solutions.

Next, we discuss the multiphysics solutions in Figs. (6.5)-(6.7). We note that these figures can be directly compared to the single-physics model's Figs. (6.2)-(6.4). Figs. (6.5a) and (6.5b) show the axial-dependent scalar flux solution $\phi(z)$ for pin-cell i.d.'s 4–7 and 8–11, respectively. We note that the units of the scalar flux $\phi(z)$ in Figs. (6.5) are [s^{-1}]. In a 3-D problem, the energy-integrated scalar flux $\phi(\boldsymbol{r})$ describes the rate of neutron path-length generation per unit volume. The 3-D scalar flux $\phi(\boldsymbol{r})$

is traditionally written in units of $[\text{cm}^{-2} \cdot \text{s}^{-1}]$, although, it may be more physically representative to instead write it in $[\text{cm} \cdot \text{cm}^{-3} \cdot \text{s}^{-1}]$. Since the discrete (and continuous) models are radially- and energy- integrated, the scalar flux $\phi(z)$ describes the rate of neutron path length generation per unit axial thickness in the pin-cell. This can be written in units of $[\text{cm} \cdot \text{cm}^{-1} \cdot \text{s}^{-1}]$, or, equivalently, in $[\text{s}^{-1}]$.

Next, we comment on the shape and magnitude of the scalar flux $\phi(z)$ solutions in Figs. (6.5). Each scalar flux curve in these figures has a "cosine-like" shape. This is expected. The analytic, single-physics, eigenvalue, *neutron diffusion* scalar flux solution in a bare, homogeneous slab with Marshak vacuum boundary conditions is a scaled and shifted cosine. Of course, the scalar flux solutions in the inhomogeneous (owing to thermal hydraulics feedback), neutron transport, discrete model problem shown in Figs. (6.5) are not perfect cosines, since the discrete model is not a homogeneous, purely diffusive slab. Even though the discrete model simulates transport physics, not diffusion, its transport solution is diffusion-like, since the system is optically thick and the neutron scattering ratio $c \equiv \Sigma_s / \Sigma_t$ is (somewhat) high everywhere (≈ 0.94). (The effective scattering ratio is near unity.) Also, the axial dependence of nuclear data in the discrete model is weak. This can be understood by observing the $O(10^{-6})$–$O(10^{-4})$ $\Sigma_{u,1,d}$ and $\Sigma_{u,1,b}$ magnitudes for Σ_t, Σ_s, and $\nu\Sigma_f$ in Table (6.2). Owing to the weak dependence of nuclear data on temperature, the discrete model is somewhat homogeneous. This is all to say that we expect the curves in Eqs. (6.5a)-(6.5b) to be somewhat cosine-like, which they are.

The Figs. (6.5) scalar flux curves' departure from a pure cosine is a product of both the use of transport physics and the presence of nonlinear feedback in the discrete model. All scalar curves in Figs. (6.5) are somewhat bottom-peaked (towards $z = 0$). This is representative of the scalar flux solution in a 3-D, reactor core, multiphysics simulation. (When we explore higher-fidelity simulations in Ch. 7 with the MPACT code, we will observe the same behavior.) We note that pin-cell i.d.'s 7–11 are somewhat more bottom-peaked than pin-cell i.d.'s 4–6. This split in behavior aligns with the split between pin-cell's representative of assemblies in [27] *with* (i.d.'s 7 – 11) and *without* (i.d.'s 4 – 6) Burnable Absorber (BA) rods in them [Table (6.1)].

The scalar flux solution magnitudes in Figs. (6.5) are different between pin-cell i.d.'s. The fission heat generation cross section $\kappa\Sigma_f$ parameters $\kappa\Sigma_{f,0}$, $\kappa\Sigma_{f,1,d}$, and $\kappa\Sigma_{f,1,b}$ change between pin-cell i.d.'s in Table (6.2). Since the discrete model uses a total system power normalization condition [Eq. (6.9d)], pin-cell i.d.'s with higher average $\kappa\Sigma_f$ values tend to require lower scalar fluxes ϕ to achieve the same total

system thermal power P.

In Figs. (6.6), we plot the bulk average fluid temperature $T(z)$ solution curves for each pin-cell i.d. The T solutions are similar between i.d.'s and the curves tend to lie on-top of one another. Fluid temperatures T monotonically increase with axial height z, as expected. (As the fluid moves up the core, it incrementally gains heat from fuel rods.)

Figs. (6.7) show the fuel Doppler temperature solutions $T_d(z)$. The general shapes of the Doppler temperature curves in Figs. (6.7) follow closely to the corresponding scalar flux shapes in Figs. (6.5). This is because the Doppler temperature T_d is strongly dependent on the local scalar flux ϕ solution through the relationship $T_d(z) \equiv T(z) + \beta h(z)$ with $h(z) \equiv \kappa \Sigma_f(z)\phi(z)$.

Table (6.7) gives λ-eigenvalue solutions for each pin-cell i.d. The relative differences between eigenvalues in Table (6.7) match with the representative assembly compositions (^{235}U enrichment, burnable absorbers) from [27] described in Table (6.1). As a reminder, the k-eigenvalue can be written in terms of the λ-eigenvalue using the definition $\lambda \equiv 1/k$. $\lambda < 1$, $\lambda = 1$, $\lambda > 1$ correspond to $k > 1$, $k = 1$, and $k < 1$, and supercritical, critical, and subcritical systems, respectively.

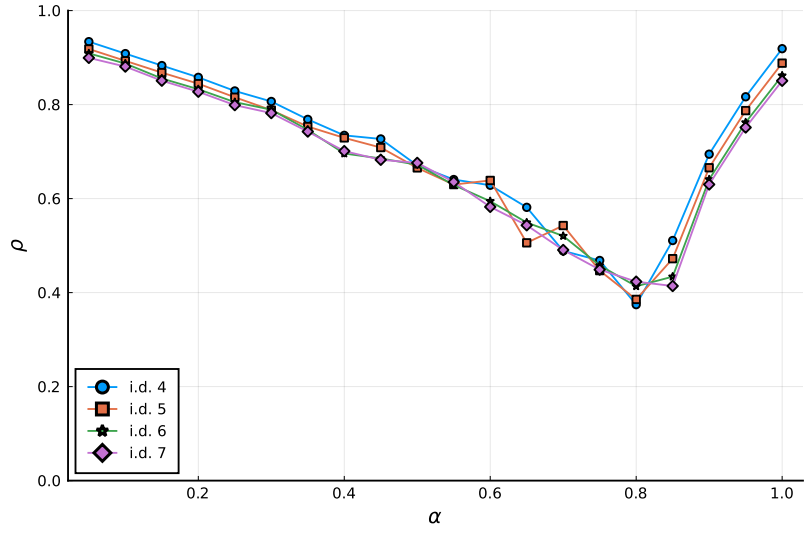
Table 6.7: Discrete 1-D Model Eigenvalue Solutions.

i.d.	4	5	6	7	8	9	10	11
λ	0.951	0.892	0.852	0.991	1.026	1.062	0.939	0.969

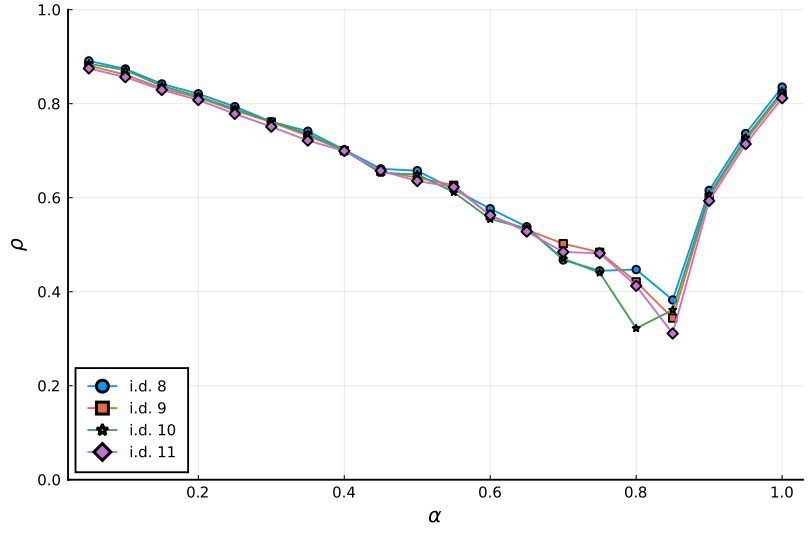
By comparing the single-physics [Figs. (6.2)-(6.4)] to the multiphysics [Figs. (6.5)-(6.7)] 1-D model solutions, we observe noticeable differences after the re-introduction of feedback. In the single-physics Figs. (6.2), all scalar flux solutions are symmetric, whereas in the multiphysics Figs. (6.5), each curve is asymmetric (bottom-peaked). Comparing the single-physics [Figs. (6.3)-(6.4)] and multiphysics [Figs. (6.6)-(6.7)] fluid and Doppler temperature solutions, we see that differences in these fields are visually indistinguishable between pin-cell i.d.'s in the single-physics model, while multiphysics solutions are noticeably different among pin-cell i.d.'s, especially when comparing those representative of assemblies with and without burnable absorbers [Table (6.1)].

6.6.3. Multiphysics Iterative Performance

To highlight the deficiencies of M-CMFD and R-CMFD, we ran a series of R-CMFD simulations for each pin-cell i.d. 4–11 with various values of the R-CMFD relaxation factor α. The relaxation factor α relaxes the fission heat source supplied to the high-order thermal hydraulics (advection-diffusion, $\mathcal{L}$) solver in R-CMFD Step 2 [Eqs. (6.21)]. $\alpha = 0$ corresponds to full relaxation. $\alpha = 1$ corresponds to no relaxation. Without relaxation ($\alpha = 1$), M-CMFD and R-CMFD are equivalent. Therefore, R-CMFD simulations that use $\alpha = 1$ constitute M-CMFD simulations.



(a) i.d. = 4–7



(b) i.d. = 8–11

Figure 6.8: R-CMFD Spectral Radius vs. Relaxation Factor.

In Figs. (6.8), we plot numerical estimates of the R-CMFD outer iteration spectral radius ρ as a function of relaxation factor α. As with the results in Figs. (6.5)-(6.7), Fig. (6.8) results were obtained using an S_{16} Gauss-Legendre angular quadrature set, $I = 500$ equal-width spatial cells, and $D_t = 10^2$. The spectral radius ρ is a measure of the convergence rate of the slowest decaying (i.e. limiting) error mode in a problem. A spectral radius $\rho \geq 1$ indicates a divergent simulation. $\rho = 0$ indicates instant convergence. The closer the spectral radius ρ is to 1 (while remaining less than 1), the more slowly converging the simulation. CMFD achieves a spectral radius $\rho \approx 0.22$ when applied to a single-physics transport problem, under nominal conditions. With reasonable convergence criteria, this usually corresponds to convergence in approximately 10–20 iterations. In Table (6.6), we observe this expected behavior for the single-physics model. When CMFD is naively applied to multiphysics transport problems using the M-CMFD and R-CMFD approaches, sub-optimal convergence is usually observed. Such behavior appears in Figs. (6.8).

In Figs. (6.8), no R-CMFD curve drops to the ideal pure-transport CMFD spectral radius $\rho \approx 0.22$. Every simulation in Figs. (6.8) converges sub-optimally, regardless of relaxation factor α. Thankfully, no R-CMFD simulation in Figs. (6.8) diverges, since $\rho < 1$ everywhere. However, as the relaxation factor α approaches either 0 or 1, the spectral radius ρ reaches towards 1. For these extreme values of α, R-CMFD runs took $O(10^2)$ outer iterations to converge, requiring as many transport sweeps, nuclear data updates, and high-order thermal hydraulics solves. We note that the "V-like" curves in Figs. (6.8) are similar to theoretical, Fourier analysis spectral radius results obtained using a proxy, flux-dependent feedback model in [50].

In Figs. (6.8), we see that the spectral radius ρ vs. relaxation factor α shape does not change significantly as we modify the pin-cell i.d. (i.e. nuclear data parameters in Table (6.2)). For the discrete model problems tested, the optimal R-CMFD relaxation factor α appears fixed around $\alpha = 0.8$. This similarity in optimal relaxation factor α is not indicative of R-CMFD behavior in general. Generally, the optimal R-CMFD relaxation factor α changes from problem to problem. We also note that even if an optimal relaxation factor α is chosen in Figs. (6.8), sub-optimal convergence is still observed.

In Table (6.8), we list representative R-CMFD spectral radii ρ from Figs. (6.8), with relaxation factor $\alpha = 0.7$. In addition, the same problems were run with X-CMFD, NILO-A, and NILO-CMFD. We show their spectral radii ρ in Table (6.8) as well. (For the NILO-CMFD simulations, the perturbation scalar $\epsilon = 1 \cdot 10^{-2}$ in

Eqs. (6.39)). We experimented with several small values of ϵ and saw little change in performance. We also give the outer iterations to convergence N for each case. In each case, a problem was considered converged when the infinity-norm of the relative error between iterations (i.e. the "relative delta") of every solution quantity in the problem (ϕ, λ, T, Σ_u) dropped below $1 \cdot 10^{-8}$. The X-CMFD, NILO-A, and NILO-CMFD spectral radii ρ remain nearly constant as the pin-cell i.d. is varied. More importantly, the X-CMFD, NILO-A, and NILO-CMFD spectral radii ρ are near the designated optimal values $\rho \approx 0.22$ obtained when applying CMFD to a single-physics transport problem. Table (6.8) indicates that for the discrete 1-D model, X-CMFD, NILO-A, and NILO-CMFD converge at the optimal rate.

Table 6.8: Discrete Model Spec. Radii (ρ) and Iterations (N) (R-CMFD $\alpha = 0.7$).

i.d. \ ρ (N)	R-CMFD	X-CMFD	NILO-A	NILO-CMFD
4	0.489 (28)	0.184 (12)	0.184 (12)	0.184 (12)
5	0.543 (31)	0.186 (12)	0.186 (12)	0.186 (12)
6	0.520 (30)	0.189 (13)	0.189 (13)	0.189 (13)
7	0.491 (27)	0.190 (13)	0.190 (13)	0.190 (13)
8	0.467 (27)	0.191 (13)	0.191 (13)	0.191 (13)
9	0.502 (30)	0.191 (13)	0.191 (13)	0.191 (13)
10	0.470 (27)	0.191 (13)	0.191 (13)	0.191 (13)
11	0.485 (29)	0.192 (13)	0.192 (13)	0.192 (13)

In Fig. (6.9), we show spectral radii ρ for a slightly modified version of the discrete model. In this "modified model", we vary the system thickness Z while keeping the total system thermal power P constant. Besides the system thickness Z, all other parameters in the modified model are consistent with Tables (6.2)-(6.4). Fig. (6.9) shows the R-CMFD spectral radius ρ on the modified model vs. problem thickness Z for two representative pin-cell i.d.'s (4 and 8) and three representative relaxation factors, $\alpha = 0.5$, 0.7, and 1.0. We also show X-CMFD and NILO-A curves for the same pin-cell i.d.'s. (All X-CMFD and NILO-A curves lie on top of one another, constant at $\rho \approx 0.22$, and cannot be visually distinguished.) Fig. (6.9) simulations used an S_{16} Gauss-Legendre angular quadrature set, $D_t = 10^2$, and spatial meshes with constant cell thickness $\Delta z_i = \Delta z = 0.25$ [cm]. In Fig. (6.9), we see that as the modified model problem thickness Z is increased, the R-CMFD spectral radius increases. This is the case for all pin-cell i.d.'s and relaxation factors α shown. We note that for simulations that did not converge within 200 outer iterations, we did not display the corresponding points in Fig. (6.9).

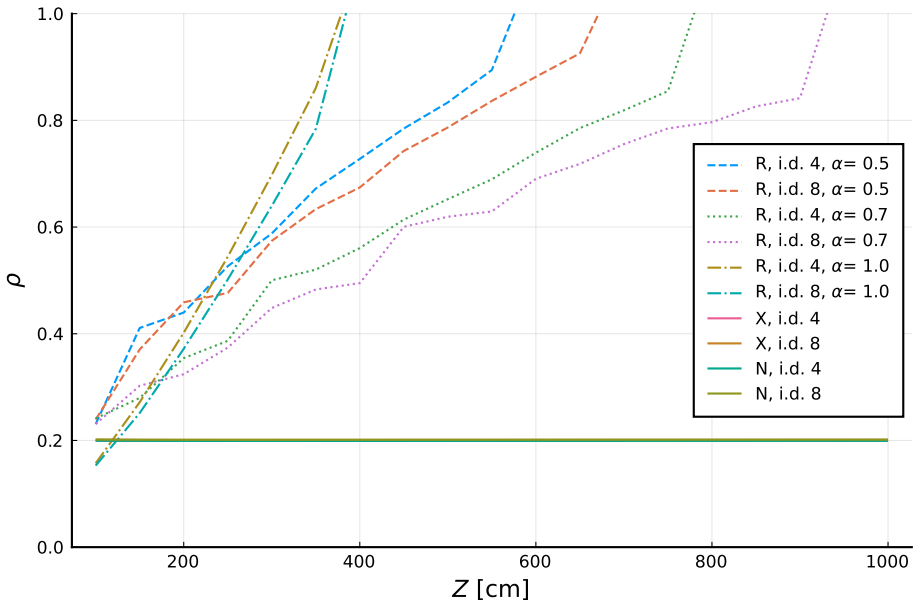
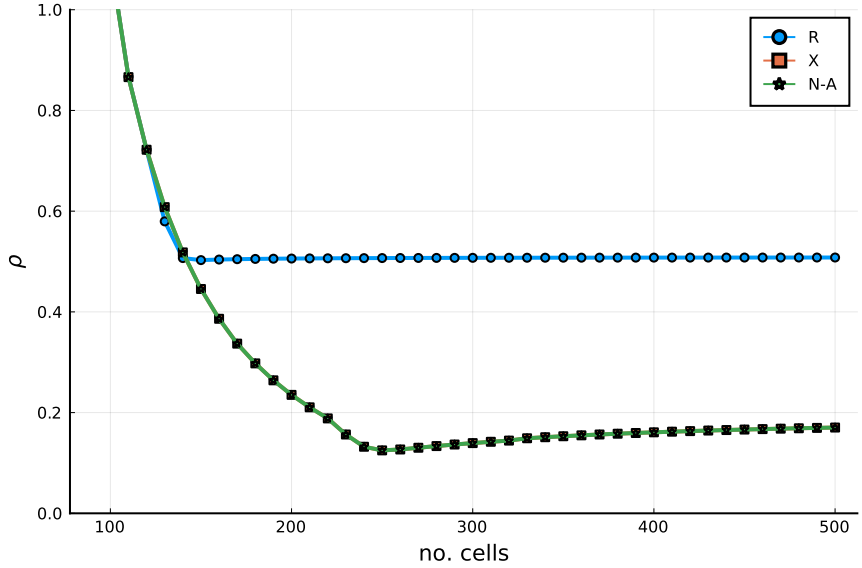


Figure 6.9: Modified Model Spectral Rad. vs. Axial Thickness (N ⇒ NILO-A).

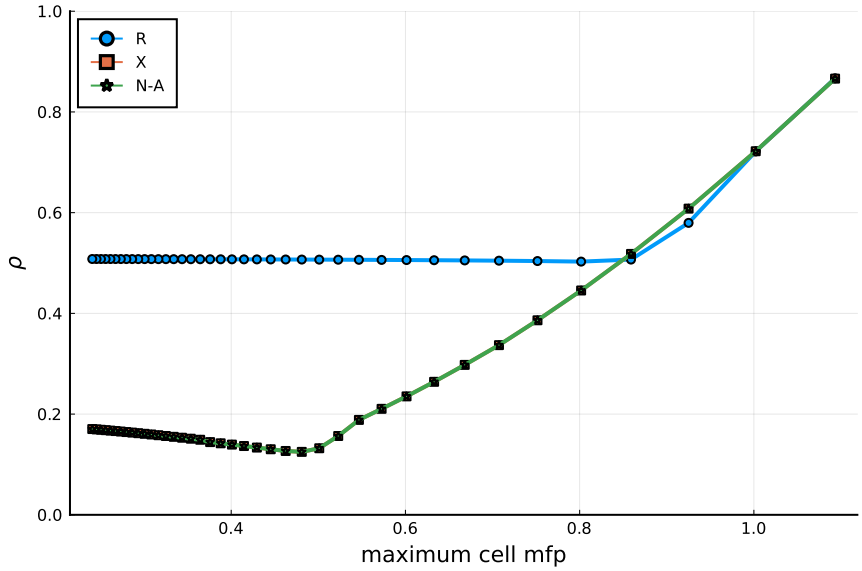
The standard Coarse Mesh Finite Difference (CMFD) method is known to suffer stability issues when mesh cells become optically thick ($\gtrapprox$ 1 mean-free-paths thick). In CMFD, these mesh-related instabilities can be addressed by using modified procedures such as partial-current based CMFD (pCMFD) [14, 15], optimally diffusive CMFD (odCMFD) [126] and linear prolongation-based CMFD (lpCMFD) [114]. In our work, we did not use any of these methods. We used standard CMFD, and so, we expect all methods (M-CMFD, R-CMFD, X-CMFD, NILO-A, and NILO-CMFD) to degrade in performance as mesh cells become optically thick. This instability is distinct from the multiphysics feedback-related instabilities of primary focus to our work.

To confirm that mesh-related instabilities persist in these multiphysics methods, we ran a series of R-CMFD ($\alpha = 0.7$), X-CMFD, and NILO-A discrete model simulations for pin-cell i.d. 8. We used an S_{16} Gauss Legendre quadrature set and $D_t = 1 \cdot 10^2$. We varied the number I of equal-width spatial cells used to discretize the domain. The spectral radius ρ versus the number of spatial cells I for each method is plotted in Fig. (6.10a). Fig. (6.10b) plots the same data, but shows the spectral radius ρ as a function of the mean-free-path thickness of the largest cell in a given problem. Figs. (6.10) confirm that each method becomes less stable as mesh cells become thicker. We note that the X-CMFD and NILO-A curves approach $\rho \approx 0.22$ when cells become optically thin, while the R-CMFD curve approaches a different, sub-optimal value. Even with a finely discretized mesh, multiphysics-related instabilities

hinder R-CMFD's performance.



(a) vs. Number of Cells I.



(b) vs. Maximum Cell Mean-Free-Path Thickness.

Figure 6.10: Discrete 1-D Model Spectral Radius vs. Spatial Mesh.

6.7. Conclusions

In Ch. 5, the continuous 1-D model was systematically derived from the 3-D neutron transport – thermal hydraulics equations that describe a Pressurized Water Reactor

(PWR) fuel pin unit cell in order to provide a simplified (1-D) test bed for exploring multiphysics iterative methods for the original 3-D equations. Numerous approximations were introduced into the 1-D model. These were purposefully done in such a way that the 1-D equations are physically reasonable, and the mathematical difficulties experienced by iterative methods for the 3-D equations are preserved as much as possible. The goal of the 1-D model is to simplify the 3-D equations to 1-D, in such a way that (i) the solutions remain qualitatively realistic, (ii) the stability (or, instability) issues encountered by standard iterative methods in 3-D are retained, and (iii) current and new iterative methods can be tested, in a way that avoids complications associated with coding in 2-D and 3-D.

Of course, one does not know for certain whether, in the simplifying done to obtain the 1-D model, some important features of the 3-D equations are lost that affect the performance of iterative methods. There is always some uncertainty about this. Nonetheless, one can implement the 1-D model. If the solutions obtained by the 1-D model qualitatively agree with the solutions of 3-D problems, that is a (good) sign that the 1-D model retains a significant amount of the 3-D physics. Also, if standard iterative methods applied to the 1-D model experience similar issues with it that they experience with 3-D problems, this is a further (good) sign that the 1-D model does what it was intended to do.

Given all this, the 1-D model was (i) discretized using standard numerical methods, (ii) implemented in a test code, and (iii) run. From the results shown in Ch. 6, the solutions do seem to be qualitatively realistic, and the R-CMFD and X-CMFD methods behave roughly as they do in 3-D. In addition, the new NILO-A and NILO-CMFD methods were developed for the 1-D model, implemented in our test code, and run. The iterative performance of NILO-A and NILO-CMFD is shown to be nearly identical to that of X-CMFD.

The results from Chs. 5 and 6 show that NILO-A and NILO-CMFD are successful when applied to the 1-D model. However, the final test of these methods requires implementation in a realistic, 3-D reactor physics code. This will be discussed next in Ch. 7.

Chapter 7.

NILO-CMFD in MPACT

7.1. Introduction

In Chs. 5 & 6, we considered a 1-D, monoenergetic, neutron transport – thermal hydraulics model. The success of NILO-A and NILO-CMFD on the 1-D model gives hope that these methods may be successfully applied to 3-D, multigroup, reactor multiphysics simulations. These more complex problems are closely aligned with reactor physics calculations performed routinely in industry. In this chapter, we attempt to verify NILO's suitability for these real-world problems.

The simplicity afforded by the 1-D model allowed us to implement our own neutron transport, transport-corrected diffusion, and advection-diffusion solvers, connecting them together with R-CMFD, X-CMFD, NILO-A, and NILO-CMFD in a self-contained test code. This approach is not viable for studying 3-D, reactor-scale simulations, since the discretized neutronics and thermal hydraulics descriptions in realistic problems are significantly more complex than our 1-D discrete model equations. Instead, in this chapter we use the Michigan Parallel Characteristics-based Transport (MPACT) code to model neutronics (transport and diffusion) and the COBRA-TF (CTF) subchannel code to model thermal hydraulics. By leveraging MPACT, CTF, and an existing implementation of the "industry standard" R-CMFD method, our implementation and testing of multiphysics iterative methods on 3-D reactor physics problems became tractable.

During the initial planning stages of our implementation of NILO-CMFD in MPACT and CTF, we worked under the (incorrect) assumption that nuclear data evaluations would be inexpensive for the multiphysics problems we were interested in investigating. This assumption led us to implement the incomplete NILO-A method in

MPACT and CTF, rather than the full NILO-CMFD method. As mentioned previously, NILO-A was designed as a simpler alternative to NILO-CMFD. The NILO-A method offers a less-involved implementation and is expected to be nearly as performant as the full NILO-CMFD method in situations where nuclear data evaluations are inexpensive.

In NILO-A, nuclear data updates in the nonlinear diffusion solve are unapproximated, whereas NILO-CMFD uses approximate expressions. As a consequence, we expect the cost of a NILO-A outer iteration to exceed that of NILO-CMFD. This runtime overhead should be more noticeable in situations where nuclear data evaluations are expensive. On the other hand, if nuclear data evaluations are inexpensive, then NILO-A and NILO-CMFD outer iteration costs should be comparable. Regardless, we expect both NILO-A and NILO-CMFD to converge at the optimal CMFD (single-physics) convergence rate. This behavior was observed in the 1-D model numerical results presented in Ch. 6. In other words, performance differences between NILO-A and NILO-CMFD should occur in comparisons of simulation runtimes, not outer iteration convergence rates.

Eventually, we realized that for several problems of interest, our original assumption of inexpensive nuclear data updates was incorrect. For example, in feedback-depletion problems, the number of unique tracked isotopes inflates nuclear data update costs. Unfortunately, after realizing this, we did not have sufficient time to extend our NILO-A implementation into the full NILO-CMFD method. Therefore, we list the following considerations that should be taken into account when assessing the numerical results presented in this chapter:

(i) For the reasons mentioned above, full NILO-CMFD results will not be included. Instead, we compare only the incomplete NILO-A method to the standard R-CMFD method.

(ii) From the 1-D numerical results in Ch. 6, we expect NILO-A to converge at the optimal, single-physics, CMFD convergence rate.

(iii) Since nuclear data evaluations are expensive in the feedback-depletion problems we test, we expect NILO-A to suffer a runtime penalty. In some cases, this runtime penalty may outweigh the gains made by NILO-A in terms of lowering the number of iterations to achieve convergence.

(iv) We view our implementation of NILO-A in MPACT and CTF as an initial proof-

of-concept. We have not spent considerable time or effort on optimization and expect other NILO-A runtime penalties as a result.

With the above caveats mentioned, we outline the remainder of this chapter:

Before discussing numerical results, we introduce the Virtual Environment for Reactor Applications (VERA), a multiphysics simulation "environment" designed as a part of the Consortium for the Advanced Simulation of Light Water Reactions (CASL) project. It is within the context of VERA that we implement NILO-A. We provide details of our NILO-A implementation, as well as the implementation of the standard R-CMFD method in VERA. The Michigan Parallel Characteristics-based Transport (MPACT) code acts as the primary deterministic transport solver within VERA as well as the main multiphysics driver program. Therefore, the source-code modifications that were required to implement NILO-A were limited to MPACT. Accordingly, we summarize MPACT (e.g., structure, methods, discretization, etc.). We also provide a few details about the COBRA-TF (CTF) subchannel and simplified thermal hydraulics (simTH) solvers, which we use as the high-order $\mathcal{L}$ and low-order L thermal hydraulics operators in our NILO-A implementation.

We then introduce a group of 3-D reactor assemblies. Later, we use these assemblies to benchmark NILO-A against R-CMFD. However, before making these performance comparisons, we take a detour to discuss the solution fields in these assemblies. In this thesis, solution information is usually of minor concern. We expect each iterative method we discuss to reach the same solution for a given multiphysics problem (within convergence tolerances). Rather, we are more interested in the efficiency with which that solution is found. However, by looking at 3-D solution information from a set of assemblies, we will observe several qualitative similarities with the solutions obtained from the discrete 1-D model in Ch. 6. These similarities act as evidence towards the veracity of the 1-D model, instilling greater confidence in the practical relevance of the numerical results presented in Ch. 6.

Following our discussion of 3-D assembly solutions, we compare the performance of NILO-A to R-CMFD on a series of feedback-depletion assembly simulations. Analyzing outer iteration counts from these simulations, we see that NILO-A converges at or near the ideal single-physics CMFD convergence rate, thereby *minimizing the number of transport sweeps and thermal hydraulics solves required to solve a given problem.* In contrast, R-CMFD converges sub-optimally and, in certain cases, diverges.

Unfortunately, when comparing NILO-A and R-CMFD runtimes for these same assembly problems, we observe that NILO-A spends a considerable amount of time

performing nuclear data updates. In certain situations, this extra runtime cost can result in worse wall-clock-time performance for NILO-A as compared to R-CMFD. As mentioned above, we expect this runtime issue to be resolved by an implementation of the full NILO-CMFD method, making use of its approximate nuclear data updates. However, due to time constraints, we were not able to verify this expectation.

Finally, we consider a full-scale, quarter-core, reactor feedback-depletion problem. As with the assembly simulations, the R-CMFD convergence rate is less than desireable, while NILO-A performs optimally. We again see that NILO-A suffers a considerable runtime penalty, originating from its excessive nuclear data evaluations.

Overall, the numerical results presented in this chapter indicate that:

(i) On realistic, 3-D reactor multiphysics problems, NILO-A succesfully minimizes the number of transport sweeps and thermal hydraulics evaluations required to reach convergence.

(ii) While NILO-A outperforms R-CMFD in terms of outer iteration counts, NILO-A's excessive nuclear data evaluations impose a significant runtime penalty. In certain cases, this leads to worse overall wall-clock-time performance compared to the standard R-CMFD method.

(iii) An implementation of the full NILO-CMFD method in MPACT and CTF should be pursued to address the performance issues mentioned above.

7.2. MPACT Overview

The Michigan Parallel Characteristics-based Transport (MPACT) code [55] was developed as part of the Consortium for the Advanced Simulation of Light Water Reactors (CASL) project [106] established by the United States Department of Energy (U.S. DOE). The goal of CASL was to foster collaboration between national labs, academia, and industry in order to develop software for the accurate and efficient simulation of operating nuclear reactors.

As a central deliverable, CASL developed a computational "environment" of single-physics packages to be connected together and solved in consort, to faithfully simulate the coupled, multiphysics environment of a reactor core. This environment was named the *Virtual Environment for Reactor Applications* (VERA) [107].

The principal deterministic neutronics code in VERA is the Michigan Parallel

Characteristics-based Transport (MPACT) code [55]. In this role, MPACT calculates the reactor-core neutron distribution and associated reaction rates. As a consequence of CASL's focus on the faithful simulation of operating reactors, MPACT is often used to solve multiphysics problems in which the nuclear data (e.g., macroscopic cross sections, etc.) themselves are problem unknowns. Through VERA, MPACT is coupled to several other single-physics solvers that supply it with the necessary utilities to update nuclear data when updated fission heat sources and related neutronically-determined solution data are available. For example, MPACT is able to send thermal energy deposition information (primarily, locally deposited fission energy) to the COBRA-TF (CTF) subchannel fluid dynamics code [87], which, in turn, supplies to MPACT updated temperatures and densities required for updating nuclear data.

In our implementation of NILO-A in the VERA suite, we use MPACT as the *driver code*. This means that, in the simulations we run, MPACT is the simulation's point of entry, and MPACT decides when to call out to external single-physics solvers (e.g., CTF). MPACT has two roles: it (i) solves for the neutron distribution in the core, and (ii) coordinates the multiphysics iterative method. With regard to its role as neutronics solver, MPACT uses the Coarse Mesh Finite Difference (CMFD) acceleration scheme by default. MPACT's use of CMFD simplified our implementation of NILO-A, since we did not need to implement our own transport-corrected diffusion solver. Because of its role as multiphysics iteration coordinator, MPACT was the only code we needed to modify to implement NILO-A in VERA, simplifying our implementation by not requiring us to to write interfacing code in multiple codebases. For example, we did not need to write any code in CTF.

For sweeping, MPACT uses the 2D-1D method [20] in which (i) the reactor core is divided into stacked radial slices, (ii) the radial slices are swept on a complex (i.e. non-Cartesian) *fine-grid* mesh using the Method of Characteristics (MOC) [66], and (iii) the radial slices are stitched together axially using leakage factors and a 1-D axial neutronics solver. We refer to the spatial mesh on which MPACT sweeps as the *fine mesh*. MPACT's transport-corrected diffusion (CMFD) solver works on a hexahedral (brick) *coarse mesh*. An illustration of the radial structure of MPACT fine and coarse meshes for a pin cell is given in Fig. (7.1). In the axial direction, the coarse mesh is extruded. We emphasize that MPACT uses a fine mesh for sweeping, and a coarse mesh for transport-corrected diffusion. (Up to this point in the thesis, we have ignored this complication. When subtleties related to the use of two meshes

arise (both fine and coarse), we address them.)

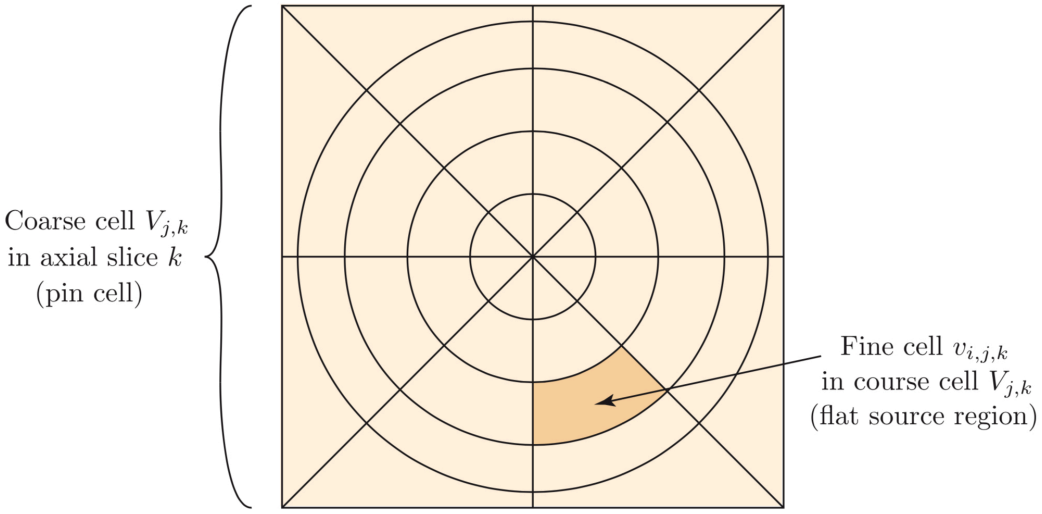


Figure 7.1: Pin-Cell Fine- and Coarse- Mesh Cells (MPACT Theory Manual).

MPACT uses Relaxed CMFD (R-CMFD) as its default iterative method for steady-state multiphysics problems. Next, we give a cursory overview of MPACT's Relaxed CMFD (R-CMFD) implementation. Although MPACT can be coupled to multiple sources of feedback in a given simulation, we only reference coupling to the CTF subchannel thermal hydraulics code, to keep in line with simulation results presented later in this chapter. For a finer-grained discussion of the R-CMFD implementation in MPACT (including equations) we refer the reader to the MPACT Theory Manual [55].

Before outlining R-CMFD in MPACT, we mention a slight difference between MPACT's outer iteration ordering and the ordering that we have used up to this point in the thesis. In MPACT, the first step of an outer iteration is a thermal hydraulics solve, the second is a nuclear data evaluation, the third is a low-order diffusion calculation, and the fourth is a transport sweep. This ordering of outer iteration steps should have no effect on the outer iteration convergence rate, compared to the "equivalent" ordering considered throughout the rest of this thesis.

We also note that MPACT treats temperatures T and densities ρ as separate, distinct quantities. In earlier chapters, we eliminated density by assuming a direct dependence of density on temperature, $\rho(T)$. In MPACT, this dependence is not always available, so we must include explicit treatments of both temperature and density in the R-CMFD outline presented below:

(MPACT) R-CMFD Step 1 : Thermal Hydraulics Solve

In the first step of an outer iteration, MPACT calls the subchannel thermal hydraulics code COBRA-TF (CTF) to update temperatures and densities in response to the most up-to-date fission heat source estimate. The heat source sent to CTF is relaxed by the user-chosen relaxation factor α (by default, $\alpha = 0.5$).

(MPACT) R-CMFD Step 2 : Nuclear Data Update

Using the CTF-updated temperatures and densities, MPACT updates fine-mesh nuclear data. Depending on whether the temperature and density changes are large enough, MPACT may perform a self-shielding calculation in order to update equivalence cross sections [64]. (In actuality, MPACT nuclear data is stored on the "cross section mesh," a mesh residing between the fine and coarse meshes in terms of spatial fidelity. To simplify this chapter, we ignore this implementation detail and discuss quantities stored on the cross section mesh as though they reside on the fine mesh.)

(MPACT) R-CMFD Step 3 : CMFD Solve

To set up the CMFD linear diffusion eigenvalue system, MPACT *homogenizes* nuclear data from the fine mesh to the coarse mesh. Using multigrid terminology, this is a *restriction* operation. Homogenization is accomplished through simple scalar flux weighting. Additionally, transport correction factors and boundary correction factors are evaluated on coarse-mesh cell surfaces. With the CMFD system defined, MPACT solves the diffusion eigenvalue problem using Wielandt Shifted Power Iteration (WS-PI) to determine updated (coarse-mesh) scalar flux and eigenvalue estimates. In order to propagate coarse-mesh scalar flux updates to the fine-mesh, a standard multiplicative fine-to-coarse-mesh weighting is used.

(MPACT) R-CMFD Step 4 : Transport Sweep

To conclude an outer iteration, a 2D-1D "transport sweep" is performed, consisting of a 2-D radial Method of Characteristics (MOC) transport sweep followed by a 1-D axial solve to obtain updated fine-grid scalar flux estimates.

Finally, MPACT checks convergence and max iteration criteria. If either condition has been met, iteration is halted. Otherwise, MPACT returns to Step 1 to begin the next outer iteration.

7.3. NILO-A Overview

Next, we describe our implementation of NILO-A in MPACT. (Again, due to time constraints, we have not implemented the full NILO-CMFD method in MPACT. Thus, we only discuss NILO-A in what follows.)

The major differences between R-CMFD and NILO-A are confined to the diffusion solve (Step 3 of MPACT's outer iteration ordering). In R-CMFD, a linear diffusion eigenvalue problem updates the coarse-mesh scalar fluxes and eigenvalue. NILO-A's version of this step involves a nonlinear diffusion solve that couples together the CMFD diffusion equations, nuclear data update utilities, and an approximate low-order thermal hydraulics solver L.

To implement NILO-A, a suitable low-order thermal hydraulics solver L is required. In the case of MPACT, a simplified Thermal Hydraulics (*simTH*) solver was already implemented that we were able to repurpose as NILO-A's L. In CTF, 3-D subchannel mass, momentum, and energy conservation equations are solved in the fluid, and 3-D heat-conduction equations are solved in the solid (fuel, gap, cladding) [87]. In simTH, a 1-D axial enthalpy balance updates fluid temperatures and densities, and solid temperatures are updated either through a 1-D cylindrical geometry heat conduction solve or via table lookup / interpolations of pre-computed fuel temperature data [55].

A simTH solve is three orders of magnitude less expensive than a CTF solve. Also, in its role as L in the MPACT numerical results presented later, the simTH operator is accurate enough to maintain NILO-A's optimal outer iteration convergence rate. Therefore, simTH has the qualities that we require of a successful NILO-CMFD low-order thermal hydraulics operator L. The suitability of simTH, coupled with the fact that we did not have to implement it ourselves, vastly simplified our implementation of NILO-A in MPACT. In general, this "reuse" of existing thermal hydraulics codes may simplify the implementation of NILO-A and NILO-CMFD in other reactor physics codes. Additionally, by reusing inexpensive thermal hydraulics solvers, NILO-A and NILO-CMFD may foster new applications for outdated thermal hydraulics codes that have been superseded by more accurate solvers.

Another modification that was required to implement NILO-A in MPACT was a minor rewrite of the diffusion power-iteration procedure. To solve the NILO-A nonlinear diffusion problem, we used the Nonlinear Wielandt Shifted Power Iteration (N-WS-PI) inner iteration scheme outlined in Chs. 4 & 6. Implementation of N-WS-PI was straightforward, since MPACT already included a WS-PI solver for linear

diffusion eigenvalue problems.

A further complication related to NILO-A inner iterations in MPACT was the distinction between fine and coarse meshes. In MPACT, thermal hydraulics (both CTF and simTH) and cross section updates are performed on the fine mesh, while transport-corrected diffusion solves occur on the coarse mesh. In our implementation of NILO-A, we used coarse-to-fine-mesh and fine-to-coarse-mesh mapping to propagate inner iteration solution updates between meshes whenever necessary.

Next, we outline our NILO-A implementation in MPACT:

(MPACT) NILO-A Step 1 : Thermal Hydraulics Solve

This step is identical to (MPACT) R-CMFD Step 1, except relaxation is **removed** ($\alpha = 1$). This step sends fully-updated fission heat source estimates to CTF, where the high-order thermal hydraulics operator $\mathcal{L}$ solves the 3-D mass, momentum, and energy conservation equations in the fluid and a 3-D heat conduction equation in the solid for updated fine-grid temperatures and densities. These updated temperatures and densities are then sent back to MPACT.

(MPACT) NILO-A Step 2 : Nuclear Data Update

This step is identical to (MPACT) R-CMFD Step 2. Fine-mesh nuclear data updates are performed in response to the updated temperature and density estimates from Step 1.

(MPACT) NILO-A Step 3 : Nonlinear Diffusion Solve

Besides the removal of relaxation in Step 1, this step is the only significant departure from R-CMFD. In NILO-A, a nonlinear transport-corrected diffusion eigenvalue problem is solved for the updated scalar flux, eigenvalue, and nuclear data estimates. The inner iteration procedure involves iteratively (i) updating fine-mesh temperatures and densities using simTH-calculated (L) corrections to the Step 1 CTF-calculated ($\mathcal{L}$) values, (ii) updating fine-grid nuclear data using the new temperature and density estimates, (iii) homogenizing updated nuclear data to the coarse mesh, (iv) performing a single shifted power iteration, and (v) projecting the resulting coarse-mesh scalar flux to the fine-mesh to calculate an updated fine-mesh fission heat source to be used in the next inner iteration.

(MPACT) NILO-A Step 4 : Transport Sweep

This step is identical to (MPACT) R-CMFD Step 4. A 2D-1D "transport sweep" updates fine-grid scalar fluxes.

The above outline loosely describes our implementation of NILO-A in MPACT. Thanks to a collection of factors, our NILO-A implementation was rather straightforward. Chief among these factors were (i) NILO-A's similarities to MPACT's default R-CMFD procedure, (ii) NILO-A's loosely-coupled nature and ability to use any suitable low-order thermal hydraulics operator L (i.e. the simplified Thermal Hydraulics (simTH) solver), and (iii) the similarities between the Nonlinear WS-PI (N-WS-PI) method and MPACT's default WS-PI inner iteration scheme.

In the next section, we describe numerical results comparing R-CMFD and NILO-A performance in MPACT on a suite of Pressurized Water Reactor (PWR) assembly problems, and a Westinghouse quarter core problem.

7.4. Numerical Results

In this section, we discuss R-CMFD and NILO-A numerical results in MPACT. We begin by considering a series of assembly problems and conclude with a larger-scale quarter-core problem.

7.4.1. Reactor Assembly Calculations

Problem Description

To benchmark NILO-A performance, we compared R-CMFD and NILO-A behavior on a suite of assembly problems. These problems were created by combining problem input parameters from the selection of Pressurized Water Reactor (PWR) benchmark problems described in the VERA Core Physics Benchmark Progression report [31]. The assembly problems we explore are 3-D, multiphysics versions of the CASL Progression Problem 2 fuel lattices. In Table (7.1), we list the identifiers (i.d.'s) used to refer to the assemblies, along with a brief characterization of each.

Table 7.1: 3-D Assembly Problem Identifiers & Characteristics.

i.d.	Characteristics
2a-3d	No Poisons
2e-3d	12 Pyrex Inserts
2f-3d	24 Pyrex Inserts
2k-3d	24 Pyres Inserts + Zoned Enrich.
2o-3d	12 Gadolinia Rods
2p-3d	24 Gadolinia Rods

Each assembly in Table (7.1) was solved as a "k-search" problem, meaning that the boric acid concentration was fixed (at 1300 [ppm]), with the eigenvalue k allowed to vary as a problem unknown. (Later, when discussing quarter-core results, we consider "boron search" problems in which this relationship is flipped; k is fixed at unity, while critical boric acid concentration becomes a dependent (solution) variable.)

In the Table (7.1) assemblies, thermal hydraulics feedback from CTF was included. In NILO-A simulations, MPACT's internal simTH solver was used for the low-order thermal hydraulics operator L.

Next, we provide some additional parameters common to each assembly in Table (7.1). Inlet fluid (moderator) temperature is 565 [K] $\approx$ 292 [°C]. Outlet pressure is 2250 [psia]. As mentioned earlier, boric acid concentration is fixed at 1300 [ppm]. Assembly pitch (side-length) and axial height are 21.5 [cm] and 406.337 [cm], respectively (consistent with the Watts Bar Unit 1 (WBU1) Westinghouse-designed core). Each assembly has quarter radial symmetry. Assembly thermal power is 17.67 [MW] and coolant mass flow rate is 0.6823 [Mlbs $\cdot$ hr^{-1}], both consistent with assembly-averaged values from WBU1. Each assembly model includes lower and upper core plates, spacer grids, lower and upper nozzles, gap, plug, and plenum. In the appendices, we provide an example VERA input file for the 2a-3d assembly.

Next, we comment on the differences between each assembly in Table (7.1). Assembly 2a-3d is a 17×17 Westinghouse assembly with 3.1 weight-percent ^{235}U enrichment (all assemblies use Zircaloy-4 cladding). Problem 2e-3d is identical to 2a-3d, except that Pyrex Burnable Absorber (BA) rods are inserted into 12 of the assembly guide tubes. Problem 2f-3d is identical to 2e-3d, except that 24 Pyrex rods are inserted instead of 12. 2k-3d is identical to 2f-3d, both with 24 Pyrex inserts, except that 2k-3d additionally includes zoned enrichment. In 2k-3d, both 3.6 and 3.1 weight-percent ^{235}U rods are present. Rather than Pyrex inserts, problems 2o-3d and 2p-3d include 12 and 24 gadolinium-infused fuel rods, respectively.

Convergence Check

Before comparing R-CMFD and NILO-A performance on the assemblies in Table (7.1), we first check the correctness of our NILO-A implementation in MPACT by comparing R-CMFD and NILO-A solutions while varying convergence criteria. We expect some difference between the R-CMFD and NILO-A solutions, because each method's outer iterations are distinct, and our problems of interest are not solved to infinite precision. However, as the convergence criteria are tightened, we expect the relative difference between R-CMFD and NILO-A solutions to decrease. To test this conjecture, we ran a series of R-CMFD and NILO-A steady-state, fresh fuel, k-search, feedback simulations of assemblies 2a-3d and 2f-3d.

In Tables (7.2)-(7.3), we show both the root-mean-squared (rms) and maximum (max) relative difference between R-CMFD and NILO-A solutions of the assembly pin-powers, fluid temperature, and fuel temperature as a function of convergence tolerance ϵ. For a given simulation, ϵ was used in both the "k tolerance" and "flux tolerance" cards in the VERA common user input file [80]. As expected, both measures of solution difference decrease as the convergence tolerance ϵ is tightened. This suggests that both R-CMFD and NILO-A converge to the same solution. Assuming that MPACT's default R-CMFD iteration scheme converges to the correct solution, we can say that our implementation of NILO-A in MPACT also reaches the correct solution.

In Tables (7.2)-(7.3), we have also included the number of outer iterations to convergence for each of the R-CMFD (R) and NILO-A (N-A) convergence check simulations. For both problems and both methods, we see that as the convergence tolerance ϵ is tightened, the number of outer iterations increases. This is expected. We also notice that R-CMFD and NILO-A iteration counts are similar for a given convergence tolerance ϵ. This is indicative of a trend we see throughout this chapter. For fresh-fuel problems (e.g., Tables (7.2)-(7.3) simulations), we see similar outer iteration convergence rates between NILO-A and R-CMFD. It appears that these fresh-fuel problems are simple enough that convergence rate differences do not occur. Later, we will see that when these same assemblies are depleted, R-CMFD performance begins to degrade. In other words, the process of depletion makes these problems considerably more difficult from the standpoint of R-CMFD. It is in these situations that the convergence rate differences between NILO-A and R-CMFD become more apparent.

Table 7.2: Problem 2a-3d Convergence Table.

	Pin Power		Fluid Temp.		Fuel Temp.		Iters.	
ϵ \ % err.	rms	max	rms	max	rms	max	R	N-A
10^{-2}	$9.915 \cdot 10^{-1}$	$1.723 \cdot 10^{0}$	$5.435 \cdot 10^{-2}$	$1.258 \cdot 10^{-1}$	$1.288 \cdot 10^{0}$	$2.706 \cdot 10^{0}$	3	3
10^{-3}	$1.897 \cdot 10^{-1}$	$7.559 \cdot 10^{-1}$	$1.755 \cdot 10^{-2}$	$3.145 \cdot 10^{-2}$	$2.854 \cdot 10^{-1}$	$5.274 \cdot 10^{-1}$	5	4
10^{-4}	$2.866 \cdot 10^{-2}$	$5.374 \cdot 10^{-2}$	$1.037 \cdot 10^{-3}$	$1.759 \cdot 10^{-3}$	$1.710 \cdot 10^{-2}$	$3.735 \cdot 10^{-2}$	7	7
10^{-5}	$3.077 \cdot 10^{-3}$	$3.190 \cdot 10^{-2}$	$3.157 \cdot 10^{-5}$	$9.941 \cdot 10^{-5}$	$8.590 \cdot 10^{-4}$	$3.855 \cdot 10^{-3}$	11	10
10^{-6}	$1.472 \cdot 10^{-4}$	$7.625 \cdot 10^{-4}$	$6.355 \cdot 10^{-6}$	$9.953 \cdot 10^{-6}$	$1.440 \cdot 10^{-4}$	$6.703 \cdot 10^{-4}$	16	16

Table 7.3: Problem 2f-3d Convergence Table.

	Pin Power		Fluid Temp.		Fuel Temp.		Iters.	
ϵ \ % err.	rms	max	rms	max	rms	max	R	N-A
10^{-2}	$7.033 \cdot 10^{-1}$	$9.678 \cdot 10^{-1}$	$4.619 \cdot 10^{-2}$	$9.286 \cdot 10^{-2}$	$7.158 \cdot 10^{-1}$	$1.720 \cdot 10^{0}$	4	4
10^{-3}	$3.701 \cdot 10^{-1}$	$1.100 \cdot 10^{0}$	$4.650 \cdot 10^{-2}$	$1.035 \cdot 10^{-1}$	$7.949 \cdot 10^{-1}$	$2.065 \cdot 10^{0}$	4	6
10^{-4}	$1.933 \cdot 10^{-2}$	$2.692 \cdot 10^{-2}$	$3.751 \cdot 10^{-4}$	$7.954 \cdot 10^{-4}$	$1.111 \cdot 10^{-2}$	$6.942 \cdot 10^{-1}$	8	8
10^{-5}	$3.504 \cdot 10^{-3}$	$5.928 \cdot 10^{-3}$	$1.150 \cdot 10^{-4}$	$2.273 \cdot 10^{-4}$	$3.597 \cdot 10^{-3}$	$7.017 \cdot 10^{-1}$	11	11
10^{-6}	$2.246 \cdot 10^{-4}$	$1.396 \cdot 10^{-3}$	$4.658 \cdot 10^{-6}$	$1.198 \cdot 10^{-5}$	$1.114 \cdot 10^{-4}$	$1.044 \cdot 10^{-3}$	16	16

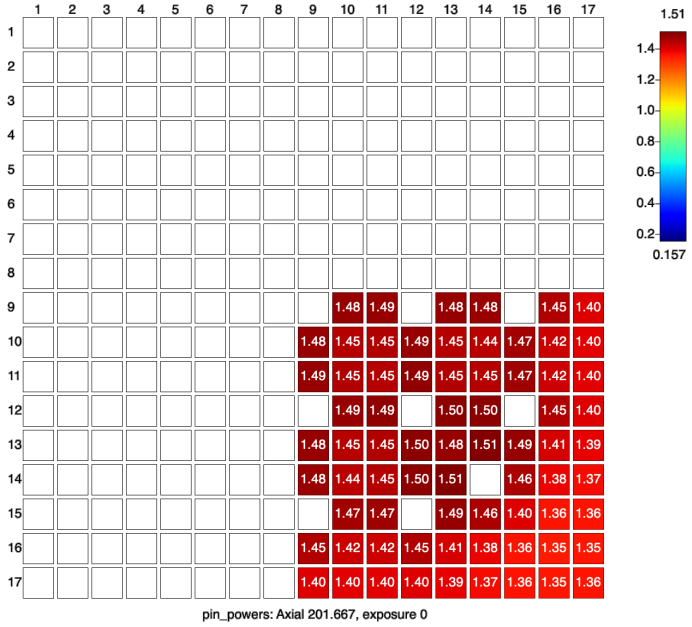
Steady-State Feedback Problem Solutions

Next, we explore the radial and axial solutions of steady-state, fresh-fuel problems for two example assemblies from Table (7.1), specifically, 2a-3d and 2f-3d. We do this for several reasons: (i) to give additional information about the assembly geometry (guide tube locations, etc.) (ii) to showcase how the variety of absorbers in Table (7.1) assemblies causes noticeable changes in the solution profiles, and (iii) to show the qualitative similarities between 3-D assembly solutions from MPACT and radially integrated solutions from the simplified 1-D model explored in Chs. 5 & 6. All solutions in this section were obtained from R-CMFD simulations using the tightest convergence tolerance from Tables (7.2)-(7.3), $\epsilon = 1 \cdot 10^{-6}$. (NILO-A solutions are visually indistinguishable; both iterative methods converge to the same solution within the error tolerance ϵ, as shown in Tables (7.2)-(7.3).)

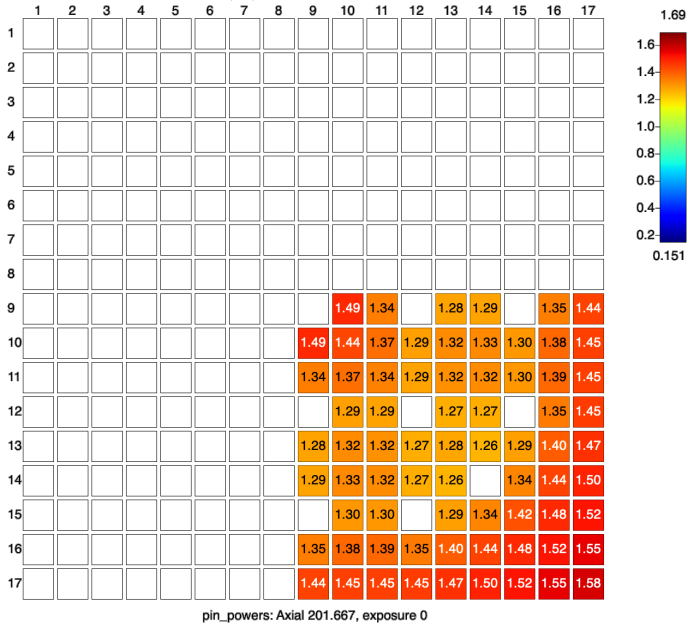
We note that the goal of this subsection is not to compare R-CMFD and NILO-A performance. By presenting 3-D solutions in order to increase our confidence in the 1-D results presented earlier, we help to strengthen our argument that the 1-D NILO-A and NILO-CMFD results from Ch. 6 are meaningful. This becomes especially important when considering that we have yet to implement the full NILO-CMFD method to a 3-D reactor multiphysics problem. After this consideration of solution profiles, we return to R-CMFD and NILO-A performance comparisons in the next subsection.

In Figs. (7.2), we show the radial pin-power power distribution (in units of normal-

ized pinwise reaction rate) for the steady-state, fresh fuel, 2a-3d and 2f-3d problems [Table (7.1)]. Figs. (7.2), as well as all other radially-dependent solutions shown in this section, are given at an axial height $z = 201.667$ [cm], approximately halfway up



(a) Assembly 2a-3d

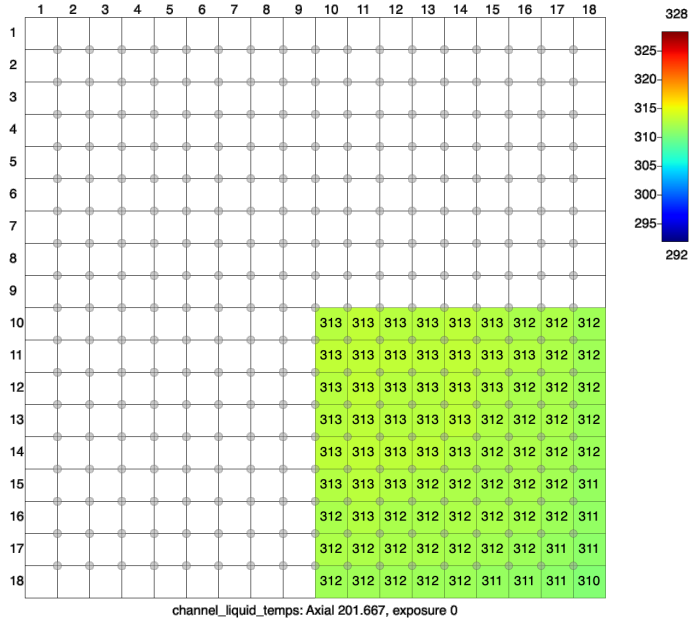


(b) Assembly 2f-3d

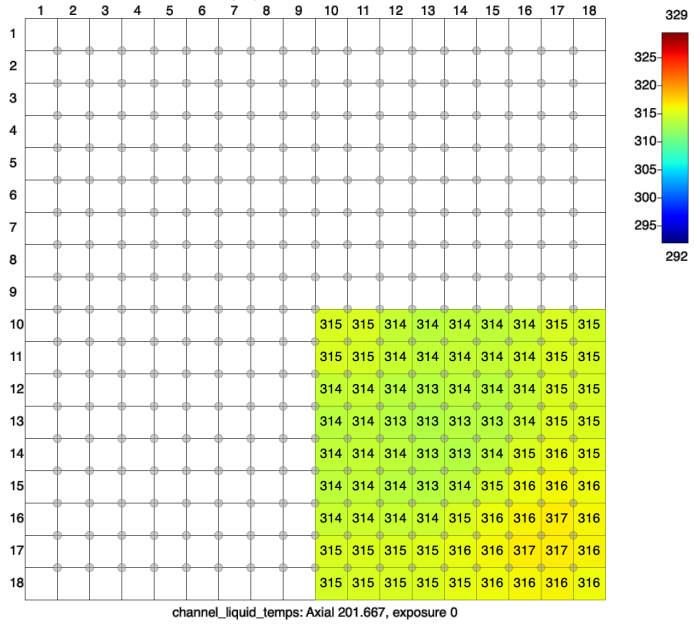
Figure 7.2: Fresh Fuel Pin Power Radial Profiles.

the core height. Solutions in Figs. (7.2) are only shown for the bottom-right quad-

rant of the assembly, owing to the quarter radial symmetry used for each assembly simulation. This display format will be mirrored in subsequent radial solution plots. Solution values for other quadrants are obtained simply by reflecting the solutions of the quadrant shown.



(a) Assembly 2a-3d



(b) Assembly 2f-3d

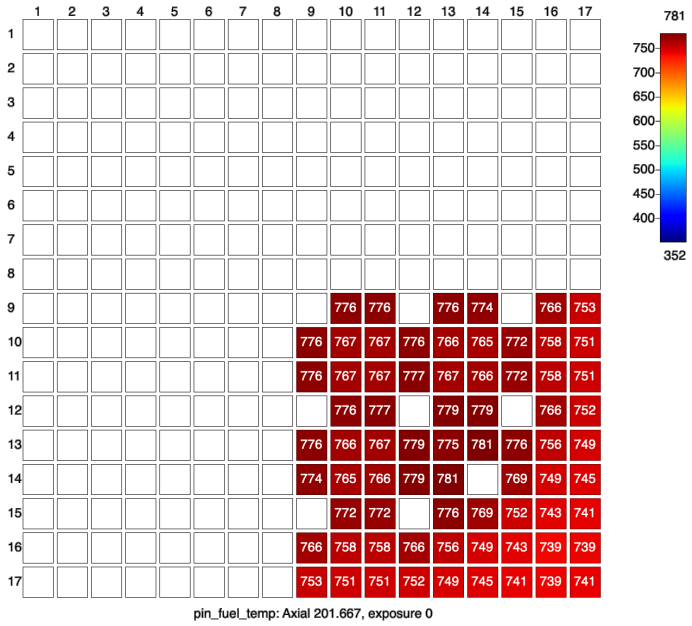
Figure 7.3: Fresh Fuel Coolant Temperature [°C] Radial Profiles.

The blank squares in the bottom-right quadrant of Figs. (7.2) represent guide tubes, which do not contain active (i.e. fissionable) fuel. Thus, their pin-wise powers are identically zero. In 2a-3d [Fig. (7.2a)], every guide tube in the assembly is "empty," meaning that they are zirconium tubes surrounded by and filled with coolant. In 2f-3d, as mentioned in Table (7.1), 24 of the assembly guide tubes (all but the center-most at location (9,9)) contain Pyrex burnable absorber rods. We note the marked depression in pin powers surrounding these Pyrex absorber filled guide tubes in Fig. (7.2b). In comparison, assembly 2a-3d's [Fig. (7.2a)] pin powers are relatively flat radially.

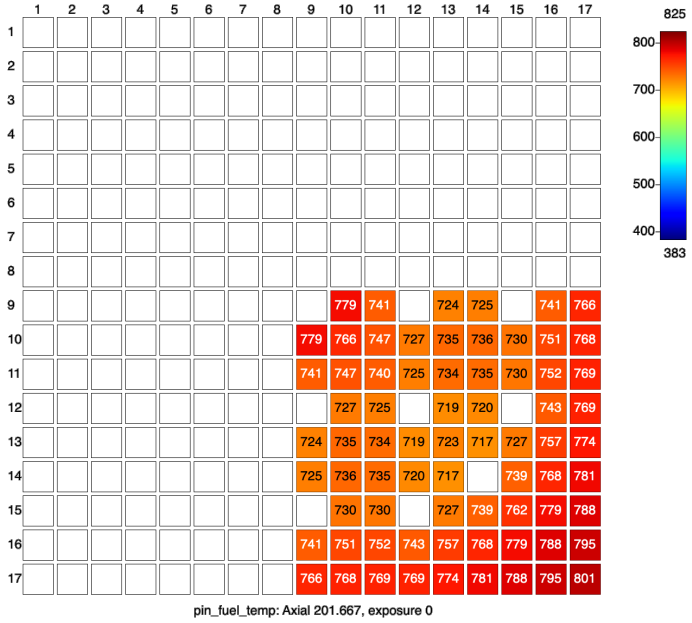
Figs. (7.3) show the radial profile ($z = 201.667$ [cm]) of the CTF-determined channel average coolant temperature [°C] for the 2a-3d and 2f-3d steady-state feedback problems. In Figs. (7.3), each numbered square represents a fluid subchannel lying in the space formed between fuel pins and guide tubes (represented by gray circles). Slightly more radial variation in coolant temperature occurs for the Pyrex rodded 2f-3d assembly in Fig. (7.3b) than for the burnable-absorber-less 2a-3d assembly in Fig. (7.3a).

Figs. (7.4) show the CTF-determined volume average fuel pellet temperatures [°C] for the steady-state, fresh fuel 2a-3d [Fig. (7.4a)] and 2f-3d [7.4b] problems ($z = 201.667$ [cm]). Since fuel temperatures are tightly linked to pin powers, we observe that the radial fuel temperature profiles in Figs. (7.4a) and (7.4b) mimic their corresponding radial pin-power shapes in Figs. (7.2a) and (7.2b). Notably, in Fig. (7.4b), we see a decrease in pellet temperatures in fuel rods immediately surrounding Pyrex filled guide tubes.

The simplified 1-D model explored in Chs. 5 & 6 is radially integrated, preventing us from obtaining radial solution profiles in Ch. 6. Next, we show the axially-dependent solutions of the steady-state, fresh fuel 2a-3d and 2f-3d multiphysics problems, which we are able to compare to the 1-D model solutions from Ch. 6. In the following, we see several qualitative similarities between solutions of the two models (1-D and 3-D). This approximate agreement instills confidence that the 1-D model explored in Ch. 5 & 6 successfully captures physics seen in realistic 3-D problems. This gives greater weight to the NILO-A and NILO-CMFD results discussed in Ch. 6.



(a) Assembly 2a-3d



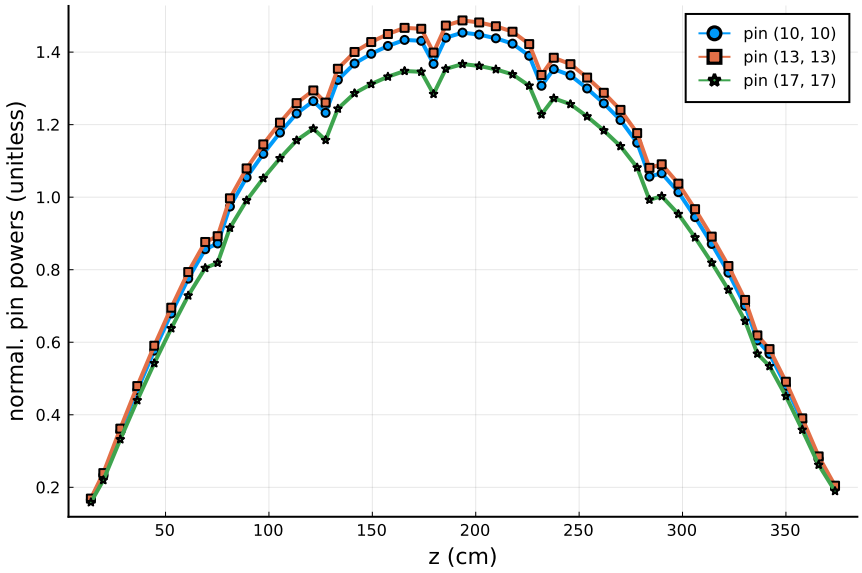
(b) Assembly 2f-3d

Figure 7.4: Fresh Fuel Pellet Temperature [°C] Radial Profiles.

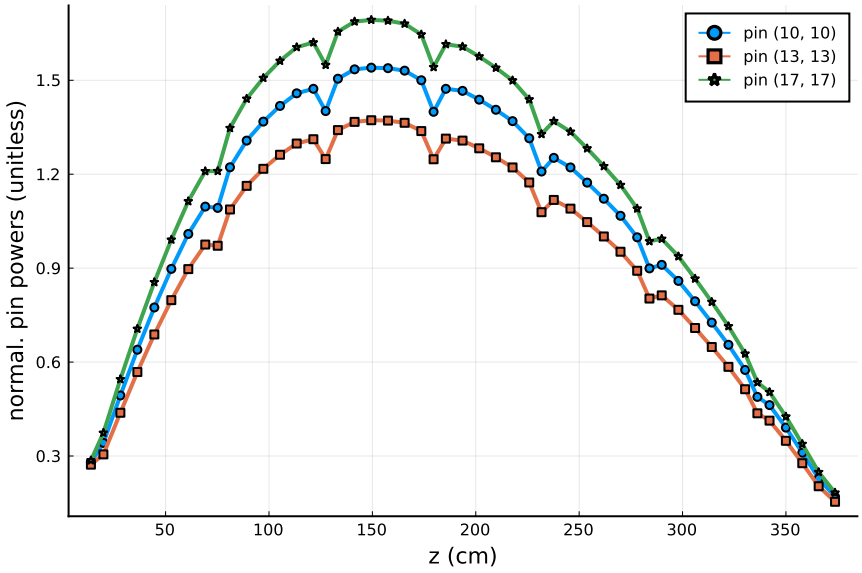
In Figs. (7.5)-(7.7), we plot the axial dependence of the normalized pin powers, channel coolant temperatures [°C], and volume average fuel pellet temperatures [°C] for a few representative fuel pins and coolant channels. In Figs. (7.5) and (7.7), axial solutions are plotted for fuel pins (10,10), (13,13), and (17.17). The pin locations corresponding to these identifiers can be viewed in the radial-dependent Figs. (7.2)

and (7.4). The radial location of coolant channels (10,10), (13,13), and (17,17) shown in Figs. (7.6) can likewise be found using the numbering in Fig. (7.3).

In Figs. (7.5), we plot the axial dependence of the normalized pin-powers for pins (10,10), (13,13), and (17,17) from the 2a-3d [Fig. (7.5a)] and 2f-3d [Fig. (7.5b)] steady-state, fresh-fuel, multiphysics problems. The noticeable power dips in Figs. (7.5) correspond to the axial placement of spacer grids. We note that the 2f-3d pin-power distributions in Fig. (7.5b) are noticeably more bottom-peaked than the 2a-3d axial



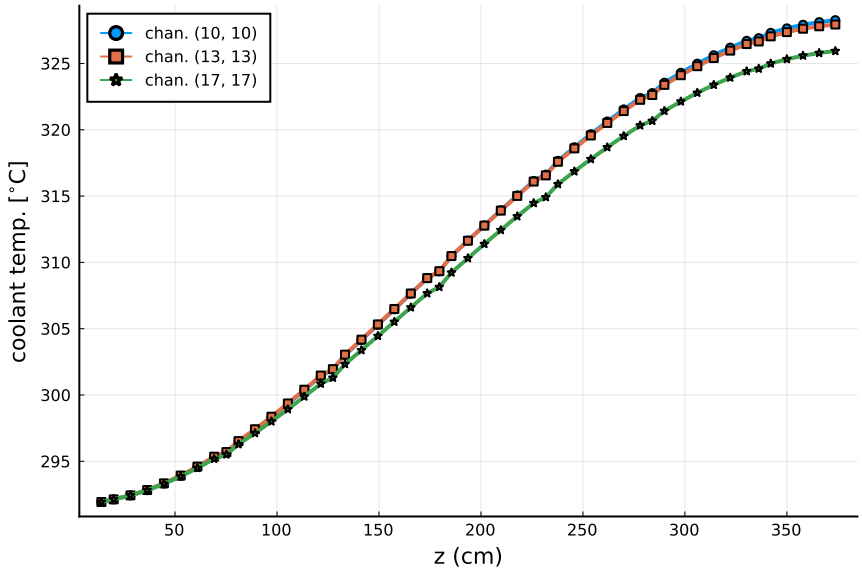
(a) Assembly 2a-3d



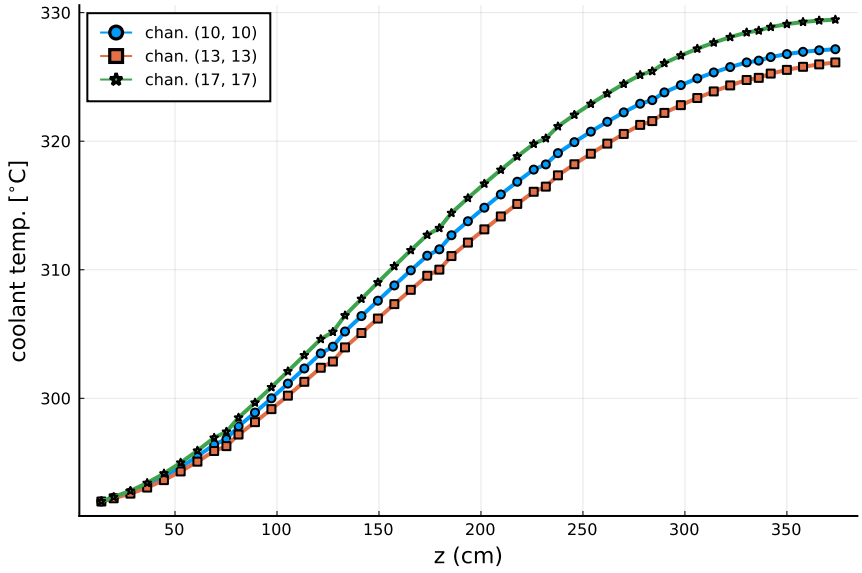
(b) Assembly 2f-3d

Figure 7.5: Fresh Fuel Pin Power Axial Profiles.

profiles in Fig. (7.5a). We also note that the 2f-3d assembly contains Pyrex burnable absorbers, while the 2a-3d assembly has no burnable absorbers. This difference in 3-D assembly results falls in line with an observation made in Ch. 6, where we noted that the scalar flux profiles in 1-D model pins based on assemblies containing burnable absorbers were noticeably more bottom-peaked than those without burnable absorbers.



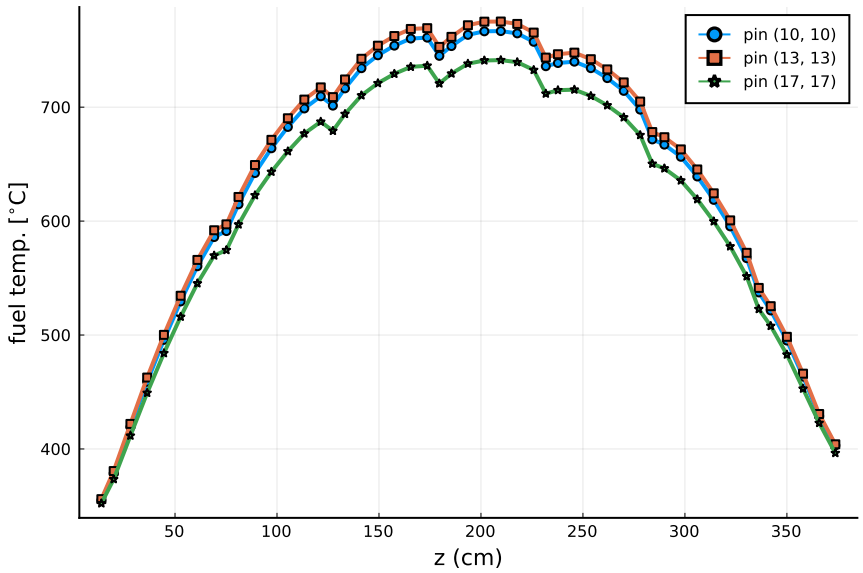
(a) Assembly 2a-3d



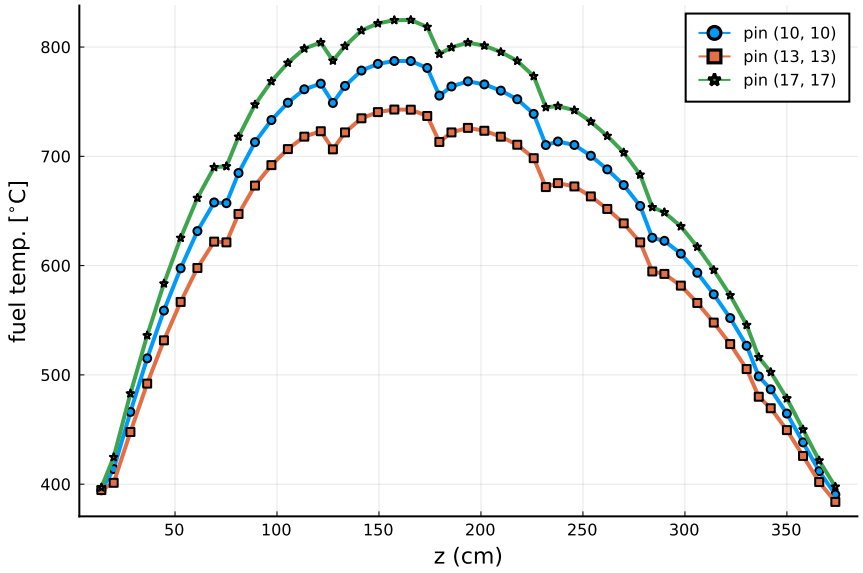
(b) Assembly 2f-3d

Figure 7.6: Fresh Fuel Coolant Temperature [°C] Axial Profiles.

Figs. (7.6) show the axial dependence of the coolant channel moderator temperature [°C] in channels (10,10), (13,13), and (17,17) for assembly 2a-3d [Fig. (7.6a)] and 2f-3d [Fig. (7.6b)]. As in the 1-D model, coolant temperatures monotonically increase as one moves up through the assembly, and fission heat produced in the fuel rods is accumulated by the coolant.



(a) Assembly 2a-3d



(b) Assembly 2f-3d

Figure 7.7: Fresh Fuel Pellet Temperature [°C] Axial Profiles.

Figs. (7.7) plot the axial dependence of the volume-average fuel pellet temperature

in a given slice for the 2a-3d [Fig. (7.7a)] and 2f-3d [Fig. (7.7b)] steady-state, fresh-fuel, multiphysics problems. As expected, owing to the tight relationship between pin power and fuel-pellet temperature, the 2f-3d fuel temperature profiles in Fig. (7.7b) are noticeably more bottom-peaked than those for 2a-3d in Fig. (7.7a).

Feedback-Depletion Calculations

The convergence Tables (7.2)-(7.3), radial solution Figs. (7.2)-(7.4), and axial solution Figs. (7.5)-(7.7) were all created from steady-state, beginning-of-cycle (fresh fuel), multiphysics simulations of the 2a-3d and 2f-3d assemblies in Table (7.1). During our initial surveys in search of problems to compare R-CMFD and NILO-A relative performance metrics, we noticed that feedback-depletion problems were particularly taxing on the standard R-CMFD method. (In this chapter, we do not show 3-D, steady-state results with increasing system power since the power required to observe R-CMFD performance impacts was unphysically high.)

A *feedback-depletion problem* is an industry-standard problem used to model the time-dependent criticality of a reactor as the fuel and other isotopes within the system are depleted or accumulated by neutron interactions with and radioactive decay of isotopes. In a feedback-depletion problem, as compared to a depletion problem, the effects of multiphysics feedback are included in the calculation. In an operating reactor core, as the criticality k of the system changes over time in response to isotope accretion and depletion, $k \approx 1$ is maintained through the movement of control rods and modification to the boric acid (poison) concentration in the coolant.

Timescales of feedback-depletion problems are too long to reasonably solve with time-dependent transport calculations. Instead, a quasi-static approximation is usually employed, in which a time-dependent feedback-depletion problem is solved through a sequence of steady-state, multiphysics transport problems. In MPACT, feedback-depletion problems may be solved using a quasi-statics-based predictor-corrector scheme, which requires two steady-state multiphysics transport problem solutions each depletion timestep. We employed this predictor-corrector approach in all assembly and quarter-core feedback-depletion problems discussed in this chapter.

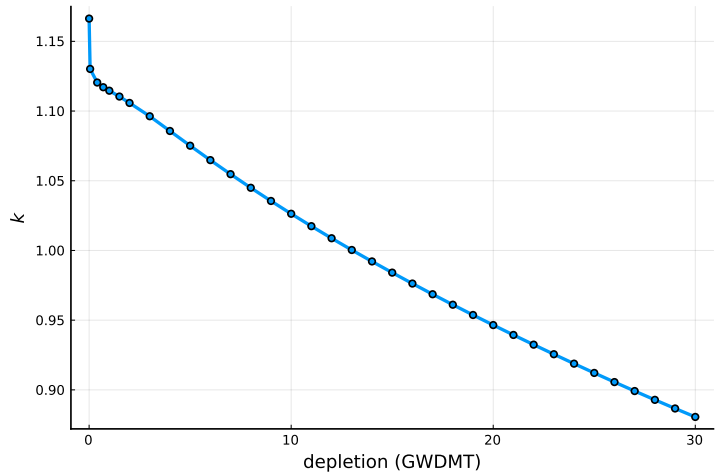
For each of the 3-D assemblies in Table (7.1), we ran a feedback-depletion problem using a predictor-corrector timestepping scheme and a fixed boric acid concentration (1300 [ppm]). All problems were run up to an burnup of 35 GWDMT (Giga-watt days per initial metric ton of heavy metal), where we use the term “burnup” to refer

to the amount that a problem has been depleted. In order to capture the beginning-of-cycle Xenon transient, initial timesteps were finer than those later-in-cycle. To be more specific, initial timesteps were $\Delta t_1 = 0.05$, $\Delta t_2 = 0.35$, $\Delta t_3 = 0.3$, $\Delta t_4 = 0.3$, $\Delta t_5 = 0.5$, and $\Delta t_6 = 0.5$ [GWDMT] for initial assembly burnups of $t_1 = 0.05$, $t_2 = 0.40$, $t_3 = 0.70$, $t_4 = 1.0$, $t_5 = 1.5$, and $t_6 = 2.0$ [GWDMT]. After these initial timesteps were completed, coarser timesteps of $\Delta t = 1$ [GWDMT] were used until an burnup of $t_{\text{end}} = 35$ [GWDMT] was reached.

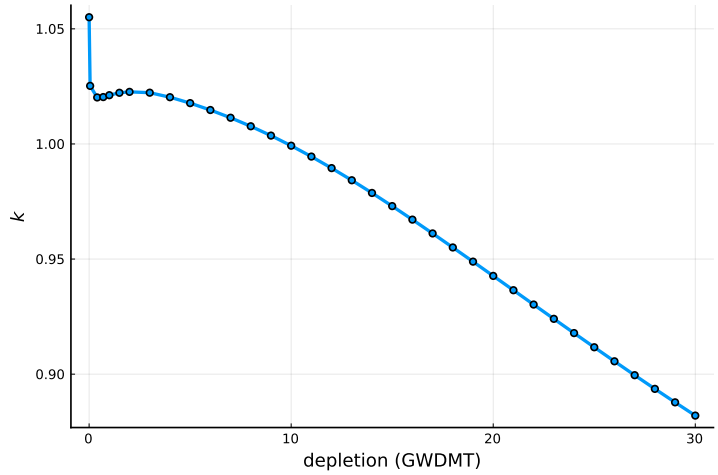
Figs. (7.8)-(7.9) show the k-eigenvalue as a function of burnup (i.e. amount depleted, in [GWDMT]) for assemblies (2a-3d, 2e-3d, 2f-3d) [Figs. (7.8)] and assemblies (2k-3d, 2o-3d, 2p-3d) [Figs. (7.9)]. In each figure, the initial sudden drop in k is caused by the Xenon transient.

In Fig. (7.8a) for assembly 2a-3d, the k-eigenvalue monotonically decreases as a function of burnup. Since assembly 2a-3d contains no burnable absorbers (Pyrex inserts of Gadolinia incorporated rods), reactivity is continuously removed from the system as ^{235}U is burned away. Besides 2a-3d [Fig. (7.8a)], all other assemblies in Figs. (7.8)-(7.9) show an upward trend in reactivity k following the initial downward Xenon transient. These upward trends in k correspond to the burning away of burnable absorbers that dominate the effect of fissionable isotope burnup. Eventually, the burnable absorbers are sufficiently burned off that the fuel burnup dominates, and the criticality k then trends downwards. By comparing 2e-3d [Fig. (7.8b)] to 2f-3d [Fig. (7.8c) with 12 and 24 Pyrex inserts, respectively, we see that this effect becomes more severe as more burnable absorbers are included in an assembly. The same relationship can be observed by comparing Fig. (7.9b) to Fig. (7.9c), with 12 and 24 Gadolinia-infused rods, respectively.

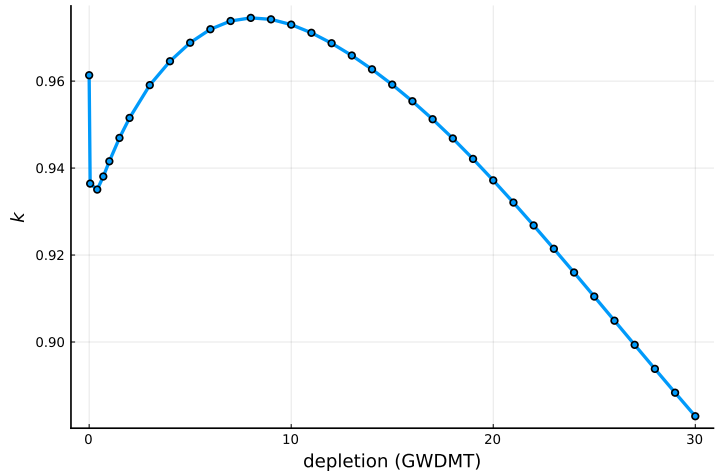
Figs. (7.10)-(7.11) show the number of outer iterations required per timestep for assemblies (2a-3d, 2e-3d, 2f-3d) [Figs. (7.10)] and (2k-3d, 2o-3d, 2p-3d) [Figs. (7.11)] feedback-depletion simulations. In these figures, we show curves for R-CMFD with three different relaxation factors: $\alpha = 0.5$ (MPACT's default), 0.7, and 0.9, along with a single NILO-A curve in each figure. All R-CMFD and NILO-A runs used CTF as the high-order thermal hydraulics solver $\mathcal{L}$. NILO-A runs used simTH for their low-order thermal hydraulics operator L. Since we used a quasi-static predictor-corrector time-discretization, the number of outer iterations shown for a given timestep is the sum of the outer iterations required by two steady-state, eigenvalue, feedback problems. In other words, the outer iterations plotted are the sum of predictor- and corrector-step outer iterations.



(a) Assembly 2a-3d

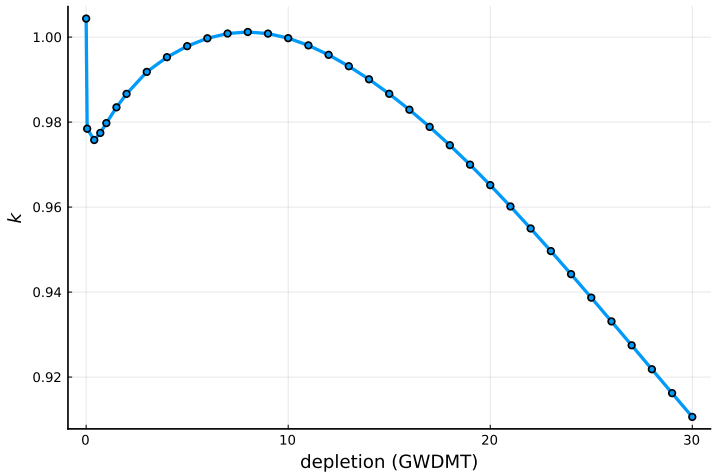


(b) Assembly 2e-3d

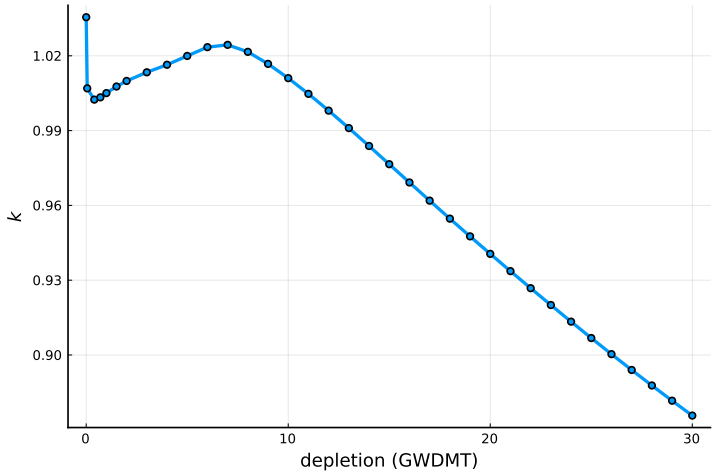


(c) Assembly 2f-3d

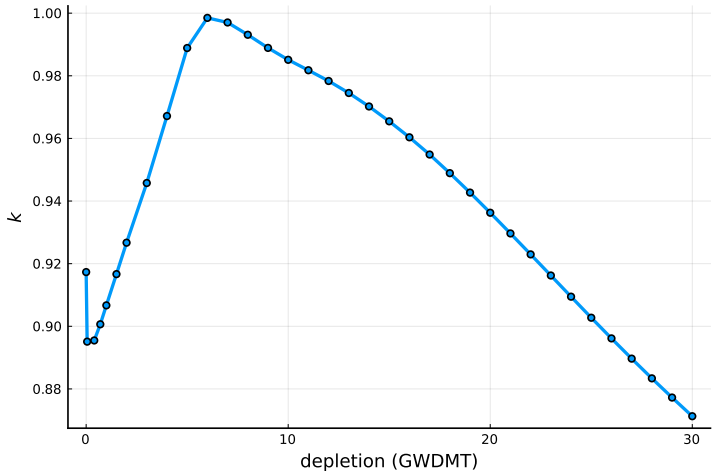
Figure 7.8: Feedback-Depletion k-Eigenvalue vs. Burnup.



(a) Assembly 2k-3d



(b) Assembly 2o-3d



(c) Assembly 2p-3d

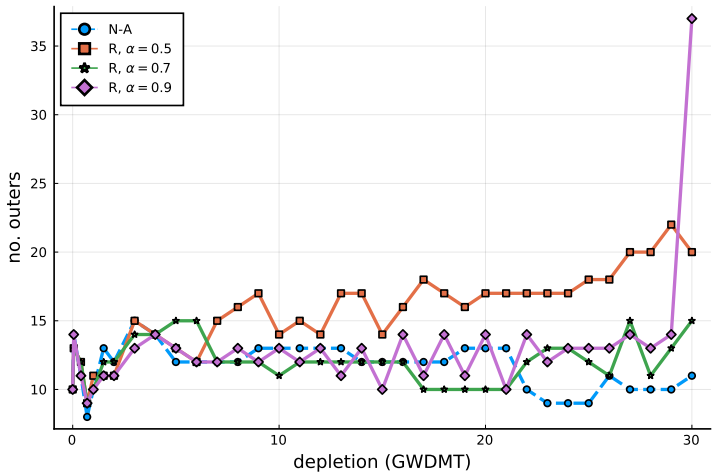
Figure 7.9: Feedback-Depletion k-Eigenvalue vs. Burnup.

There is a consistent trend among the R-CMFD curves in Figs. (7.10)-(7.11). Namely, R-CMFD outer iterations per timestep start out relatively low at the beginning of the depletion cycle, but they trend upwards as problems reach a later stage. In some cases, for example $\alpha = 0.9$ in 2f-3d, 2k-3d, and 2p-3d and $\alpha = 0.5$ in 2p-3d, simulations were halted since they triggered our max threshold of 100 outer iterations per predictor- or corrector- step. For these simulations, R-CMFD would likely diverge and never complete the depletion problem. Another R-CMFD trend in Figs. (7.10)-(7.11) is that $\alpha = 0.7$ seems to be the best relaxation factor among the tested $\alpha = 0.5, 0.7, 0.9$. (This indicates that for the problems tested, $\alpha = 0.7$ is a better relaxation factor than MPACT's current default of $\alpha = 0.5$.)

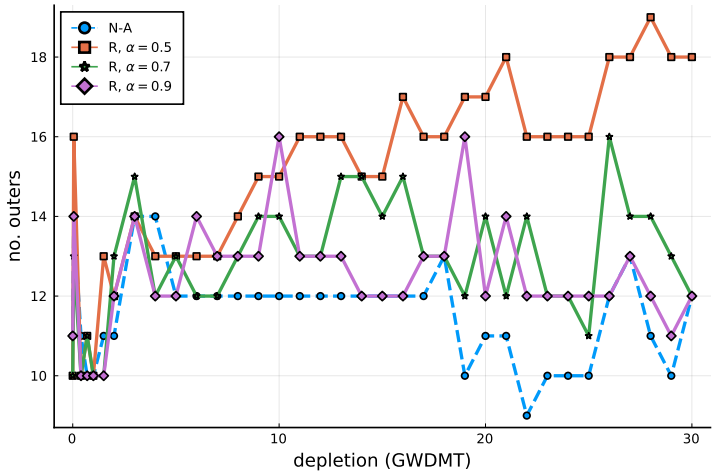
NILO-A curves in Figs. (7.10)-(7.11) show the behavior that we expect from the 1-D numerical results presented in Ch. 6. Namely, NILO-A outer iteration curves remain relatively flat at a low outer iteration count throughout the entire depletion problem. This shows that NILO-A succesfully minimizes the number of transport sweeps and high-order thermal hydraulics evaluations required to solve the feedback-depletion problems. Unlike R-CMFD, NILO-A is unaffected by changing isotopics as the depletion problem progresses. In fact, NILO-A outer iterations per timestep tend to decrease slightly late in the depletion problem. In terms of outer iterations per timestep, NILO-A bests all R-CMFD relaxation factors $\alpha = 0.5, 0.7, 0.9$ for every assembly in Figs. (7.10)-(7.11).

We mention that the stable, optimal convergence rate of NILO-A and the unstable, sub-optimal convergence rate of R-CMFD seen in Figs. (7.10)-(7.11) match with our 1-D numerical results from Ch. 6. Experience tells us that R-CMFD behaves poorly on certain problem types. It appears as though the late-in-cycle isotopics of these assembly feedback-depletion problems make for difficult R-CMFD simulations. We do not fully understand the reason behind this. Further studies are needed to ascertain why fresh-fuel presents little difficulty to R-CMFD while depleted fuel causes issues. We also mention that if we had started our assembly feedback-depletion problems with an already-depleted assembly, there would be an even wider gap in iterative performance between R-CMFD and NILO-A than what is shown in Figs. (7.10)-(7.11).

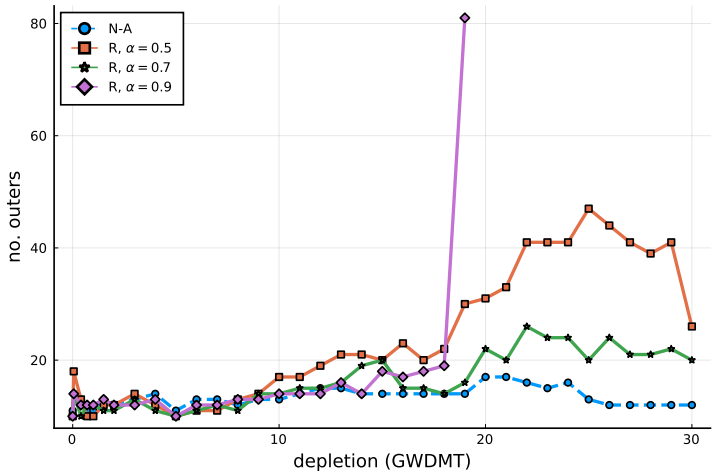
However, outer iteration counts are not the sole metric used to measure the efficiency of a method. A different measure of performance is retrieved through program runtimes. We consider this next.



(a) Assembly 2a-3d

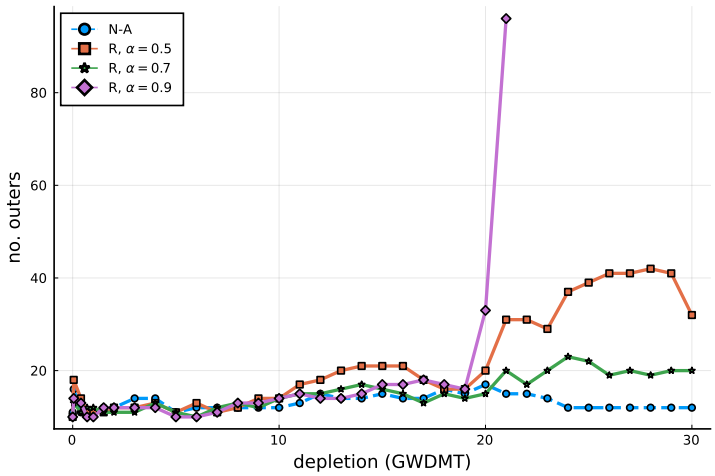


(b) Assembly 2e-3d

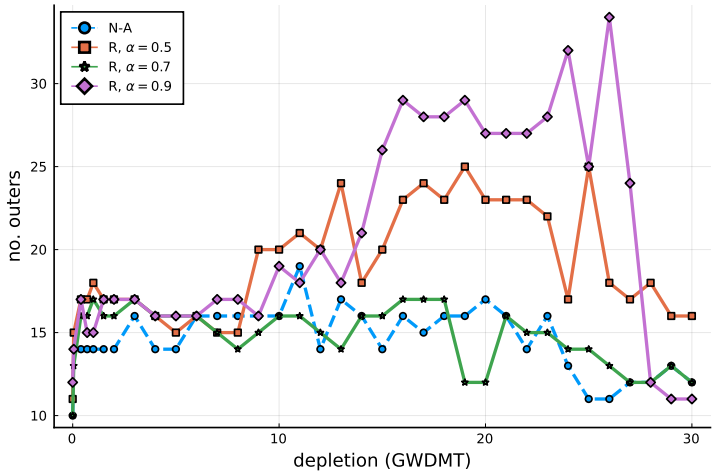


(c) Assembly 2f-3d

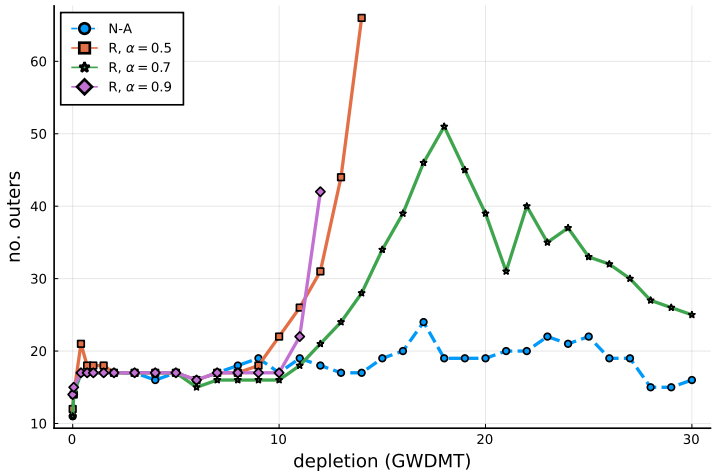
Figure 7.10: Outer Iterations / Feedback-Depletion Timestep (1/2).



(a) Assembly 2k-3d



(b) Assembly 2o-3d



(c) Assembly 2p-3d

Figure 7.11: Outer Iterations / Feedback-Depletion Timestep (2/2).

In Table (7.4), we summarize the results from Figs. (7.10)-(7.11) by showing the total number of outer iterations required by each assembly feedback-depletion simulation (summed over all timesteps). Missing entries correspond to divergent simulations.

Table 7.4: Assembly Feedback-Depletion Outer Iterations (Total).

i.d.	R ($\alpha = 0.5$)	R ($\alpha = 0.7$)	R ($\alpha = 0.9$)	NILO-A
2a-3d	540	420	454	408
2e-3d	526	449	435	404
2f-3d	805	565	–	469
2k-3d	739	527	–	458
2o-3d	662	517	716	508
2p-3d	–	902	–	627

Next, we give representative timing results comparing R-CMFD and NILO-A. These results were taken from the same feedback-depletion simulations that were used to produce Figs. (7.10)-(7.11). In Table (7.5), we list timing results for the assembly 2a-3d feedback-depletion problem solved using R-CMFD ($\alpha = 0.5$) and NILO-A. The same timing information for assembly 2f-3d appears in Table (7.6).

Table 7.5: Feedback-Depletion Problem Timing, 2a-3d.

	R-CMFD ($\alpha = 0.5$)		NILO-A	
Outer Iters.	540		408	
Procedure	Time [hours]	Time/It [sec]	Time [hours]	Time/It [sec]
MOC	0.36	2.4	0.66	5.8
Nodal	0.07	0.5	0.05	0.4
CMFD	2.08	13.9	0.83	7.3
COBRA-TF	0.96	6.4	0.48	4.2
Depletion	0.04	0.3	0.04	0.4
XS and etc.	0.57	3.8	3.52	31.1
Total	4.07	27.1	5.59	49.3

Table 7.6: Feedback-Depletion Problem Timing, 2f-3d.

	R-CMFD ($\alpha = 0.5$)		NILO-A	
Outer Iters.	805		469	
Procedure	Time [hours]	Time/It [sec]	Time [hours]	Time/It [sec]
MOC	0.60	2.7	0.71	5.4
Nodal	0.09	0.4	0.06	0.5
CMFD	3.60	16.1	0.84	6.4
COBRA-TF	1.58	7.1	0.62	4.8
Depletion	0.04	0.2	0.04	0.3
XS and etc.	0.89	4.0	3.56	27.3
Total	6.80	30.4	5.82	44.7

Before comparing timing results between R-CMFD ($\alpha = 0.5$) and NILO-A in Tables (7.5)-(7.6), we mention a small inconsistency originating from a NILO-A implementation detail. In MPACT's implementation of R-CMFD, cross section self-shielding calculations, which require transport sweep evaluations, are optionally performed each outer iteration, depending on whether temperatures and/or densities have changed appreciably enough between outer iterations. Change thresholds may be set as a user-specified input parameter; otherwise, they are set at default values. In our NILO-A implementation, the self shielder is called every outer iteration. We did not attempt to limit the number of self shielder evaluations in our NILO-A implementation. Because of this, the NILO-A timing results in Tables (7.5)-(7.6) have inflated Method of Characters (MOC) solver times as compared to R-CMFD.

Comparing outer iteration curves between R-CMFD ($\alpha = 0.5$) and NILO-A for assemblies 2a-3d and 2f-3d in Figs. (7.10a) and (7.10c), we see that for each problem, NILO-A requires fewer outer iterations in total to reach the end of the depletion problem. This would suggest that NILO-A simulations would finish faster than their R-CMFD ($\alpha = 0.5$) counterparts. However, this is not necessarily the case, as can be observed in Tables (7.5)-(7.6). In Table (7.5), we see that R-CMFD ($\alpha = 0.5$) took 4.07 hours to finish the simulation, while NILO-A took 5.59 hours. Looking more closely at Table (7.5) we see that while NILO-A saved time on Nodal, CMFD, and CTF solves, it spent considerably more time than R-CMFD on cross section updates. Since we are considering depletion problems, there are many isotopes being tracked in the system, and this increases the cost of cross section updates. Since NILO-A's nonlinear diffusion solve includes unapproximated cross section updates, and these are of significant expense, they place a noticeable penalty on NILO-A's

runtime performance. In 2f-3d [Table (7.5)], the increased time for NILO-A cross section updates is counteracted enough by lower outer iteration counts that NILO-A (5.82 hours) finishes in less time than R-CMFD ($\alpha = 0.5$) (6.80 hours).

We also mention that even though NILO-A required fewer outer iterations to solve the 2a-3d [Fig. (7.10a)] and 2f-3d [Fig. (7.10c)] feedback-depletion problems, the Method of Characteristics (MOC)-based transport sweep runtimes for NILO-A in Tables (7.5)-(7.6) are greater than those for R-CMFD. One would expect that fewer outer iterations would lead to fewer MOC sweeps. However, the increased NILO-A MOC runtime is caused by our lack of optimization to limit the number of self-shielding evaluations in our NILO-A implementation. In R-CMFD simulations, self-shielding is only performed when necessary.

The outer iteration counts in Figs. (7.10)-(7.11) and timing results in Tables (7.5)-(7.6) show that while NILO-A successfully limits the number of outer iterations required to solve assembly feedback-depletion problems, the additional runtime burden from cross section evaluations in NILO-A's nonlinear diffusion problem may lead to the method being less efficient than R-CMFD in terms of overall runtime performance. This strongly suggests that an implementation of the full NILO-CMFD method in MPACT, including approximate cross section evaluations in the nonlinear diffusion solve, would lead to significant improvements over the standard R-CMFD approach. Unfortunately, due to time constraints, we were not able to implement the full NILO-CMFD method in MPACT, in order to test this hypothesis.

7.4.2. Quarter Core

In order to compare R-CMFD ($\alpha = 0.5$) and NILO-A on a full-scale reactor core problem, we ran each method on CASL Progression Problem 9, the Watts Bar Unit 1 Cycle 1 Feedback-Depletion problem. We refer to this problem as the "quarter core" problem throughout this section. Like the assembly feedback-depletion problems presented above, the quarter-core problem used CTF for the high-order thermal hydraulics operator $\mathcal{L}$. In the NILO-A run, we used the simTH operator as the low-order thermal hydraulics L. Unlike the assembly feedback-depletion problems covered in the previous section, the quarter core problem is a boron search problem. Thus, at each timestep the boric acid concentration required to reach criticality ($k = 1$) is determined as a problem unknown rather than the k-eigenvalue itself. Since the quarter core problem models an operating reactor, it includes a detailed operating his-

tory, with operational parameters changing throughout the depletion cycle. In Tables (7.7)-(7.8), we list problem information including system rated power, inlet coolant temperature, control rod bank (D) position, and burnup (in [EFPD], Effective Full Power Days) for each of the 32 statepoints in the quarter core simulation.

Table 7.7: P9 (Quarter Core) Statepoint Information (1/2).

STATE	Power (%)	Inlet Temp. [°F]	Rodbank (D) Pos.	Burnup [EFPD]
0	0.0	557.0	186	0.0
1	65.7	557.6	192	9.0
2	99.7	558.1	219	32.0
3	98.0	558.2	218	50.0
4	100.0	558.6	219	64.0
5	99.7	558.7	215	78.0
6	99.7	558.6	217	92.7
7	99.8	558.8	220	105.8
8	99.8	558.4	220	120.9
9	99.5	557.9	219	133.8
10	98.0	558.0	214	148.4
11	95.1	557.9	216	163.3
12	94.8	557.9	214	182.2
13	99.8	557.8	220	194.3
14	93.9	557.5	218	207.7
15	100.1	558.0	222	221.1

Table 7.8: P9 (Quarter Core) Statepoint Information (2/2).

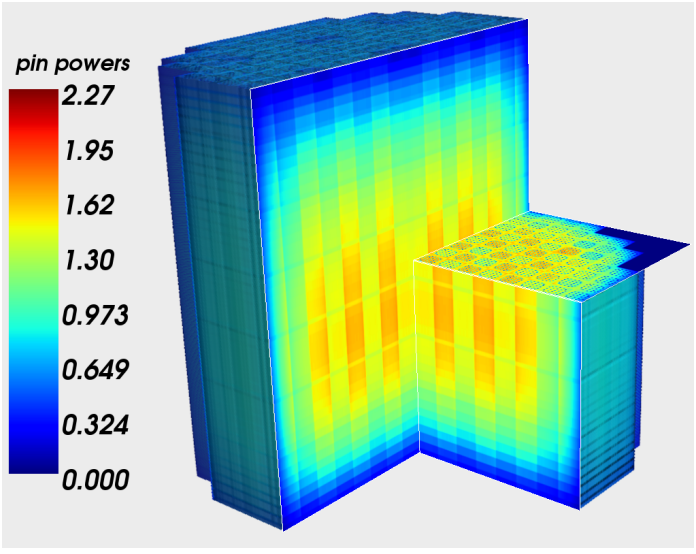
STATE	Power (%)	Inlet Temp. [°F]	Rodbank (D) Pos.	Burnup [EFPD]
16	99.7	557.7	220	238.0
17	100.2	557.6	222	250.0
18	95.6	557.9	211	269.3
19	96.4	558.1	215	282.3
20	93.4	557.4	211	294.6
21	99.7	557.5	217	312.1
22	98.0	557.6	215	326.8
23	99.4	557.7	220	347.8
24	99.9	557.8	219	373.2
25	86.9	556.7	202	373.2
26	86.9	556.7	202	392.3
27	99.6	558.0	220	392.3
28	99.6	558.0	220	398.6
29	89.9	557.1	224	410.7
30	78.8	556.3	228	423.6
31	64.5	554.9	230	441.0

Figs. (7.12) show representative quarter-core solution profiles from STATE 4 [Table (7.7)]. We plot normalized pin powers, moderator temperatures [°C], and volume average fuel temperatures [°C] in Figs. (7.12a), (7.12b), and (7.12c), respectively.

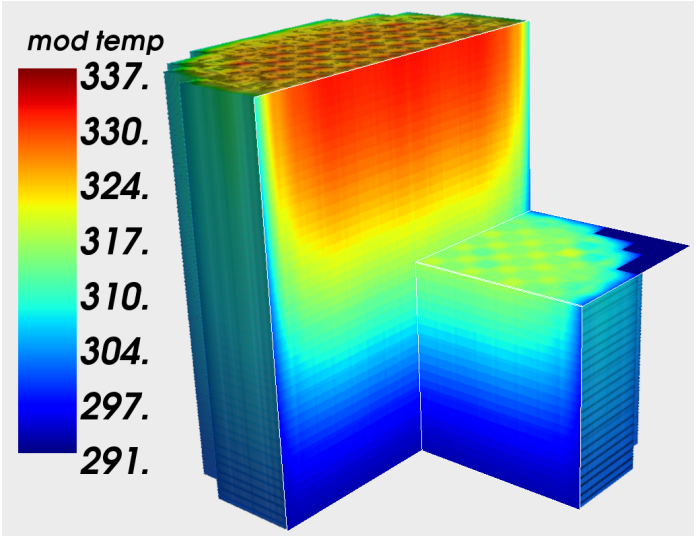
Figs. (7.12) demonstrate that the assembly axial solution profiles shown in Figs. (7.5)-(7.7) are consistent throughout the entire core. In particular:

- The axial dependence of the pin-powers is roughly cosine-shaped.
- The axial dependence of the moderator (fluid) temperatures is monotonically-increasing with z.
- The axial dependence of the fuel temperatures closely follows that of the pin powers.
- There are two characteristic temperatures (fuel and fluid) that have distinct axial dependences.

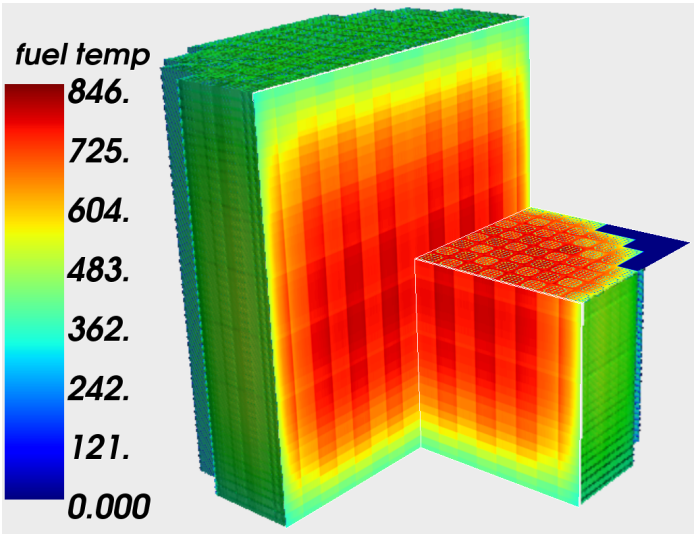
As we noted earlier, these are all features of the 1-D model solutions discussed in Ch. 6. Thus, the 3-D quarter-core solutions again demonstrate the validity of the 1-D model, as a qualitatively realistic tool for analyzing computational methods (discretization and iterative) for 3-D PWR multiphysics problems.



(a) Pin Powers



(b) Moderator Temp. [°C]



(c) Volume Average Fuel Temp. [°C]

Figure 7.12: Quarter-Core 3-D Solution Profiles.

Fig. (7.13) plots the boric acid concentration as a function of burnup for the quarter-core feedback-depletion problem. This is commonly referred to as a "boron letdown curve". The initial sharp decrease in critical boric acid concentration is due to the initial Xenon transient. We note that in the last few timesteps of the problem, a critical boron concentration of 0 is calculated. (For these burnups, the core is subcritical.) Since the operating Watts Bar Unit 1 Cycle 1 core remained critical during these times, this indicates some minor inconsistencies between the operating physical core and the simulation.

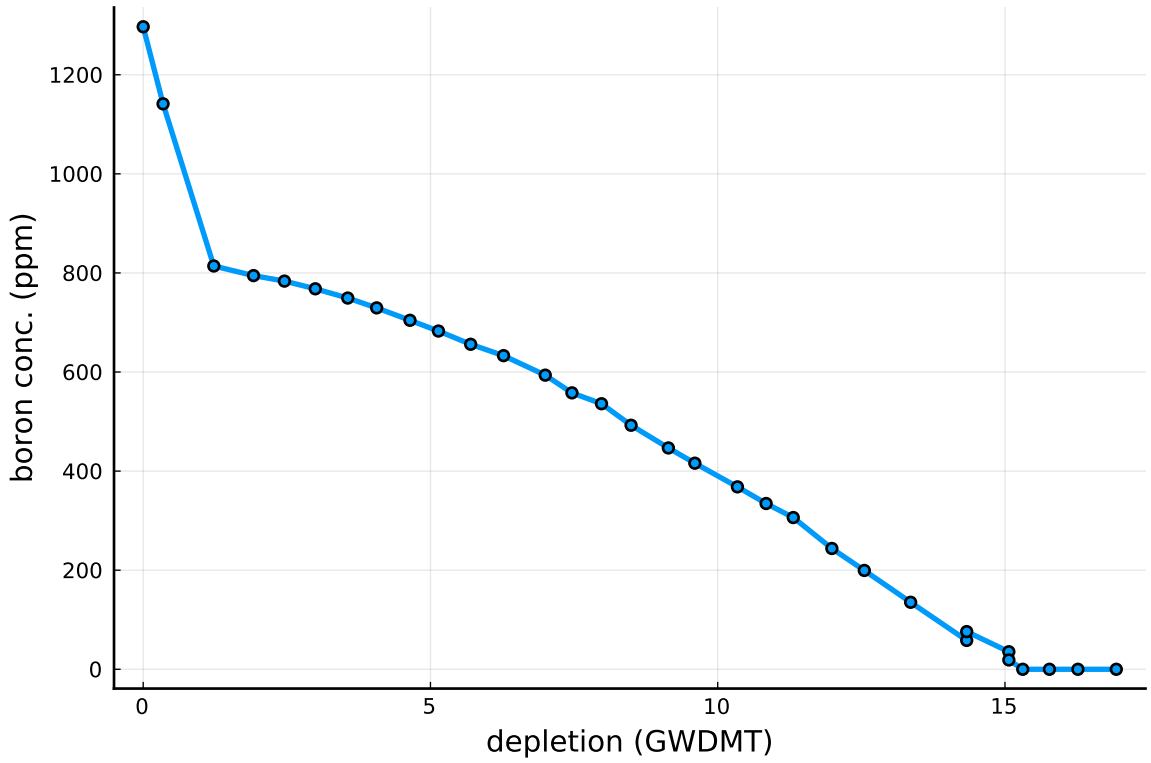


Figure 7.13: Quarter Core (P9) Critical Boron Conc. vs. Burnup.

Fig. (7.14) shows the number of outer iterations required by each statepoint of the quarter core simulation [Tables (7.7)-(7.8)] as a function of burnup [GWDMT]. Both R-CMFD (using the default $\alpha = 0.5$) and NILO-A outer iteration curves are plotted. As with the assembly outer iterations discussed earlier in Figs. (7.10)-(7.11), R-CMFD outer iterations tend to increase later in the depletion problem, whereas NILO-A outer iterations remain low and fairly constant. We note that at certain burnups in Fig. (7.14), there are two markers each for R-CMFD and NILO-A. This is because certain statepoints in Tables (7.7)-(7.8), particularly those following larger changes in system parameters (e.g., power, control rod bank position), were not tasked with performing a predictor-corrector depletion timestep, but instead with calculating a single steady-state solve to update solution information prior to the next timestep. Overall, Fig. (7.14) shows, for a full-scale reactor-core problem, that NILO-A successfully limits the total number of outer iterations as compared to R-CMFD.

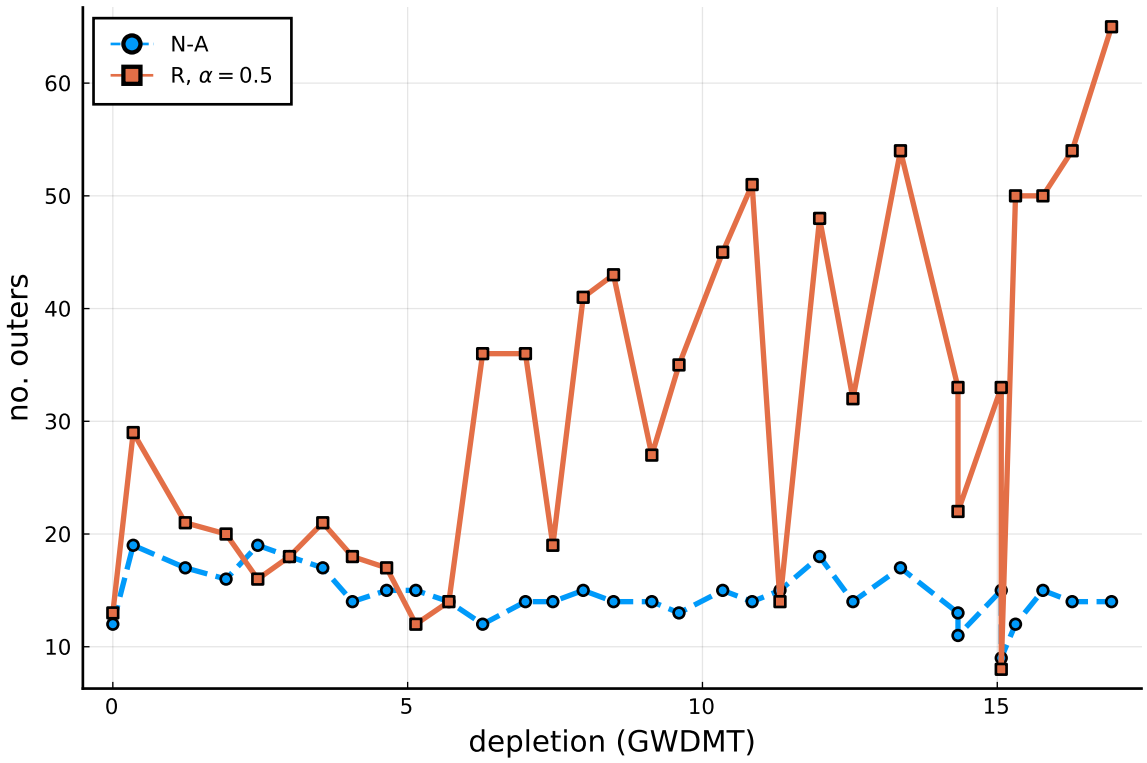


Figure 7.14: Quarter Core (P9) Outer Iters. / Statepoint vs. Burnup.

As with the assembly results discussed previously, the outer iteration counts in Fig. (7.14) do not tell the entire story. In Table (7.9), we show timing results for the R-CMFD and NILO-A quarter-core simulations. We see that even though NILO-A required considerably fewer outer iterations, as observed in Fig. (7.14), its runtime more than doubled that of R-CMFD. Looking more closely at the runtimes of individual routines in Table (7.9), we see similar behavior to what was observed in the assembly Tables (7.5)-(7.6). Again, NILO-A's excessive use of nuclear data evaluations led to a major runtime expense. Again, we mention that MOC timings in the NILO-A columns of Table (7.9) are inflated, due to our lack of optimization to limit NILO-A self-shielding evaluations. Also, we mention another inconsistency affecting CMFD timings in Table (7.9). In our NILO-A run, we set the maximum number of allowed inner iterations per outer iteration at 100 to ensure convergence of the NILO-A nonlinear diffusion problem. In the R-CMFD run, this parameter was set at its default value of 20.

As with the assembly problems, we expect NILO-A cross-section-related runtime issues to be corrected by the full NILO-CMFD method's use of approximate cross section updates. However, we cannot show results confirming this hypothesis, as

we have only implemented NILO-A in MPACT at this point in time. Further code optimization may also help to improve NILO timing results.

Table 7.9: Feedback-Depletion Problem Timing, P9 (Quarter Core).

	R-CMFD ($\alpha = 0.5$)		NILO-A	
Outer Iters.	995		468	
Procedure	Time [hours]	Time/It [sec]	Time [hours]	Time/It [sec]
MOC	17.99	65.1	17.73	136.4
Nodal	4.56	16.5	1.96	15.1
CMFD	4.31	15.6	11.50	88.5
COBRA-TF	5.20	18.8	3.72	28.6
Depletion	0.53	1.9	0.87	6.7
XS and etc.	4.60	16.6	62.79	483.0
Total	37.19	134.6	98.56	758.2

7.5. Step 4 Iteration Performance

The previous paragraphs show that the computational expense of performing the Step 4 operations is significantly greater for NILO-A than for R-CMFD, and we have argued that a major reason for this is the excess number of expensive cross section evaluations required in NILO-A outer iterations. (This should be adequately dealt with when the full NILO-CMFD method is implemented.)

However, a related question can be considered, concerning the basic fundamental iteration process used by MPACT to solve the R-CMFD and NILO-A Step 4 equations: Do the extra calculations in Step 4 of NILO-A (approximate TH solve, cross section evaluation) affect the rate of convergence of these basic iterations?

MPACT refers to these basic iterations as *Linear Solver Iterations* (LSI)'s. (These have not been previously discussed in this thesis, and we will not discuss them in detail here.) A single LSI involves only simple algebraic manipulations and is inexpensive. (Multiple LSI's are required to perform a single Power Iteration. Multiple Power Iterations are required to solve the R-CMFD Step 4 diffusion equation; and likewise, multiple Power Iterations are required to "solve" the NILO-A diffusion equation in each inner iteration.) In general, MPACT can employ several thousand LSI's to solve typical coarse-grid diffusion problems.

MPACT provides the number of LSI's per outer iteration for each calculation. As an example, Table (7.10) shows the number of LSI's per outer iteration required by

R-CMFD and NILO-A runs on five of the 3-D PWR assembly feedback-depletion problems explored in this chapter. The results in Table (7.10) show how the extra calculations in Step 4 of NILO-A affect the number of LSI's required to converge:

Table 7.10: Linear Solver Iterations / Outer Iteration.

assem.	R-CMFD ($\alpha = 0.5$)	NILO-A
2a-3d	2807	1400
2e-3d	2723	1369
2f-3d	3418	1363
2k-3d	3270	1400
2o-3d	2987	1429

This table shows that the number of LSI's required to solve Step 4 of NILO-A is roughly half the number required to solve Step 4 of R-CMFD! This result is unexpected and will require further study to understand. However, it indicates that the added complexity of the Step 4 NILO-A equations should enhance, rather than impede, the basic MPACT LSI process for solving these equations.

7.6. Conclusions

In this chapter, we have discussed the implementation and performance of NILO-A on 3-D multiphysics problems run using the MPACT neutronics and CTF thermal hydraulics codes.

By comparing NILO-A and R-CMFD solution fields for fresh-fuel, steady-state, assembly feedback problems while varying problem convergence criteria, we confirmed that our NILO-A implementation converges to the proper (R-CMFD) solution. Outer iteration counts were nearly identical between NILO-A and R-CMFD for these fresh-fuel simulations, suggesting that R-CMFD solves these problems with little difficulty. (We note that R-CMFD's simplicity, efficiency, and ability to solve many problems of practical interest can make this approach difficult to outperform. This appears to be the case for the aforementioned fresh-fuel assembly calculations.)

However, for more difficult problems, we expect NILO-A and NILO-CMFD convergence rates to outpace R-CMFD. The reason for this is that while R-CMFD's use of relaxation is rather ad hoc, the principles underlying NILO-A and NILO-CMFD are firmly based in the physics of the problem being solved. This concept appears in our motivation and derivation of NILO-A and NILO-CMFD presented in earlier

chapters. On more difficult problems, we expect NILO-A and NILO-CMFD to be more efficient and robust than R-CMFD.

By considering a series of assembly feedback-depletion problems, we found this to be the case. By comparing outer iteration curves for several feedback-depletion problems, we noticed the erratic and sub-optimal behavior of R-CMFD, especially during later timesteps. At the same time, NILO-A converged smoothly and quickly for all problems tested.

Unfortunately, owing to the large cost of nuclear data evaluations in feedback-depletion problems (something we did not forsee when planning our initial implementation in MPACT), NILO-A simulations suffered a significant runtime penalty. We expect this issue to be resolved by implementation of the full NILO-CMFD method in MPACT. However, we have not had the time to test this hypothesis.

By considering quarter-core feedback-depletion results, we arrived at the same conclusions drawn from the assembly feedback-depletion problems mentioned above. Namely, while NILO-A outperforms R-CMFD in terms of outer iteration performance, it suffers significant runtime penalties over R-CMFD in problems where nuclear data evaluations are costly. Again, we expect NILO-CMFD to combat this issue.

Overall, this chapter shows that while NILO-A shows stability and optimal convergence, in situations where nuclear data evaluations are expensive, the full NILO-CMFD method would be preferable. In the next chapter, we propose this extension of the current chapter as "low-hanging" fruit for future work.

Chapter 8.

Summary, Conclusions, & Future Work

8.1. Summary & Conclusions

In this thesis, we have proposed, implemented, and tested two Nonlinearly Implicit Low Order (NILO)-CMFD multiphysics iteration schemes: the incomplete NILO-A method and the full NILO-CMFD method. NILO-A and NILO-CMFD were designed as approximations to the X-CMFD method for the robust and efficient iterative solution of neutron transport – thermal hydraulics multiphysics problems. For a suite of discrete, 1-D, model multiphysics problems, we showed that both NILO-A and NILO-CMFD preserve the optimal single-physics CMFD outer iteration spectral radius of $\rho \approx 0.22$. Also, for a series of 3-D reactor assembly and quarter-core feedback-depletion simulations, we again showed that NILO-A converges optimally and outperforms the standard R-CMFD method in terms of outer iterations to convergence. Below, we collate and summarize the topics covered in each chapter of this thesis.

In Ch. 1, we motivated our interest in neutron transport – thermal hydraulics multiphysics problems. First, we discussed the nonlinear interaction between neutron radiation and matter and explained why this physical process must be included to accurately simulate operating nuclear reactor cores.

Then, we presented a historical review of single-physics neutron transport iterative methods. While the main focus of this thesis is on multiphysics iterative methods, the NILO-A and NILO-CMFD methods are both based upon the Coarse Mesh Finite Difference (CMFD) single-physics transport acceleration scheme. Thus, a review of

the single-physics-focused research leading up to the proposal of CMFD was necessary. Afterwards, we introduced multiphysics iterative methods. We split this broad topic into two subcategories: tightly- and loosely-coupled approaches. After summarizing tight coupling, we quickly dismissed this topic in favor of loose coupling. In our view, tight coupling's requirement of residual access limits the ability to repurpose existing single-physics thermal hydraulics codes for multiphysics simulations. Thus, we focused on loose coupling for its ease-of-implementation. We covered the main sources of inspiration for NILO-A and NILO-CMFD, including Shen's [91, 93, 95] and Walker's [110, 111] separate work on X-CMFD-like multiphysics iterative methods.

In Ch. 2, we detailed the single-physics components that make up the neutron transport – thermal hydraulics multiphysics problem. We illustrated the dependence of neutron macroscopic cross sections (i.e. neutron interaction probabilities) on background material temperature and density. By assuming that a direct dependence of density on temperature was available, we were able to eliminate an explicit treatment of density in most of the thesis. Instead, in the equations we write, density dependence is implied by the temperature dependence. We introduced the continuous, λ-eigenvalue formulation of the neutron transport equation and also introduced our general notation for describing thermal hydraulics using the operator $\mathcal{L}$.

In Ch. 3, we focused on iterative methods for single-physics neutron transport problems. We described Source Iteration (SI) as iterated inversions of the transport leakage-collision operator. We commented on SI's poor convergence rate for optically thick eigenvalue problems, and we introduced the Coarse Mesh Finite Difference (CMFD) method as an acceleration scheme for SI.

In Ch. 4, we outlined a collection of CMFD-based multiphysics iterative methods. First, we outlined the standard Multiphysics CMFD (M-CMFD) and Relaxed CMFD (R-CMFD) methods as multiphysics modifications to the single-physics CMFD approach. We noted that the inclusion of a relaxation factor in R-CMFD acts as a user-tunable "knob" designed to stabilize the M-CMFD method in the presence of strong feedback. We discussed the drawbacks of relaxation, noting that it is not a panacea. In certain situations, regardless of the relaxation factor chosen, R-CMFD will fail (become unstable).

Next, we introduced two purely theoretical methods: Theoretical Method 0 (TM-0) and Theoretical Method 1 (TM-1). Both TM-0 and TM-1 converge in one iteration (i.e. their spectral radii are zero). However, neither approach can be efficiently implemented in a computer-code. Rather, they serve a pedagogical role.

We used TM-1 to introduce the concept of nonlinearly implicit parameters in transport-corrected diffusion equations. We defined a term as being *nonlinearly implicit* if it resides in a transport-corrected diffusion equation and depends, in any way, on the scalar flux solution to that same diffusion problem. When transport-corrected diffusion problems include nonlinearly implicit temperatures and nuclear data, a *nonlinear* transport-corrected diffusion eigenvalue problem must be solved.

We also defined fully nonlinearly implicit temperatures and nuclear data as being evaluated with high-order (i.e. unapproximated) thermal hydraulics (i.e. $\mathcal{L}$) and nuclear data update physics. We noted that TM-1's "instant" convergence originates from its fully nonlinearly implicit treatment of all parameters in its nonlinear diffusion problem: transport correction vector, temperatures, and nuclear data.

We then derived the X-CMFD method, as an approximation to TM-1, by explicitly evaluating (i.e. lagging) the transport-correction vector in TM-1's nonlinear diffusion problem. We commented on X-CMFD's strong outer iteration performance, but noted that the approach typically places an excessive computational burden on thermal hydraulics solvers and nuclear data update routines.

We then proposed the NILO-A and NILO-CMFD methods as approximations to X-CMFD, where the fully nonlinearly implicit evaluation of temperatures and nuclear data is loosened by instead evaluating them using approximate update physics. Rather than being fully nonlinearly implicit, in NILO, these terms are *approximately nonlinearly implicit*. We proposed NILO-A as an incomplete form of the full NILO-CMFD method, useful for when unapproximated nuclear data updates are relatively inexpensive.

In Chs. 5 & 6, we shifted our focus towards 1-D, simplified, neutron transport – thermal hydraulics model problems derived from the equations describing a 3-D Pressurized Water Reactor (PWR) fuel-pin unit cell. Using standard discretization techniques, we described the M-CMFD, R-CMFD, X-CMFD, NILO-A, and NILO-CMFD methods as they applied to the discretized 1-D model. After outlining each approach, we introduced a set of problem parameters derived from 3-D PWR benchmark problems. Examining solution information from the 1-D model, we noticed qualitative similarities with the results observed in our 3-D MPACT simulations presented in Ch. 7. This gave us confidence in the validity and usefulness of the 1-D model. Testing NILO-A and NILO-CMFD on the discrete 1-D model, we showed that their outer iteration convergence rates appear to be nearly identical to X-CMFD. This was our hope; that the approximations introduced into X-CMFD to form NILO-A

and NILO-CMFD would not affect X-CMFD's optimal outer iteration convergence rate. At the same time, the standard R-CMFD method faltered on our 1-D model problems, regardless of the chosen relaxation factor.

In Ch. 7, we focused on realistic 3-D reactor simulations using the Michigan Parallel Characteristic-based Transport (MPACT) code coupled to the COBRA-TF (CTF) subchannel thermal hydraulics code. We introduced MPACT as the primary deterministic transport solver in the Virtual Environment for Reactor Applications (VERA) multiphysics environment, developed as a part of the Consortium for the Advanced Simulation of Light Water Reactors (CASL) project. We described MPACT's implementation of the standard Relaxed CMFD (R-CMFD) method and then discussed our implementation of NILO-A in MPACT. Due to time constraints, and some incorrect assumptions about the computational expense of nuclear data updates in the feedback-depletion problems we were interested in testing, we did not implement the full NILO-CMFD method in MPACT. We save this implementation for future work, discussed below.

Testing R-CMFD and NILO-A on a series of realistic 3-D assembly feedback-depletion problems, we observed that NILO-A achieved an optimal convergence rate throughout the depletion cycle. In fact, NILO-A iterations to convergence trended downwards as we approached the end of a cycle. In contrast, R-CMFD iterative performance degraded as the depletion problems progressed. We observed similar behavior for NILO-A and R-CMFD on a full-scale, quarter-core, feedback-depletion problem.

Unfortunately, when comparing R-CMFD and NILO-A runtime performance, NILO-A was less efficient than R-CMFD in terms of problem runtime in several instances. NILO-A's runtime slowdown was due to the method's over-reliance on nuclear data evaluations. This was especially true in the feedback-depletion problems we tested, since nuclear data evaluations were rather expensive in these problems. We expect that an implementation of the full NILO-CMFD method, which employs approximate nuclear data updates, should address these runtime issues.

Overall, this thesis proposed a new class of Nonlinearly Implicit Low-Order Coarse Mesh Finite Difference (NILO-CMFD) multiphysics iterative methods for neutron transport problems with nonlinear thermal hydraulics feedback. In developing NILO-A and NILO-CMFD, we attempted to make the methods as easy to implement as possible. In order to do this, we based NILO-CMFD on the standard Relaxed CMFD (R-CMFD) multiphysics iterative method. Additionally, both NILO-A and NILO-

CMFD are loosely coupled, to facilitate the use of off-the-shelf thermal hydraulics solvers. As a consequence, our implementation of NILO-A in MPACT was rather straightforward.

We have several ideas to extend the current work to address the shortcomings of NILO-A as it is now implemented in MPACT. These are described next:

8.2. Future Work

Next, we list some thoughts on ways in which the work covered in this thesis can be extended:

3-D NILO-CMFD implementation

As mentioned in Ch. 7, due to incorrect assumptions related to nuclear data evaluation costs (coupled with time constraints), we were unable to implement the full NILO-CMFD method in MPACT. While the NILO-A results in Ch. 7 showed robust outer iteration performance, excessive nuclear data evaluations ultimately led to NILO-A losing out to standard R-CMFD in terms of overall runtime performance. We suggest that an implementation of the full NILO-CMFD method in MPACT should be pursued, in order to present a clearer improvement over R-CMFD and NILO-A.

Why are feedback-depletion problems difficult?

In the 3-D feedback-depletion problems presented in Ch. 7, we noticed that as the depletion cycles progressed, R-CMFD performance degraded. It appears that the change in isotopics during depletion led to a more difficult multiphysics problem for R-CMFD later in cycle. We do not fully understand the reason for this. An exploration of this phenomenon would be an interesting topic of future research. One current hypothesis is that the nearly bi-modal scalar flux shape that appears in significantly depleted reactor cores may be the cause of R-CMFD's late-in-cycle performance issues [48].

Beyond Pressurized Water Reactors (PWR)'s

Both the 1-D and 3-D numerical experiments presented in this thesis were performed on Pressurized Water Reactors (PWR) problems. However, beyond PWR's, other nuclear reactor types exist. For example, Boiling Water Reactors

(BWR)'s [30], High Temperature Gas-cooled Reactors (HTGR)'s [125], Molten Salt Reactors (MSR)'s [69], etc. Considering the number of Boiling Water Reactor (BWR)'s operating within the United States and abroad, as well as the difficulty of BWR simulations [17], we believe an application of NILO-CMFD towards a BWR problem is the next step in the method's development and testing (after implementation of the full NILO-CMFD method in MPACT). The derivations of NILO-A and NILO-CMFD presented in Ch. 4 were quite general and were not tailored to PWR's. Our hope is that NILO-A and NILO-CMFD can be successfully applied to other reactor types. In fact, we suspect that NILO-CMFD may prove even more useful when applied to reactors with more complex thermal hydraulics descriptions. For example, the use of a low-order thermal hydraulics operator may be useful when solving BWR problems with complex multiphase flow.

Comparing L's

In our implementation of NILO-A in MPACT, we used the COBRA-TF (CTF) subchannel code for the high-order thermal hydraulics operator $\mathcal{L}$ and MPACT's internal simplified Thermal Hydraulics (simTH) solver for the low-order L. As the numerical results in Ch. 7 show, this choice of L is suitable for the PWR assembly and quarter-core problems that we tested. Using simTH for L also greatly simplified our implementation of NILO-A in MPACT, since we were able to repurpose this existing thermal hydraulics solver. However, we have not explored how simple L can be made, while still preserving an optimal outer iteration convergence rate for NILO-A and NILO-CMFD. We suspect that as long as L captures low-spatial-frequency thermal hydraulics behavior it should work effectively, but we have not answered the question: "how approximate can L actually be made?"

Theoretical (Fourier) analysis

We did not include the Fourier analysis of NILO-A or NILO-CMFD in this thesis. During the early development of NILO-CMFD, we performed a Fourier analysis on a prototype version of the method that used a different outer iteration ordering. Since this analysis is not strictly consistent with the outlines of NILO-A and NILO-CMFD presented in Chs. 4 & 6, we did not present it here. The results of this analysis indicated that our prototype NILO-CMFD method

quickly eliminated low-spatial-frequency (i.e. "flat") error modes when applied to a variation of the continuous 1-D model. Identical behavior was observed for X-CMFD. In contrast, a Fourier analysis of R-CMFD applied to the same problems showed that this standard method's slow convergence and instability were caused by the misbehavior of these same low-frequency error modes. We suggest that a new Fourier analysis of NILO-A and NILO-CMFD be performed, using the ordering of the methods as outlined in this thesis. We have not done this ourselves, due to time constraints.

Data-Driven L

There is an ongoing research fervor into data-driven numerical methods for the representation and solution of complex, often nonlinear, systems [9, 10, 52, 53, 76]. We envision that data-driven methods could be used to generate suitable low-order thermal hydraulics operators (L) in NILO-A and NILO-CMFD.

Additional feedback mechanisms

This thesis focuses on neutron transport – thermal hydraulics multiphysics problems. However, beyond thermal hydraulics, other feedback mechanisms influence the behavior of reactors. For example, fuel thermal expansion and performance [33, 68, 81, 85, 102], CRUD Induced Power Shifts (CIPS) [18, 127], and CRUD Induced Localized Corrosion (CILC) [86] each affect the reactor multiphysics problem to varying degrees. We envision that the NILO-A and NILO-CMFD methods can be generalized to include these physics as well.

Inner iteration performance

We did not spend significant effort on designing an efficient inner iteration procedure for NILO-A and NILO-CMFD. Our use of the Power Iteration (PI)-based Nonlinear Wielandt Shifted Power Iteration (N-WS-PI) method in Chs. 6 & 7 was mainly done out of convenience. N-WS-PI was simple to implement in our 1-D test code, and most of the components required by the method were already implemented in MPACT. Further work should be done on developing efficient inner iteration schemes for NILO-A and NILO-CMFD. Perhaps this could be pursued using some form of multigrid in energy and space, e.g., the Multilevel in Space and Energy Diffusion (MSED) methodology developed by Yee [123]. In MSED, a Grey (single-group) collapse in energy is used to more efficiently iteratively solve a (transport-corrected) diffusion eigenvalue problem. Since the

fission heat source (h) in NILO-A and NILO-CMFD inner iterations is integrated over energy (i.e. it is Grey), we envision that MSED can be straightforwardly modified to solve the NILO-A and NILO-CMFD nonlinear diffusion eigenvalue problems.

Benefits of partial convergence?

In all of the NILO-A and NILO-CMFD results presented in this thesis, the nonlinear diffusion eigenvalue problem was converged to a suitably tight tolerance. Recent research by Shen [91, 92] has indicated that partially converging the linear diffusion eigenvalue problem in Multiphysics CMFD (M-CMFD) has positive effects on the method's outer iteration convergence rate. Shen showed that partial convergence acts in a similar (but more efficient) manner to a relaxation factor [92]. Combining Shen's Nearly Optimally Partially Converged CMFD (NOPC-CMFD) method [94, 96] with aspects of the NILO-A and NILO-CMFD methods may lead to interesting results.

NILO-CMFD performance with many energy groups

The implementation of the full NILO-CMFD involves the use of a Taylor series approximation to model the behavior of nuclear data that depends on temperature and density. Because of the strong behavior of certain cross sections at resonances, it is not always true that continuous-energy cross sections depend smoothly on temperature. However, because multigroup cross sections are based on integrals of continuous-energy cross sections over energy bins, the behavior of multigroup cross sections should be "smoother" than that of continuous-energy cross sections. The point is that the NILO-CMFD method may become less efficient as the number of energy groups is increased, and the width of the energy bins is decreased. This might be the subject of a future study.

Overall, we view this thesis as a proof-of-concept for the NILO approach. We envision this work as the first step in the application of NILO-CMFD to a wide assortment of neutron transport-based multiphysics problems.

Appendix A.

VERA Input Deck

For reference, we provide the VERA Input Deck that was used to run the Ch. 7 2a-3d feedback-depletion problem with R-CMFD ($\alpha = 0.7$):

```
[CASEID]
title 'Feedback-Depletion (2a, assembly, k-search)'


[STATE] ! 1
  power    100.0 ! perc. rated power
  flow     100.0 ! perc. rated flow
  tinlet   565 K ! ~557 F, ~ 292 C
  pressure 2250  ! psia
  boron    1300  ! ppm
  sym      qtr
  feedback on
  thexp    off
  search   keff
[STATE] ! 2
  deplete GWDMT 0.05
  restart_write eos_0.05.h5 "eos"
[STATE] ! 3
  deplete GWDMT 0.4
  restart_write eos_0.4.h5 "eos"
[STATE] ! 4
  deplete GWDMT 0.7
  restart_write eos_0.7.h5 "eos"
[STATE] ! 5
  deplete GWDMT 1.0
  restart_write eos_1.0.h5 "eos"
[STATE] ! 6
  deplete GWDMT 1.5
```

```
  restart_write eos_1.5.h5 "eos"
[STATE] ! 7
  deplete GWDMT 2.0
  restart_write eos_2.0.h5 "eos"
[STATE] ! 8
  deplete GWDMT 3.0
  restart_write eos_3.0.h5 "eos"
[STATE] ! 9
  deplete GWDMT 4.0
  restart_write eos_4.0.h5 "eos"
[STATE] ! 10
  deplete GWDMT 5.0
  restart_write eos_5.0.h5 "eos"
[STATE] ! 11
  deplete GWDMT 6.0
  restart_write eos_6.0.h5 "eos"
[STATE] ! 12
  deplete GWDMT 7.0
  restart_write eos_7.0.h5 "eos"
[STATE] ! 13
  deplete GWDMT 8.0
  restart_write eos_8.0.h5 "eos"
[STATE] ! 14
  deplete GWDMT 9.0
  restart_write eos_9.0.h5 "eos"
[STATE] ! 15
  deplete GWDMT 10.0
  restart_write eos_10.0.h5 "eos"
[STATE] ! 16
  deplete GWDMT 11.0
  restart_write eos_11.0.h5 "eos"
[STATE] ! 17
  deplete GWDMT 12.0
  restart_write eos_12.0.h5 "eos"
[STATE] ! 18
  deplete GWDMT 13.0
  restart_write eos_13.0.h5 "eos"
[STATE] ! 19
  deplete GWDMT 14.0
  restart_write eos_14.0.h5 "eos"
[STATE] ! 20
  deplete GWDMT 15.0
```

```
  restart_write eos_15.0.h5 "eos"
[STATE] ! 21
  deplete GWDMT 16.0
  restart_write eos_16.0.h5 "eos"
[STATE] ! 22
  deplete GWDMT 17.0
  restart_write eos_17.0.h5 "eos"
[STATE] ! 23
  deplete GWDMT 18.0
  restart_write eos_18.0.h5 "eos"
[STATE] ! 24
  deplete GWDMT 19.0
  restart_write eos_19.0.h5 "eos"
[STATE] ! 25
  deplete GWDMT 20.0
  restart_write eos_20.0.h5 "eos"
[STATE] ! 26
  deplete GWDMT 21.0
  restart_write eos_21.0.h5 "eos"
[STATE] ! 27
  deplete GWDMT 22.0
  restart_write eos_22.0.h5 "eos"
[STATE] ! 28
  deplete GWDMT 23.0
  restart_write eos_23.0.h5 "eos"
[STATE] ! 29
  deplete GWDMT 24.0
  restart_write eos_24.0.h5 "eos"
[STATE] ! 30
  deplete GWDMT 25.0
  restart_write eos_25.0.h5 "eos"
[STATE] ! 31
  deplete GWDMT 26.0
  restart_write eos_26.0.h5 "eos"
[STATE] ! 32
  deplete GWDMT 27.0
  restart_write eos_27.0.h5 "eos"
[STATE] ! 33
  deplete GWDMT 28.0
  restart_write eos_28.0.h5 "eos"
[STATE] ! 34
  deplete GWDMT 29.0
```

```
  restart_write eos_29.0.h5 "eos"
[STATE] ! 35
  deplete GWDMT 30.0
  restart_write eos_30.0.h5 "eos"

[CORE]
  size   1 ! one assembly
  apitch 21.50 ! assembly pitch (cm)
  height 406.337 ! (CASL p9)
  rated  17.67 0.6823 ! MW, Mlbs/hr (CASL p6)

  core_shape
    1

  assm_map
    ASSY

  lower_plate ss  5.0 0.5   ! mat, thickness, vol frac
  upper_plate ss  7.6 0.5

  bc_rad reflecting

[ASSEMBLY]
  npin   17
  ppitch 1.26

  ! CASL p2 materials
  fuel U31 10.257 94.5 / 3.1 u-234=0.026347

  ! cells (CASL p2-defined)
  cell 1 0.4096 0.418 0.475 / U31 he zirc4 ! fuel rod
  cell 2        0.561 0.602 /    mod zirc4 ! guide/instrument
  ! cells (CASL p6-defined)
  cell p6_3            0.561 0.602 / mod    zirc4      ! guide/instrument tube
  cell p6_4            0.418 0.475 /     he zirc4      ! plenum
  cell p6_5                  0.475 /        zirc4      ! end plug
  cell p6_6                  0.475 /        mod        ! empty

  lattice FUEL
    2
    1 1
    1 1 1
```

```
  2 1 1 2
  1 1 1 1 1
  1 1 1 1 1 2
  2 1 1 2 1 1 1
  1 1 1 1 1 1 1 1
  1 1 1 1 1 1 1 1 1

lattice PLEN
  p6_3
  p6_4 p6_4
  p6_4 p6_4 p6_4
  p6_3 p6_4 p6_4 p6_3
  p6_4 p6_4 p6_4 p6_4 p6_4
  p6_4 p6_4 p6_4 p6_4 p6_4 p6_3
  p6_3 p6_4 p6_4 p6_3 p6_4 p6_4 p6_4
  p6_4 p6_4 p6_4 p6_4 p6_4 p6_4 p6_4 p6_4
  p6_4 p6_4 p6_4 p6_4 p6_4 p6_4 p6_4 p6_4 p6_4

lattice PLUG
  p6_3
  p6_5 p6_5
  p6_5 p6_5 p6_5
  p6_3 p6_5 p6_5 p6_3
  p6_5 p6_5 p6_5 p6_5 p6_5
  p6_5 p6_5 p6_5 p6_5 p6_5 p6_3
  p6_3 p6_5 p6_5 p6_3 p6_5 p6_5 p6_5
  p6_5 p6_5 p6_5 p6_5 p6_5 p6_5 p6_5 p6_5
  p6_5 p6_5 p6_5 p6_5 p6_5 p6_5 p6_5 p6_5 p6_5

lattice GAP
  p6_3
  p6_6 p6_6
  p6_6 p6_6 p6_6
  p6_3 p6_6 p6_6 p6_3
  p6_6 p6_6 p6_6 p6_6 p6_6
  p6_6 p6_6 p6_6 p6_6 p6_6 p6_3
  p6_3 p6_6 p6_6 p6_3 p6_6 p6_6 p6_6
  p6_6 p6_6 p6_6 p6_6 p6_6 p6_6 p6_6 p6_6
  p6_6 p6_6 p6_6 p6_6 p6_6 p6_6 p6_6 p6_6 p6_6

! from p6
axial ASSY   6.053
```

```
        GAP   10.281
        PLUG  11.951
        FUEL 377.711
        PLEN 393.711
        PLUG 395.381
        GAP  397.51

  ! from p6
  grid END inc    3.866 1017 / loss=0.9070 ! grid height (cm), mass (g), loss coef
  grid MID zirc4  3.810 875 / loss=0.9065 ! grid height (cm), mass (g), loss coef

  ! from p6
  grid_axial
      END  13.884
      MID  75.2
      MID 127.4
      MID 179.6
      MID 231.8
      MID 284.0
      MID 336.2
      END 388.2

  lower_nozzle  ss 6.053 6250.0  ! mat, height, mass (g)
  upper_nozzle  ss 8.827 6250.0  ! mat, height, mass (g)

[EDITS]
  axial_edit_bounds
      11.951
      15.817
      24.028
      32.239
      40.45
      48.662
      56.873
      65.084
      73.295
      77.105
      85.17
      93.235
      101.3
      109.365
      117.43
```

```
      125.495
      129.305
      137.37
      145.435
      153.5
      161.565
      169.63
      177.695
      181.505
      189.57
      197.635
      205.7
      213.765
      221.83
      229.895
      233.705
      241.77
      249.835
      257.9
      265.965
      274.03
      282.095
      285.905
      293.97
      302.035
      310.1
      318.165
      326.23
      334.295
      338.105
      346.0262
      353.9474
      361.8686
      369.7898
      377.711

[MPACT]
! parallelization
   num_space 29
! cmfd
   cmfd_shift_method adaptive
   cmfd_num_outers   100
```

```
   cmfd_rtol         1e-6
! iteration control
   k_tolerance    1e-5
   flux_tolerance 1e-5
   num_outers     100
! thermal hydraulics
   coupling_method ctf
! depletion
   xs_filename      mpact51g_71_4.3m2_03262018.fmt
   dep_filename     MPACT.dpl
   dep_kernel       internal
   depl_time_method p-c

[COUPLING]
   rlx_power 0.7

[COBRATF]
   parallel 1
```

Appendix B.

Estimating NILO-CMFD Runtime

Below, we give some initial thoughts on the expected performance of a full NILO-CMFD implementation in a 3-D reactor physics code, as compared to R-CMFD. Although we were unable to complete this implementation in MPACT, we can still make some rough estimates about its expected performance.

One may ask, "What is the estimated computational cost of an optimized NILO-CMFD implementation compared to that of R-CMFD?" Unfortunately, this question does not have a unique answer. The reason for this is that the number of outer iterations required by R-CMFD to converge is heavily problem-dependent, whereas the number of outer iterations required by NILO-CMFD is consistently around 10. An additional complication in comparing NILO-CMFD and R-CMFD runtime performance is the inclusion of a user-chosen "knob" in R-CMFD by way of its relaxation factor α.

Next, we address the more answerable question: "What is an accurate way to think about the computational cost of an optimized NILO-CMFD?" To answer this question, let us first consider the simpler question: "What is the typical cost of solving a single-physics neutron transport problem using CMFD?" To answer this question, let us define:

$$C_{\text{tr}} = \text{the cost of performing a single transport sweep.} \tag{B.1}$$

For this discussion, we will estimate the cost of solving CMFD's low-order, transport-corrected diffusion equation as roughly 20% that of a transport sweep. Using this estimate, the cost of a typical single-physics CMFD outer iteration is $1.2 \cdot C_{\text{tr}}$. Since single-physics CMFD typically converges in about 10 iterations, then the total cost

of a typical single-physics CMFD calculation is:

$$\text{total cost (single-physics CMFD)} = 10 \cdot 1.2 \cdot C_{\text{tr}} = 12 \cdot C_{\text{tr}} \ . \tag{B.2}$$

Thus, a single iteration of CMFD is more costly (by about 20%) than the cost of a single transport sweep, but we gladly pay this cost because the number of outer iterations moving from Source Iteration (SI) to CMFD is effectively reduced to about 10.

Next, to answer the stated question for NILO-CMFD, let us define, in addition to C_{tr}:

$$C_{\text{th}} = \text{the cost of a thermal hydraulics solve} \ , \tag{B.3a}$$

$$C_{\text{nd}} = \text{the cost of a nuclear data evaluation} \ . \tag{B.3b}$$

As in the single-physics CMFD estimate given above, one can estimate the cost of solving the NILO-CMFD nonlinear transport-corrected diffusion problem as about 20% the cost of all the high-order calculations. Then, the estimated cost of a single NILO-CMFD outer iteration is: $1.2 \cdot (C_{\text{tr}} + C_{\text{th}} + C_{\text{nd}})$. Since NILO-CMFD requires about 10 iterations to converge, then the total cost for a typical multiphysics calculation solved by NILO-CMFD is:

$$\text{total cost (multiphysics NILO-CMFD)} = 12(C_{\text{tr}} + C_{\text{th}} + C_{\text{nd}}) \ . \tag{B.4}$$

This estimate [Eq. (B.4)] argues that NILO-CMFD should be thought of as the logical extension of single-physics CMFD [Eq. (B.2)] to multiphysics problems. Eqs. (B.2) and (B.4) can be rewritten in a similar form as:

$$\begin{aligned} \text{total cost} = 12 \ (\ &\text{sum of high-order single-physics} \\ &\text{calculations in an iteration} \ ) \ . \end{aligned} \tag{B.5}$$

NILO-CMFD preserves the fast convergence rate of CMFD for single-physics problems: problems converge in about 10 iterations. Like CMFD for single-physics problems, each high-order physics package is called only once, and the estimated cost of a NILO-CMFD outer iteration is about 20% greater than the sum of the costs of all the high-order calculations. (The 20% overhead goes into the cost of solving the low-order transport-corrected diffusion problem.)

We note that if the estimate of 20% overhead were modified to 30% or 40%, Eqs.

(B.2), (B.4), and (B.5) would change in small, very predictable ways. Consequently, the overhead estimate of 20% is not important.

We conclude that it is inaccurate to think about the cost of NILO-CMFD in terms of how it compares in number of iterations or clock time to R-CMFD. (That comparison is too strongly problem-dependent.) Instead, one should think of the "performance" of NILO-CMFD for multiphysics problems as being comparable to the "performance" of CMFD for single-physics problems [Eqs. (B.2), (B.4), and (B.5)].

In other words, we assert that in terms of performance, NILO-CMFD is the effective generalization of single-physics CMFD to mulitphysics problems. (In spite of its name, R-CMFD is **not** the effective generalization of single-physics CMFD to multiphysics problems.)

Bibliography

[1] M. Adams and W. Martin. "Diffusion Synthetic Acceleration of Discontinuous Finite Element Transport Iterations". In: *Nuclear Science and Engineering* **111**.2 (1992), pp. 145–167.

[2] M. Adams and E. Larsen. "Fast Iterative Methods for Discrete-Ordinates Particle Transport Calculations". In: *Progress in Nuclear Energy* **40**.1 (2002), pp. 3–159.

[3] R. Alcouffe. "Diffusion Synthetic Acceleration Methods for the Diamond-Differenced Discrete-Ordinates Equations." In: *Nuclear Science and Engineering* **64**.2 (1977), pp. 344–355.

[4] D. Anderson. "Iterative Procedures for Nonlinear Integral Equations". In: *Journal of the ACM (JACM)* **12**.4 (1965), pp. 547–560.

[5] D. Anistratov and V. Gol'din. "Multilevel Quasidiffusion Methods for Solving Multigroup Neutron Transport k-Eigenvalue Problems in One-Dimensional Slab Geometry". In: *Nuclear Science and Engineering* **169**.2 (2011), pp. 111–132.

[6] Y. Azmy and E. Sartori. *Nuclear Computational Science: A Century in Review*. Springer, 2010.

[7] W. Boyd, S. Shaner, et al. "The OpenMOC Method of Characteristics Neutral Particle Transport Code". In: *Annals of Nuclear Energy* **68** (2014), pp. 43–52.

[8] P. Brown and Y. Saad. "Hybrid Krylov Methods for Nonlinear Systems of Equations". In: *SIAM Journal on Scientific and Statistical Computing* **11**.3 (1990), pp. 450–481.

[9] S. Brunton and J. Kutz. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press, 2019.

[10] S. Brunton, J. Proctor, J. Kutz, and W. Bialek. "Discovering Governing Equations from Data by Sparse Identification of Nonlinear Dynamical Systems". In: *Proceedings of the National Academy of Sciences of the United States of America* **113**.15 (2016), pp. 3932–3937.

[11] D. Cacuci (Editor). *Handbook of Nuclear Engineering*. (Chapters 1, 2, 3, and 9 are relevant to the theory and application of neutron cross sections for nuclear reactor problems.) Springer, 2010.

[12] A. Calloo, R. Le Tellier, and D. Couyras. "Comparison of Chebyshev and Anderson Accelerations". In: *PHYSOR2020 – International Conference on Physics of Reactors: Transition to a Scalable Nuclear Future*. 2020.

[13] G. Cefus and E. Larsen. "Stability Analysis of Coarse-Mesh Rebalance". In: *Nuclear Science and Engineering* **105**.1 (1990), pp. 31–39.

[14] N. Cho. "The Partial Current-Based CMFD (p-CMFD) Method Revisited". In: *Proceedings of the KNS Autumn Meeting*. 2012.

[15] N. Cho, G. Lee, and C. Park. "Partial Current-Based CMFD Acceleration of the 2D/1D Fusion Method for 3D Whole-Core Transport Calculations". In: *Transactions of the American Nuclear Society* **88** (2003).

[16] J. Choi, C. Chang, et al. "Coupling Exascale Multiphysics Applications: Methods and Lessons Learned". In: *Proceedings - IEEE 14th International Conference on eScience*. 2018, pp. 442–452.

[17] S. Choi, Q. Shen, D. Jabaay, Y. Liu, B. Kochunas, and T. Downar. *MPACT Efficiency and Robustness Enhancements for Full-Core BWR Modeling*. Tech. rep. NURAM-2021-002-00. University of Michigan, 2021.

[18] B. Collins, J. Galloway, R. Salko, K. Clarno, A. Wysocki, B. Okhuysen, and D. Andersson. "Whole Core CRUD-Induced Power Shift Simulations using VERA". In: *Proc. PHYSOR 2018*. 2018, pp. 3818–3833.

[19] B. Collins, S. Hamilton, and S. Stimpson. "Use of Generalized Davidson Eigenvalue Solver for Coarse Mesh Finite Difference Acceleration". In: *International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering, M & C 2017*. 2017.

[20] B. Collins, S. Stimpson, et al. "Stability and Accuracy of 3D Neutron Transport Simulations Using the 2D/1D Method in MPACT". In: *Journal of Computational Physics* **326** (2016), pp. 612–628.

[21] L. Cornejo and D. Anistratov. "Nonlinear Diffusion Acceleration Method with Multigrid in Energy for k-Eigenvalue Neutron Transport Problems". In: *Nuclear Science and Engineering* **184**.4 (2016), pp. 514–526.

[22] M. Daeubler, A. Ivanov, et al. "High-Fidelity Coupled Monte Carlo Neutron Transport and Thermal-Hydraulic Simulations Using Serpent 2/SUBCHANFLOW". In: *Annals of Nuclear Energy* **83** (2015), pp. 352–375.

[23] I. Dincer. "Green Methods for Hydrogen Production". In: *International Journal of Hydrogen Energy* **37**.2 (2012), pp. 1954–1971.

[24] M. Dorr and C. Still. "Concurrent Source Iteration in the Solution of Three-Dimensional, Multigroup Discrete Ordinates Neutron Transport Equations". In: *Nuclear Science and Engineering* **122**.3 (1996), pp. 287–308.

[25] J. Duderstadt and L. Hamilton. *Nuclear Reactor Analysis*. 1st ed. 1973.

[26] R. Ferrer. "Stability Analysis of Coarse-Mesh Finite Difference Acceleration Schemes and Moment-Based, Spatially-Linear Discretizations". In: *Annals of Nuclear Energy* **158** (2021).

[27] H. Finnemann and A. Galati. *NEACRP 3-D LWR Transient Benchmark. Final Specifications*. Tech. rep. 1991.

[28] A. Foderaro. *The Elements of Neutron Interaction Theory*. MIT Press, 1971.

[29] E. Gelbard and L. Hageman. "Synthetic Method As Applied To S_n Equations". In: *Nuclear Science and Engineering* **37**.2 (1969), pp. 288–298.

[30] E. Gialdi, S. Grifoni, C. Parmeggiani, and C. Tricoli. "Core Stability in Operating BWR: Operational Experience". In: *Progress in Nuclear Energy* **15**.C (1985), pp. 447–459.

[31] A. Godfrey. *VERA Core Physics Benchmark Progression Problem Specifications*. Tech. rep. CASL-U-2012-0131-004. Revision 4. 2014.

[32] A. Graham, T. Downar, B. Collins, R. Salko, and S. Palmtag. "Assessment of Thermal-Hydraulic Feedback Models". In: *Physics of Reactors 2016, PHYSOR 2016: Unifying Theory and Experiments in the 21st Century*. 2016.

[33] Y. Guo, Z. Li, S. Huang, M. Liu, and K. Wang. "A New Neutronics-Thermal-Mechanics Multi-Physics Coupling Method for Heat Pipe Cooled Reactor Based on RMC and OpenFOAM". In: *Progress in Nuclear Energy* **139** (2021).

[34] M. Gutknecht and S. Röllin. "The Chebyshev Iteration Revisited". In: *Parallel Computing* **28**.2 (2002), pp. 263–283.

[35] S. Hamilton, M. Berrill, et al. "An Assessment of Coupling Algorithms for Nuclear Reactor Core Physics Simulations". In: *Journal of Computational Physics* **311** (2016), pp. 241–257.

[36] B. Herman. "Monte Carlo and Thermal Hydraulic Coupling Using Low-Order Nonlinear Diffusion Acceleration". PhD thesis. Massachusetts Institute of Technology, 2014.

[37] A. Ivanov, V. Sanchez, R. Stieglitz, and K. Ivanov. "High Fidelity Simulation of Conventional and Innovative LWR with the Coupled Monte-Carlo Thermal-Hydraulic System MCNP-SUBCHANFLOW". In: *Nuclear Engineering and Design* **262** (2013), pp. 264–275.

[38] M. Jarrett, B. Kochunas, A. Zhu, and T. Downar. "Analysis of Stabilization Techniques for CMFD Acceleration of Neutron Transport Problems". In: *Nuclear Science and Engineering* **184**.2 (2016), pp. 208–227.

[39] J. Jones, B. Collins, and M. Asgari. "Anderson Acceleration Applied to Multiphysics Simulation of Boiling Water Reactors Using VERA". In: *ANS Mathematics & Computation (M&C) 2021*. 2021, pp. 1946–1951.

[40] H. Joo, D. Barber, G. Jiang, and T. Downar. *PARCS: A Multi-Dimensional Two-Group Reactor Kinetics Code Based on the Nonlinear Analytic Nodal Method*. Tech. rep. PU/NE-98-26. 1998.

[41] P. Joskow and J. Parsons. "The Economic Future of Nuclear Power". In: *Dadealus* **138**.4 (2009), pp. 45–59.

[42] B. Kelley and E. Larsen. "CMFD Acceleration of Spatial Domain-Decomposed Neutron Transport Problems". In: *PHYSOR 2012: Conference on Advances in Reactor Physics - Linking Research, Industry, and Education*. 2012.

[43] C. Kelley. *Iterative Methods for Linear and Nonlinear Equations*. SIAM, 1995.

[44] C. Kelley. *Solving Nonlinear Equations with Newton's Method*. SIAM, 2003.

[45] D. Kelly, A. Kelly, et al. "MC21/CTF and VERA Multiphysics Solutions to VERA Core Physics Benchmark Progression Problems 6 and 7". In: *Nuclear Engineering and Technology* **49** (6 2017), pp. 1326–1338.

[46] K. S. Kim, M. L. Williams, D. Wiarda, and K. T. Clarno. "Development of the Multigroup Cross Section Library for the CASL Neutronics Simulator MPACT: Method and Procedure". In: *Annals of Nuclear Energy* 133 (2019). DOI: 10.1016/j.anucene.2019.05.010.

[47] D. Knoll and D. Keyes. "Jacobian-Free Newton-Krylov Methods: A Survey of Approaches and Applications". In: *Journal of Computational Physics* **193**.2 (2004), pp. 357–397.

[48] B. Kochunas. private communication. Mar. 2022.

[49] B. Kochunas, B. Collins, et al. "VERA Core Simulator Methodology for Pressurized Water Reactor Cycle Depletion". In: *Nuclear Science and Engineering* **185**.1 (2017), pp. 217–231.

[50] B. Kochunas, A. Fitzgerald, and E. Larsen. "Fourier Analysis of Iteration Schemes for k-Eigenvalue Transport Problems with Flux-Dependent Cross Sections". In: *Journal of Computational Physics* **345** (2017), pp. 294–307.

[51] H. Kopp. "Synthetic Method Solution of the Transport Equation". In: *Nuclear Science and Engineering* **17**.1 (1963), pp. 65–74.

[52] J. Kutz. *Data-Driven Modeling & Scientific Computation: Methods for Complex Systems & Big Data*. Oxford University Press, 2013.

[53] J. Kutz et al. *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*. SIAM, 2016.

[54] H. Langtangen and S. Linge. *Finite Difference Computing with PDEs: A Modern Software Approach*. Springer Open, 2017.

[55] E. Larsen et al. *MPACT Theory Manual*. Tech. rep. Version 4.1. 2019.

[56] E. Larsen. "Unconditionally Stable Diffusion-Synthetic Acceleration Methods for the Slab Geometry Discrete Ordinates Equations. Part I: Theory." In: *Nuclear Science and Engineering* **82**.1 (1982), pp. 47–63.

[57] E. Larsen and B. Kelley. "The Relationship Between the Coarse-Mesh Finite Difference and the Coarse-Mesh Diffusion Synthetic Acceleration Methods". In: *Nuclear Science and Engineering* **178**.1 (2014), pp. 1–15.

[58] J. Lee and H. Joo. "Convergence Analysis of Fixed-Point Iteration with Anderson Acceleration on a Simplified Neutronics/Thermal-Hydraulics System". In: *Nuclear Engineering and Technology* (2021). in press.

[59] M. Lee, H. Joo, D. Lee, and K. Smith. "Investigation of CMFD Accelerated Monte Carlo Eigenvalue Calculation with Simplified Low Dimensional Multigroup Formulation". In: *Proc. PHYSOR 2010*. 2010, pp. 9–14.

[60] J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, 2002.

[61] E. Lewis. *Fundamentals of Nuclear Reactor Physics*. Academic Press, 2008.

[62] E. Lewis. "Progress in Multidimensional Neutron Transport Computation". In: *Nuclear Science and Engineering* **64**.2 (1977), pp. 279–293.

[63] E. Lewis and W. Miller. *Computational Methods of Neutron Transport*. John Wiley & Sons, 1984.

[64] Y. Liu and W. Martin. "Pin-Resolved Resonance Self-Shielding Methods in LWR Direct Transport Calculations". In: *Annals of Nuclear Energy* **110** (2017), pp. 1165–1175.

[65] Y. Liu, R. Salko, et al. "Improved MPACT Energy Deposition and Explicit Heat Generation Coupling with CTF". In: *Annals of Nuclear Energy* **152** (2021).

[66] Z. Liu, H. Wu, et al. "A New Three-Dimensional Method of Characteristics for the Neutron Transport Calculation". In: *Annals of Nuclear Energy* **38**.2-3 (2011), pp. 447–454.

[67] A. Lovins. "Nuclear Weapons and Power-Reactor Plutonium". In: *Nature* **283** (1980), pp. 817–823.

[68] Y. Ma, M. Liu, et al. "Neutronic and Thermal-Mechanical Coupling Analyses in a Solid-State Reactor using Monte Carlo and Finite Element Methods". In: *Annals of Nuclear Energy* **151** (2021).

[69] H. MacPherson. "The Molten Salt Reactor Adventure". In: *Nuclear Science and Engineering* **90**.4 (1985), pp. 374–380.

[70] A. Manera, U. Rohde, H. M. Prasser, and T. H. Van Der Hagen. "Modeling of Flashing-Induced Instabilities in the Start-Up Phase of Natural-Circulation BWRs using the Two-Phase Flow Code FLOCAL". In: *Nuclear Engineering and Design* 235.14 (2005), pp. 1517–1535.

[71] E. Marquis, J. Hyde, et al. "Nuclear Reactor Materials at the Atomic Scale". In: *Materials Today* **12**.11 (2009), pp. 30–37.

[72] R. McClarren and T. Haut. "Acceleration of Source Iteration Using the Dynamic Mode Decomposition". In: *International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering, M & C 2019*. 2019, pp. 1681–1689.

[73] R. McClarren and T. Haut. "Data-Driven Acceleration of Thermal Radiation Transfer Calculations with the Dynamic Mode Decomposition and a Sequential Singular Value Decomposition". In: *Journal of Computational Physics* **448** (2022).

[74] D. McCoy and E. Larsen. "Unconditionally Stable Diffusion-Synthetic Acceleration Methods for the Slab Geometry Discrete Ordinates Equations. Part Ii: Numerical Results." In: *Nuclear Science and Engineering* **82**.1 (1982), pp. 64–70.

[75] W. Miller. "Generalized Rebalance: a Common Framework for Transport Acceleration Methods." In: *Nuclear Science and Engineering* **65**.2 (1978), pp. 226–236.

[76] F. Montáns, F. Chinesta, R. Gómez-Bombarelli, and J. Kutz. "Data-Driven Modeling and Learning in Science and Engineering". In: *Comptes Rendus - Mecanique* **347**.11 (2019), pp. 845–855.

[77] S. Nisan, G. Caruso, et al. "Sea-Water Desalination with Nuclear and Other Energy Sources: The EURODESAL Project". In: *Nuclear Engineering and Design* **221**.1-3 (2003), pp. 251–275.

[78] A. Novak, P. Romano, et al. "Preliminary Coupling of OpenMC and Nek5000 Within the MOOSE Framework". In: *International Conference on Physics of Reactors, PHYSOR 2018: Reactor Physics Paving the Way Towards More Efficient Systems*. 2018, pp. 2162–2173.

[79] A. Al-Othman, N. Darwish, et al. "Nuclear Desalination: A State-of-the-Art Review". In: *Desalination* **457** (2019), pp. 39–61.

[80] S. Palmtag, A. Godfrey, M. Baird, and E. Walker. *VERA Common Input User Manual*. Tech. rep. 2019.

[81] S. Palmtag, B. Kochunas, et al. "Modeling Thermal Expansion in VERA-CS". In: *Proc. M&C 2017*. 2017.

[82] J. Parrington, H. Knox, et al. "Chart of the nuclides. Fifteenth edition, revised to 1996". In: ().

[83] W. Reed. "The Effectiveness of Acceleration Techniques for Iterative Methods in Transport Theory". In: *Nuclear Science and Engineering* **45**.3 (1971), pp. 245–254.

[84] J. Rhodes, K. Smith, and D. Lee. "CASMO-5 Development and Applications". In: *Proccedings of PHYSOR 2006 - American Nuclear Society's Topical Meeting on Reactor Physics*. 2006.

[85] M. Rose, T. Downar, and F. Gleicher. "Redwing: A MOOSE Application for Coupling MPACT and BISON". In: *Transactions of the American Nuclear Society* **111** (2014), pp. 616–619.

[86] R. Salko, S. Slattery, et al. *Development of Preliminary VERA-CS Crud-Induced Localized Corrosion Modeling Capability*. Tech. rep. CASL-U-2018-1617-000. 2018.

[87] R. Salko and M. Avramova. *CTF Theory Manual*. Tech. rep. CASL-U-2018-1715-000. 2019.

[88] S. Schunert, Y. Wang, et al. "A Flexible Nonlinear Diffusion Acceleration Method for the S_N Transport Equations Discretized with Discontinuous Finite Elements". In: *Journal of Computational Physics* **338** (2017), pp. 107–136.

[89] J. Senecal and W. Ji. "Approaches for Mitigating Over-Solving in Multiphysics Simulations". In: *International Journal for Numerical Methods in Engineering* **112**.6 (2017), pp. 503–528.

[90] J. Senecal and W. Ji. "Development of an Efficient Tightly Coupled Method for Multiphysics Reactor Transient Analysis". In: *Progress in Nuclear Energy* **103** (2018), pp. 33–44.

[91] Q. Shen. "Robust and Efficient Methods in Transient Whole-Core Neutron Transport Calculations". PhD thesis. University of Michigan, Ann Arbor, 2021.

[92] Q. Shen, N. Adamowicz, and B. Kochunas. "Relationship Between Relaxation and Partial Convergence of Nonlinear Diffusion Acceleration for Problems with Feedback". In: *International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering, M&C 2019*. 2019, pp. 2248–2257.

[93] Q. Shen, N. Adamowicz, B. Kochunas, and E. Larsen. "X-CMFD: A Robust Iteration Scheme for CMFD-Based Acceleration of Neutron Transport Problems with Nonlinear Feedback". In: *International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering, M&C 2019*. 2019, pp. 2238–2247.

[94] Q. Shen, S. Choi, and B. Kochunas. "A Robust Relaxation-Free Multiphysics Iteration Scheme for CMFD–Accelerated Neutron Transport k–Eigenvalue Calculations – II: Numerical Results". In: *Nuclear Science and Engineering* **195**.11 (2021), pp. 1202–1235.

[95] Q. Shen, S. Choi, and B. Kochunas. "Robust Relaxation-Free Multiphysics Simulation via Power-Iteration-Level-Coupled CMFD Acceleration in MPACT". In: *ANS Mathematics & Computation (M&C) 2021*. 2021, pp. 2064–2073.

[96] Q. Shen and B. Kochunas. "A Robust Relaxation-Free Multiphysics Iteration Scheme for CMFD-Accelerated Neutron Transport k-Eigenvalue Calculations — I: Theory". In: *Nuclear Science and Engineering* **195**.11 (2021), pp. 1176–1201.

[97] Q. Shen, Y. Xu, and T. Downar. "Stability Analysis of the CMFD Scheme with Linear Prolongation". In: *Annals of Nuclear Energy* **129** (2019), pp. 298–307.

[98] K. Smith. "Nodal Method Storage Reduction by Nonlinear Iteration". In: *Transactions of the American Nuclear Society* **44** (1983), p. 265.

[99] K. Smith and J. Rhodes. "Full-Core, 2-D, LWR Core Calculations with CASMO-4E". In: *Proceedings of the PHYSOR 2002 - International Conference on the New Frontiers of Nuclear Technology : Reactor Physics, Safety and High-Performance Computing - The ANS 2002 RPD Topical Meeting*. 2002.

[100] R. Stammler and M. Abbate. *Methods of Steady-State Reactor Physics in Nuclear Design*. Academic Press, 1983.

[101] C. Stanley and F. Marshall. "Advanced Test Reactor: A National Scientific User Facility". In: *International Conference on Nuclear Engineering*. 2008, pp. 367–372.

[102] S. Stimpson, K. Clarno, R. Pawlowski, R. Gardner, J. Powers, B. Collins, A. Toth, S. Novascone, S. Pitts, J. Hales, and G. Pastore. "Coupled Fuel Performance Calculations in VERA and Demonstration on Watts Bar Unit 1, Cycle 1". In: *Annals of Nuclear Energy* **145** (2020).

[103] A. Toth. "A Theoretical Analysis of Anderson Acceleration and Its Application in Multiphysics Simulation for Light-Water Reactors". PhD thesis. North Carolina State University, 2016.

[104] A. Toth, C. Kelley, et al. "Analysis of Anderson Acceleration on a Simplified Neutronics/Thermal Hydraulics System". In: *NS MC 2015 - Joint International Conference on Mathematics and Computation (M&C), Supercomputing in Nuclear Applications (SNA) and the Monte Carlo (MC) Method*. 2015.

[105] A. Toth and C. Kelley. "Convergence Analysis for Anderson Acceleration". In: *SIAM Journal on Numerical Analysis* **53**.2 (2015), pp. 805–819.

[106] P. Turinsky and D. Kothe. "Modeling and Simulation Challenges Pursued by the Consortium for Advanced Simulation of Light Water Reactors (CASL)". In: *Journal of Computational Physics* **313** (2016), pp. 367–376.

[107] J. Turner, K. Clarno, et al. "The Virtual Environment for Reactor Applications (VERA): Design and Architecture". In: *Journal of Computational Physics* **326** (2016), pp. 544–568.

[108] S. Van Der Marck, A. Koning, and K. Charlton. "The Options for the Future Production of the Medical Isotope 99Mo". In: *European Journal of Nuclear Medicine and Molecular Imaging* **37** (2010), pp. 1817–1820.

[109] R. Varga. *Matrix Iterative Analysis*. 2nd. Springer, 2000.

[110] E. Walker. "Low-Order Multiphysics Coupling Techniques for Nuclear Reactor Applications". PhD thesis. The University of Tennessee, Knoxville, 2017.

[111] E. Walker, B. Collins, and J. Gehin. "Low-Order Multiphysics Coupling Techniques for Nuclear Reactor Applications". In: *Annals of Nuclear Energy* **132** (2019), pp. 327–338.

[112] H. Walker and P. Ni. "Anderson Acceleration for Fixed-Point Iterations". In: *SIAM Journal on Numerical Analysis* **49**.4 (2011), pp. 1715–1735.

[113] D. Walter, V. Petrov, N. Adamowicz, and A. Manera. "MIMIC: Michigan Interface for Multi-state In-Memory Coupling with STAR-CCM+ for Nuclear Applications". In: *M&C 2017 - International Conference on Mathematics & Computational Methods Applied to Nuclear Science & Engineering, Jeju, Korea*. 2017.

[114] D. Wang and S. Xiao. "A Linear Prolongation Approach to Stabilizing CMFD". In: *Nuclear Science and Engineering* **190**.1 (2018), pp. 45–55.

[115] D. Wang and Z. Zhu. "A Revisit to CMFD Schemes: Fourier Analysis and Enhancement". In: *Energies* **14**.2 (2021).

[116] J. Wang, Q. Wang, and M. Ding. "Review on Neutronic/Thermal-Hydraulic Coupling Simulation Methods for Nuclear Reactor Analysis". In: *Annals of Nuclear Energy* **137** (2020).

[117] Y. Wang. "Nonlinear Diffusion Acceleration for the Multigroup Transport Equation Discretized with S_N and Continuous FEM with Rattlesnake". In: *Proceedings of the 2013 International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering - M&C 2013*. 2013.

[118] T. Wareing. "Asymptotic Diffusion Accelerated Discontinuous Finite Element Methods for Transport Problems". PhD thesis. The University of Michigan, 1991.

[119] J. Willert, H. Park, and W. Taitano. "Using Anderson Acceleration to Accelerate the Convergence of Neutron Transport Calculations with Anisotropic Scattering". In: *Nuclear Science and Engineering* **181**.3 (2015), pp. 342–350.

[120] E. Wolters, E. Larsen, and W. Martin. "Hybrid Monte Carlo-CMFD methods for Accelerating Fission Source Convergence". In: *Nuclear Science and Engineering* **174**.3 (2013), pp. 286–299.

[121] S. Yalçin. "A Review of Nuclear Hydrogen Production". In: *International Journal of Hydrogen Energy* **14**.8 (1989), pp. 551–561.

[122] M. Yavuz and E. Larsen. "Spatial Domain Decomposition for Neutron Transport Problems". In: *Transport Theory and Statistical Physics* **18**.2 (1989), pp. 205–219.

[123] B. Yee, B. Kochunas, and E. Larsen. "A Multilevel in Space and Energy Solver for 3-D Multigroup Diffusion and Coarse-Mesh Finite Difference Eigenvalue Problems". In: *Nuclear Science and Engineering* **193**.7 (2019), pp. 722–745.

[124] B. Yee, B. Kochunas, E. Larsen, and Y. Xu. "Space-Dependent Wielandt Shifts for Multigroup Diffusion Eigenvalue Problems". In: *Nuclear Science and Engineering* **188**.2 (2017), pp. 140–159.

[125] Z. Zhang and S. Yu. "Future HTGR Developments in China after the Criticality of the HTR-10". In: *Nuclear Engineering and Design* **218** (2002), pp. 249–257.

[126] A. Zhu, M. Jarrett, et al. "An Optimally Diffusive Coarse Mesh Finite Difference Method to Accelerate Neutron Transport Calculations". In: *Annals of Nuclear Energy* **95** (2016), pp. 116–124.

[127] L. Zou, H. Zhang, J. Gehin, and B. Kochunas. "Coupled Thermal-Hydraulic/Neutronics/Crud Framework in Prediction of Crud-Induced Power Shift Phenomenon". In: *Nuclear Technology* **183**.3 (2013), pp. 535–542.

[128] E. Zukoski. *A Review of Flows Driven by Natural Convection in Adiabatic Shafts*. Tech. rep. NIST-GCR-95-679. California Institute of Technology, 1995.