Parallel 3-D Method of Characteristics with Linear Source and Advanced Transverse Integration

by

Andrew P. Fitzgerald

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
2019

Doctoral Committee:

Professor Brendan Kochunas, Chair,
Professor Thomas Downar,
Professor Edward Larsen,
Professor Venkat Raman,
Dr. Rodolfo Ferrer,

apfitzge@umich.edu
ORCID ID: 0000-0002-4066-0949

2019

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

List of Algorithms

LIST OF APPENDICES

LIST OF ABBREVIATIONS

MoC Method of Characteristics

FS flat source

LS linear source

FSA flat-source approximation

LSA linear-source approximation

FSMoC flat-source method of characteristics

LSMoC linear-source method of characteristics

LIFA linear-isotropic-flat-anisotropic

CASL Consortium for the Advanced Simulation of Light Water Reactors

LWR Light Water Reactor

PWR Pressurized Water Reactor

NEAMS Nuclear Energy Advanced Modeling and Simulation Program

PDE Partial Differential Equation

ODE Ordinary Differential Equation

P_N Spherical Harmonics

SSpherical Harmonics (P_N) Simplified P_N

CP Collision Probability

CDP method of Characteristic Direction Probabilities

S_N Discrete Ordinates

NDA non-linear diffusion acceleration

CMFD coarse mesh finite-difference

T/H thermal-hydraulic

TCP0 transport-corrected P_0

DNPL direct neutron path linking

MRMB memory reduction technique for macroband

MRT modular ray-tracing

LEAF Legendre polynomial expansion of angular flux

VERA Virtual Environment for Reactor Analysis

VERA-CS Virtual Environment for Reactor Analysis Core Simulator

UO_2 uranium oxide

MOX mixed oxide

CPU central processing unit

GPU graphics processing unit

GPGPU general purpose graphics processing unit

RSB recursive spectral bisection

RIB recursive intertial bisection

REB recursive expansion bisection

SOI sphere of influence

MMR maximum-to-minimum ratio

ARA axially and radially aligned

RA radially aligned

UR unrestricted

B&W Babcox and Wilcox

FLOP floating point operation

EFPD effective full power days

GWDMT gigawatt-days per metric ton heavy metal

pcm per cent mille

HFP hot full power

CTF COBRA-TF

KAIST Korea Advanced Institute of Science and Technology

KAERI Korea Atomic Energy Research Institute

UM University of Michigan

IFBA integral fuel burnable absorber

MIT Massachusetts Institute of Technology

FMA fused multiply add

ABSTRACT

In the design and analysis of nuclear fission reactor systems, simulations are an essential tool for improving efficiency as well as safety. Neutronics simulations have always been limited by the available computational resources. This is because of the large discretizations that are needed for the neutron transport equation, which has a 6-dimensional phase space for steady-state eigenvalue problems.

The "gold standard" for 3-D neutron transport simulations is Monte Carlo with explicit geometry representation because it treats all dependent variables continuously. However, there are significant remaining challenges for Monte Carlo methods that prohibit their widespread use, and put them at a disadvantage compared to deterministic methods. The "gold standard" for deterministic 3-D neutron transport is the 3-D MoC. Numerous deterministic methods exist for solving the 3-D transport equation. Each of them has their own drawback. 3-D MoC is considered the "best" due to its ability to accurately model the exact geometry and approximate anisotropic scattering (other methods do just one of these well or become undesirably complex). The downside of the 3-D MoC method is the substantial computational resources required to discretize the problem.

Over the past decade, there has been renewed interest in assessing the state of the art for 3-D MoC and the tractability of this problem on the newest computer architectures. Previous work made significant strides in parallelizing the MoC algorithm for 100,000's of processors, but ultimately did not prove viable due to the extreme compute resources required. Since then there has been progress in making 3-D MOC less computationally burdensome by adopting more advanced discretization methods that lead to fewer spatial mesh regions and rays; namely the linear source approximation, and chord-classification or on-the-fly ray-tracing. The goal of this thesis

is to continue progress in reducing the computational burden of Method of Characteristics (MoC) calculations, with a particular focus on three-dimensional calculation. This thesis makes an effort to reach this goal through three related contributions: the utilization of graph-theory for spatial decomposition, improvements to the linear-source approximation (LSA) for multiphysics calculations, and a novel 3-D ray-tracing method with advanced transverse integration.

Spatial decomposition is typically very beneficial, if not necessary, for whole-core direct transport methods. Previous works on 3-D MoC calculations have used simple spatial decomposition schemes, that often resulted in poor load-balancing, particularly when using the LSA. This work addresses this issue by utilizing graph partitioning methods to give a better load-balance, even in cases where the number of computational cells is very different in different regions of the reactor.

The LSA has previously been shown to allow for the use of a coarser mesh while maintaining accuracy in pure neutronics calculations. However, typically the problems of interest involve multiple physics such as isotopic depletion and thermal-hydraulic (T/H) feedback. This work improves the LSA method for such problems by re-formulating the equations to eliminate an inefficiency in cases with non-constant cross sections. This is shown to significantly improve run-times and reduce memory usage, even in such cases.

Finally, a novel 3-D ray-tracing method, based on the macroband, is developed to reduce the number of characteristic tracks necessary for an accurate result. The method is compared against a traditional ray-tracing method for several benchmark problems. In several of these cases, the method is shown to significantly reduce the number of segments necessary for similar accuracy from a traditional ray-tracing method. The ray-tracing method is also shown to have very desirable properties such as near-monotonic convergence, and is able to act as more of a "black-box" solver.

CHAPTER 1

Introduction

1.1 Motivation

Computer simulations have played an important role in the design and analysis of nuclear reactor systems over the past 60 years [1]. The methods used by these simulations have always been limited by the available computational resources; as such, in the 1950's two-group diffusion theory was used as a basis for simulation tools [1]. As computers became more powerful, multigroup diffusion calculations became the method of choice for Light Water Reactor (LWR) design calculations.

More accurate and detailed simulation tools allow for designs to have higher power density, and thus be more profitable, without compromising safety. However, computational resources have always limited the level of detail of simulation tools. Exponential increases in computing power, and high-performance computing clusters have made whole-core transport calculations possible [2–9]. Programs such as Consortium for the Advanced Simulation of Light Water Reactors (CASL) and Nuclear Energy Advanced Modeling and Simulation Program (NEAMS) have focused on development of modern advanced simulation tools to address certain challenge problems. Large computing clusters are generally unavailable to reactor analysts in industry, and so using direct whole-core 3-D transport methods is not common outside academia or national laboratories.

The "gold standard" of deterministic methods has been the 3-D Method of Characteristics (MoC) [10] due to its' ability to accurately model complicated geometries. At the time of writing, whole-core 3-D MoC calculations are generally not possible without the use of large computing clusters. This is due to the large discretizations that are necessary for the neutron transport equation, which has a 6-dimensional phase space for steady-state eigenvalue problems. In the past decade, there has been renewed interest in making 3-D MoC more efficient and performant by using parallelism [11], modern graphics processing unit (GPU) architectures [7], and ray-tracing storage techniques [12, 13]. Work has also been done to make MoC faster by improving the efficiency of the calculations, using higher-order approximations [9, 14].

The bulk of this thesis work is comprised of three distinct, yet connected, topics, all with a focus

on improving the feasibility of 3-D MoC calculations. The first topic is related to improving the parallel efficiency of MoC calculations. It is the author's opinion that improving the efficiency of 3-D MoC calculations should be the primary focus of current/future research, as it is not currently feasible for industry to use thousands of processors. Thus, two techniques are utilized as part of this thesis work: the linear-source approximation (LSA), and the macroray.

The first contribution of this thesis is work in improving parallel efficiency. While large scale parallelism on thousands of processors may not be feasible for industry, some degree of parallelism is necessary for whole-core calculations due to memory constraints. An automated spatial decomposition scheme based on graph theory, is developed leading to significantly improved parallel efficiency [15, 16].

The second contribution focuses on improvements to a method previously researched by other groups, the linear-source approximation (LSA) [9, 14, 17]. Specifically, this work has led to improvements of the method for stability in near-void regions [18], and efficiency in multi-physics simulations [19]. The LSA is an approximation that is used to improve MoC efficiency by reducing the number of computational cells required for accurate results.

The macroray is a novel ray-tracing technique developed as part of this thesis work; this technique is an extension of the two-dimensional macroband [20] ray-tracing technique. The macroband technique has been shown to reduce the number of characteristic rays required for accurate results in two-dimensional flat-source calculations [21, 22]. However, prior to this work, there has been no effort to extend this method to 3-D MoC calculations. The author seeks to fill this gap, as the number of rays in 3-D MoC calculations is generally much higher than in 2-D, so the possible benefits of the method are larger. Additionally, the method is expected to be even more efficient when using a coarse mesh with the LSA.

1.2 Outline

The remainder of this dissertation is structured as follows. Chapter 2 gives an overview of neutron transport theory, with a focus on what is relevant to this work. This chapter also gives an overview of the current state-of-the-art 3-D transport methods. The focus of this work, the MoC, is derived in full detail in Chapter 3. Techniques for making more efficient use of the available computational resources in MoC calculations, by improving spatial decomposition, are given in Chapter 4. Chapter 5 lists significant improvements to the moment-based linear-source approximation (LSA) made by this work. A novel ray-tracing technique, using advanced transverse integration, intended to decrease the amount of computational work while maintaining accuracy is presented in Chapter 6. Finally, Chapter 7 gives a summary of the work done as part of this dissertation, the conclusions that can be made from these studies, and possible paths for future research on related topics.

Bibliography

[1] G. C. Bilodeau et al. *PDQ: An IBM-704 Code to Solve the Two-dimensional Few-group Neutron-Diffusion Equations*. Tech. rep. Bettis Plant, Westinghouse Electrice Corporation, 1957.

[2] Kord Smith and Joel D. Rhodes. "CASMO-4 Characteristic Methods for Two-Dimensional PWR and BWR Core Calculations". In: *Transactions of the American Nuclear Society*. 2000.

[3] Richard Sanchez et al. "APOLLO2 Year 2010". In: *Journal of Nuclear Engineering and Technology* 42.5 (2010), pp. 474–499.

[4] Han Gyu Joo et al. "Methods and performance of a three-dimensional whole-core transport code DeCART". In: *PHYSOR 2004*. 2004, pp. 21–34. ISBN: 0894486837. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-22344449157{\&}partnerID=tZOtx3y1.

[5] Thomas M. Evans et al. "Denovo: A New Three-Dimensional Parallel Discrete Ordinates Code in SCALE". In: *Nuclear Technology* 171.2 (2010), pp. 171–200. ISSN: 0029-5450. DOI: 10.13182/nt171-171.

[6] W. S. Yang et al. "Neutronics modeling and simulation of sharp for fast reactor analysis". In: *Nuclear Engineering and Technology* 42.5 (2010), pp. 520–545. ISSN: 17385733. DOI: 10.5516/NET.2010.42.5.520.

[7] William Boyd et al. "The OpenMOC method of characteristics neutral particle transport code". In: *Annals of Nuclear Energy* 68 (2014), pp. 43–52. ISSN: 03064549. DOI: 10.1016/j.anucene.2013.12.012.

[8] Benjamin Collins 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. ISSN: 10902716. DOI: 10.1016/j.jcp.2016.08.022.

[9] Geoffrey Alexander Gunow. "Full Core 3D Neutron Transport Simulation Using the Method of Characteristics with Linear Sources". Doctoral. Massachusetts Institute of Technology, 2018.

[10] J.R. Askew. *A Characteristics Formulation of the Neutron Transport Equation in Complicated Geometries*. 1972.

[11] Brendan Matthew Kochunas. "A Hybrid Parallel Algorithm for the 3-D Method of Characteristics Solution of the Boltzmann Transport Equation on High Performance Compute Clusters". Doctoral. University of Michigan, 2013, pp. 1–194.

[12] Daniele Sciannandrone, Simone Santandrea, and Richard Sanchez. "Optimized tracking strategies for step MOC calculations in extruded 3D axial geometries". In: *Annals of Nuclear Energy* 87 (2016), pp. 49–60. ISSN: 18732100. DOI: 10.1016/j.anucene.2015.05.014.

[13] Geoffrey Gunow et al. "Reducing 3D MOC Storage Requirements with Axial On- the-fly Ray Tracing". In: *Physics of Reactors 2016, PHYSOR 2016: Unifying Theory and Experiments in the 21st Century*. Sun Valley: American Nuclear Society, 2016. URL: http://hdl.handle.net/1721.1/109864{\%}0ACreative.

[14] Rodolfo M. Ferrer and Joel D. Rhodes. "A Linear Source Approximation Scheme for the Method of Characteristics". In: *Nuclear Science and Engineering* 182.2 (2016), pp. 151–165. ISSN: 0029-5639. DOI: 10.13182/NSE15-6.

[15] Andrew Fitzgerald et al. "Automated Decomposition of a Structured Grid". In: *Transactions of the American Nuclear Society*. Vol. 117. Washington D.C., 2017, pp. 731–734.

[16] Andrew P Fitzgerald et al. "Spatial decomposition of structured grids for nuclear reactor simulations". In: *Annals of Nuclear Energy* 132 (2019), pp. 686–701. ISSN: 0306-4549. DOI: 10.1016/j.anucene.2019.06.054.

[17] Rodolfo M. Ferrer and Joel D. Rhodes. "The linear source approximation and particle conservation in the Method of Characteristics for isotropic and anisotropic sources". In: *Annals of Nuclear Energy* 115 (2018), pp. 209–219. ISSN: 03064549. DOI: 10.1016/j.anucene.2018.01.023.

[18] Andrew Fitzgerald and Brendan Kochunas. "Fast Exponential Function Approximations for the Method of Characteristics with Linear Source". In: *Transactions of the American Nuclear Society*. Orlando, USA, 2018, pp. 645–648.

[19] Andrew P Fitzgerald, Brendan Kochunas, and Thomas Downar. "Improved Formulation of the Method of Characteristics with Linear Source for 2D/1D and Multiphysics Calculations". In: *M&C*. Portland, OR, 2019, pp. 1093–1103.

[20] Eduardo A. Villarino et al. "HELIOS: Angularly Dependent Collision Probabilities". In: *Nuclear Science and Engineering* 112.1 (1992), pp. 16–31. ISSN: 0029-5639. DOI: 10.13182/NSE112-16.

[21] Akio Yamamoto et al. “Non-Equidistant Ray Tracing for the Method of Characteristics”. In: *M&C 2005* (2005), pp. 1–10.

[22] François Févotte, Simone Santandrea, and Richard Sanchez. “Advanced Transverse Integration for the Method of Characteristics”. In: *M&C 2007*. 2007, pp. 1–12. ISBN: 0894480596.

CHAPTER 2

Neutron Transport Theory

In this chapter, the basic theory behind the neutron transport equation, and the numerical methods used to solve it, are introduced.

2.1 Neutron Transport Equation

The mean behavior of neutrons in a (steady-state) system is described by the Boltzmann transport equation:

$$\begin{aligned}\Big[\widehat{\boldsymbol{\Omega}}\cdot\boldsymbol{\nabla}+\Sigma_t(\boldsymbol{x},E)\Big]\psi(\boldsymbol{x},\widehat{\boldsymbol{\Omega}},E)=&\\ \frac{1}{4\pi}\Bigg[&Q(\boldsymbol{x},\widehat{\boldsymbol{\Omega}},E)\\ &+\int_0^\infty\int_{4\pi}\Sigma_s(\boldsymbol{x},\widehat{\boldsymbol{\Omega}}'\cdot\widehat{\boldsymbol{\Omega}},E'\to E)\psi(\boldsymbol{x},\widehat{\boldsymbol{\Omega}}',E')\,\mathrm{d}\Omega'\,\mathrm{d}E'\\ &+\chi(\boldsymbol{x},E)\int_0^\infty\nu\Sigma_f(\boldsymbol{x},E')\int_{4\pi}\psi(\boldsymbol{x},\widehat{\boldsymbol{\Omega}}',E')\,\mathrm{d}\Omega'\,\mathrm{d}E'\Bigg],\end{aligned} \tag{2.1a}$$

$$\begin{aligned}\forall\boldsymbol{x}\in V,\quad\forall\widehat{\boldsymbol{\Omega}}\in 4\pi,\quad\forall E\in[0,\infty),&\\ \psi(\boldsymbol{x},\widehat{\boldsymbol{\Omega}},E)=\psi^b(\boldsymbol{x},\widehat{\boldsymbol{\Omega}},E),&\\ \forall\boldsymbol{x}\in\delta V,\quad\forall\widehat{\boldsymbol{\Omega}}\cdot\hat{\boldsymbol{n}}<0,\quad,\forall E\in[0,\infty),&\end{aligned} \tag{2.1b}$$

where $\boldsymbol{x}$ is the position vector, $\widehat{\boldsymbol{\Omega}}$ is the direction vector, E is the neutron energy, Σ quantities are the macroscopic cross sections, ψ is the angular flux, ν is the average number of neutrons produced per fission, χ is the fission neutron energy spectrum, δV is the problem boundary, and $\hat{\boldsymbol{n}}$ is the outward surface normal vector.

The position vector, $\boldsymbol{x}$, is a column vector of the spatial coordinates:

$$\boldsymbol{x} \equiv \begin{bmatrix} x \\ y \\ z \end{bmatrix} . \tag{2.2}$$

The direction vector, $\widehat{\boldsymbol{\Omega}}$, is a column unit-vector which gives the direction of flight for neutrons, and is defined by

$$\widehat{\boldsymbol{\Omega}} \equiv \begin{bmatrix} \Omega_x \\ \Omega_y \\ \Omega_z \end{bmatrix} = \begin{bmatrix} \sqrt{1-\mu^2}\cos(\varphi) \\ \sqrt{1-\mu^2}\sin(\varphi) \\ \mu \end{bmatrix} , \tag{2.3a}$$

where φ is the azimuthal angle, and μ is the cosine of the polar angle θ,

$$\mu \equiv \cos(\theta). \tag{2.3b}$$

The spatial and angular coordinates systems are depicted visually in Fig. 2.1.

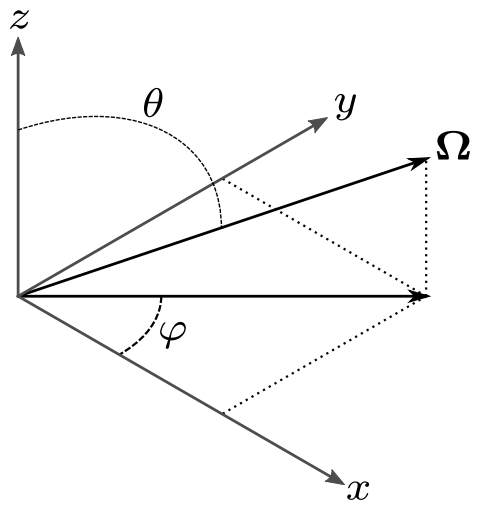


Figure 2.1: Depiction of the spatial and directional coordinate system used in the neutron transport equation.

The transport equation, given by Eq. (2.1a), is an equation that represents the balance of neutrons. The streaming term, $\widehat{\boldsymbol{\Omega}} \cdot \boldsymbol{\nabla}\psi(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, E)$, gives the rate at which neutrons are moving in or out of the of an infinitesimal volume in phase-space due to motion. The collision term, $\Sigma_t(\boldsymbol{x}, E)\psi(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, E)$, gives the rate at which neutrons have interactions (collisions) with a nucleus of the surrounding material. The source terms make up the right-hand side of the equation, and are separated into three components: an external source, the scattering source, and the fission

source. The scattering source, $\int_0^\infty \int_{4\pi} \Sigma_s(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}' \cdot \widehat{\boldsymbol{\Omega}}, E' \to E)\psi(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}', E')\,\mathrm{d}\Omega'\,\mathrm{d}E'$, gives the rate at which neutrons are scattered into the given direction and energy at a set point in space. The fission source, $\chi(\boldsymbol{x}, E)\int_0^\infty \nu\Sigma_f(\boldsymbol{x}, E')\int_{4\pi}\psi(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}', E')\,\mathrm{d}\Omega'\,\mathrm{d}E'$, gives the production rate of neutrons due to fission events. The vast majority of fission events are prompt, though a small fraction of fission events emit *delayed* neutrons. Generally, in steady-state calculations the difference between prompt and delayed fission neutrons is ignored. However, for transient calculations, capturing this difference is essential. The external source, $Q(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, E)$, is a generic term that accounts for neutrons produced by all other processes that are not dependent on the angular flux.

Generally, reactor physicists are interested in reaction rates, that are useful for determining power production, rather than the angular flux. A reaction rate at a specific point, direction, and energy can be computed as the product of the reaction cross section and the angular flux,

$$R_r(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, E) = \Sigma_r(\boldsymbol{x}, E)\psi(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, E). \tag{2.4}$$

Integration over a volume, energy range, and direction gives a total reaction rate. These quantities are the primary figures of merit for engineering applications. For convenience, it is useful to define derived quantities that are used in these calculations. The *scalar flux*

$$\phi(\boldsymbol{x}, E) \equiv \int_{4\pi} \psi(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, E)\,\mathrm{d}\Omega, \tag{2.5}$$

is the zeroth order angular moment. The neutron *current* is a vector quantity, and is the first order angular moment of the angular flux

$$\boldsymbol{J}(\boldsymbol{x}, E) \equiv \int_{4\pi} \widehat{\boldsymbol{\Omega}}\psi(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, E)\,\mathrm{d}\Omega. \tag{2.6}$$

Generally, the angular moments of the angular flux are defined as

$$\Phi_\ell^n(\boldsymbol{x}, E) \equiv \int_{4\pi} R_\ell^n(\widehat{\boldsymbol{\Omega}})\psi(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, E)\,\mathrm{d}\Omega, \tag{2.7}$$

where $R_\ell^n(\widehat{\boldsymbol{\Omega}})$ are the real spherical harmonics functions defined by

$$R_\ell^n(\widehat{\boldsymbol{\Omega}}) \equiv \sqrt{(2-\delta_{n,0})\frac{(\ell-|n|)!}{(\ell+|n|)!}}P_\ell^{|n|}(\mu)\mathcal{T}(\varphi), \tag{2.8a}$$

where $P_\ell^{|n|}(\mu)$ is the Ferrer definition [1] of the associated Legendre Polynomial defined as

$$P_\ell^{|n|}(\mu) \equiv \left(1-\mu^2\right)^{n/2} \frac{\mathrm{d}^n}{\mathrm{d}\mu^n} P_\ell(\mu), \quad n \geq 0, \tag{2.8b}$$

and

$$\mathcal{T}(\varphi) \equiv \begin{cases} \cos(n\varphi), & \text{if } n \geq 0, \\ \sin(|n|\varphi), & \text{otherwise.} \end{cases} \tag{2.8c}$$

2.2 k-Eigenvalue Problems

One of the most common calculations in reactor analysis is the simulation of reactor systems at operating conditions. A reactor operating at normal conditions is effectively unchanging in time. The common technique for solving this class of problems is to transform Eq. (2.1a) into an eigenvalue problem, such that the fission source is scaled to preserve neutron balance:

$$\begin{aligned} \left[\widehat{\boldsymbol{\Omega}} \cdot \boldsymbol{\nabla} + \Sigma_t(\boldsymbol{x}, E)\right] \psi(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, E) = \frac{1}{4\pi} \Bigg[& Q(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, E) \\ & + \int_0^\infty \int_{4\pi} \Sigma_s(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}' \cdot \widehat{\boldsymbol{\Omega}}, E' \to E) \psi(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}', E') \, \mathrm{d}\Omega' \, \mathrm{d}E' \\ & + \frac{\chi(\boldsymbol{x}, E)}{k_{\text{eff}}} \int_0^\infty \nu\Sigma_f(\boldsymbol{x}, E') \phi(\boldsymbol{x}, E') \, \mathrm{d}E' \Bigg], \end{aligned} \tag{2.9a}$$

$$\begin{gathered} \forall \boldsymbol{x} \in V, \quad \forall \widehat{\boldsymbol{\Omega}} \in 4\pi, \quad \forall E \in [0, \infty), \\ \psi(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, E) = \psi^b(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, E), \\ \forall \boldsymbol{x} \in \delta V, \quad \forall \widehat{\boldsymbol{\Omega}} \cdot \hat{\boldsymbol{n}} < 0, \quad , \forall E \in [0, \infty), \end{gathered} \tag{2.9b}$$

where k_{eff} is the inverse of the largest eigenvalue of the system λ_1. The multiplication factor, k_{eff}, indicates the criticality of the system. If k_{eff} is one, then the system is *critical* and will remain at the current conditions unless otherwise changed. A k_{eff} less than one means that the system is *subcritical* and indicates the reactor system is unable to sustain the chain reaction of nuclear fission reactions to produce power. Finally, a k_{eff} greater than one indicates that a system is *supercritical* and, if not changed, the neutron population will increase.

Generally, this class of problems are solved iteratively. This will be discussed in more detail in Section 2.5, which lists algorithms for this iteration process. Still, Eq. (2.9a) has a six-dimensional phase space and cannot, in general systems, be solved exactly. Numerical techniques must be used

to obtain approximate solutions to this equation in calculations for realistic reactor systems. In the Section 2.3, an overview of several methods for solving this equation, or approximate forms of this equation, is provided.

2.3 Computational Transport Methods

Generally, transport methods are divided into two broad categories: stochastic and deterministic. Stochastic methods, also called "Monte Carlo" methods, rely on random sampling to emulate the "life" of individual neutrons. Deterministic methods rely on discretization of the transport equation. The process of discretization introduces approximations. An overview of these different approaches is given in the following subsections.

2.3.1 Monte Carlo

Stochastic, or "Monte Carlo" methods are methods that simulate individual neutrons in the system. The simulation of each neutron relies on the random sampling of probability distributions for all aspects such as, where the *free* neutron is born, which direction it is traveling in, the energy of the neutron, the distance to the next collision, and the type of collision event. This process is repeated until the neutron leaks out of the system or is absorbed, possibly inducing a fission event with other neutrons to simulate, for many different neutrons.

Monte Carlo methods give a probabilistic estimate of the true solution as well as an associated uncertainty in that result. This class of methods is generally considered to be the most accurate because it is capable of representing the phase-space continuously. As more particles are simulated, the uncertainty in the estimated solution is reduced.

For whole-core reactor analysis, the quantities of interest would typically require an extremely large number of individual neutron histories to be simulated, typically trillions. Variance reduction techniques are an area of active research that allow for quantities of interest to be estimated accurately with fewer histories. However, Monte Carlo methods generally remain too expensive for whole-core calculations, and have challenges with multiphysics and time-dependent problems.

2.3.2 Deterministic Methods

In deterministic methods, it is generally not possible to represent the phase-space continuously. Thus these methods rely on discretization of the transport equation. In particular, spatial, directional, and energy discretization are common in these methods.

2.3.2.1 The Multigroup Approximation

The multigroup approximation is common in nearly every deterministic neutron transport method. This approximation discretizes the energy variable into discrete energy groups. Generally, cross sections have strong dependence on the energy of incident neutrons; this dependence is typically not smooth, due to the presence of resonances. Around resonance energies, the cross sections increase significantly, as observed in Fig. 2.2. The data of this plot was provided by the ENDF/B-VIII.0 nuclear reaction data library [2].

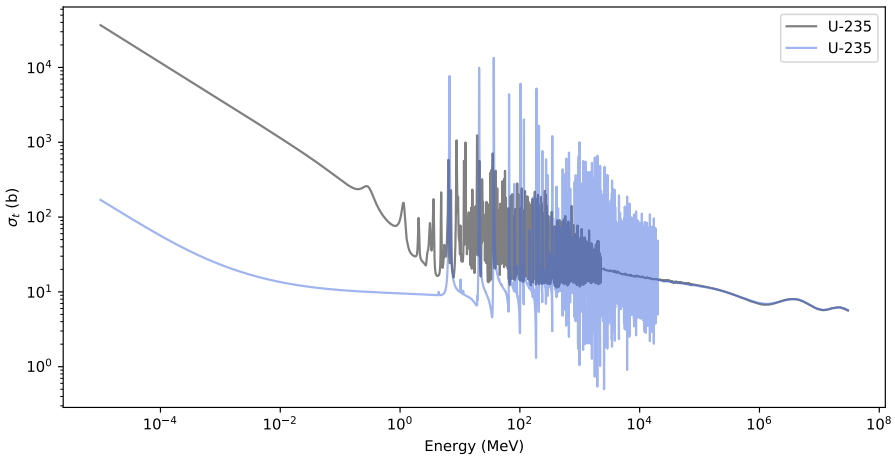


Figure 2.2: Uranium 235 and 238 total microscopic cross sections as a function of energy. Data provided through the ENDF/B-VIII.0 nuclear reaction data library.

The complicated dependence on energy would require hundreds of thousands of energy points to faithfully represent all the resonances of interest in thermal reactors. Modeling of this many energy points in whole-core simulations would be computationally impractical. The multigroup eigenvalue transport equation can be found by integrating the Eq. (2.9a) over an energy energy interval $[E_g, E_{g-1})$, where $E_g > E_{g-1}$.

$$\left[\widehat{\boldsymbol{\Omega}} \cdot \boldsymbol{\nabla} + \Sigma_t^g(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}})\right] \psi^g(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}) = \frac{1}{4\pi}\left[\sum_{g'=1}^{G} \int_{4\pi} \Sigma_s^{g' \to g}(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, \widehat{\boldsymbol{\Omega}}') \psi^{g'}(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}') \, \mathrm{d}\Omega' + \frac{\chi^g(\boldsymbol{x})}{k_{\text{eff}}} \sum_{g'=1}^{G} \nu \Sigma_f^{g'}(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}) \phi^{g'}(\boldsymbol{x})\right], \tag{2.10}$$

$$\forall \boldsymbol{x}, \quad \forall \widehat{\boldsymbol{\Omega}} \in 4\pi, \quad \forall g \in \{1, 2, \ldots, G\},$$

where the multigroup quantities are defined by

$$\psi^g(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}) \equiv \int_{E_g}^{E_{g-1}} \psi(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, E)\, \mathrm{d}E, \tag{2.11a}$$

$$\chi^g(\boldsymbol{x}) \equiv \int_{E_g}^{E_{g-1}} \chi(\boldsymbol{x}, E)\, \mathrm{d}E, \tag{2.11b}$$

$$\Sigma_t^g(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}) \equiv \frac{\int_{E_g}^{E_{g-1}} \Sigma_t(\boldsymbol{x}, E)\psi(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, E)\, \mathrm{d}E}{\psi^g(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}})}, \tag{2.11c}$$

$$\nu\Sigma_f^g(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}) \equiv \frac{\int_{E_g}^{E_{g-1}} \nu\Sigma_f(\boldsymbol{x}, E)\psi(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, E)\, \mathrm{d}E}{\psi^g(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}})}, \tag{2.11d}$$

$$\Sigma_s^{g'\to g}(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, \widehat{\boldsymbol{\Omega}}') \equiv \frac{\int_{E_g}^{E_{g-1}} \int_{E_{g'}}^{E_{g'-1}} \Sigma_s(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}' \cdot \widehat{\boldsymbol{\Omega}}, E' \to E)\psi(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}', E')\, \mathrm{d}E'\, \mathrm{d}E}{\psi^{g'}(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}')}. \tag{2.11e}$$

By defining the cross sections in this way, no approximations have been made, and the reaction rates of each energy group are preserved. However, this approach has two issues: the cross sections are dependent on the angular flux which is not known *a priori*, and have dependence on the neutron direction of flight. Generally, the dependence on the angular flux is addressed by solving a *spatially* simplified problem to generate a continuous or fine-group neutron energy spectrum. This spectrum is then used as the weighting function (in place of $\psi^g(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}})$) to "collapse" the cross sections into coarser multigroup values [3]. This introduces an approximation into the transport equation.

To eliminate the directional dependence of the multigroup cross sections, an additional approximation is made: isotropic angular flux spectrum,

$$\psi(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}, E) \approx \frac{1}{4\pi}\Phi(\boldsymbol{x}, E). \tag{2.12}$$

Using this approximate angular flux as the weighting function for multigroup cross sections in Eq. (2.10) and Eqs. (2.11) can be simplified to

$$\begin{aligned}\left[\widehat{\boldsymbol{\Omega}} \cdot \boldsymbol{\nabla} + \Sigma_t^g(\boldsymbol{x})\right] \psi^g(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}) = \frac{1}{4\pi}\Bigg[&\sum_{g'=1}^{G} \int_{4\pi} \Sigma_s^{g'\to g}(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}' \cdot \widehat{\boldsymbol{\Omega}})\psi^{g'}(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}')\, \mathrm{d}\Omega' \\ &+ \frac{\chi^g(\boldsymbol{x})}{k_{\text{eff}}} \sum_{g'=1}^{G} \nu\Sigma_f^{g'}(\boldsymbol{x}) \int_{4\pi} \psi^{g'}(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}')\, \mathrm{d}\Omega'\Bigg], \\ \forall \boldsymbol{x}, \quad \forall \widehat{\boldsymbol{\Omega}} \in 4\pi, \quad &\forall g \in \{1, 2, \ldots, G\},\end{aligned} \tag{2.13}$$

where the approximated multigroup cross sections are defined as

$$\Sigma_t^g(\boldsymbol{x}) \equiv \frac{\int_{E_g}^{E_{g-1}} \Sigma_t(\boldsymbol{x}, E)\Phi(\boldsymbol{x}, E)\,\mathrm{d}E}{\int_{E_g}^{E_{g-1}} \Phi(\boldsymbol{x}, E)\,\mathrm{d}E}, \tag{2.14a}$$

$$\nu\Sigma_f^g(\boldsymbol{x}) \equiv \frac{\int_{E_g}^{E_{g-1}} \nu\Sigma_f(\boldsymbol{x}, E)\Phi(\boldsymbol{x}, E)\,\mathrm{d}E}{\int_{E_g}^{E_{g-1}} \Phi(\boldsymbol{x}, E)\,\mathrm{d}E}, \tag{2.14b}$$

$$\Sigma_s^{g'\to g}(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}' \cdot \widehat{\boldsymbol{\Omega}}) \equiv \frac{\int_{E_g}^{E_{g-1}} \int_{E_{g'}}^{E_{g'-1}} \Sigma_s(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}' \cdot \widehat{\boldsymbol{\Omega}}, E' \to E)\Phi(\boldsymbol{x}, E')\,\mathrm{d}E'\,\mathrm{d}E}{\int_{E_{g'}}^{E_{g'-1}} \Phi(\boldsymbol{x}, E')\,\mathrm{d}E'}. \tag{2.14c}$$

For thermal reactors, the weighting spectrum, $\Phi(\boldsymbol{x}, E)$, is well approximated by a spatially uniform spectrum, $\Phi(E)$. This justifies the use of the spectrum of the spatially simplified system. However, this is not the case for other systems, such as fast reactors. In such systems, the spatial dependence of the weighting spectrum plays is important to capture accurately, and the above methods cannot be used.

2.3.2.2 Spatial Discretization

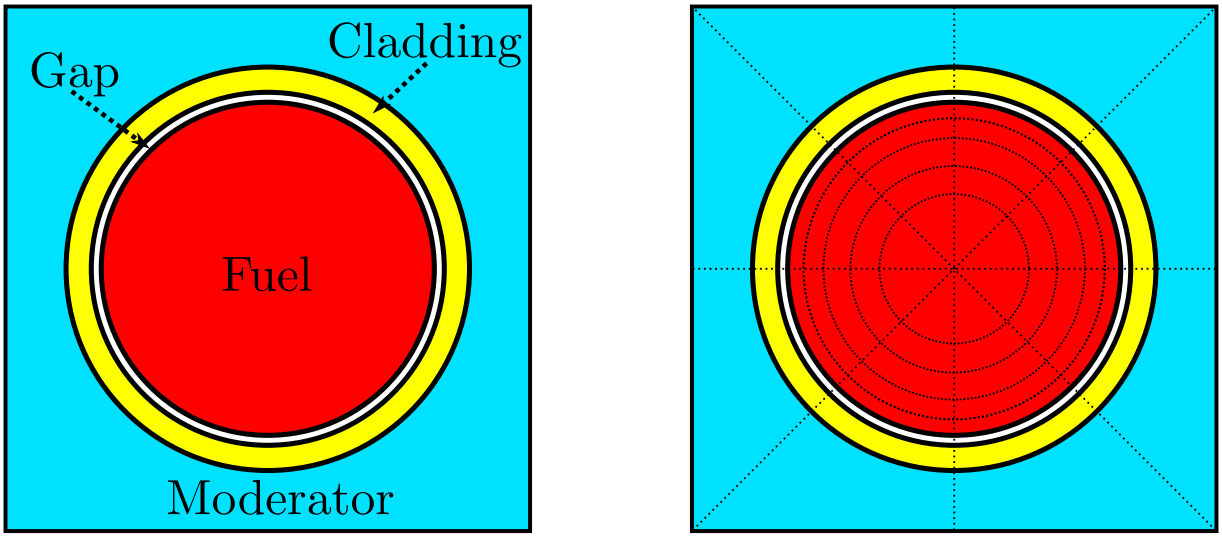
Nearly all computational transport methods involve some form of spatial discretization. Reactor designs include many different material regions, and nearly all simulation tools will discretize the spatial domain into these different material regions. Deterministic methods will generally apply a finer meshing within these material regions, to discretize them into *transport cells*. For the purposes of this work, a cell $\mathcal{R}_i$ is indexed with i. A visualization of the material and hypothetical meshing of a characteristics based transport method for a single pin-cell are shown in Fig. 2.3. In deterministic codes, the typical assumption is that material properties (cross sections) are constant within each computational cell.

2.3.2.3 Directional Discretization

Typically, the directional variable cannot be treated exactly in deterministic methods. There are two common methods of approximating the solution as a function of direction $\widehat{\boldsymbol{\Omega}}$:

1. Spherical Harmonics (P_N) Expansion
2. Discrete Ordinates (S_N)

Spherical Harmonics (P_N) Expansion Expansion in spherical harmonics, often referred to as P_N, is one of the oldest transport methods, where N indicates the order of the expansion. In this



(a) Material regions

(b) Hypothetical computational cell mesh

Figure 2.3: Material and mesh spatial discretization examples for a single pin cell.

method, the angular flux is expanded as a linear combination of spherical harmonics moments:

$$\psi_m^g(\boldsymbol{x}) = \sum_{l=0}^{\infty} \sum_{n=-l}^{l} \Phi_\ell^n(\boldsymbol{x}, E) R_\ell^n(\widehat{\boldsymbol{\Omega}}), \tag{2.15}$$

where the spherical harmonics,$R_\ell^n(\widehat{\boldsymbol{\Omega}})$, are defined by Eqs. (2.8). No approximation has been introduced at this point, but in practice this series is truncated at some finite number N:

$$\psi_m^g(\boldsymbol{x}) \approx \sum_{l=0}^{N} \sum_{n=-l}^{l} \Phi_\ell^n(\boldsymbol{x}, E) R_\ell^n(\widehat{\boldsymbol{\Omega}}). \tag{2.16}$$

The P_N equations can be obtained by multiplying the multi-group transport equation (Eq. (2.13)) by $R_\ell^n(\widehat{\boldsymbol{\Omega}})$ for each valid (l, n) pair under the specified order, and integrating over 4π. This yields a system of $(N+1)^2$ equations for each energy group; the number of equations increases quadratically with increasing orders, making P_N methods less feasible for large calculations.

Discrete Ordinates (S_N) The Discrete Ordinates (S_N) method is a discretization of the directional variable $\widehat{\boldsymbol{\Omega}}$ by a quadrature. Let the $\mathcal{M}_N$ be the set of discrete directions, and weights,

$$\mathcal{M}_N \equiv \left\{ \widehat{\boldsymbol{\Omega}}_m \in \{\widehat{\boldsymbol{\Omega}}_1, \widehat{\boldsymbol{\Omega}}_2, \ldots, \widehat{\boldsymbol{\Omega}}_N\}, w_m \in \{w_1, w_2, \ldots, w_N\} \right\}, \tag{2.17a}$$

such that a directional integration can be approximated as

$$\int_{4\pi} f(\widehat{\mathbf{\Omega}})\,\mathrm{d}\Omega \approx 4\pi \sum_{m\in\mathcal{M}_N} w_m f(\widehat{\mathbf{\Omega}}_m), \tag{2.17b}$$

where

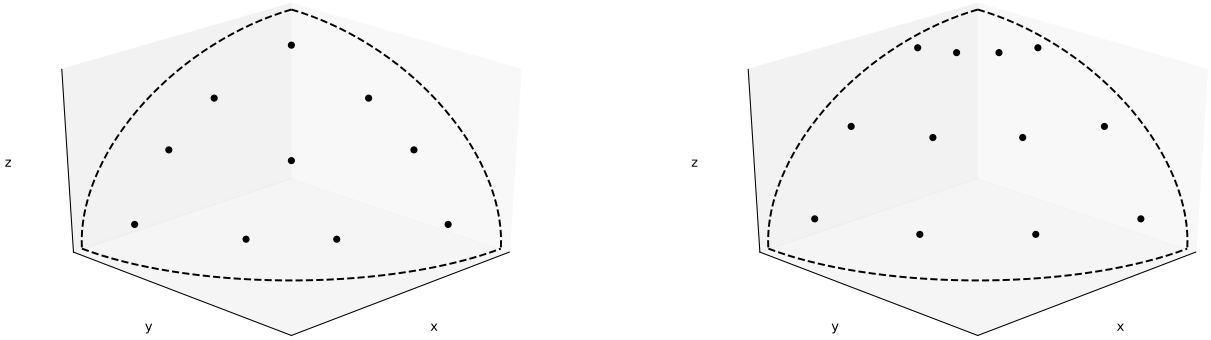
$$\sum_{m\in\mathcal{M}_N} w_m = 1. \tag{2.17c}$$

There are two common forms of quadrature sets that are commonly used in transport calculations: level-symmetric and product quadratures. The level-symmetric quadratures include directions that are (roughly) evenly distributed over the unit-sphere; this is optimal in situations where each direction has similar variation. However, typical reactor designs have significantly less variation in the axial (z) direction, along which fuel rods are oriented. In this situation, neutrons with directions close to the z-axis are modeled poorly because there are few azimuthal angles at these polar levels, as is demonstrated in Fig. 2.4a. These steep polar angles are important in reactor analysis due to self-shielding effects, which are strongly dependent on polar angle.

Product quadratures are generated by a multiplicative combination of separate quadrature sets in the azimuthal and polar directions. The azimuthal quadrature set is generated over the domain $[0, 2\pi]$, while the polar quadrature set is generated for the polar cosine, μ, over the domain $[-1, 1]$. This quadrature generation technique does not suffer from the same issue for steep polar directions, as each polar level has the same number of azimuthal directions. A common choice for the azimuthal quadrature generation is the Chebyshev quadrature, which gives evenly spaced azimuthal angles with equal weights. The polar cosine quadrature set typically uses a Gauss-Legendre quadrature, or an optimized quadrature such as the Tabuchi-Yamamoto quadrature [4]. Figure 2.4b shows an example of a product quadrature's set of directions using a Chebyshev azimuthal quadrature and Gauss-Legendre polar quadrature.

2.4 State-of-the-Art 3-D Computational Transport Methods

Until recently, whole-core neutronics calculations were carried out primarily through a two-step procedure. First, a transport method was used to compute homogenized cross section data for assemblies, and then 3-D diffusion was used to solve the problem. More recently, research has focused on a one-step approach, so-called direct whole-core transport. In this approach, the 3-D reactor is directly modeled using transport methods. There are many different transport methods that are currently being researched; this section seeks to give an overview of several state-of-the-art 3-D transport methods.



(a) Level-Symmetric Quadrature (S_8)

(b) Chebyshev-Gauss Product Quadrature with 4 azimuthal and 3 polar angles

Figure 2.4: (a) Level-Symmetric and (b) product quadrature direction set examples shown for a single octant of the unit-sphere.

2.4.1 SP_N

The Simplified P_N (SP_N) method was introduced by Gelbard [5] in 1961, and was seen as a middle ground between diffusion and transport [6]. The SP_N equations were first derived by examining the form of the 1-D P_N equations; in fact, the 1-D SP_N and P_N equations are equivalent. While the original derivation Gelbard [5] lacked theoretical justification, the SP_N method has since been shown to be an asymptotic correction to standard diffusion theory [7].

The mono-energetic planar geometry P_N equations can be written as

$$\frac{\mathrm{d}}{\mathrm{d}z}\left[\frac{l}{2l+1}\phi_{l-1}+\frac{l+1}{2l+1}\phi_{l+1}\right]+\Sigma_t\phi_l=\Sigma_{s,l}\phi_l+Q\delta_{l,0},\quad \text{for}\quad 0\leq l\leq N, \tag{2.18a}$$

where $\Sigma_{s,l}$ is the lth order scattering moment and Q is either an external source or fission source. The expansion is generally truncated by assuming that $\phi_{N+1}=0$.

The 1-D P_1 equation can be written as

$$-\frac{\mathrm{d}}{\mathrm{d}z}D\frac{\mathrm{d}}{\mathrm{d}z}\phi_0+\Sigma_t\phi_0=\Sigma_{s,0}\phi_0+Q, \tag{2.19}$$

where

$$D\equiv\frac{1}{3(\Sigma_t-\Sigma_{s,1})}. \tag{2.20}$$

The 3-D P_1 equations simply replace the derivative term operator of Eq. (2.19) with the 3-D diffusion operator,

$$\frac{\mathrm{d}}{\mathrm{d}z}D\frac{\mathrm{d}}{\mathrm{d}z}\rightarrow\boldsymbol{\nabla}\cdot D\boldsymbol{\nabla},$$

yielding the 3-D P_1 (diffusion) equation:

$$-\boldsymbol{\nabla}\cdot D\boldsymbol{\nabla}\phi_0+\Sigma_t\phi_0=\Sigma_{s,0}\phi_0+Q. \tag{2.21}$$

This simple relation between 1-D and 3-D equations only holds for the special P_1 case. The SP_N method uses a similar modification for higher-order P_N equations. This results in some lost accuracy compared to P_N, but the SP_N equations are significantly simpler to solve [5, 6]. Unlike P_N, SP_N equations do not converge to the transport solution as $N\to\infty$, but they do generally have higher accuracy than diffusion (for orders of $N>1$). Generally, SP_3 or SP_5 are considered to have sufficient accuracy, and are generally less computationally intensive than transport methods.

The mono-energetic SP_N equations can be written as

$$-\boldsymbol{\nabla}\cdot\frac{1}{3\Sigma_{tr,1}}\boldsymbol{\nabla}\phi_0-\boldsymbol{\nabla}\cdot\frac{2}{3\Sigma_{tr,1}}\boldsymbol{\nabla}\phi_2+\Sigma_{tr,0}\phi_0=Q, \tag{2.22a}$$

$$\begin{aligned}
&-\boldsymbol{\nabla}\cdot\left(\frac{n(n-1)}{(2n+1)(2n-1)\Sigma_{tr,n-1}}\right)\boldsymbol{\nabla}\phi_{n-2}\\
&-\boldsymbol{\nabla}\cdot\left(\frac{(n+1)(n+2)}{(2n+1)(2n+3)\Sigma_{tr,n+1}}\right)\boldsymbol{\nabla}\phi_{n+2}\\
&-\boldsymbol{\nabla}\cdot\left(\frac{n^2}{(2n+1)(2n-1)\Sigma_{tr,n-1}}+\frac{(n+1)^2}{(2n+1)(2n+3)\Sigma_{tr,n+1}}\right)\boldsymbol{\nabla}\phi_n\\
&+\Sigma_{tr,n}\phi_n=0,\quad \text{for}\quad n=2,4,...,N-1,
\end{aligned} \tag{2.22b}$$

where

$$\Sigma_{tr,n}\equiv\Sigma_t-\Sigma_{s,n}. \tag{2.22c}$$

2.4.2 Method of Characteristics

The Method of Characteristics (MoC) is a technique used to solve first-order Partial Differential Equations (PDEs), by transforming the PDE into a system of Ordinary Differential Equations (ODEs). The method was first applied to the neutron transport problem by Askew in 1972 [8], but it only began to see real use in the 1980's [9]. The MoC transforms the transport equation into the characteristic form, by examining the equation along straight neutron paths through the spatial domain. By examining the equation along one of these characteristic "tracks" or "rays", the average angular flux along the track within a cell can be calculated. The scalar flux can then be found by collecting the average angular flux along all tracks passing through this region, in a numerical integration over space and angle.

Like the Collision Probability (CP) method, MoC is able to handle completely arbitrary geometry; however, unlike the CP method, it is also able to account for anisotropic scattering in a

straightforward manner. Additionally, the MoC does not produce the large matrices in realistic applications as the CP method does. For problems that contain more than a few hundred cells, the MoC is generally preferred over CP methods [10].

The method of Characteristic Direction Probabilities (CDP) is a method similar to both CP method and the MoC [11, 12]. The CDP uses ray-tracing to evaluate transmission probabilities between cells. However, it only considers transmission probabilities between cells that are traversed by a shared characteristic ray, rather than considering the transmission probability between all cells as in the CP methods. This significantly reduces the computational resources required by traditional CP methods. This method has also shown improvements over MoC in cases with few unique geometries and constant material properties throughout the simulation; however these conditions are not applicable in problems of interest to industry.

MoC has seen extensive use as a 2-D transport method in lattice codes [13, 14], and as a radial transport method in 2D/1D method [15–18]. However, due to the large number of segments generated by 3-D ray-tracing, the MoC has been viewed as too expensive for realistic applications [19]. Nonetheless, 3-D MoC is typically seen as the "gold-standard" of transport methods due to its general treatment of geometry, ability to handle anisotropic scattering, and accuracy. For this reason, there has been active research on improving the 3-D MoC.

Kochunas [20] developed a hybrid shared/distributed parallel algorithm for 3-D MoC. This contribution significantly increased the ability to parallelize 3-D MoC on modern computer clusters, and showed good parallel scaling on thousands of cores. Other works investigated the linear-source approximation (LSA) for 3-D MoC applications [21], which reduces the number of segments generated during ray-tracing. Sciannandrone et al. [22] developed optimized tracking strategies (chord-classification) for locally extruded geometries for 3-D MoC calculations. These optimized strategies did not reduce the number of tracks or segments generated, but they significantly reduced the memory required to store them. Still, 3-D MoC remains too expensive for usage outside work in academia or national laboratories.

The MoC is the primary subject of this thesis work. Chapter 3 is devoted to the details of the method, and Section 3.4 expands upon the details of current ray-tracing techniques used in MoC. Chapter 5 details improvements made to the MoC in this thesis work, and Chapter 6 describes a newly investigated ray-tracing method.

2.4.3 2D/1D Methods

The 2D/1D methods were first developed by researchers at Korea Advanced Institute of Science and Technology (KAIST) [23] and Korea Atomic Energy Research Institute (KAERI) [17], in the CRX and DeCART codes, respectively. Though different, these two methods followed the same

fundamental approach to solving 3-D reactor transport problems:

1. Divide the core into separate axial slices/planes,
2. Perform 2-D transport calculations within each plane,
3. Couple the planes with transverse leakages.

These methods were based on the assumption that reactors may be very heterogeneous in the radial direction, but in the axial direction they are relatively homogeneous. The primary difference between the two methods is in the transverse leakage terms; in CRX the transverse leakages are anisotropic, but in DeCART the leakages are isotropic. To distinguish these methods, they will be referred to as anisotropic and isotropic 2D/1D methods.

Following the relative success of these methods, other research groups have followed in their paths. nTRACER [24], and MPACT [15] used the isotropic 2D/1D method. The STREAM [25], and APOLLO3 [26] have also implemented 2D/1D methods.

Stimpson [27] implemented an anisotropic 2D/1D method in MPACT using a Fourier expansion of the azimuthal angles for the axial and radial transverse leakages. Jarrett [28] further improved upon Stimpson's [27] work by introducing a 2D/1D method using P_3 in the axial direction. 2D/1D methods are fundamentally based on the idea that heterogeneity in the axial direction is low; in cases where this is not true these methods have been observed to break down, or yield solutions with larger errors.

2.4.3.1 Derivation

The details of the 2D/1D equations have implications on results in Chapter 5. In this section, the 2D/1D equations are derived. We begin with the mono-energetic transport equation

$$\boldsymbol{\Omega}\cdot\boldsymbol{\nabla}\psi(\boldsymbol{x},\boldsymbol{\Omega})+\Sigma_t(\boldsymbol{x})\psi(\boldsymbol{x},\boldsymbol{\Omega})=\frac{Q(\boldsymbol{x})}{4\pi}, \tag{2.23}$$

where $Q(\boldsymbol{x})$ is the source.

2D/1D methods divide the transport equation into radial (2-D) and axial (1-D) equations. The streaming term, $\boldsymbol{\Omega}\cdot\boldsymbol{\nabla}\psi$ can be divided into the radial and axial components:

$$\boldsymbol{\Omega}\cdot\boldsymbol{\nabla}\psi=(\boldsymbol{\Omega}\cdot\boldsymbol{\nabla})_{xy}\psi+\mu\frac{\partial\psi}{\partial z}. \tag{2.24}$$

The radial equation is obtained by first separating the axial derivative from the streaming term:

$$\left[(\boldsymbol{\Omega}\cdot\boldsymbol{\nabla})_{xy}+\Sigma_t\right]\psi(\boldsymbol{x},\boldsymbol{\Omega})=\frac{Q(\boldsymbol{x})}{4\pi}-\mu\frac{\partial\psi}{\partial z}. \tag{2.25}$$

Equation (2.25) is then integrated over an axial plane k, over the interval $[z_{k-1/2}, z_{k+1/2}]$, yielding

$$\left[(\boldsymbol{\Omega}\cdot\boldsymbol{\nabla})_{xy} + \Sigma_{t,k}(\boldsymbol{x}_{xy})\right]\psi_k(\boldsymbol{x}_{xy},\boldsymbol{\Omega}) = \frac{Q_k(\boldsymbol{x}_{xy})}{4\pi} - \widetilde{J}_{z,k}(\boldsymbol{x}_{xy},\boldsymbol{\Omega}), \tag{2.26a}$$

where

$$\psi_k(\boldsymbol{x}_{xy},\boldsymbol{\Omega}) \equiv \frac{1}{h_k}\int_{z_{k-1/2}}^{z_{k+1/2}} \psi(\boldsymbol{x},\boldsymbol{\Omega})\,\mathrm{d}z, \tag{2.26b}$$

$$Q_k(\boldsymbol{x}_{xy}) \equiv \frac{1}{h_k}\int_{z_{k-1/2}}^{z_{k+1/2}} Q(\boldsymbol{x},\boldsymbol{\Omega})\,\mathrm{d}z, \tag{2.26c}$$

$$\widetilde{J}_{z,k}(\boldsymbol{x}_{xy},\boldsymbol{\Omega}) = \frac{\mu}{h_k}\left[\psi(\boldsymbol{x}_{xy},z_{k+1/2},\boldsymbol{\Omega}) - \psi(\boldsymbol{x}_{xy},z_{k-1/2})\right], \tag{2.26d}$$

and $h_k = z_{k+1/2} - z_{k-1/2}$. Generally, the difference between isotropic and anisotropic 2D/1D methods is in the treatment of the transverse leakages, $\widetilde{J}_{z,k}$. In MPACT, the axial transverse leakage is typically discretized over the coarse radial mesh (typically a pin cell), and is assumed to be isotropic.

The axial equations are found by moving the radial streaming term to the right hand side of Eq. (2.23),

$$\left[\mu\frac{\partial}{\partial z} + \Sigma_t\right]\psi(\boldsymbol{x},\boldsymbol{\Omega}) = \frac{Q(\boldsymbol{x})}{4\pi} - (\boldsymbol{\Omega}\cdot\boldsymbol{\nabla})_{xy}\psi(\boldsymbol{x},\boldsymbol{\Omega}), \tag{2.27}$$

and integrating over a radial area. This radial area can be a fine cell (transport cell), but is typically a coarser discretization such as a pin cell [28]. We operate on Eq. (2.27) by

$$\frac{1}{A_{ij}}\int_{x_{i-1/2}}^{x_{i+1/2}}\int_{y_{j-1/2}}^{y_{j+1/2}}(\cdot)\,\mathrm{d}x\,\mathrm{d}y = \frac{1}{A_{ij}}\iint_{ij}(\cdot)\,\mathrm{d}x\,\mathrm{d}y,$$

to obtain:

$$\left[\mu\frac{\partial}{\partial z} + \Sigma_{t,k,ij}(z,\boldsymbol{\Omega})\right]\psi_{ij}(z,\boldsymbol{\Omega}) = \frac{Q_{ij}(z)}{4\pi} - \frac{1}{A_{ij}}\sum_{s\in\{N,S,E,W\}}(\boldsymbol{\Omega}\cdot\widehat{n}_s)\psi_{ij,s}(z,\boldsymbol{\Omega}), \tag{2.28a}$$

where

$$\Sigma_{t,k,ij}(z,\boldsymbol{\Omega}) \equiv \frac{\iint_{ij}\Sigma_{t,k}(\boldsymbol{x}_{xy})\psi_k(\boldsymbol{x}_{xy},\boldsymbol{\Omega})\,\mathrm{d}x\,\mathrm{d}y}{\iint_{ij}\psi_k(\boldsymbol{x}_{xy},\boldsymbol{\Omega})\,\mathrm{d}x\,\mathrm{d}y}, \tag{2.28b}$$

where s indicates a surface for the cardinal directions $\{N,S,E,W\}$. For hexagonal pins there would instead be six directions to loop over; for simplicity this work assumes cartesian pin cells. Typically the coarse cell cross sections are taken to be isotropic, but polar dependence has been investigated [28].

The radial equations, Eqs. (2.26), are solved by a 2-D transport method, such as MoC, while

the axial equations, Eqs. (2.28), are solved by either a transport method or approximate method such as diffusion [16, 28]. The radial equations yield surface fluxes on the pin faces, while the axial equations yield surface fluxes on the axial planar faces. The radial equations are, however, integrated over an axial plane which results in an axial flat solution within each plane. These methods then require many axial planes to resolve the axial behavior in a system, particularly if there is significant axial variation in the solution.

2.4.3.2 Transverse Leakage Splitting

Eqs. (2.26) give no guarantee that the right-hand-sides are positive. Negative sources present considerable challenges to iteration stability when using non-linear acceleration methods such as coarse mesh finite-difference (CMFD) [28]. The transverse leakage splitting method [27, 29] was developed to "split" the transverse leakage term such that the total source $(Q + \widetilde{J})$ is non-negative. A split term, $\widetilde{L}_z$, is defined

$$\widetilde{L}_z \equiv -\left[\frac{Q_k(\boldsymbol{x}_{xy})}{4\pi} - \widetilde{J}_{z,k}(\boldsymbol{x}_{xy}, \boldsymbol{\Omega})\right], \tag{2.29}$$

and is added to both sides of Eq. (2.26a). The left hand side (the 2-D transport operator) cannot have a free-source term, so it is necessary to merge this split term into the cross section term. The angular flux is not known a priori, so a approximation is used, typically isotropic flux:

$$\psi_k(\boldsymbol{x}_{xy}, \boldsymbol{\Omega}) \approx \frac{\phi_k(\boldsymbol{x}_{xy})}{4\pi}. \tag{2.30}$$

This approximation is introduced only to combine the left-hand-side leakage term into the cross section term as

$$\widetilde{\Sigma}_{t,k}(\boldsymbol{x}_{xy}) \equiv \Sigma_{t,k}(\boldsymbol{x}_{xy}) + 4\pi\frac{\widetilde{L}_z}{\phi_k(\boldsymbol{x}_{xy})}. \tag{2.31}$$

The radial equation then becomes

$$\left[(\boldsymbol{\Omega}\cdot\boldsymbol{\nabla})_{xy} + \widetilde{\Sigma}_{t,k}(\boldsymbol{x}_{xy})\right]\psi_k(\boldsymbol{x}_{xy}, \boldsymbol{\Omega}) = 0. \tag{2.32}$$

Because splitting introduces an approximation, it generally decreases the accuracy of the solution. Thus, the transverse splitting method is typically use only in cells that have a negative source. More recently, alternative methods have been investigated [30], but they will not be covered in this work.

2.4.4 Extruded 3-D Methods

Due to the wide adoption of 2D/1D methods in recent years, some current research paths have turned towards *extruded mesh* 3-D transport methods. An axially extruded mesh is a mesh that uses the same radial discretization for all axial positions in the domain. These methods generally rely on the mesh of the problem to be an axially extruded mesh, and use 3-D transport methods that rely on that assumption. This an active area of research and while they usually rely on similar assumptions, many different approaches have been taken.

The PROTEUS [31] code introduced a new transport methodology that combined 2-D MoC with a discontinuous Galerkin method (finite-elements) for treatment of the axial variable. Other groups have used similar approaches, but with different axial basis functions; STREAM [25] uses linear orthogonal functions, and recent work at University of Michigan (UM) [32] has utilized the Legendre polynomials. Recent work on STREAM has used a diamond-difference method for approximating axial behavior.Generally, these methods do not suffer from the same inaccuracies of standard 2D/1D methods, but they do still assume the use of extruded meshes.

The Legendre polynomial expansion of angular flux (LEAF) method, developed by Yamamoto et al. [33] in the GENESIS code, is an extruded method which is distinct from the PROTEUS-like methods. This method is the successor of the ASMOC3D method [34]. The LEAF method performs 2-D radial ray-tracing to generate a set of characteristic planes. The angular flux on the left, right, top, and bottom surfaces of each characteristic plane is given an assumed shape as a linear combination of Legendre polynomials. A transport calculation is then carried out in each of these characteristic planes by the method of short characteristics.

The extruded 3-D transport methods seem to have accuracy benefits over the more traditional 2D/1D methods; however, they are usually more computationally expensive, though reducing this cost is an active area of research. Unlike general 3-D transport methods, such as 3-D MoC, these extruded methods are unable to handle general geometries. This may be of concern in future reactor designs, but in most modern reactor designs this is an appropriate simplification to make.

2.5 Source Iteration

Generally, the k-eigenvalue transport problems, introduced in Section 2.2, are solved iteratively. Given an initial guess for the k-eigenvalue, boundary conditions, and interior flux-moments, an estimate of the source can be computed. A transport "sweep" can be performed, in which updated boundary conditions and flux-moments are computed. Given these updated flux-moments a new estimate of the eigenvalue can be calculated. This process can be repeated until the eigenvalue and flux-moments are converged within some tolerance. This procedure is referred to as either

source iteration, or power iteration; typically, power iteration is used to refer to this procedure in problems with a fixed source. For simplicity, this process is shown for a isotropic mono-energetic, continuous-space, one-dimensional transport problem in Algorithm 1.

Algorithm 1 Source Iteration algorithm for the k-eigenvalue transport problem.

1: Begin iteration j with a known boundary conditions, scalar flux estimate, $\phi^{(j)}(x)$, and a k-eigenvalue estimate, k_{eff}^{j}.

2: Perform a transport sweep:

$$\left[\mu\frac{\partial}{\partial x} + \Sigma_t(x)\right]\psi^{(j+1)}(x,\mu) = \frac{1}{2}\left[\Sigma_s(x) + \frac{1}{k_{\text{eff}}^{(j)}}\nu\Sigma_f(x)\right]\phi^{(j)}(x), \tag{2.33a}$$

$$\forall x \in [0,X], \quad \forall \mu \in [-1,1],$$

$$\psi^{(j)}(0,\mu) = \psi^{b}(\mu), \quad \mu \in [0,1], \tag{2.33b}$$

$$\psi^{(j)}(X,\mu) = \psi^{b}(\mu), \quad \mu \in [-1,0]. \tag{2.33c}$$

3: Update the scalar flux, and the eigenvalue for the next iteration:

$$\phi^{(j+1)}(x) = \left(\int_{-1}^{1}\psi^{(j+1)}(x,\mu)\,\mathrm{d}\mu\right)\frac{\Phi_0}{\frac{1}{X}\int_0^X\int_{-1}^{1}\psi^{(j+1)}(x',\mu)\,\mathrm{d}\mu\,\mathrm{d}x'}, \tag{2.34a}$$

$$k_{\text{eff}}^{(j+1)} = \frac{\int_0^X \nu\Sigma_f\phi^{(j+1)}(x)\,\mathrm{d}x}{\int_0^X \Sigma_a\phi^{(j+1)}(x)\,\mathrm{d}x + \int_{-1}^{0}\psi^{(j+1)}(0,\mu)\,\mathrm{d}\mu + \int_0^1\psi^{(j+1)}(X,\mu)\,\mathrm{d}\mu}. \tag{2.34b}$$

4: Repeat steps 1. - 3. until sufficient convergence.

2.5.1 Transport Acceleration

While Algorithm 1 is valid, it typically converges very slowly, requiring many iterations to obtain reasonable results. In full-core calculations, a single transport sweep is typically the most computationally expensive operation. For this reason, solving practical problems using Algorithm 1 is not feasible. There has been considerable effort in developing methods that accelerate the convergence of transport calculations by using a lower-order calculation, typically based on the diffusion approximation. These methods often reduce the number of iterations from O(1000) to O(10) for thermal reactor systems.

While other acceleration methods exist [35], the most common for k-eigenvalue problems are non-linear diffusion acceleration (NDA) methods [36], typically using the CMFD method [37]. Algorithm 2 lists the standard NDA algorithm for a 1-D mono-energetic k-eigenvalue problem. The primary difference between CMFD and NDA is that CMFD introduces the concept of a second, coarser spatial grid, while NDA uses the same spatial mesh as the transport problem.

The CMFD acceleration method has been shown to significantly reduce computational transport run-times [16, 36, 38]. There have been several improvements upon the original CMFD formulation [37], such as pCMFD [23] and odCMFD [39]. The pCMFD method preserves partial currents rather than net currents, and has been shown to be unconditionally stable for transport problems at fixed conditions [23]. The odCMFD method generalizes the CMFD and pCMFD methods by adding an artificial term to the diffusion coefficient, and has faster convergence properties than pCMFD [39]. Recently, it was shown that the theoretical reason for this is the that most MoC methods do not preserve the linear solutions of the transport equation, resulting in the need for an unphysical correction to the diffusion coefficient to account for truncation error in the transport discretization.

Utilization of CMFD acceleration in transport calculations with thermal-hydraulic (T/H) feedback has not had the favorable stability and convergence properties as calculations without feedback [40]. Many transport codes have required under-relaxation of the scalar flux in the iteration schemes for stability in these calculations; there is ongoing research investigating a less ad-hoc approach [40]. This instability and convergence slow-down in problems with feedback has prevented full utilization of more advanced multi-level solvers [40, 41].

Algorithm 2 Non-linear Diffusion Acceleration (NDA) algorithm for the k-eigenvalue transport problem.

1: Begin iteration i with known boundary conditions, scalar flux estimate, $\phi^{(j)}(x)$, a net current estimate, $\boldsymbol{J}^{(j)}(x)$, and a k-eigenvalue estimate, k_{eff}.

2: Compute linear and non-linear correction factors for low-order diffusion equation:

$$\widehat{D}^{(j)}(x) = \frac{\frac{\mathrm{d}}{\mathrm{d}x}\left[J^{(j)}(x) + \frac{1}{3\Sigma_t(x)}\frac{\mathrm{d}\phi^{(j)}(x)}{\mathrm{d}x}\right]}{\phi^{(j)}(x)} \tag{2.35}$$

3: Solve the low-order diffusion eigenvalue problem:

$$-\frac{\mathrm{d}}{\mathrm{d}x}\frac{1}{3\Sigma_t(x)}\frac{\mathrm{d}\phi^{(j+1/2)}(x)}{\mathrm{d}x} + \left[\Sigma_a(x) + \widehat{D}^{(j)}(x)\right]\phi^{(j+1/2)}(x) = \frac{1}{k_{\text{eff}}}\nu\Sigma_f(x)\phi^{(j+1/2)}(x). \tag{2.36}$$

4: Perform a transport sweep using the scalar flux, and eigenvalue estimates from the low-order diffusion calculation:

$$\left[\mu\frac{\partial}{\partial x} + \Sigma_t(x)\right]\psi^{(j+1)}(x,\mu) = \frac{1}{2}\left[\Sigma_s(x) + \frac{1}{k_{\text{eff}}^{(j)}}\nu\Sigma_f(x)\right]\phi^{(j)}(x), \tag{2.37}$$

$$\forall x \in [0, X], \quad \forall \mu \in [-1, 1].$$

5: Update the scalar flux, current, and eigenvalue estimates for next iteration:

$$\phi^{(j+1)}(x) = \int_{-1}^{1}\psi^{(j+1)}(x,\mu)\,\mathrm{d}\mu\frac{\Phi_0}{\frac{1}{X}\int_0^X\int_{-1}^{1}\psi^{(j+1)}(x',\mu)\,\mathrm{d}\mu\,\mathrm{d}x'}, \tag{2.38a}$$

$$\boldsymbol{J}^{(j+1)}(x) = \int_{-1}^{1}\mu\psi^{(j+1)}(x,\mu)\,\mathrm{d}\mu\frac{\Phi_0}{\frac{1}{X}\int_0^X\int_{-1}^{1}\psi^{(j+1)}(x',\mu)\,\mathrm{d}\mu\,\mathrm{d}x'}, \tag{2.38b}$$

$$k_{\text{eff}}^{(j+1)} = \frac{\int_0^X \nu\Sigma_f\phi^{(j+1)}(x)\,\mathrm{d}x}{\int_0^X \Sigma_a\phi^{(j+1)}(x)\,\mathrm{d}x + \int_{-1}^{0}\psi^{(j+1)}(0,\mu)\,\mathrm{d}\mu + \int_0^1\psi^{(j+1)}(X,\mu)\,\mathrm{d}\mu}. \tag{2.38c}$$

Bibliography

[1] Ladislav Trlifaj. "Some Aspects of the Spherical Harmonics Method for Neutron-Transport Problems in Cylindrical Geometry". In: *Czechoslovak Journal of Physics* 8 (1958), pp. 390–395.

[2] D. A. Brown et al. "ENDF/B-VIII.0: The 8th Major Release of the Nuclear Reaction Data Library with CIELO-project Cross Sections, New Standards and Thermal Scattering Data". In: *Nuclear Data Sheets* 148 (2018), pp. 1–142. ISSN: 00903752. DOI: 10.1016/j.nds.2018.02.001.

[3] Dave Knott and Akio Yamamoto. "Lattice Physics Computations". In: *Handbook of Nuclear Engineering* (2010), pp. 913–1239. DOI: 10.1007/978-0-387-98149-9_9.

[4] Akio Yamamoto et al. "Derivation of optimum polar angle quadrature set for the method of characteristics based on approximation error for the bickley function". In: *Journal of Nuclear Science and Technology* 44.2 (2007), pp. 129–136. ISSN: 00223131. DOI: 10.1080/18811248.2007.9711266.

[5] E. M. Gelbard. *Simplified Spherical Harmonics equations and their use in shielding problems*. Tech. rep. Bettis Atomic Power Laboratory, 1961.

[6] Ryan G. Mcclarren. "Theoretical aspects of the simplified pn equations". In: *Transport Theory and Statistical Physics* 39.2-4 (2010), pp. 73–109. ISSN: 00411450. DOI: 10.1080/00411450.2010.535088.

[7] Edward W. Larsen. "Asymptotic diffusion and simplified pn approximations for diffusive and deep penetration problems. Part 1: Theory". In: *Transport Theory and Statistical Physics* 39.2-4 (2010), pp. 110–163. ISSN: 00411450. DOI: 10.1080/00411450.2010.531878.

[8] J.R. Askew. *A Characteristics Formulation of the Neutron Transport Equation in Complicated Geometries*. 1972.

[9] M.J. Halsall. "CACTUS, a characteristics solution to the neutron transport equations in complicated geometries". In: *Transactions of the American Nuclear Society*. 1980. URL: https://inis.iaea.org/search/search.aspx?orig{_}q=RN:12592444.

[10] Alain Hébert. “Multigroup Neutron Transport and Diffusion Computations”. In: *Handbook of Nuclear Engineering* (2010), pp. 751–911. DOI: 10.1007/978-0-387-98149-9_8.

[11] Ser Gi Hong and Nam Zin Cho. “Method of Characteristic Direction Probabilities for Heterogeneous Lattice Calculation”. In: *Nuclear Science and Engineering* 132.1 (2017), pp. 65–77. ISSN: 0029-5639. DOI: 10.13182/nse99-a2049.

[12] Zhouyu Liu et al. “Theory and analysis of accuracy for the method of characteristics direction probabilities with boundary averaging”. In: *Annals of Nuclear Energy* 77 (2015), pp. 212–222. ISSN: 0306-4549. DOI: 10.1016/J.ANUCENE.2014.11.016.

[13] Kord Smith and Joel D. Rhodes. “CASMO-4 Characteristic Methods for Two-Dimensional PWR and BWR Core Calculations”. In: *Transactions of the American Nuclear Society*. 2000.

[14] Rodolfo M. Ferrer and Joel D. Rhodes. “A Linear Source Approximation Scheme for the Method of Characteristics”. In: *Nuclear Science and Engineering* 182.2 (2016), pp. 151–165. ISSN: 0029-5639. DOI: 10.13182/NSE15-6.

[15] MPACT Team. *MPACT Theory Manual v2.2.0*. Tech. rep. Consortium for Advanced Simulation of Light Water Reactors, 2016.

[16] Benjamin Collins 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. ISSN: 10902716. DOI: 10.1016/j.jcp.2016.08.022.

[17] Han Gyu Joo et al. “Methods and performance of a three-dimensional whole-core transport code DeCART”. In: *PHYSOR 2004*. 2004, pp. 21–34. ISBN: 0894486837. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-22344449157{\&}partnerID=tZOtx3y1.

[18] Gil Soo Lee and Nam Zin Cho. “2D/1D fusion method solutions of the three-dimensional transport OECD benchmark problem C5G7 MOX”. In: *Progress in Nuclear Energy* 48.5 (2006), pp. 410–423. ISSN: 01491970. DOI: 10.1016/j.pnucene.2006.01.010.

[19] Richard Sanchez. “Prospects in deterministic three-dimensional whole-core transport calculations”. In: *Nuclear Engineering and Technology* 44.5 (2012), pp. 113–150. ISSN: 17385733. DOI: 10.5516/NET.01.2012.501.

[20] Brendan Matthew Kochunas. “A Hybrid Parallel Algorithm for the 3-D Method of Characteristics Solution of the Boltzmann Transport Equation on High Performance Compute Clusters”. Doctoral. University of Michigan, 2013, pp. 1–194.

[21] Geoffrey Alexander Gunow. “Full Core 3D Neutron Transport Simulation Using the Method of Characteristics with Linear Sources”. Doctoral. Massachusetts Institute of Technology, 2018.

[22] Daniele Sciannandrone, Simone Santandrea, and Richard Sanchez. "Optimized tracking strategies for step MOC calculations in extruded 3D axial geometries". In: *Annals of Nuclear Energy* 87 (2016), pp. 49–60. ISSN: 18732100. DOI: 10.1016/j.anucene.2015.05.014.

[23] Nam Zin Cho. "Fusion of method of characteristics and nodal method for 3-D whole-core transport calculation". In: *American Nuclear Society*. 2002, p. 322.

[24] Yeon Sang Jung and Han Gyu Joo. "Decoupled Planar MOC Solution for Dynamic Group Constant Generation in Direct Three-Dimensional Core". In: *M&C 2009* (2009).

[25] Youqi Zheng, Sooyoung Choi, and Deokjung Lee. "A new approach to three-dimensional neutron transport solution based on the method of characteristics and linear axial approximation". In: *Journal of Computational Physics* 350 (2017), pp. 25–44. ISSN: 10902716. DOI: 10.1016/j.jcp.2017.08.026.

[26] Bastien Faure et al. "A 2D/1D Algorithm for Effective Cross-Section Generation in Fast Reactor Neutronic Transport Calculations". In: *Nuclear Science and Engineering* 192.1 (2018), pp. 40–51. ISSN: 1943748X. DOI: 10.1080/00295639.2018.1480190.

[27] S.G. Stimpson. "An Azimuthal, Fourier Moment-Based Axial SN Solver for the 2D/1D Scheme". PhD thesis. University of Michigan, 2015.

[28] Michael Gregory Jarrett. "A 2D/1D Neutron Transport Method with Improved Angular Coupling". Doctoral. University of Michigan, 2018.

[29] Blake W. Kelley and Edward W. Larsen. "A consistent 2D/1D approximation to the 3D neutron transport equation". In: *Nuclear Engineering and Design* 295 (2015), pp. 598–614. ISSN: 00295493. DOI: 10.1016/j.nucengdes.2015.07.026.

[30] Chen Zhao et al. "Improved leakage splitting method for the 2D/1D transport calculation". In: *Progress in Nuclear Energy* 105.January (2018), pp. 202–210. ISSN: 01491970. DOI: 10.1016/j.pnucene.2018.01.007.

[31] Abel Marin-Lafleche, Micheal A. Smith, and Changho Lee. "Proteus-MOC: A 3D deterministic solver incorporating 2D method of characteristics". In: *M&C 2013*. Vol. 4. 2013, pp. 2759–2770. ISBN: 9781627486439. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-84883349083{\&}partnerID=tZOtx3y1.

[32] Nicholas Herring and Brendan Kochunas. *Personal Communications*. 2019.

[33] Akio Yamamoto et al. "GENESIS: A Three-Dimensional Heterogeneous Transport Solver Based on the Legendre Polynomial Expansion of Angular Flux Method". In: *Nuclear Science and Engineering* 49.6 (2017), pp. 1143–1156. ISSN: 2234358X. DOI: 10.1016/j.net.2017.06.016.

[34] Akinori Giho et al. “Development of Axially Simplified Method of Characteristics in Three-Dimensional Geometry”. In: *Journal of Nuclear Science and Technology* 45.10 (2008), pp. 985–996. ISSN: 0022-3131. DOI: 10.3327/jnst.45.985.

[35] Alain Hébert. “Acceleration of step and linear discontinuous schemes for the method of characteristics in DRAGON5”. In: *Nuclear Engineering and Technology* 49.6 (2017), pp. 1135–1142. ISSN: 2234358X. DOI: 10.1016/j.net.2017.07.004.

[36] K.S. Smith and J.D. Rhodes III. “Full-Core, 2-D, LWR Core Calculations with CASMO-4E”. In: *Physor* 1 (2002), pp. 1–13.

[37] Kord S. Smith. “Nodal method storage reduction by nonlinear iteration”. In: *Transactions of the American Nuclear Society*. Vol. 44. 1983, pp. 265–266.

[38] Dmitriy Y Anistratov. “Multi-level nonlinear diffusion acceleration method for multigroup transport k-eigenvalue problems”. In: *International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering, Rio de Janeiro, RJ, Brazil, May* 2 (2011), pp. 8–12.

[39] Ang Zhu 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. ISSN: 18732100. DOI: 10.1016/j.anucene.2016.05.004.

[40] Brendan Kochunas, Andrew Fitzgerald, and Edward 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. ISSN: 10902716. DOI: 10.1016/j.jcp.2017.05.028.

[41] Ben C. Yee, Brendan Kochunas, and Edward W. 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* 00.00 (2019), pp. 1–24. ISSN: 0029-5639. DOI: 10.1080/00295639.2018.1562777.

CHAPTER 3

The Method of Characteristics

3.1 Fundamentals

The Method of Characteristics (MoC) is a technique used to solve first-order PDEs, by transforming the PDE into a system of ODEs. The method was first applied to the neutron transport equation by Askew in 1972 [1], but only began to see real use in the 1980's [2]. The MoC transforms the transport equation into the characteristic form, by following the equation along straight neutron paths through the spatial domain. For brevity, the derivation of this method will begin with the multigroup S_N k-eigenvalue transport equation with spatially discretized mesh with constant material properties within each cell. Here, the spatial derivatives have not yet been discretized.

$$\left[\widehat{\boldsymbol{\Omega}}_m \cdot \boldsymbol{\nabla} + \Sigma_{t,i}^g\right] \psi_{mi}^g(\boldsymbol{x}) = \frac{1}{4\pi} q_{mi}^g(\boldsymbol{x}) \,, \tag{3.1}$$

$$\forall \boldsymbol{x} \in \mathcal{R}_i, \quad \forall m \in \mathcal{M}_N, \quad \forall i, g,$$

where $\mathcal{R}_i$ is the spatial cell, $\mathcal{M}_N$ is the directional quadrature, as described in Section 2.3.2.3, and the fixed-source, $q_{mi}^g(\boldsymbol{x})$ can be found by applying the discrete-to-moment operator, $\mathcal{S}_{i,m}^g$, defined by

$$\mathcal{S}_{i,m}^g(f) \equiv \sum_{g'} \sum_{\ell=0}^{L} \sum_{n=-\ell}^{\ell} R_\ell^n(\widehat{\boldsymbol{\Omega}}_m) \Sigma_{s,\ell,i}^{g' \to g} f_{n,i}^{\ell,g'}(\boldsymbol{x}) + \frac{\chi_i^g}{k_{\text{eff}}} \sum_{g'} \nu \Sigma_{f,i}^{g'} f_i^{g'}(\boldsymbol{x}), \tag{3.2}$$

to get

$$q_{mi}^g(\boldsymbol{x}) \equiv \left[\sum_{g'} \sum_{\ell=0}^{L} \sum_{n=-\ell}^{\ell} R_\ell^n(\widehat{\boldsymbol{\Omega}}) \Sigma_{s,\ell,i}^{g' \to g} \Phi_{i,n}^{\ell,g}(\boldsymbol{x}) + \frac{\chi_i^g}{k_{\text{eff}}} \sum_{g'} \nu \Sigma_{f,i}^{g'} \phi_i^{g'}(\boldsymbol{x}) \right], \tag{3.3}$$

where L is the maximum scattering order. If L were infinite, there would not any additional approximation to the scattering source; however, the first several orders have the most effect on the figures of merit, and in practice the sum is truncated with L typically being less than five.

Consider a point, $\boldsymbol{x}_0$, and a line passing through this point in direction $\widehat{\boldsymbol{\Omega}}_m$. Any location along

this *characteristic* line (also referred to as a ray, or track), can be described as

$$\boldsymbol{x} = \boldsymbol{x}_0 + s\widehat{\boldsymbol{\Omega}}_m, \tag{3.4}$$

where s is the distance along the track from $\boldsymbol{x}_0$. Applying this transformation, Eq. (3.1) is put into the characteristic form

$$\left[\frac{\mathrm{d}}{\mathrm{d}s} + \Sigma_{t,i}^g\right] \psi_{mi}^g\Big(\boldsymbol{x}_0 + s\widehat{\boldsymbol{\Omega}}_m\Big) = \frac{1}{4\pi} q_{mi}^g\Big(\boldsymbol{x}_0 + s\widehat{\boldsymbol{\Omega}}_m\Big)\,. \tag{3.5}$$

As stated in Section 2.1, reactor physicists are generally interested in spatially and directionally integrated angular flux quantities rather than the angular flux along a single path. Thus typical in the MoC to have many different characteristic tracks through our problem; in this work separate tracks will be subscripted with the index k. Each track is broken up into track-segments by considering the segments contained within each computational cell. The characteristic form of the transport equation then becomes

$$\left[\frac{\mathrm{d}}{\mathrm{d}s} + \Sigma_{t,i}^g\right] \psi_{mki}^g(s) = \frac{1}{4\pi} q_{mi}^g(s)\,, \tag{3.6}$$
$$\forall s \in [0, s_{mki}], \forall m \in \mathcal{M}_N, \forall i, k, g,$$

where s_{mki} is the total length of the track-segment, as depicted in Fig. 3.1.

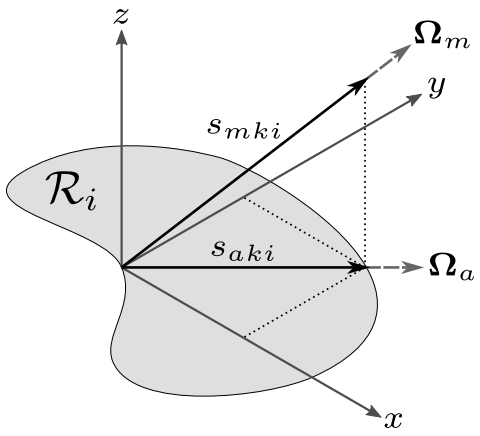


Figure 3.1: Depiction of a single characteristic track through a cell i.

Equation (3.6) can be solved analytically along a characteristic track-segment using an integrating factor,

$$M(s) = \exp\left(\int_0^s \Sigma_{t,i}^g \,\mathrm{d}s'\right) = \exp(\tau_m^g)\,, \tag{3.7}$$

where the *optical thickness*, τ_m^g, can be simplified as

$$\tau_m^g \equiv \Sigma_{t,i}^g s, \tag{3.8}$$

$$\forall s \in [0, s_{mki}],$$

assuming constant properties along the track-segment. Using this integrating factor, the generic solution to the MoC equation, given in Eq. (3.6), is

$$\psi_{mki}^g(s) = \psi_{mki}^{g,\text{in}} \exp(-\tau_m^g) + \int_0^s \frac{1}{4\pi} q_{mi}^g(s') \exp\left(-\Sigma_{t,i}^g \left[s - s'\right]\right) \mathrm{d}s', \tag{3.9}$$

where $\psi_{mki}^{g,\text{in}}$ is the incident angular flux, $\psi_{mki}^g(0)$. If a source shape is provided, Eq. (3.9) can be evaluated for every track-segment in the problem. The next subsection introduces formal methods to approximate the integration of quantities over both space and direction. These procedures can be used to determine the scalar flux or other quantities necessary in MoC calculations.

3.1.1 Track-Based Integration

Determining the angular flux along a single characteristic track is typically not the goal of reactor physics calculations. It is most often necessary to evaluate reaction rates, and therefore the scalar flux through integration of the angular flux. This section aims to provide a formal basis for the integration process used in the MoC for transport calculations.

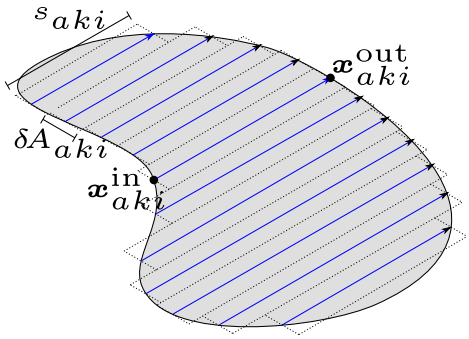


Figure 3.2: Example characteristic tracks (2D) through a cell for a single direction.

The MoC is based on the Discrete Ordinates (S_N) approximation; integration over the directional variable simply becomes a quadrature integration:

$$\int_{4\pi} f(\widehat{\boldsymbol{\Omega}}) \, \mathrm{d}\Omega \approx 4\pi \sum_m w_m f(\widehat{\boldsymbol{\Omega}}_m). \tag{3.10}$$

Within a cell, $\mathcal{R}_i$, there are many characteristic track-segments for each direction in the directional quadrature, as is shown for a single direction in Fig. 3.2. Thus, the spatial discretization is different for each direction, and spatial integration is linked with the directional integration. For a single direction, the integration over the spatial domain can be approximated by the weighted summation of track-averaged values, with the weight being equal to the area of the track-segment. The average value of a function, $f(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}_m)$, along a track-segment is denoted as

$$\Big\langle f(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}_m) \Big\rangle_{mki} \equiv \frac{1}{s_{mki}} \int_0^{s_{mki}} f(s, \widehat{\boldsymbol{\Omega}}_m)\, \mathrm{d}s, \tag{3.11}$$

where s_{mki} is the total length of the track-segment. The spatial integration for a single direction becomes

$$\frac{1}{V_i} \int_{\boldsymbol{x} \in \mathcal{R}_i} f(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}_m)\, \mathrm{d}^3\boldsymbol{x} \approx \Big\langle f(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}_m) \Big\rangle_{mi} \equiv \frac{1}{V_i} \sum_k \delta A_{mki} s_{mki} \Big\langle f(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}_m) \Big\rangle_{mki}, \tag{3.12}$$

where δA_{mki} is the cross-sectional area of the track (width in 2-D). In this notation, the integral is divided by the volume such that $\langle f \rangle_{mi}$ is approximately the mean value in the region, for the direction $\widehat{\boldsymbol{\Omega}}_m$. Finally, an integration over both space and angle can be defined as

$$\Big\langle f(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}) \Big\rangle_i = 4\pi \sum_m w_m \Big\langle f(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}_m) \Big\rangle_{mi}. \tag{3.13}$$

These integrations have been expressed as 3-D MoC equations. The general form remains the same for 2-D calculations, with minor changes. The spatial integration, Eq. (3.12), requires an additional scaling factor ($\sin(\theta_p)$), and the volume, V_i, is the area of the cell:

$$\frac{1}{V_i} \int_{\boldsymbol{x} \in \mathcal{R}_i} f(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}_m)\, \mathrm{d}^3\boldsymbol{x} \approx \Big\langle f(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}_m) \Big\rangle_{mi} \equiv \frac{\sin(\theta_p)}{V_i} \sum_k \delta A_{mki} s_{mki} \Big\langle f(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}_m) \Big\rangle_{mki} \tag{3.14}$$

The scaling factor is necessary (but not sufficient) for the integrated cell area to be preserved for each polar angle; if $\sin(\theta_p)$ is factored into the summation, s_{mki} becomes s_{aki} (the 2-D track-length).

3.1.2 Track-Length Renormalization

In general, the spatial integration described in Section 3.1.1 does not preserve the cell volume; this is visually apparent in Fig. 3.2. In order to preserve spatial volumes within a cell, track-lengths are often "renormalized". As the area of each ray, δA_{mki} approaches zero, the renormalization becomes irrelevant; thus it is only of consequence when rays have relatively coarse spacing between each other. There are three renormalization methods that become obvious through the notation presented

in Section 3.1.1:

1. segment-volume preservation
2. direction-volume preservation
3. volume preservation

Track-length renormalization involves adjusting the lengths of track-segments such that volume is preserved. Let us define a renormalization factor, ξ_{mki}, such that the renormalized track-length is given by

$$t_{mki} = \xi_{mki} s_{mki}. \tag{3.15}$$

The spatial integration schemes given by Eqs. (3.11) and (3.12) become

$$\left\langle f(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}_m) \right\rangle_{mki} \equiv \frac{1}{t_{mki}} \int_0^{t_{mki}} f(s, \widehat{\boldsymbol{\Omega}}_m) \, \mathrm{d}t_m, \tag{3.16a}$$

and

$$\left\langle f(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}_m) \right\rangle_{mi} \equiv \frac{1}{V_i} \sum_k \delta A_{mki} t_{mki} \left\langle f(\boldsymbol{x}, \widehat{\boldsymbol{\Omega}}_m) \right\rangle_{mki}, \tag{3.16b}$$

where the spatial variable $\boldsymbol{x}$ can now be written as a function of the renormalized track-distance, t_m, as

$$\boldsymbol{x} = \boldsymbol{x}^{\text{in}}_{mki} + t_m \widehat{\boldsymbol{\Omega}}_m / \xi_{mki}, \tag{3.16c}$$

where $\boldsymbol{x}^{\text{in}}_{mki}$ is the starting point of the track-segment.

Segment-volume preservation is a renormalization method in which the track-length is adjusted such that the analytic volume within the cross-sectional area of each track-segment is preserved. This renormalization technique is the most "correct" method of renormalization, but is very expensive as each track is renormalized separately. It is also more difficult to implement, as the analytic area of each track-segment must be found. To the best of our knowledge, this method is not implemented in any production-level MoC code.

Direction-volume preservation is the next "most-correct" renormalization technique. In this method, every mono-directional spatial integration should preserve the cell volume, i.e.

$$\langle 1 \rangle_{mi} = 1. \tag{3.17}$$

This constraint leads to the renormalization factor given by

$$\xi_{mi} = \frac{V_i}{\sum_k \delta A_{mki} s_{mki}}. \tag{3.18}$$

This method is significantly less expensive in terms of memory, computational time, and difficulty of implementation.

The simplest renormalization technique, volume preservation, only preserves the volume over the spatial and directional integration, i.e.

$$\langle 1 \rangle_i = 4\pi. \tag{3.19}$$

This constraint leads to the renormalization factor given by

$$\xi_i = \frac{V_i}{\sum_m w_m \sum_k \delta A_{mki} s_{mki}}. \tag{3.20}$$

Renormalization is not the only technique used for volume preservation. Another method is to use the numerical volume, $\sum_k \delta A_{mki} s_{mki}$ in place of V_i in Eq. (3.12). A detailed comparison of these different approaches has not taken place, to the best of our knowledge. The renormalization technique generally seems to be the faster approach, and is the approach used in MPACT [3], which is used extensively in this work.

3.2 The Flat-Source Approximation

The simplest approximation to the spatial shape of the source, $q^g_{mi}(\boldsymbol{x})$, within each cell is the flat-source approximation (FSA). The MoC has been widely used in lattice physics and neutron transport codes [4], many of which have utilized the flat-source method of characteristics (FSMoC) [2, 3, 5–10].

3.2.1 Derivation

The FSA is simply the assumption that within each cell, $\mathcal{R}_i$, the source, $q^g_{mi}(\boldsymbol{x})$, is uniform. This can be expressed as

$$q^g_{mi}(\boldsymbol{x}) \approx q^g_{mi} = q^g_i + \sum_{\ell=0}^{L} \sum_{n=-\ell}^{\ell} R^n_\ell(\widehat{\boldsymbol{\Omega}}_m) q^{g,n}_{i,\ell} \tag{3.21}$$

To obtain a source in this form, Eq. (3.3) requires that the region averaged scalar flux and higher-order angular moments (up to order L) be determined. In mathematical terms, the flat-source can be determined as

$$q^g_{mi} = \left[\sum_{g'} \sum_{\ell=0}^{L} \sum_{n=-\ell}^{\ell} R^n_\ell(\widehat{\boldsymbol{\Omega}}_m) \Sigma^{g'\to g}_{s,\ell,i} \Phi^{\ell,g'}_{i,n} + \frac{\chi^g_i}{k_{\text{eff}}} \sum_{g'} \nu \Sigma^{g'}_{f,i} \phi^{g'}_i \right], \tag{3.22}$$

where the $\phi_i^{g'}$ is the region-averaged scalar flux, and $\Phi_{i,n}^{\ell,g'}$ are the region-averaged angular moments of the flux.

In order to get these region-averaged flux moments, the spatial and directional integration operators, introduced in Section 3.1.1, are used. The region-averaged scalar flux is given by

$$\phi_i^g = \langle \psi^g \rangle_i = \frac{4\pi}{V_i} \sum_m w_m \sum_k t_{mki} \delta A_{mki} \langle \psi^g \rangle_{mki}, \tag{3.23a}$$

and the higher-order angular moments of the flux are given by

$$\Phi_{i,n}^{\ell,g} = \left\langle R_\ell^n(\widehat{\mathbf{\Omega}}) \psi^g \right\rangle_i = \frac{4\pi}{V_i} \sum_m w_m R_\ell^n(\widehat{\mathbf{\Omega}}_m) \sum_k t_{mki} \delta A_{mki} \langle \psi^g \rangle_{mki}. \tag{3.23b}$$

To evaluate these flux moments, the track-averaged angular flux, $\langle \psi^g \rangle_{mki}$, must be found. By applying the FSA, Eq. (3.6) becomes

$$\left[\frac{\mathrm{d}}{\mathrm{d}t_m} + \Sigma_{t,i}^g \right] \psi_{mki}^g(t_m) = \overline{q}_{mi}^g, \tag{3.24}$$

where

$$\overline{q}_{mi}^g \equiv \frac{1}{4\pi} q_{mi}^g. \tag{3.25}$$

This can be solved analytically for the angular flux along the track,

$$\psi_{mki}^g(t_m) = \psi_{mki}^{g,\text{in}} + \left(\frac{\overline{q}_{mi}^g}{\Sigma_{t,i}^g} - \psi_{mki}^{g,\text{in}} \right) F_1(\tau_m^g), \tag{3.26a}$$

where

$$F_1(\tau_m^g) \equiv 1 - \exp(-\tau_m^g), \tag{3.26b}$$

and τ_m^g is the (renormalized) optical thickness,

$$\tau_m^g \equiv t_m \Sigma_{t,i}^g. \tag{3.26c}$$

One approach to find $\langle \psi^g \rangle_{mki}$, is to perform integration of Eq. (3.26a) to evaluate the track-average value, resulting in

$$\langle \psi^g \rangle_{mki} = \frac{\overline{q}_{mi}^g}{\Sigma_{t,i}^g} - \left(\frac{\overline{q}_{mi}^g}{\Sigma_{t,i}^g} - \psi_{mki}^{g,\text{in}} \right) \frac{F_1(\tau_{mki}^g)}{\tau_{mki}^g}. \tag{3.27}$$

This approach will be referred to as *explicit*, as the moment's integral is explicitly evaluated. Another, approach, which in the author's opinion is simpler, is to use the track-averaging operator on the

characteristic form of the transport equation, Eq. (3.24), which simplifies to

$$\langle \psi^g \rangle_{mki} = \frac{\overline{q}^g_{mi}}{\Sigma^g_{t,i}} + \frac{\psi^{g,\text{in}}_{mki} - \psi^{g,\text{out}}_{mki}}{\tau^g_{mki}}. \tag{3.28}$$

This approach will be referred to as *implicit*, as the moment's integral is not explicitly evaluated. We note that the resulting forms of these two approaches are equivalent; by evaluating the outgoing flux in Eq. (3.26a) at the outgoing position, Eq. (3.28) can be put into the form of Eq. (3.27). By substituting the track-averaged angular flux in Eqs. (3.23), the flux moments can be evaluated, and used to compute the source. A transport calculation may be carried out using the source iteration algorithm defined by Algorithm 1.

3.2.2 Particle Conservation

The neutron transport equation, Eq. (2.1a), is a statement of particle balance within the defined phase-space. Previous works [11, 12] have examined the FSMoC with respect to *particle conservation*. Le Tellier and Hébert [11] defined necessary constraints on the directional quadrature and the characteristic tracks (trajectories) in order to ensure particle conservation for the anisotropic FSMoC. The constraints can be found by requiring

$$\frac{1}{4\pi}\left\langle R^n_\ell(\widehat{\mathbf{\Omega}}) q^g_{mi} \right\rangle_i = q^{g,n}_{i,\ell}. \tag{3.29}$$

Substituting Eq. (3.21) into Eq. (3.29), requires that

$$\sum_m w_m R^n_\ell(\widehat{\mathbf{\Omega}}_m) R^{n'}_{\ell'}(\widehat{\mathbf{\Omega}}_m) = \delta_{\ell\ell'}\delta_{nn'}, \tag{3.30a}$$

and

$$\sum_k t_{mki}\delta A_{mki} = V_i. \tag{3.30b}$$

Equation (3.30a) is a constraint on the directional quadrature, requiring orthogonality of the real spherical harmonics [11]. Equation (3.30b) requires that *at least* direction-dependent renormalization, Eq. (3.18), be used.

If the constraints on directional quadrature and characteristic tracks are satisfied, several simplifications to Eqs. (3.23) are possible.

$$\phi^g_i = \frac{q^g_i}{\Sigma^g_{t,i}} + \frac{4\pi}{V_i \Sigma^g_{t,i}} \sum_m w_m \sum_k \delta A_{mki} \Delta\psi^g_{mki}, \tag{3.31a}$$

$$\Phi_{i,n}^{\ell,g} = \frac{q_{i,\ell}^{g,n}}{\Sigma_{t,i}^{g}} + \frac{4\pi}{V_i \Sigma_{t,i}^{g}} \sum_m w_m R_\ell^n(\widehat{\mathbf{\Omega}}_m) \sum_k \delta A_{mki} \Delta\psi_{mki}^{g}, \tag{3.31b}$$

where

$$\Delta\psi_{mki}^{g} \equiv \psi_{mki}^{g,\text{in}} - \psi_{mki}^{g,\text{out}}. \tag{3.31c}$$

3.2.3 Isotropic Simplifications

While anisotropic scattering is necessary for accurate calculations, it is also common for isotropic source calculations to be performed. Typically, these account for anisotropic behavior by using the transport-corrected P_0 (TCP0) approximation [13]. While not as accurate as truly anisotropic calculations, use of an isotropic source results in significantly fewer calculations, and allows for additional simplifications to be made.

Equation (3.29) is now only of concern for the isotropic component of the source. This results in the following constraint,

$$\sum_m w_m \sum_k t_{mki} \delta A_{mki} = V_i, \tag{3.32}$$

which is equivalent to the direction-independent renormalization, given by Eq. (3.20). These isotropic calculations become significantly less expensive, as only the scalar flux needs to be computed.

3.2.4 Applications

The FSMoC has been utilized in many MoC production codes [2, 3, 5–10]. However, previous studies on the FSMoC have found that a fine mesh must be used to obtain accurate results, particularly in the presence of control rods or blades, strong absorber rods, gadolinia poisoned fuel rods [14], and large reflector regions (such as in critical experiments) [15]. As the number of mesh elements increase, so does the number of track-segments (on which the MoC computations are performed). This results in large run-times, and has motivated the development of linear-source approximations (LSAs) to the MoC, that are discussed in detail in Section 3.3.

3.3 The Linear-Source Approximation

3.3.1 Overview

The linear-source approximation (LSA), in the MoC, assumes the shape of the source along a characteristic track-segment is linear. There has long been motivation for the development of LSAs for the MoC, as previous work [16] indicated that a spatially linear source was able to achieve faster computational performance in S_N calculations. There have been many different variants of this approximation. The first instance of the LSA was the *gradient source approximation* introduced by Halsall [17]. This early linear-source method of characteristics (LSMoC) was based on the averaging of the angular flux gradient along tracks, and was implemented in the WIMS [17], and PEACH [18] MoC transport codes. These averaged gradients were then used as estimates to the gradient of the scalar flux, which were used to compute the source shape as spatially linear.

Petkov and Takeda devised a LSA that estimated the gradient of the scalar flux based on the P_1 approximation in the MARIKO code [14, 19]. In this approximation, the gradient of the scalar flux is computed from the neutron current, the total cross section, and the linearly anisotropic scattering matrix:

$$\boldsymbol{\nabla}\phi_i^g \approx -3\left(\Sigma_{t,i}^g \boldsymbol{J}_i^g - \sum_{g'} \Sigma_{s,1,i}^{g'\to g}\boldsymbol{J}_i^{g'}\right). \tag{3.33}$$

A similar approach, using the diffusion approximation to compute the scalar flux gradient, was used in the so called "quasi-linear" source implemented by Rabiti et al. [20]. In this approach, the $\Sigma_{s,1,i}^{g'\to g}$ matrix is diagonalized, turning the P_1 approximation into the diffusion approximation. Due to their basis on the P_1 and diffusion approximations, these early LSAs are inaccurate in situations where more transport-like effects are present. It can be shown, even in simple cases, that this approximation can be predict the opposite direction for the scalar flux gradient.

Santandrea and Sanchez [21] introduced the positive linear and nonlinear surface characteristics scheme, which constructed a linear source by interpolating between source values on the surfaces of cell regions. Various improvements have been made to this surface characteristics scheme for conservation [21], as well as coupling in APOLLO2 [22]. Le Tellier and Hébert [23] introduced a simplification to the linear characteristics scheme for conservation, by using a diamond-differencing scheme. This work was extended by Hébert [24], to include higher-order diamond difference schemes, as well as allowing for acceleration [25].

The most recent LSA examined in this work was introduced as a 2-D general high-order method for unstructured meshes by Masiello et al. [26]. The approximation uses track-based integration, defined in Section 3.1.1, in order to compute spatial moments of the angular flux. This LSA was shown to reduce memory and computation times in 3-D MoC calculations [27]. The general method

was simplified in the case of the isotropic and anisotropic linear source (LS) by Ferrer and Rhodes [15]; this also introduced the "LS-P0" method in which the isotropic source is spatially linear, but the anisotropic source components are spatially uniform within each cell. This LSA was also shown to be consistent with particle conservation, under certain constraints, and to be compatible with CMFD acceleration [12].

This thesis work has made extensive use of this LSA, and has made improvements upon the method. For this reason, in the following section the formulation prior to the work of this thesis is derived.

3.3.2 Derivation

The moment-based LSA assumes the shape of the source, $q^g_{mi}(\boldsymbol{x})$, is spatially linear within each cell, $\mathcal{R}_i$. This can be expressed as

$$q^g_{mi}(\boldsymbol{x}) \approx q^g_{mi} + \boldsymbol{x} \cdot \widehat{\underline{\boldsymbol{q}}}^g_{mi}, \tag{3.34a}$$

where $\widehat{\underline{\boldsymbol{q}}}^g_{mi}$ is a column vector of source spatial expansion coefficients,

$$\widehat{\underline{\boldsymbol{q}}}^g_{mi} \equiv \begin{bmatrix} \widehat{q}^g_{mi,x} \\ \widehat{q}^g_{mi,y} \\ \widehat{q}^g_{mi,z} \end{bmatrix}, \tag{3.34b}$$

and $\boldsymbol{x}$ is the position in *local* coordinates. A similar spatial expansion of the angular moments of the flux can be performed,

$$\phi^{\ell,g}_{i,n}(\boldsymbol{x}) = \overline{\phi}^{\ell,g}_{i,n} + \boldsymbol{x} \cdot \widehat{\underline{\boldsymbol{\phi}}}^{\ell,g}_{i,n}, \tag{3.35}$$

the source can then be expressed as

$$q^g_{mi}(\boldsymbol{x}) = \sum_{g'} \sum_{\ell=0}^{L} \sum_{n=-\ell}^{\ell} R^n_\ell(\widehat{\boldsymbol{\Omega}}_m) \Sigma^{g' \to g}_{s,\ell,i} \phi^{\ell,g'}_{i,n}(\boldsymbol{x}) + \frac{\chi^g_i}{k_{\text{eff}}} \sum_{g'} \nu \Sigma^{g'}_{f,i} \phi^{g'}_i(\boldsymbol{x}), \tag{3.36}$$

and the linear expansion coefficients are explicitly given by

$$\widehat{\underline{\boldsymbol{q}}}^g_{mi} = \sum_{g'} \sum_{\ell=0}^{L} \sum_{n=-\ell}^{\ell} R^n_\ell(\widehat{\boldsymbol{\Omega}}_m) \Sigma^{g' \to g}_{s,\ell,i} \widehat{\underline{\boldsymbol{\phi}}}^{\ell,g'}_{i,n} + \frac{\chi^g_i}{k_{\text{eff}}} \sum_{g'} \nu \Sigma^{g'}_{f,i} \widehat{\underline{\boldsymbol{\phi}}}^{g}_{g'}. \tag{3.37}$$

In the spatial moment-base LSA, it is convenient to define the spatially linear source (and flux) in terms of a cell-local coordinate system. We allow $\boldsymbol{X}$ to be the position variable in the global

coordinate system, the local coordinates are then defined as

$$\boldsymbol{x} = \boldsymbol{X} - \boldsymbol{X}^{\mathrm{c}}_{mi}, \tag{3.38}$$

where $\boldsymbol{X}^{\mathrm{c}}_{mi}$ is the numerical centroid of the cell i.

These numerical centroids can be defined as either direction-dependent, or direction-independent, which will have implications on particle conservation, as is discussed in Section 3.3.3. The direction-dependent centroids are defined by

$$\boldsymbol{X}^{\mathrm{c}}_{mi} \equiv \langle \boldsymbol{X} \rangle_{mi} = \frac{1}{V_i} \sum_k \delta A_{mki} t_{mki} \boldsymbol{X}^{\mathrm{c}}_{mki}, \tag{3.39}$$

where $\boldsymbol{X}^{\mathrm{c}}_{mki}$ is the global coordinate vector of the track-segment mid-point. Similarly, the direction-independent centroids are defined by

$$\boldsymbol{X}^{\mathrm{c}}_{i} \equiv \frac{1}{4\pi} \langle \boldsymbol{X} \rangle_{i} = \frac{1}{V_i} \sum_m w_m \sum_k \delta A_{mki} t_{mki} \boldsymbol{X}^{\mathrm{c}}_{mki}. \tag{3.40}$$

Following the same approach as the FSMoC derivation, in Section 3.2.1, computing the source requires the region-averaged flux moments, $\overline{\phi}^{\ell,g}_{i,n}$, and the flux expansion coefficients, $\underline{\widehat{\phi}}^{\ell,g}_{i,n}$. The region-averaged flux moment can be found using the same definition as previously,

$$\overline{\phi}^{\ell,g}_{i,n} \equiv \left\langle R^n_\ell(\widehat{\boldsymbol{\Omega}}) \psi^g \right\rangle_i = \frac{4\pi}{V_i} \sum_m w_m R^n_\ell(\widehat{\boldsymbol{\Omega}}_m) \sum_k \delta A_{mki} t_{mki} \langle \psi^g \rangle_{mki}. \tag{3.41a}$$

In order to determine the spatial expansion coefficients of the flux moments, Eq. (3.35) is operated on by $\left\langle R^n_\ell(\widehat{\boldsymbol{\Omega}}) \boldsymbol{x}(\cdot) \right\rangle_i$. Recognizing that this should be directly proportional to angular flux operated on by $\left\langle R^n_\ell(\widehat{\boldsymbol{\Omega}}) \boldsymbol{x} \psi^g \right\rangle_i$, a system of equations is found

$$\boldsymbol{M}_i \underline{\widehat{\boldsymbol{\phi}}}^{\ell,g}_{i,n} = \left\langle R^n_\ell(\widehat{\boldsymbol{\Omega}}) \boldsymbol{x} \psi^g \right\rangle_i, \tag{3.41b}$$

where

$$\boldsymbol{M}_i \equiv \left\langle \boldsymbol{x}^T \boldsymbol{x} \right\rangle_i. \tag{3.41c}$$

The spatial angular flux moments, $\left\langle R^n_\ell(\widehat{\boldsymbol{\Omega}}) \boldsymbol{x} \psi^g \right\rangle_i$, are then defined as

$$\left\langle R^n_\ell(\widehat{\boldsymbol{\Omega}}) \boldsymbol{x} \psi^g \right\rangle_i = \frac{4\pi}{V_i} \sum_m w_m R^n_\ell(\widehat{\boldsymbol{\Omega}}_m) \sum_k \delta A_{mki} t_{mki} \left(\boldsymbol{x}^{\mathrm{in}}_{mki} \langle \psi^g \rangle_{mki} + \widehat{\boldsymbol{\Omega}}_m \langle t_m \psi^g \rangle_{mki} / \xi_{mi} \right). \tag{3.41d}$$

In order to evaluate the flux moments defined in Eqs. (3.41), the track-averaged angular flux

values, $\langle \psi^g \rangle_{mki}$, and $\langle t_m \psi^g \rangle_{mki}$, must be determined. First, the transport equation must be put into characteristic form, using Eq. (3.16c) the spatially expanded source, Eq. (3.34a), can be defined along the characteristic. The characteristic transport equation becomes

$$\left[\frac{\mathrm{d}}{\mathrm{d}t_m} + \Sigma_{t,i}^g\right] \psi_{mki}^g(s) = \overline{q}_{mki}^g + \widehat{q}_{mi}^g \left(t_m - \frac{t_{mki}}{2}\right), \tag{3.42a}$$

where

$$\overline{q}_{mki}^g \equiv \frac{1}{4\pi}\left[q_{mi}^g + \boldsymbol{x}_{mki}^{\mathrm{c}} \cdot \underline{\widehat{\boldsymbol{q}}}_{mi}^g\right], \tag{3.42b}$$

$$\widehat{q}_{mi}^g \equiv \frac{1}{4\pi}\left[\frac{\widehat{\boldsymbol{\Omega}}_m \cdot \underline{\widehat{\boldsymbol{q}}}_{mi}^g}{\xi_{mi}}\right], \tag{3.42c}$$

and $\boldsymbol{x}_{mki}^{\mathrm{c}}$ is the local-coordinate centroid of the track-segment. Substituting this assumed source shape (linear) into the generic MoC solution, given by Eq. (3.9), the angular flux along a track-segment is found to be

$$\psi_{mki}^g(s) = \psi_{mki}^{g,\mathrm{in}} + \left(\frac{\overline{q}_{mki}^g}{\Sigma_{t,i}^g} - \psi_{mki}^{g,\mathrm{in}}\right) F_1(\tau_m^g) + \frac{\widehat{q}_{mi}^g}{2(\Sigma_{t,i}^g)^2} F_2(\tau_m^g), \tag{3.43a}$$

where

$$F_1(\tau_m^g) \equiv 1 - \exp(-\tau_m^g), \tag{3.43b}$$

and

$$F_2(\tau_m^g) \equiv 2[\tau_m^g - F_1(\tau_m^g)] - \tau_{mki}^g F_1(\tau_m^g). \tag{3.43c}$$

As discussed in Section 3.2.1, there are two *equivalent* methods with which one could determine the track-averaged angular flux values. The original derivation of the LSA method by Ferrer and Rhodes [15] used the *implicit* definition for the track-average angular flux, but the *explicit* definition for the first spatial moment of the angular flux. In Chapter 5, the implicit definition is taken for the first spatial moment of the angular flux as well, which allows for additional improvements for the method in multi-physics and 2D/1D applications. For the remainder of this section, the formulation as it was originally derived by Ferrer and Rhodes [15] is shown, using the explicit form of $\langle t_m \psi^g \rangle_{mki}$.

The implicitly defined track-average flux is given by operating on Eq. (3.42a) by $\langle (\cdot) \rangle_{mki}$, and the explicitly defined first spatial moment of the angular flux is given by operating on Eq. (3.43a) by $\langle t_m (\cdot) \rangle_{mki}$. These are shown in Eqs. (3.44).

$$\langle \psi^g \rangle_{mki} = \frac{\overline{q}_{mki}^g}{\Sigma_{t,i}^g} + \frac{\Delta\psi_{mki}^g}{\tau_{mki}^g}, \tag{3.44a}$$

and

$$\langle t_m \psi^g \rangle_{mki} = \psi^{g,\text{in}}_{mki} \frac{t_{mki}}{2} + \left(\frac{\overline{q}^g_{mki}}{\Sigma^g_{t,i}} - \psi^{g,\text{in}}_{mki} \right) \frac{G_1(\tau^g_{mki})}{\Sigma^g_{t,i}} + \frac{\widehat{q}^g_{mi}}{2(\Sigma^g_{t,i})^2} t_{mki} G_2(\tau^g_{mki}). \tag{3.44b}$$

Here

$$G_1(\tau^g_{mki}) \equiv 1 + \frac{\tau^g_{mki}}{2} - \left(1 + \frac{1}{\tau^g_{mki}} \right) F_1(\tau^g_{mki}), \tag{3.45a}$$

and

$$G_2(\tau^g_{mki}) \equiv \frac{2}{3} \tau^g_{mki} - \left(1 + \frac{2}{\tau^g_{mki}} \right) G_1(\tau^g_{mki}). \tag{3.45b}$$

The original derivation further simplified Eqs. (3.41) into the following forms.

$$\overline{\phi}^{\ell,g}_{i,n} = \frac{4\pi}{\Sigma^g_{t,i} V_i} \sum_m w_m R^n_\ell(\widehat{\boldsymbol{\Omega}}_m) \Sigma^g_{t,i} \Psi^g_{mi}, \tag{3.46a}$$

$$\left\langle \boldsymbol{x} R^n_\ell(\widehat{\boldsymbol{\Omega}}) \psi^g \right\rangle_i = \frac{4\pi}{\Sigma^g_{t,i} V_i} \sum_m w_m R^n_\ell(\widehat{\boldsymbol{\Omega}}_m) \Sigma^g_{t,i} \left(\boldsymbol{\Psi}^g_{mi} + \widehat{\boldsymbol{\Omega}}_m \widehat{\Psi}^g_{mi} / \xi_i \right), \tag{3.46b}$$

where

$$\Psi^g_{mi} \equiv \frac{1}{\Sigma^g_{t,i}} \left[\frac{q^g_{mi}}{4\pi} \sum_k \delta A_{mki} t_{mki} + \frac{\widehat{\boldsymbol{q}}^g_{mi}}{4\pi} \cdot \sum_k \delta A_{mki} \boldsymbol{x}^{\text{c}}_{mki} t_{mki} + \sum_k \delta A_{mki} \Delta \psi^g_{mki} \right], \tag{3.47a}$$

$$\begin{aligned} \boldsymbol{\Psi}^g_{mi} \equiv \frac{1}{\Sigma^g_{t,i}} \Bigg[& \left(\sum_k \delta A_{mki} t_{mki} \boldsymbol{x}^{\text{in}}_{mki} \right) \frac{q^g_{mi}}{4\pi} + \left(\sum_k \delta A_{mki} t_{mki} \boldsymbol{x}^{\text{in}}_{mki} (\boldsymbol{x}^{\text{c}}_{mki})^T \right) \frac{\widehat{\boldsymbol{q}}^g_{mi}}{4\pi} \\ & + \sum_k \delta A_{mki} \boldsymbol{x}^{\text{in}}_{mki} \Delta \psi^g_{mki} \Bigg], \end{aligned} \tag{3.47b}$$

and

$$\widehat{\Psi}^g_{mi} \equiv \frac{1}{\Sigma^g_{t,i}} \left[\frac{q^g_{mi}}{4\pi} C^g_{mi} + \frac{\widehat{\boldsymbol{q}}^g_{mi}}{4\pi} \cdot \boldsymbol{C}^g_{mi} + \sum_k \delta A_{mki} t_{mki} \psi^{g,\text{in}}_{mki} H(\tau^g_{mki}) \right], \tag{3.47c}$$

where

$$C^g_{mi} \equiv \frac{1}{\Sigma^g_{t,i}} \sum_k \delta A_{mki} t_{mki} G_1(\tau^g_{mki}), \tag{3.48a}$$

$$\boldsymbol{C}^g_{mi} \equiv \frac{1}{\Sigma^g_{t,i}} \sum_k \delta A_{mki} t_{mki} \left(\boldsymbol{x}^{\text{c}}_{mki} G_1(\tau^g_{mki}) + \widehat{\boldsymbol{\Omega}}_m \frac{s_{mki}}{2} G_2(\tau^g_{mki}) \right), \tag{3.48b}$$

and

$$H(\tau^g_{mki}) \equiv \frac{\tau^g_{mki}}{2} - G_1(\tau^g_{mki}). \tag{3.49}$$

The C^g_{mi} and $\boldsymbol{C}^g_{mi}$ are dependent on both the energy group, g, and direction m, for each region, i.

If these terms are stored, this leads to considerable memory usage, and will often use more memory than the flux moments that are of interest. If cross sections are constant in the problem (no feedback, or 2D/1D transverse leakage splitting), these coefficients can be pre-computed once at the outset of the simulation. If this is not the case, these coefficients must be re-evaluated each time there is a change in cross sections. The re-evaluation of these coefficients can lead to significant overhead. In cases with T/H feedback, cross sections typically change each iteration.

3.3.3 Particle Conservation

When considering particle conservation, use of the LSA results in additional constraints on the calculations. Similarly to Section 3.2.2, the track-based integration of the source must exactly integrate to the spatial and angular moments of the source. The conservation of spatial moments is the basis of this LSA [12], so this constraint is satisfied without additional constraints on the method. The angular moment constraint is expressed as

$$\frac{1}{4\pi}\left\langle R_\ell^n(\widehat{\boldsymbol{\Omega}})q_{mi}^g(\boldsymbol{x})\right\rangle_i = q_{i,\ell}^{g,n}. \tag{3.50}$$

In addition to the constraints introduced in Section 3.2.2, namely direction-dependent renormalization, and directional quadrature restrictions, there is a constraint on the definition of the local coordinate system:

$$\langle \boldsymbol{x}\rangle_{mi} = 0. \tag{3.51}$$

This is equivalent to stating that the local coordinate system must be defined with respect to direction-dependent global centroids, as is given by Eq. (3.39).

3.3.4 Isotropic Simplifications

Ferrer and Rhodes [15] suggested that allowing only the flat source components to consider anisotropic scattering has performance benefits, while not significantly affecting accuracy. It was demonstrated for the Babcox and Wilcox (B&W) experiments [28] that considering anisotropic scattering only with spatially flat flux moments resulted in approximately 10 pcm error. Furthermore, by making this simplification, run-times were reduced significantly (up to 45%), while memory savings were even more significant (up to 89%) [15].

As stated in Section 3.2.3, it is very common in reactor simulations to use TCP0 cross-sections (which are isotropic). Assuming isotropic scattering, Eqs. (3.46) become

$$\phi_i^g = \frac{q_i^g}{\Sigma_{t,i}^g} + \frac{4\pi}{\Sigma_{t,i}^g V_i}\sum_m w_m \sum_k \delta A_{mki}\Delta\psi_{mki}^g, \tag{3.52a}$$

$$\langle \boldsymbol{x}\psi^g \rangle_i = \boldsymbol{C}_i^g \frac{\widehat{\boldsymbol{q}}_i^g}{\Sigma_{t,i}^g} + \frac{4\pi}{\Sigma_{t,i}^g V_i} \sum_m w_m \sum_k \delta A_{mki} \left[\boldsymbol{x}_{mki}^{\text{in}} \Delta\psi_{mki}^g + \widehat{\boldsymbol{\Omega}}_m s_{mki} \psi_{mki}^{g,\text{in}} H(\tau_{mki}^g) \right], \quad (3.52b)$$

where

$$\boldsymbol{C}_i^g \equiv \frac{1}{\Sigma_{t,i}^g V_i} \sum_{m/2} w_m \sum_k \delta A_{mki} \widehat{\boldsymbol{\Omega}}_m \widehat{\boldsymbol{\Omega}}_m^T s_{mki}^2 G_2(\tau_{mki}^g) + \frac{2}{V_i} \sum_{m/2} wt \sum_k \delta A_{mki} \boldsymbol{x}\boldsymbol{x}^T t_{mki}. \quad (3.53)$$

These $\boldsymbol{C}_i^g$ coefficients are no longer dependent on the direction, but still require significant memory usage. They require more memory than the scalar flux coefficients. If cross sections are not constant through the simulation, the previously mentioned inefficiencies are still present.

3.3.5 Applications

Various different LSAs to the MoC have been developed and implemented in transport codes [10, 14, 15, 17, 18, 20, 21, 29]. Results have indicated that by using a LSA, the spatial mesh discretization can be made coarser, relative to the FSA, while maintaining transport accuracy. Although each segment calculation is more expensive when using a LSA, the number of calculations (due to the coarser spatial mesh) can be significantly reduced, leading to reduced run-times. Additionally, the reduction in spatial mesh elements generally reduces the amount of memory required by the calculation.

3.4 Ray-Tracing

The Method of Characteristics (MoC) [1] is based on solving the transport equation along many characteristic tracks or rays. These rays are followed through the reactor geometry in a process generally referred to as "ray-tracing". The placement and storage of these tracks is significant with respect to both calculation accuracy as well as computational performance. This section serves to give an overview of the current state-of-the-art ray-tracing methods used for MoC-like transport calculations.

3.4.1 Fundamentals

This section covers the fundamentals of why ray-tracing is necessary for MoC calculations, what data is necessary, and possible constraints. Unless a neutron has an interaction with surrounding media, it will travel in a straight line. This idea is represented in the characteristic form of the transport equation (Eq. (3.6)), which follows a generic characteristic through the problem domain.

The MoC is based on solving the Eq. (3.6) along many of these characteristic paths, performing spatial/directional integrations using Eqs. (3.16).

For a MoC iteration, it is necessary to compute the integrated quantities of the angular flux (Eqs. 3.31 and 3.46). This requires that the angular flux along a track-segment, the path-segment of a characteristic path bounded within a spatial cell, is known. Determining the angular flux along a track-segment (Eq. (3.9)) can be done only if the incident angular flux is known, and the length of the track-segment is known. Generally, characteristic paths will not be constructed independently for each spatial cell because then the incident flux on the surface would need to be approximated. Thus, a long characteristic path (ray) is generated to pass through the domain (or sub-system of the domain), and the lengths of each segment (bounded by spatial cells) are recorded. This way, given the incident flux at one end of the characteristic ray, the ray can be followed and the angular flux can be determined for each segment of the ray. Because the incident flux of one segment is the outgoing flux of the previous segment, the segments must be "swept" in order.

3.4.2 Modular Ray-Tracing

The most straight-forward approach to perform a MoC calculation is to create rays that span the domain of the transport problem being solved. However, in this approach, the information of each ray-segment must be stored, or computed on-the-fly. In large problems the number of ray-segments can become exceedingly large, and this approach is not feasible due to memory constraints.

This led to the development of so-called "modular" ray-tracing methods [6, 30–32], in which the regularity of reactor designs is utilized to reduce memory requirements. In typical reactor designs, certain geometries (like assemblies) are repeated throughout the core. Rather than laying tracks down for the global geometry, the transport problem is partitioned into "modules" that represent small often-repeated geometries in the problem. The ray-tracing data is generated for each module, in such a way there is direct linking of tracks on module interfaces; this is the direct neutron path linking (DNPL) technique devised by Kosaka and Saji [6]. This significantly reduces the amount of ray-tracing data that needs to be stored in MoC calculations, and has been widely adopted in MoC transport codes [5, 18, 33–37]. Tracks spanning the global domain are then constructed by connecting multiple modular rays, as is depicted in Fig. 3.3.

The modular ray-tracing technique, with DNPL, requires that the number of tracks on a modular boundary be an integer. This additionally requires that all modules have the same spatial dimensions, given by the pitches P_x and P_y, and that the spacing between all tracks in a direction are constant. Let δA_{a0} be the desired ray-spacing for an azimuthal angle a, and φ_{a0} be the desired azimuthal

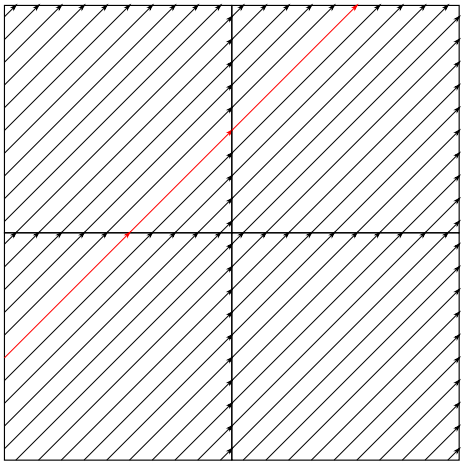


Figure 3.3: Depiction of modular ray-tracing method. Global (long) rays can be constructed by connecting multiple modular rays, as is shown in red.

angle. The number of rays on the x and y module boundaries can be determined as

$$N_x = \left\lceil \frac{P_x \sin(\varphi_{a0})}{\delta A_{a0}} \right\rceil, \tag{3.54a}$$

and

$$N_y = \left\lceil \frac{P_x \cos(\varphi_{a0})}{\delta A_{a0}} \right\rceil. \tag{3.54b}$$

The x and y distance between rays is then determined by

$$\delta_x = \frac{P_x}{N_x}, \tag{3.55a}$$

and

$$\delta_y = \frac{P_y}{N_y}. \tag{3.55b}$$

The azimuthal angle is then "corrected" to represent the true angle at which the rays are placed,

$$\varphi_a = \tan^{-1}\left(\frac{\delta_y}{\delta_x}\right), \tag{3.56}$$

and the corrected ray-spacing is then given by

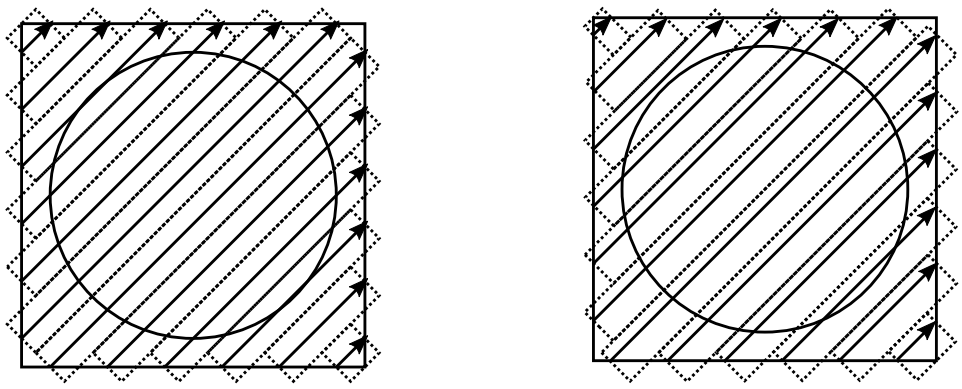
$$\delta A_a = \delta_x \sin(\varphi_a). \tag{3.57}$$

Another approach that has been taken in MPACT is to use a rational fraction to approximate $\tan(\frac{\delta_y}{\delta_x})$.

Due to the constraint of DNPL, the directional quadrature is perturbed in the process of ray-tracing. As described in Section 3.2.2, this has implications on particle conservation in calculations. In order to maintain accuracy, it may become necessary to use a higher-order directional quadrature than would have been necessary if the directions were not perturbed by modular ray-tracing (MRT).

3.4.3 Mobile Chords

The mobile chord method was introduced by Villarino et al. [38] in the HELIOS code for CP calculations, and adapted to the MoC by Yamamoto [39]. In the typical equidistant ray-tracing method, a ray is placed at the center of the ray-width. The mobile chord method offsets the ray from the center, with differing offsets in each direction. The offset is typically a fraction of the ray width that increases linearly with the azimuthal angle index. This has generally shown to be more accurate than the typical equidistant ray-tracing method [39], but is not directly compatible with the DNPL technique. While ray widths are still linked, the ray-traces are not; though, this does not seem to introduce significant discretization errors [39]. An example illustration of mobile chords is given in Fig. 3.4.



(a) Centered Chords (b) Mobile Chords

Figure 3.4: Example illustration of centered chords (traditional approach) and mobile chords for a single direction.

3.4.4 Random Rays

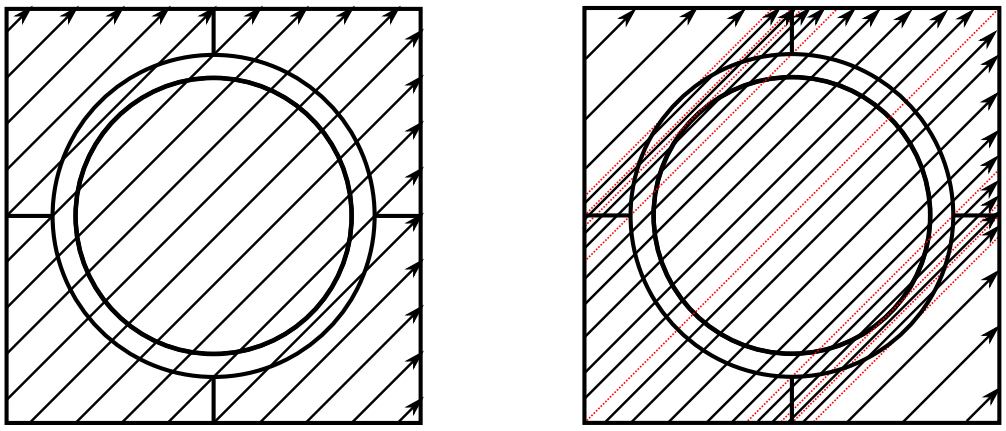
The random ray method was developed in the ARRC code by Tramm et al. [40], and is a hybrid method between Monte Carlo and MoC methods. This ray-tracing method uses stochastic discretizations of the directions and space (characteristic rays). The method uses a unique directional

quadrature and ray-trace each iteration; this significantly decreases the memory usage, as tracking storage is completely eliminated. The use of random rays on boundaries also allows for the storage of boundary/interface flux to be eliminated entirely; this is due to the consideration of cyclic-tracks, and cancellation of errors due to stochastic ray placement [40].

This also significantly increases the amount of work that needs to be done for each track, as intersection data needs to be recomputed for each track each iteration. There are claims that the cache benefits of not storing the rays out-weighs the increased computational work [40]. Initial results for this method have been promising, but no CMFD acceleration for this method has been utilized at this time.

3.4.5 Macroband

The *macroband* method was originally proposed by Villarino et al. [38] for CP calculations in HELIOS. In this method, characteristic rays placed within "macrobands" are separated by tangential and intersection points in the mesh. There is no material or geometric discontinuities within each macroband segment (macrosegment), thus the direction-of-flight averaged angular flux in each macrosegment is smooth with regards to the transverse direction. Since integration in the MoC is akin to a quadrature integration, this indicates that a more advanced quadrature, for ray placement and width, can be used to reduce discretization error [41].



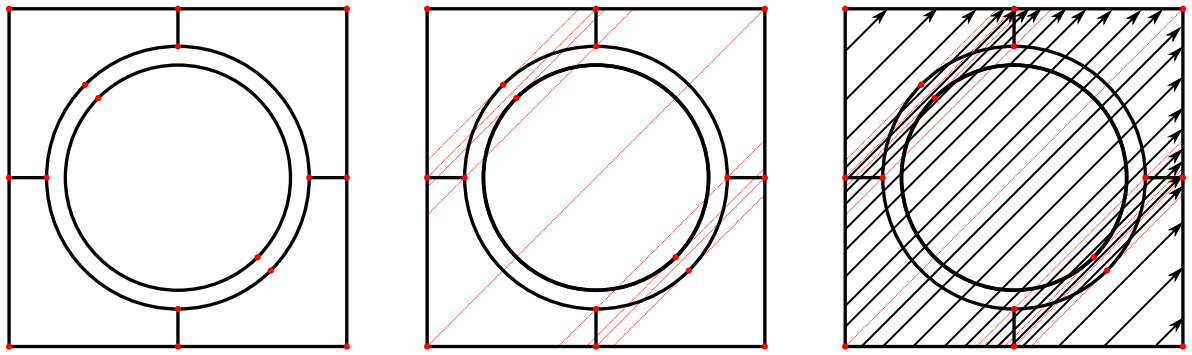
(a) Equidistant (b) Macroband

Figure 3.5: Visualization of hypothetical tracks for (a) equidistant and (b) macroband ray-tracing methods. Boundaries between macrobands are shown as red dotted lines.

The MRT ray-tracing technique does not consider the internal geometry when laying down tracks. For a specified ray-spacing and module size, the tracks are the same regardless of the internal

mesh. The macroband takes an entirely different approach, and the internal mesh is the primary guide for laying down the tracks.

Macrobands are determined by the computational mesh; for large heterogeneous assemblies, this results in very thin macrobands, which would result in significantly increased computation time. In the related method of Characteristic Direction Probabilities (CDP) Hong and Cho [42] proposed that macroband ray-tracing data only be generated on unique subsystems. Similarly, Yamamoto et al. [41] proposed the memory reduction technique for macroband (MRMB) in which macroband ray-tracing data is only generated for unit-cells. These techniques are similar to the modular ray-tracing technique in that ray-tracing data is only generated for unique subsystems, which significantly reduces the amount of ray-tracing data. The macroband ray-tracing process is displayed for a single pin-cell mesh in Fig. 3.6.



(a) Determine intersection and tangent points

(b) Determine macroband boundaries (through identified points)

(c) Perform ray-tracing within macroband boundaries

Figure 3.6: Ray-tracing process for macroband.

However, these techniques are fundamentally incompatible with the DNPL technique, thus an approximation of angular flux must be made on subsystem interfaces. Yamamoto et al. [41] proposed linearly interpolating the angular flux on cell boundaries, though other techniques involving averaging the angular flux on sub-boundaries have been utilized in the CDP [43]. Additionally, these methods do not require adjustment to the angular quadrature.

Yamamoto et al. [41] found that the macroband method (using a Gauss-Legendre quadrature for ray placement), was more accurate than conventional ray-tracing methods with equidistant ray-spacing. Févotte et al. [44] proposed a new tracking technique similar to the macroband method, in which rays (placed equidistantly) are divided into sub-bands, which are effectively *locally* projected macrobands, and the average flux of the sub-bands is propagated along each ray. Studies of ray-spacing with macroband, and Févotte et al.'s [44] method, have indicated that coarser ray-spacing can be used while maintaining accuracy [39, 41, 44].

3.4.5.1 Interface Flux Approximations

The macroband ray-tracing method is fundamentally incompatible with direct neutron path linking (DNPL); therefore, it requires an approximation at the subsystem interfaces. Although ray-tracing can be performed on small unique subsystems, similarly to the MRT method, the rays on module interfaces are no longer guaranteed to align. This makes it necessary for an approximation of the spatial dependence of the angular flux on these interfaces. However, there are multiple different options for performing this approximation.

Yamamoto et al. [41] proposed performing a linear of the flux on these interfaces. However, without adjustment, linear interpolation will not preserve the total flux passing through the surface; which is an important aspect when considering acceleration techniques such as CMFD. For 3-D CDP with uniformly spaced rays, Liu et al. [43] proposed a sub-boundary averaging method. This method partitioned each interface into sub-boundaries, with the size depending on the interface and the direction; each sub-boundary represented a single angular flux value for each energy group, that was the average of the angular flux of all rays with centroids that intersected the sub-boundary. However, because only the centroids of each ray are considered using ray-spacing that is coarse compared to the sub-boundary size it not feasible.

An earlier work [42] used a similar 2-D sub-boundary averaging (though not directionally dependent); but rather than only considering the ray centroids, it considered the "projection" of rays. The "projection" being the area on the surface which would be intersected by the ray if it were infinite in length. Then, partial intersections of the projection and sub-boundaries are considered. This allows for coarse ray-spacing to be used without as much accuracy on the interfaces as the previously mentioned method.

Figure 3.7 visually depicts the surface projections of 2-D rays on a single surface. This figure will be used as an example for the equations of this sub-boundary averaging method. ψ_s^1 should be a linear combination of the ray fluxes ψ_r^1 and ψ_r^2, based on their fractional areas, A_1^1 and A_2^1, respectively. Let us denote a sub-boundary as S_i where i is some index, and a ray projection as R_j where j is some index. We expect the surface flux to be a linear combination of the ray fluxes that intersect it; this can be expressed mathematically as

$$\psi_s^i = \sum_j \frac{A(S_i \cap R_j)}{A(S_i)} \psi_r^j, \tag{3.58}$$

where A denotes a function returning the area of the argument, and $\cap$ indicates the intersection of

two objects. To show that this preserves the leakage through the surface, consider

$$\begin{aligned} L &= \widehat{\boldsymbol{\Omega}} \cdot \hat{\boldsymbol{n}} \sum_i A(S_i)\psi_s^i \\ &= \widehat{\boldsymbol{\Omega}} \cdot \hat{\boldsymbol{n}} \sum_i A(S_i) \sum_j \frac{A(S_i \cap R_j)}{A(S_i)} \psi_r^j \\ &= \widehat{\boldsymbol{\Omega}} \cdot \hat{\boldsymbol{n}} \sum_i \sum_j A(S_i \cap R_j)\psi_r^j \\ &= \widehat{\boldsymbol{\Omega}} \cdot \hat{\boldsymbol{n}} \sum_j A(R_j)\psi_r^j, \end{aligned} \tag{3.59}$$

where this last line is simply the summation of all ray fluxes. The reverse problem, computing the ray fluxes from the sub-boundary fluxes, is found to be

$$\psi_r^j = \sum_i \frac{A(S_i \cap R_j)}{A(R_j)} \psi_s^i. \tag{3.60}$$

Figure 3.7 also highlights a problem with only considering the centroids of each ray. Ray 2 is split nearly evenly between sub-boundaries 1 and 2. If only the centroids are considered, all ray 2's flux would be put into sub-boundary 2. If the flux is relatively uniform along the surface direction, this is fine; however, if there are discontinuities in the flux then this can lead to considerable inaccuracies.

Liu et al. [43] introduced a useful concept to sub-boundary averaging, with the direction-dependence in the number of sub-boundaries on each surface. Rather than using a fixed number of sub-boundaries on each surface, it is expected that directions that are closer to being perpendicular to the surface will have larger variation, and thus need more sub-boundaries. The number of sub-boundaries on each surface can be computed using the dimensions of the surface, and the direction. Similarly to MRT, the number of sub-boundaries on a surface is constrained to be an integer. The MRT equations, Eqs. (3.54), can be used to determine the number of sub-boundaries on the x and y surfaces.

3.4.6 Three-Dimensional Ray-Tracing Techniques

The MoC is naturally extended to three-dimensional calculations along characteristic tracks spanning three-dimensions. However, 3-D MoC presents significant computational challenges. This has led to significant research effort into developing more efficient approaches to three-dimensional MoC [32, 45–48]. One of the main focuses of this research has been in reducing the complexity introduced by three-dimensional ray-tracing.

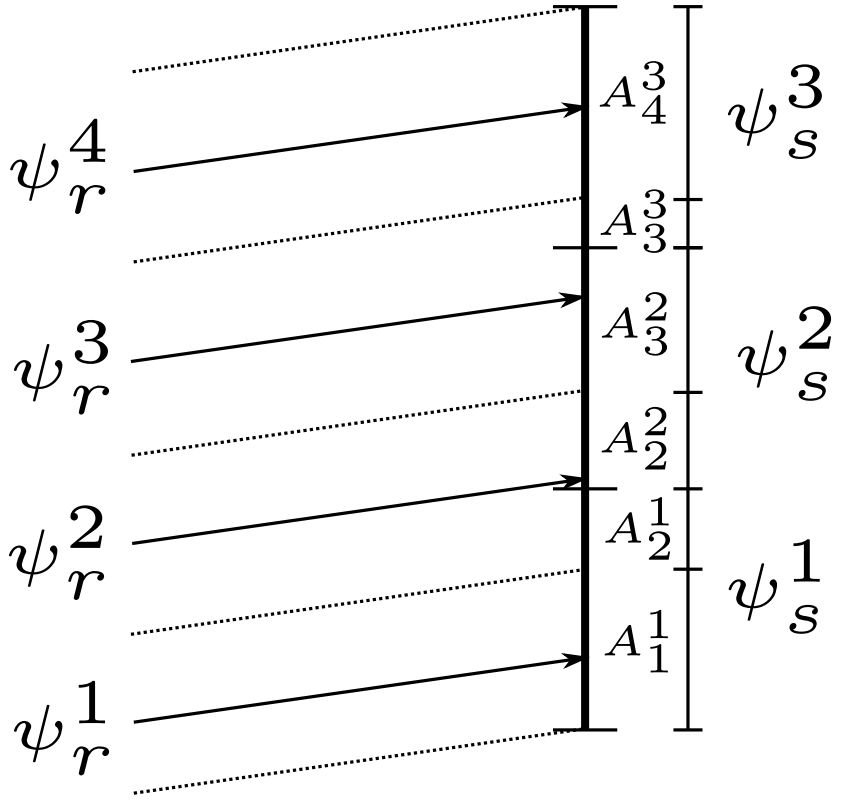


Figure 3.7: Example of sub-boundary surface flux for a single surface. Ray flux denoted with the r subscript, and surface flux denoted with s subscript. Ray projection areas are split by sub-boundaries in the line to the right of the impacted surface.

3.4.6.1 3-D Modular Ray-Tracing

In three-dimensional MoC calculations, characteristic rays must be laid down through the three-dimensional domain; Generally, three-dimensional tracks are generated by first creating a set of two-dimensional tracks, and generating three-dimensional tracks that project onto these two-dimensional tracks [32, 49]. The generation of the two-dimensional tracks can be simplified as viewing each of the two-dimensional tracks as a plane in the axial and characteristic directions, z and s respectively. Three-dimensional tracks are then produced by performing two-dimensional ray-tracing on this characteristic plane [32], as is shown in Fig. 3.8

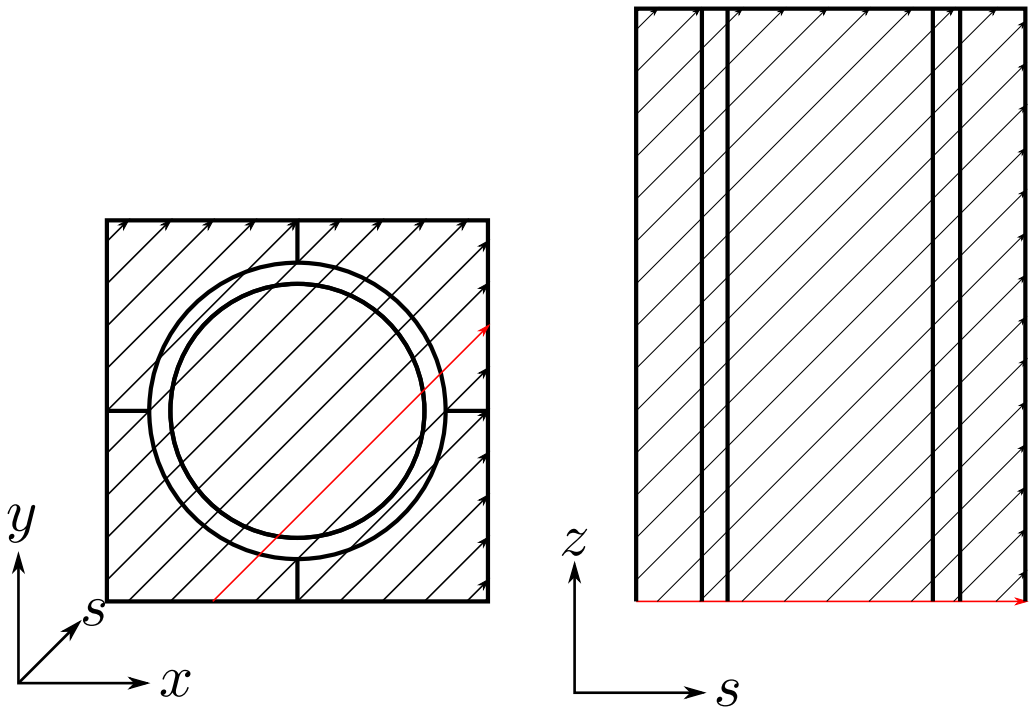


Figure 3.8: 3-D ray-tracing process. Generate 2-D tracks, these become characteristic planes. Along each plane, perform 2-D ray-tracing. The highlighted (red) characteristic track in the 2-D pin-cell on the left becomes the characteristic plane on the right.

There are subtleties in the generation of three-dimensional tracks, which have led to the development of different 3-D modular ray-tracing techniques [32, 49, 50]. Previous work had reported that direct use of the MRT method in 3-D, required that tracks be stored separately for the forward and backward directions [32]; this has, however, been shown not to be the case [49]. The simplified MRT was developed to avoid this issue [32], but it generates significantly more characteristic tracks [49].

As discussed in Section 3.4.2, 2-D MRT perturbs the azimuthal angles in the directional quadrature to ensure DNPL. For 3-D MRT, this also perturbs the polar angles in the directional

quadrature. Kochunas [32] and Jarrett et al. [50] also found that modularization of the directional quadratures led to clustering of the discrete directions, which introduces significant error in the integration of spherical harmonics moments [32], and has implications on particle conservation in anisotropic calculations (Section 3.2.2). In order to avoid this issue, the axial ray-spacing can be reduced until the modularized polar angle is only perturbed within some error criteria [32]; however, this leads to significant increases in the number of tracks (and thus increases the computational costs).

3.4.6.2 Chord-Classification

While MRT significantly reduces the storage requirements of ray-tracing data, in realistic calculations, the data cannot fully be stored in a processors cache. Loading this data from main memory is slow, and in general, the movement of this ray-tracing data into different levels of cache uses significant amounts of time in these calculations. This led to the development of a technique called *chord-classification* for locally axially extruded geometries [51]. Chord-classification recognizes that reactors typically have regularities in the 3-D geometries; this allows for characteristic track-segments to be classified into sets which share the same length. As shown in Fig. 3.9, rays on the same characteristic plane that intersect two vertical mesh boundaries will have the same lengths. Similarly for rays that intersect horizontal planes (though, as axial meshes are usually tall, this is not as common).

By classifying chords of the same length, the length can be stored only once. Additionally, this means that the exponential functions ($F_1(\tau^g_{mki})$ for FSMoC), only need to be calculated once as well. This adds irregularity to the data accessing, that is generally not optimal for computation, and thus is only expected to improve calculation speeds if a large number of chords can be classified [51]. Indeed, it was observed that most rays (96%) could be categorized as "V-chords", which that intersect two vertical surfaces. This led to a 40% reduction in transport sweep time [51]. Additionally, chord-classification allows for reduced track-storage, reducing memory by up to 93%.

3.4.6.3 On-the-Fly Ray-Tracing

The on-the-fly ray-tracing technique [52] uses some of the ideas of the chord-classification method [51], but only stores two-dimensional track information. All three-dimensional tracks are generated and temporarily stored during the transport sweep (on-the-fly), leading to significant memory savings (94% reduction), with minimal computational overhead [52]. While leading to significant (compared to the storage of all rays) memory reduction, this approach does not see the 40% run-time improvement that the chord-classification method has. It would also be a fairer comparison to compare the memory storage compared to the chord-classification method, which also showed

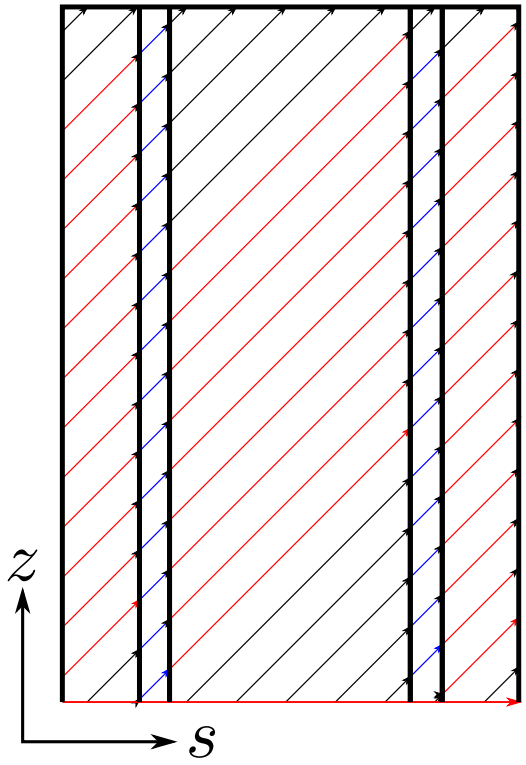


Figure 3.9: 3-D example of chord-classification. Colored (red and blue) characteristic tracks represent groups of "V-chords", rays between two vertical surfaces.

significant improvement over traditional storage techniques. The chord-classification technique already boasted a 93% reduction in track-storage memory [51]; while on-the-fly ray-tracing allows for additional reduction, it is not a significant improvement over chord-classification.

3.4.7 Transport Sweeping with the Method of Characteristics

The MoC is solved iteratively, each iteration is typically referred to as a *transport sweep*. For given boundary conditions and a source, a transport sweep is used to compute estimates of the scalar flux and other moments using equations in the form Eqs. (3.31) for FSMoC and Eqs. (3.46) for LSMoC. This is an overly generalized description of a transport sweep, and there are considerations that arise from the different ray-tracing techniques discussed.

In global ray-tracking procedures, calculations are most often carried out by examining a ray from end to end. The angular flux at the incident end can be found from the boundary conditions. Along each segment, a transmission calculation can be carried out (Eqs. 3.26 and 3.43), and flux moments can be accumulated. This procedure is shared by the MRT, which constructs global rays (long rays) by linking modular rays, as shown in Fig. 3.3. This allows for each ray calculation to be

carried out in parallel, since each ray calculation is effectively independent [32]; though special considerations must be taken to avoid race-conditions in the accumulation of moments.

However, in the macroband method, transport sweeping is carried out in a different manner. It is not possible to generate a continuous characteristic track spanning the global domain when using MRMB techniques. Thus, transport calculations are carried out in a pin-by-pin (assuming pins are the subsystems on which track data is generated) basis. For each pin calculation, boundary or interface angular fluxes can be computed as described in Section 3.4.5.1, and transmission/accumulation calculations are carried out by sweeping over the tracks. This can be done by considering each track separately, or by considering each ray with a macroband that is guaranteed to pass through the same regions. The resulting angular flux on the exiting interface must be then approximated, this can be done as described in Section 3.4.5.1.

Pin-by-pin transport calculations must be carried out in a specific order, by considering the dependencies of the angular flux. Figure 3.10 displays the sweeping order for a 2×2 array of pins in 2-D; this order is significantly different than the ray-by-ray order used in traditional MRT-based MoC calculations. Sweeping in this manner limits parallelism, though it may be possible to construct a dependency graph to get an optimal sweeping order considering macrorays individually, rather than on the pin-by-pin basis; a similar idea was recently implemnted in a S_N code[53]. The dependency graph will be different for each direction of flight [1].

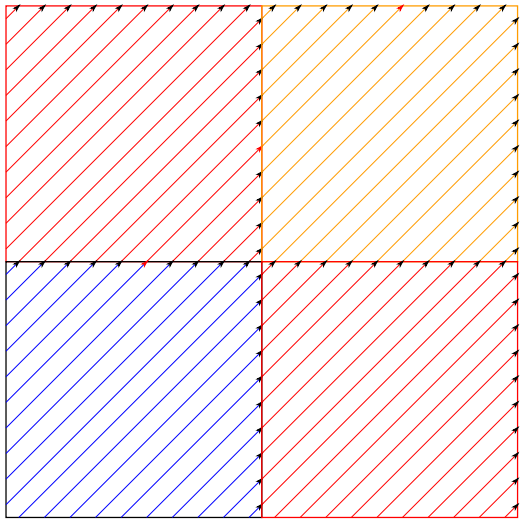


Figure 3.10: Pin-by-pin sweeping order for a 2×2 domain with colors representing the order starting from the bottom left pin.

[1] if done in a pin-by-pin manner, this can be done per octant

3.5 Parallelism

High-fidelity transport methods, such as the MoC, can require significant computational resources for full core calculations; this is particularly true for 3-D calculations. While processing power has increased exponentially since the MoC was first conceived in 1972 [1], since the early 2000's, single-core processing power has largely leveled off. System architectures, as well as code design, have become more focused on *parallel* computations. Previous works [32] made significant progress in the efficient parallelization of the MoC.

Kochunas [32] developed a hybrid-parallel algorithm for the MoC that included thread-based parallelism over characteristic tracks, as well as spatial and angular decomposition. This work showed that the MoC was able to scale well up to 10000's of processors. While this work is important, and has led to significant advancement, the use of 1000's of processors is not feasible for industrial use. It is the author's opinion that the primary focus of research on 3-D MoC techniques should be on serial efficiency, such as the linear-source approximation (LSA) (Chapter 5), and macroray (Chapter 6); however, moderate levels of parallelism are feasible for industry, and so more efficient use of parallel resources should also be a focus of research (Chapter 4).

In MoC calculations, each characteristic track calculation is nearly independent from others; previous works have indicated that loops over characteristic tracks can be parallelized efficiently by using threads on traditional central processing units (CPUs) [32] or on general purpose graphics processing units (GPGPUs) [10]. This type of parallelism is called *shared-data parallelism*, as data is shared between the parallel threads.

Large neutronics calculations may require significant amounts of memory, and thus *distributed-data parallelism* is necessary. In general, this type of parallelism separates (partitions) a domain of the problem, and separate computing nodes are assigned a subdomain. Only data for the assigned subdomain is stored, and thus whole-core simulations become possible; additionally, because each subdomain can be processed in parallel, overall runtimes typically decrease with increasing numbers of subdomains (processors).

MPACT has the capabilities for domain decomposition/parallelism over two separate domains: space and direction. In MPACT, each discrete direction has an easily calculable amount of work, and the decomposition is trivial; in general the same cannot be said of the spatial domain. As part of this thesis work, a more efficient method of spatial decomposition has been investigated and developed in MPACT [54]. Details on the spatial decomposition techniques used in this work are given in Chapter 4.

Bibliography

[1] J.R. Askew. *A Characteristics Formulation of the Neutron Transport Equation in Complicated Geometries*. 1972.

[2] M.J. Halsall. "CACTUS, a characteristics solution to the neutron transport equations in complicated geometries". In: *Transactions of the American Nuclear Society*. 1980. URL: https://inis.iaea.org/search/search.aspx?orig{_}q=RN:12592444.

[3] Benjamin Collins 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. ISSN: 10902716. DOI: 10.1016/j.jcp.2016.08.022.

[4] Dave Knott and Akio Yamamoto. "Lattice Physics Computations". In: *Handbook of Nuclear Engineering* (2010), pp. 913–1239. DOI: 10.1007/978-0-387-98149-9_9.

[5] Ser Gi Hong and Nam Zin Cho. "CRX: A code for Rectangular and Hexagonal Lattices Based on the Method of Characteristics". In: *Annals of Nuclear Energy* 25.97 (1998), pp. 547–565.

[6] Shinya Kosaka and Etsuro Saji. "Transport theory calculation for a heterogeneous multi-assembly problem by characteristics method with direct neutron path linking technique". In: *Journal of Nuclear Science and Technology* 37.12 (2000), pp. 1015–1023. ISSN: 00223131. DOI: 10.1080/18811248.2000.9714987.

[7] K.S. Smith and J.D. Rhodes III. "Full-Core, 2-D, LWR Core Calculations with CASMO-4E". In: *Physor* 1 (2002), pp. 1–13.

[8] Naoki Sugimura et al. "Neutron Transport Models of AEGIS: An Advanced Next-Generation Neutronics Design System". In: *Nuclear Science and Engineering* 155.2 (2006), pp. 276–289. ISSN: 0029-5639. DOI: 10.13182/nse155-276.

[9] Emiliano Masiello, Richard Sanchez, and Igor Zmijarevic. "New Numerical Solution with the Method of Short Characteristics for 2-D Heterogeneous Cartesian Cells in the APOLLO2 Code: Numerical Analysis and Tests". In: *Nuclear Science and Engineering* 161.3 (2008), pp. 257–278. ISSN: 0029-5639. DOI: 10.13182/nse161-257.

[10] William Boyd et al. “The OpenMOC method of characteristics neutral particle transport code”. In: *Annals of Nuclear Energy* 68 (2014), pp. 43–52. ISSN: 03064549. DOI: 10.1016/j.anucene.2013.12.012.

[11] R. Le Tellier and A. Hébert. “Anisotropy and Particle Conservation for Trajectory-Based Deterministic Methods”. In: *Nuclear Science and Engineering* 158.1 (2008), pp. 28–39. ISSN: 0029-5639. DOI: 10.13182/NSE08-A2736.

[12] Rodolfo M. Ferrer and Joel D. Rhodes. “The linear source approximation and particle conservation in the Method of Characteristics for isotropic and anisotropic sources”. In: *Annals of Nuclear Energy* 115 (2018), pp. 209–219. ISSN: 03064549. DOI: 10.1016/j.anucene.2018.01.023.

[13] Akio YAMAMOTO, Yasunori KITAMURA, and Yoshihiro YAMANE. “Simplified Treatments of Anisotropic Scattering in LWR Core Calculations”. In: *Journal of Nuclear Science and Technology* 45.3 (2008), pp. 217–229. ISSN: 0022-3131. DOI: 10.1080/18811248.2008.9711430.

[14] Petko Petkov, Toshikazu Takeda, and Takamasa Mori. “Comparison of the Flat and Linear Source Variants of the Method of Characteristics”. In: *Annals of Nuclear Energy* 26.10 (1999), pp. 935–942. ISSN: 03064549. DOI: 10.1016/S0306-4549(98)00109-1.

[15] Rodolfo M. Ferrer and Joel D. Rhodes. “A Linear Source Approximation Scheme for the Method of Characteristics”. In: *Nuclear Science and Engineering* 182.2 (2016), pp. 151–165. ISSN: 0029-5639. DOI: 10.13182/NSE15-6.

[16] Edward W. Larsen and Warren F. Miller. “Convergence Rates of Spatial Difference Equations for the Discrete-Ordinates Neutron Transport Equations in Slab Geometry”. In: *Nuclear Science and Engineering* 73.1 (1980), pp. 76–83. ISSN: 0029-5639. DOI: 10.13182/nse80-3.

[17] M. J. Halsall. “Neutron Transport in WIMS by the Characteristics Methods”. In: *Transactions of the American Nuclear Society1*. 1993, pp. 454–455.

[18] Chuntao Tang and Shaohong Zhang. “Development and verification of an MOC code employing assembly modular ray tracing and efficient acceleration techniques”. In: *Annals of Nuclear Energy* 36.8 (2009), pp. 1013–1020. ISSN: 03064549. DOI: 10.1016/j.anucene.2009.06.007.

[19] Petko T. Petkov and Toshikazu Takeda. “Transport calculations of MOX and UO2 pin cells by the method of characteristics”. In: *Journal of Nuclear Science and Technology* 35.12 (1998), pp. 874–885. ISSN: 00223131. DOI: 10.1080/18811248.1998.9733960.

[20] C Rabiti et al. “Quasi Linear Representation of the Isotropic Scattering Source for the Method of Characteristics”. In: *International Conference on Mathematics, Computational Methods & Reactor Physics* January (2009), pp. 3–7.

[21] S Santandrea and R Sanchez. “POSITIVE LINEAR AND NONLINEAR SURFACE CHARACTERISTIC SCHEMES FOR THE NEUTRON TRANSPORT EQUATION IN UNSTRUCTURED GEOMETRIES”. In: *Physor*. 2002.

[22] Simone Santandrea, Richard Sanchez, and Pietro Mosca. “A linear surface characteristics approximation for neutron transport in unstructured meshes”. In: *Nuclear Science and Engineering* 160.1 (2008), pp. 23–40. ISSN: 00295639. DOI: 10.13182/NSE07-69.

[23] Romain Le Tellier and Alain Hébert. “On the integration scheme along a trajectory for the characteristics method”. In: *Annals of Nuclear Energy* 33.14-15 (2006), pp. 1260–1269. ISSN: 03064549. DOI: 10.1016/j.anucene.2006.07.010.

[24] Alain Hébert. “High-Order Linear Discontinuous and Diamond Differencing Schemes Along Cyclic Characteristics”. In: *Nuclear Science and Engineering* 184.4 (2016), pp. 591–603. ISSN: 0029-5639. DOI: 10.13182/nse16-82.

[25] Alain Hébert. “Acceleration of step and linear discontinuous schemes for the method of characteristics in DRAGON5”. In: *Nuclear Engineering and Technology* 49.6 (2017), pp. 1135–1142. ISSN: 2234358X. DOI: 10.1016/j.net.2017.07.004.

[26] Emiliano Masiello, Roberto Clemente, and Simone Santandrea. “High-Order Method of Characteristics for 2-D Unstructured Meshes”. In: *International Conference on Mathematics, Computational Methods & Reactor Physics* June 2015 (2009). URL: http://mathematicsandcomputation.cowhosting.net/MC09/pdfs/202865.pdf{\%}5Cnpapers3://publication/uuid/217E92CC-587D-4349-843B-58B5693D625C.

[27] X. M. Chai, K. Wang, and D Yao. “The linear source approximation in three dimension characteristics method”. In: *International Conference on Mathematics, Computational Methods \& Reactor Physics (M\&C 2009)* February (2009), pp. 1–14.

[28] Y. I. Chang. “Technical rationale for metal fuel in fast reactors”. In: *Nuclear Engineering and Technology* 39.3 (2007), pp. 161–170. ISSN: 1738-5733. URL: http://gehitachiprism.com/wp-content/themes/geh{_}prism/resources/Technical{_}Rationale{_}for{_}Metal{_}Fuel{_}in{_}Fast{_}Reactors.pdf.

[29] Andrew Fitzgerald and Brendan Kochunas. “Fast Exponential Function Approximations for the Method of Characteristics with Linear Source”. In: *Transactions of the American Nuclear Society*. Orlando, USA, 2018, pp. 645–648.

[30] W. L. Filippone, S. Woolf, and R. J. Lavigne. “Particle Transport Calculations with the Method of Streaming Rays”. In: *Nuclear Science and Engineering* 77.2 (1980), pp. 119–136. ISSN: 0029-5639. DOI: 10.13182/nse81-a21346.

[31] G. J. Wu and R. Roy. “A new characteristics algorithm for 3D transport calculations”. In: *Annals of Nuclear Energy* 30.1 (2003), pp. 1–16. ISSN: 03064549. DOI: 10.1016/S0306-4549(02)00046-4.

[32] Brendan Matthew Kochunas. “A Hybrid Parallel Algorithm for the 3-D Method of Characteristics Solution of the Boltzmann Transport Equation on High Performance Compute Clusters”. Doctoral. University of Michigan, 2013, pp. 1–194.

[33] Yeon Sang Jung and Han Gyu Joo. “Decoupled Planar MOC Solution for Dynamic Group Constant Generation in Direct Three-Dimensional Core”. In: *M&C 2009* (2009).

[34] Han Gyu Joo et al. “Methods and performance of a three-dimensional whole-core transport code DeCART”. In: *PHYSOR 2004*. 2004, pp. 21–34. ISBN: 0894486837. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-22344449157{\&}partnerID=tZOtx3y1.

[35] Franck Gabriel et al. “APOLLO3: CEA/DEN DETERMINISTIC MULTI-PURPOSE CODE FOR REACTOR PHYSICS ANALYSIS”. In: May 2016. 2016.

[36] MPACT Team. *MPACT Theory Manual v2.2.0*. Tech. rep. Consortium for Advanced Simulation of Light Water Reactors, 2016.

[37] Alain Hébert. “Acceleration of Step and Linear Discontinuous Schemes for the Method of Characteristics in DRAGON5”. In: *M&C 2017*. Jeju, Korea, 2017.

[38] Eduardo A. Villarino et al. “HELIOS: Angularly Dependent Collision Probabilities”. In: *Nuclear Science and Engineering* 112.1 (1992), pp. 16–31. ISSN: 0029-5639. DOI: 10.13182/NSE112-16.

[39] Akio Yamamoto. “Reduction in spatial discretization error in the method of characteristics by using the mobile-chord ray tracing method”. In: *Annals of Nuclear Energy* 35.5 (2008), pp. 783–789. ISSN: 03064549. DOI: 10.1016/j.anucene.2007.09.018.

[40] John R. Tramm et al. “ARRC: A random ray neutron transport code for nuclear reactor simulation”. In: *Annals of Nuclear Energy* 112 (2018), pp. 693–714. ISSN: 18732100. DOI: 10.1016/j.anucene.2017.10.015.

[41] Akio Yamamoto et al. “Non-Equidistant Ray Tracing for the Method of Characteristics”. In: *M&C 2005* (2005), pp. 1–10.

[42] Ser Gi Hong and Nam Zin Cho. “Method of Characteristic Direction Probabilities for Heterogeneous Lattice Calculation”. In: *Nuclear Science and Engineering* 132.1 (2017), pp. 65–77. ISSN: 0029-5639. DOI: 10.13182/nse99-a2049.

[43] Zhouyu Liu et al. “Theory and analysis of accuracy for the method of characteristics direction probabilities with boundary averaging”. In: *Annals of Nuclear Energy* 77 (2015), pp. 212–222. ISSN: 0306-4549. DOI: 10.1016/J.ANUCENE.2014.11.016.

[44] François Févotte, Simone Santandrea, and Richard Sanchez. “Advanced Transverse Integration for the Method of Characteristics”. In: *M&C 2007*. 2007, pp. 1–12. ISBN: 0894480596.

[45] John R. Tramm et al. “A task-based parallelism and vectorized approach to 3D Method of Characteristics (MOC) reactor simulation for high performance computing architectures”. In: *Computer Physics Communications* 202 (2016), pp. 141–150. ISSN: 00104655. DOI: 10.1016/j.cpc.2016.01.007.

[46] Akio Yamamoto et al. “GENESIS: A Three-Dimensional Heterogeneous Transport Solver Based on the Legendre Polynomial Expansion of Angular Flux Method”. In: *Nuclear Science and Engineering* 49.6 (2017), pp. 1143–1156. ISSN: 2234358X. DOI: 10.1016/j.net.2017.06.016.

[47] Laurent Graziano et al. “Polynomial Characteristics Method for Neutron Transport in 3D extruded geometries”. In: *M&C 2017*. Jeju, Korea, 2017.

[48] Geoffrey Gunow et al. “Accuracy and Performance of 3D MOC for Full-Core PWR Problems”. In: *M&C 2017*. Jeju, Korea, 2017.

[49] S. Shaner et al. “Theoretical analysis of track generation in 3D method of characteristics”. In: *Mathematics and Computations, Supercomputing in Nuclear Applications and Monte Carlo International Conference, M&C+SNA+MC 2015* 4.Mc (2015), pp. 2667–2681.

[50] Michael Jarrett et al. *Improvements to the 3D MOC Solver in MPACT*. Tech. rep. CASL-U-2016-1232-000. CASL, 2016.

[51] Daniele Sciannandrone, Simone Santandrea, and Richard Sanchez. “Optimized tracking strategies for step MOC calculations in extruded 3D axial geometries”. In: *Annals of Nuclear Energy* 87 (2016), pp. 49–60. ISSN: 18732100. DOI: 10.1016/j.anucene.2015.05.014.

[52] Geoffrey Gunow et al. “Reducing 3D MOC Storage Requirements with Axial On- the-fly Ray Tracing”. In: *Physics of Reactors 2016, PHYSOR 2016: Unifying Theory and Experiments in the 21st Century*. Sun Valley: American Nuclear Society, 2016. URL: http://hdl.handle.net/1721.1/109864{\%}0ACreative.

[53] M P Adams et al. “Provably Optimal Parallel Transport Sweeps on Regular Grids”. In: *International Conference on Mathematics and Computational Methods Applied to Nuclear Science & Engineering Sun Valley* (2013). URL: https://e-reports-ext.llnl.gov/pdf/718952.pdf.

[54] Andrew P Fitzgerald et al. “Spatial decomposition of structured grids for nuclear reactor simulations”. In: *Annals of Nuclear Energy* 132 (2019), pp. 686–701. ISSN: 0306-4549. DOI: 10.1016/j.anucene.2019.06.054.

CHAPTER 4

Spatial Decomposition

4.1 Introduction

Until relatively recently, the method of choice for whole-core neutronics calculations has been neutron diffusion. Neutron diffusion codes can perform whole-core calculations, on a typical workstation, in relatively short run-times. However, with the recent shift towards higher fidelity methods such as S_N, and MoC [1], more significant computational resources are necessary. These high fidelity methods allow for more detailed analysis, through finer resolution and use of fewer approximations, but typically take far more time to perform calculations. Although processor clock-speeds have significantly improved, in the past several years computer cores have, for the most part, not become faster. Instead, computers have adopted multi-core architectures to continue to improve performance through parallel computations. Therefore, to reduce the run-times (real-time) of high fidelity simulations, it is thus necessary to rely on parallelism.

There are many different aspects of parallelism, and thread-based parallelism has been discussed previously Section 3.5. Shared-data parallelism (threads), is limited to the resources of a single computational node. In order to utilize more resources, it is necessary to use a technique called *domain decomposition*. Even without considerations for run-times, these high-fidelity simulations typically use more memory than is available on a single node, and domain decomposition becomes necessary. In general, domain decomposition involves decomposing one domain of the problem into smaller subdomains; the domains to decompose are the independent variables of the solution e.g. space, direction, and energy. Each smaller subdomain is assigned to a separate processor, and run in parallel. This model also requires communication between the processors/subdomains.

In Monte-Carlo simulations, it is common that spatial decomposition involves duplication of some spatial locations. However, in deterministic transport methods, each domain is typically *partitioned*; the spatial domain is split with no overlap between subdomains. The MPACT [2] code has the ability to decompose two domains: space and direction. In MPACT, each discrete direction has a calculable amount of work, and the decomposition can simply be performed by a

greedy algorithm; in general, the same cannot be said of the spatial domain. This chapter focuses on improvements to the spatial decomposition techniques used in MPACT. These techniques are also applicable to other deterministic transport codes and similar results would be expected. The contents of this chapter are adapted from an article published on this work [3].

As diffusion has been the method of choice for so long in the reactor physics field, spatial partitioning techniques common in other fields have, largely, not been applied. Transport codes such as MPACT [2], or OpenMOC [4] used simple spatial partitioning methods that divided the core into uniformly sized blocks. However, the spatial partitioning of a reactor can be abstracted to a graph partitioning problem [5], which has been well-studied in computer science [6] and applied to other simulation fields such as computational fluid dynamics [7]. In general, the graph partitioning problem is *NP-complete*, meaning that a partitioning cannot be easily verified as optimal; therefore, graph partitioning relies on approximate heuristic methods. Many different methods have been developed for graph partitioning, several of which are discussed in Section 4.3.

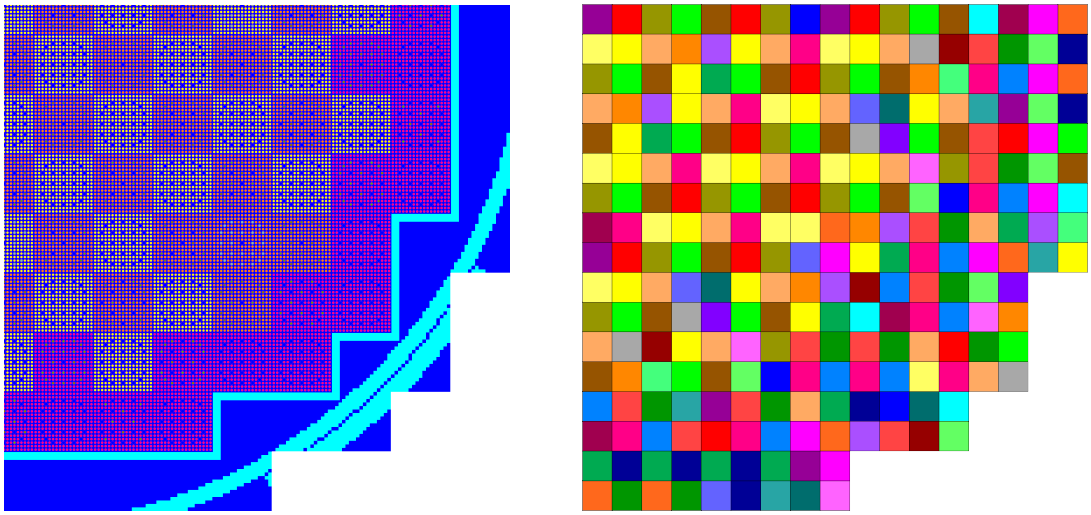
There are also existing libraries for graph partitioning, such as METIS[8] and Zoltan [9]. These libraries have been used by codes such as PROTEUS and Rattlesnake [10, 11]. However, due to constraints on the spatial decomposition that may not be satisfied by these libraries, this work has opted not to use these libraries. This is discussed in detail in Section 4.2.

The remainder of this chapter is structured as follows: in Section 4.2, a description of spatial decomposition in MPACT is given. Section 4.3 introduces relevant graph theory concepts, and the methods used for spatial partitioning in this work. Section 4.4 describes the applications of these graph theory methods in MPACT. Section 4.5 compares methods for 2-D and 3-D reactor simulations. Finally, Section 4.7 lists the conclusions that are drawn from this work.

4.2 Spatial Decomposition in MPACT

The MPACT code uses the MoC for neutronics calculations, and was originally developed for direct whole-core simulation of LWRs. As mentioned in Section 3.4.2, MPACT uses the modular ray-tracing [12] to significantly reduce track memory usage. The ray-tracing modules are small geometries that are often repeated in the reactor; these ray-tracing modules are used as the smallest unit for spatial decomposition in MPACT [13]. The ray-tracing modules are typically an axial slice of a quarter of a full Pressurized Water Reactor (PWR) fuel assembly, as shown in Fig. 4.1 The core consists of a structured grid of these modules in which each module has the same dimensions but may have different numbers of computational cells. Therefore, in MPACT, the spatial decomposition is a structured grid partitioning problem.

In general, it is possible to use the computational cells as the smallest unit in the decomposition. However, this causes the decomposition problem to become an unstructured mesh partitioning



(a) fine mesh (b) ray-tracing module mesh

Figure 4.1: Example quarter core configuration and corresponding ray-tracing modular mesh in MPACT.

problem. This has not been done in MPACT because communication would become significantly more complicated. Additionally, there would be more re-entrant rays, which would negatively affect the rate of convergence.

MPACT has had two spatial decomposition methods in the past: manual decomposition, and assembly-based decomposition. A user may manually enter a decomposition [13], but it is time-consuming to construct a balanced decomposition, and the result will likely be suboptimal. An automated method exists that recursively bisects the core using Morton-ordering [14] applied to the reactor assembly geometries. While this method is automated, it often yields very imbalanced domains, and it also restricts the number of subdomains that can be used.

Previous work has shown that spatial decomposition of reactors can be abstracted to a graph partitioning problem [5]. The use of graph partitioning methods in MPACT is expected to solve the issues encountered in each of the two approaches described above. These methods can be used to decompose into an arbitrary number of domains with high quality results, without user input.

Existing graph partitioning libraries such as METIS [8] partition graphs efficiently and have high quality results. To use all given processors, MPACT requires that each spatial subdomain contains at least one module, i.e. no partition can be empty. However, in some cases, particularly when the number of partitions is high, METIS may generate empty partitions. This means METIS cannot be used to decompose the core into an arbitrary number of subdomains without modifying the resulting partitions. For this reason, MPACT does not rely on third-party libraries for graph partitioning in the spatial decomposition process.

4.3 Applied Graph Theory

The spatial decomposition of a reactor core can be abstracted to the partitioning of a graph. Specifically, this would be a weighted graph, $G(V, E)$, that is comprised of a set of vertices, V, and a set of edges, E, that connect pairs of vertices. In general, these vertices and edges may have weights; a vertex v_i will have weight w_i, and an edge e_i between vertices v_i and v_j will have weight c_{ij}. In MPACT, a vertex represents a ray-tracing module, and the edges represent communication between adjacent modules in the MoC. The graphs are undirected because communication between ray-tracing modules is two-way.

Previous work [5] applied unweighted graph partitioning techniques to the reactor spatial decomposition problem; the work presented here applies generalizations and improvements to the methods used for graphs with weighted vertices and edges. A vertex's weight indicates the amount of computational work that is needed; as one might expect, this is highly correlated with the number of computational cells. This is shown in Section 4.5. In general, the edges may also be weighted to account for different amounts of data transfer. This is discussed in more detail in Section 4.4.

The goal of partitioning is for each subdomain to have equal weight, with minimal weight of edges cut by partition boundaries. This is equivalent to each subdomain having the same amount of computational work with minimized communication between processes. If each process has roughly the same amount of work to perform, then less time will be spent waiting for other processes, thus improving parallel efficiency. Also, with less communication, less time will be spent passing data between processes, so the parallel overhead will be reduced.

In this work, methods were separated into two distinct categories: partitioning methods and partition refinement (improvement) methods. Partitioning methods give a near-balanced partitioning for a given graph. Refinement methods attempt to reduce communication between existing partitions in a graph. As applied in MPACT, refinement methods typically did not significantly reduce communication. These methods and results are presented in Section 4.6.

4.3.1 Graph Partitioning Methods

In this work, recursive partition methods were considered due to their capability to partition into arbitrary numbers of domains. Each of these recursive partitioning methods sorts the graph, using different methods, and then divides or "cuts" the graph into two sub-graphs with approximately equal vertex weights. Once a graph's vertices are sorted into a list, V_s, the graph can be bisected using Algorithm 3.

Multi-level partitioning methods are widely used in other fields such as networking, where graphs can become very large; however, in MPACT, the number of ray-tracing modules is on the order of a few hundred to several thousand, which directly correlates to the size of the graph. Additionally, for

MPACT, the decomposition problem is static, so the computation time for partitioning is expected to be negligible as it can simply be performed one time at the outset. Due to the small graph size, multi-level methods were not considered as part of this work.

Algorithm 3 The algorithm used to determine how to cut a graph, $G(V, E)$, into two sub-graphs based on a sorted vertex list V_s, and that the graph will be recursively partitioned into N groups.

procedure GRAPH CUT($G(V, E), V_s, N$)
  $N_1 \leftarrow \lfloor N/2 \rfloor$ ▷ Desired number of recursive partitions for first subgraph
  $W_1 \leftarrow \frac{N_1}{N} \sum_{v_i \in V} w_i$ ▷ Ideal weight of first subgraph
  Let V_1 be a set of vertices such that:
    - $V_1 \subset V$
    - The vertices V_1 are taken in order from V_s
    - $W_1 - \sum_{v_i \in V_1} w_i$ is minimized
  Let V_2 be the subset $V \setminus V_1$
  Optionally call a refinement method
  Create a graph G_1 from V_1
  Create a graph G_2 from V_2
end procedure

4.3.1.1 Recursive Spectral Bisection

The recursive spectral bisection (RSB) method, originally developed by Pothen et al. [15], has been highly successful and widely used in graph partitioning [16, 17]. This method relies entirely on the connectivity of the graph and not on its geometry. The RSB method has been improved to allow for partitioning of *weighted* graphs into any number of domains [18].

The RSB method makes use of the Laplacian matrix of a graph; specifically the second-smallest eigenvalue of this matrix, referred to by Fiedler as the *algebraic connectivity* [19]. The eigenvector associated with this eigenvalue has also been known as the *Fiedler vector*. For weighted graphs, the weighted Laplacian matrix is used in lieu of the Laplacian matrix; matrix elements are given by

$$L_{ij} = \begin{cases} d_i, & i = j, \\ -c_{ij}, & i \neq j, \\ 0, & \text{else}, \end{cases} \tag{4.1}$$

where d_i is the sum of edge weights from vertex v_i, and c_{ij} is the weight of the edge between vertices v_i and v_j. The Fiedler vector is found from this weighted Laplacian matrix; by sorting the values of the Fiedler vector, the vertices can be reordered in a one-dimensional list V_s. This list of

vertices is then divided into two sets, based on weight and total number of partitions needed (see Algorithm 3). The recursive spectral bisection algorithm is listed in Algorithm 4.

Algorithm 4 The recursive spectral bisection (RSB) algorithm.

1: **procedure** RSB($G(V, E)$)
2: Let L be the weighted Laplacian of $G(V, E)$
3: Compute eigenvectors of L
4: Use the Fiedler vector to sort $V \rightarrow V_s$ ▷ If tie, use larger eigenvectors
5: Cut graph into $G_1(V_1, E_1), G_2(V_2, E_2)$: Algorithm 3
6: RSB($G_1(V_1, E_1)$)
7: RSB($G_2(V_2, E_2)$)
8: **end procedure**

4.3.1.2 Recursive Inertial Bisection

Another class of recursive partitioning methods are coordinate or geometric methods. There are many different geometric partitioning methods in existence; in this study, the recursive intertial bisection (RIB) method [6, 20] was investigated. This method uses only the geometry of the graph to construct a bisector and does not consider the connectivity (edges) in any way.

The RIB method determines a bisector that cuts the graph into two approximately equally-sized subdomains. This is easily generalized for weighted graphs. The bisector should have approximately equal amounts of weight on each side. The RIB makes no assumption of the orientation of the graph in space, unlike some other coordinate partitioning methods. The principle axes of the graph are equivalent to the eigenvectors of the inertial matrix given by

$$\boldsymbol{I} \equiv \sum_{i=1}^{n} w_i \left(\boldsymbol{x}_i - \overline{\boldsymbol{x}}\right)^T \left(\boldsymbol{x}_i - \overline{\boldsymbol{x}}\right), \tag{4.2}$$

where n is the number of vertices, $\boldsymbol{x}_i$ is a row-vector containing coordinates of vertex v_i, and $\overline{\boldsymbol{x}}$ is the mean coordinate vector given by

$$\overline{\boldsymbol{x}} \equiv \frac{\sum_{i=1}^{n} w_i \boldsymbol{x_i}}{\sum_{i=1}^{n} w_i}. \tag{4.3}$$

An approximate bisector is given as passing through the weighted centroid with normal vector given as one of the eigenvectors of $\boldsymbol{I}$.

Other works [6, 20] have used the smallest eigenvalue's eigenvector as a normal vector to minimize the mean-square distance of vertices from the bisecting line or plane. However, in this work, the largest eigenvalue's eigenvector is used, so a smaller cut-size is typically given while still

bisecting the graph into two subdomains of approximately equal weight. This is the case because in the MoC, communication scales with the surface area between adjacent ray-tracing modules. This may not be the case for other computational methods.

In general, a line or plane passing through the weighted centroid with the eigenvector normals will not cut the graph into two equally weighted subdomains. Instead, the vertices will be sorted according to their distance from the approximate bisectors, and then a cut will be made so that near equal amounts of weight are in each set using Algorithm 3. This sorting and cutting based on weights is equivalent to shifting the bisector in the direction of the normal vector. An example is visualized in Fig. 4.2. The RIB algorithm is listed in Algorithm 5.

Algorithm 5 The basic recursive intertial bisection (RIB) algorithm.

procedure RIB($G(V, E)$)
    Compute the weighted centroid of the graph $\overline{\boldsymbol{x}}$, given by Eq. (4.3)
    Shift coordinates relative to centroid: $\boldsymbol{x}_i^c = \boldsymbol{x}_i - \overline{\boldsymbol{x}} \quad \forall i \in V$
    Compute inertial matrix $\boldsymbol{I}$, given by Eq. (4.2)
    Compute eigenvectors of $\boldsymbol{I}$. Largest eigenvalue's eigenvector $\boldsymbol{e}_1$
    Compute distance from largest eigen-pair bisector: $d_i = \boldsymbol{x}_i^c \cdot \boldsymbol{e}_1$
    Sort $V \rightarrow V_s$ based on d_i. ▷ In ties use smaller eigenvalue's eigenvector
    Cut graph into $G_1(V_1, E_1), G_2(V_2, E_2)$: Algorithm 3
    RIB($G_1(V_1, E_1)$)
    RIB($G_2(V_2, E_2)$)
end procedure

4.3.1.3 Recursive Expansion-Based Methods

The recursive expansion bisection (REB) methods comprise the last class of partitioning methods examined in this work. These methods begin a bisection step by selecting a vertex as the starting point of a subdomain. This subdomain is then expanded until it is approximately half the size of the graph [5, 6, 21, 22]. In this work, the method outlined by Fitzgerald et al. [5] was slightly modified and generalized to weighted graphs. For the remainder of this work, the acronym *REB* will be used to denote this specific expansion-based method rather than the entire class of methods.

This REB method considers both the geometry and connectivity of the graph. The method begins by choosing a starting vertex for the subdomain and then expands based on a set of prioritized rules. At each expansion step, the next vertex is chosen to be geometrically close to the vertices within the subdomain, and to minimize edges between the subdomain and the remaining graph. This method makes the assumes that the mesh is structured, and that every mesh element is the same shape and size. For the application in MPACT, this is always true.

This REB method uses the concept of a *sphere of influence (SOI)* around a vertex. The SOI

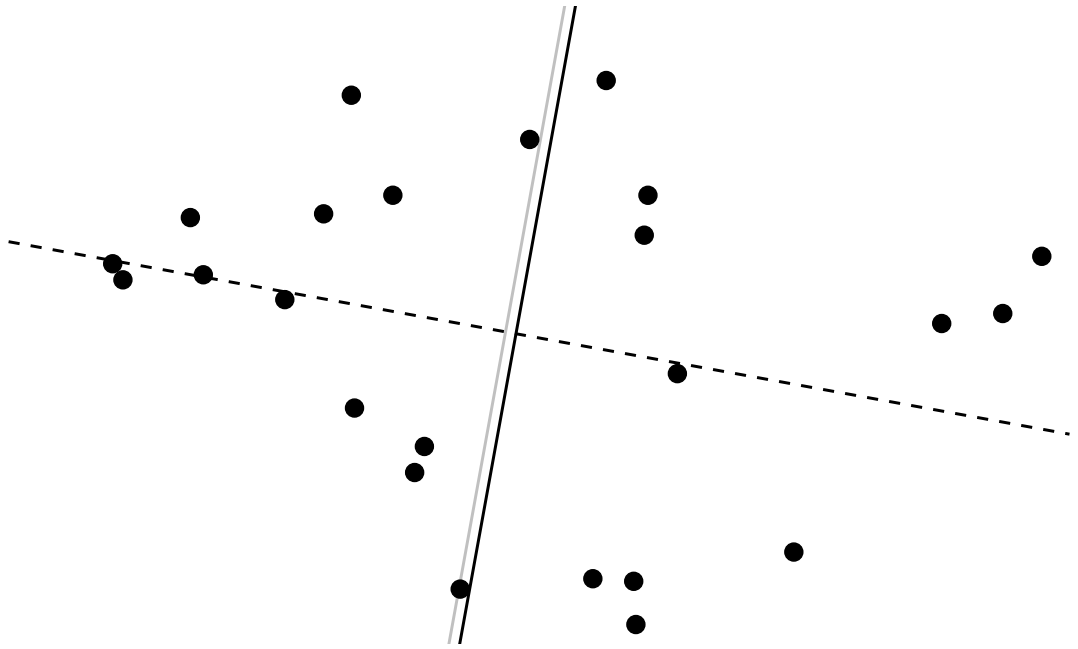


Figure 4.2: Example of an inertial bisection. Vertices are shown as black points, the bisectors of the largest eigen-pair is shown by the black solid, and the bisector of the smallest eigen-pair is shown by the black dashed line. The "shifted" bisector used in the partitioning is shown in grey. While the communication between vertices is not drawn, it is clear that the length (proportional to cut size) of the smallest eigen-pair bisector is larger than that of the largest eigen-pair bisector.

includes directly neighboring vertices and vertices that neighbor more than one of the direct neighbors or that would if the direct neighbor were present in each structured position around the primary vertex. This is shown for 2-D rectangular structured mesh in Fig. 4.3. In the implementation, the sphere of influence is calculated using a geometric distance rather than connectivity.

The starting vertex in this REB method is chosen using a set of prioritized rules:

1. it must be on a graph boundary, i.e. at least one direct neighbor is not present,
2. it must have the lowest summed weight of edges, and
3. it must be located furthest from the weighted centroid (given by Eq. (4.3)).

These rules prioritize the domain beginning its expansion in a "corner" of the graph; this helps to avoid the creation of disjoint subdomains. Vertices within the expanding subdomain are considered internal vertices, and the remaining vertices are considered to be external vertices. During expansion, the next vertex is determined using a set of prioritized rules:

1. it must be neighboring at least one internal vertex,
2. it must have the highest summed weight of edges with internal vertices,

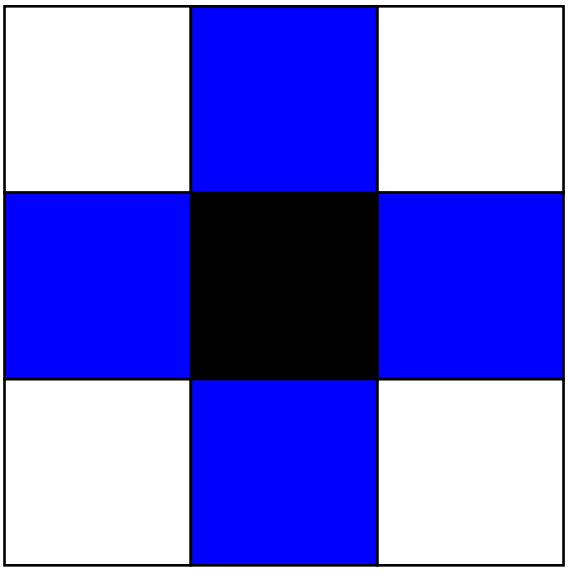


Figure 4.3: "Sphere of influence" example for 2-D rectangular structured grid. The primary vertex is shown in black, direct neighbors are blue, and additional vertices in the sphere are white.

3. it must have the lowest summed weight of edges with external vertices,
4. it must have the largest number of internal SOI vertices,
5. it must have the largest number of external SOI vertices, and
6. it must have the smallest distance from reference vertex.

The reference vertex is in the expanding subdomain, which begins as the first vertex but changes during expansion; the reference vertex is the most recently added vertex with less external communication than the previously added vertex. An example of the expansion order is shown in Fig. 4.4. The rules ensure that the next vertex is connected to the current domain, and prioritizes minimizing communication, and avoids creating disjoint subdomains.

Algorithm 6 The chosen Recursive Expansion Bisection (REB) algorithm.

1: **procedure** REB($G(V, E)$)
2: Compute weighted centroid of the graph
3: Choose a starting vertex for the expanding domain: See rules in Section 4.3.1.3
4: Expand the domain from the starting vertex. Let V_s be the list of vertices in order of the expansion: See rules in Section 4.3.1.3
5: Cut graph into $G_1(V_1, E_1), G_2(V_2, E_2)$: Algorithm 3
6: REB($G_1(V_1, E_1)$)
7: REB($G_2(V_2, E_2)$)
8: **end procedure**

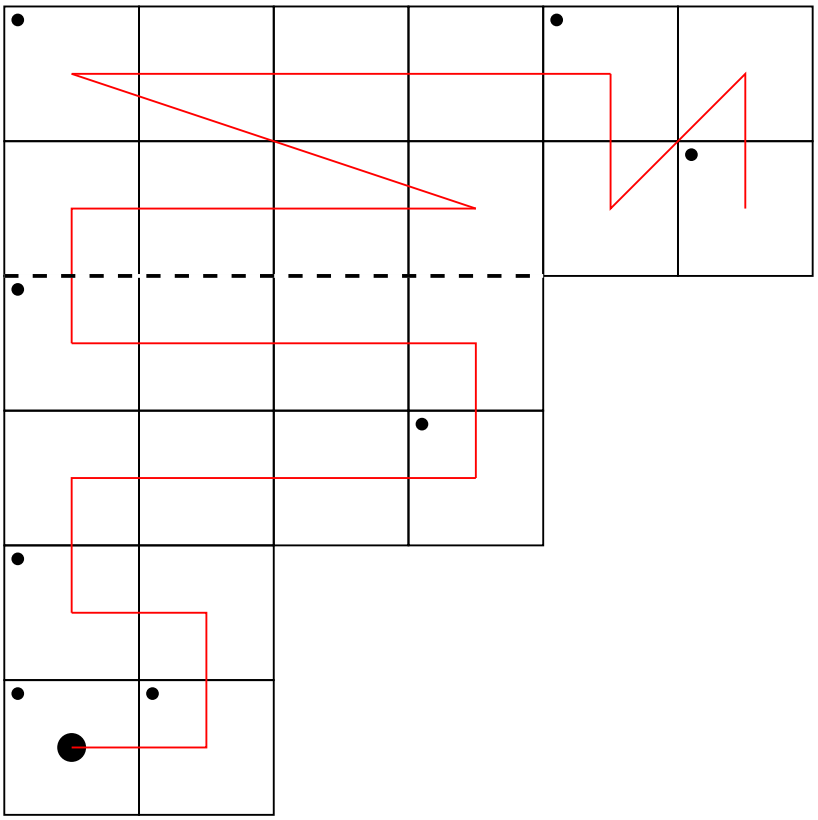


Figure 4.4: An example of the REB method expansion on a small graph. The black lines show the square mesh cells. The expansion begins at the large black point in the center of a cell, and the red line from this shows the expansion's order. Each small black dot in the upper left of a cell indicates the reference vertices during expansion. The thick black dashed line shows the bisecting cut.

4.4 Applications for MPACT

As described in Section 4.2, MPACT's spatial decomposition is performed on the ray-tracing module mesh. However, there are several restrictions on spatial subdomains in MPACT. Each spatial subdomain must be contiguous, and cannot wrap around other spatial subdomains. To account for these restrictions, adaptations are made to the graph partitioning process.

Due to restrictions in MPACT, at each recursive step, each subdomain in a bisection is made contiguous. If a partitioning method results in a noncontiguous subdomain, then each noncontiguous group of modules will be moved into the other subdomain except for the largest group. This fix is done at the expense of load-balance, but is necessary for these methods to be robust in MPACT. To ensure that no subdomain wraps around another, a fix is applied after the graph partitioning process. If a subdomain wraps around another (3 surrounding sides), the portion of the subdomain which is wrapped is moved into the wrapping subdomain.

The ray-tracing modules that make up the core in MPACT can be abstracted into a weighted graph, where vertex weights are the number of computational cells in a module, and edges connecting neighboring modules. These edges represent the communication in MPACT's MoC solver. However, MPACT typically relies on the CMFD acceleration method [23], which involves solving a sparse linear system. The work-load and communication of the CMFD solver, in MPACT this is a linear system solver from PETSc [24], are different than than of the MoC. Thus, what may be an optimal decomposition for the MoC solver, is likely not for the CMFD solver. As the MoC solver is expected to take longer than the CMFD solver, the graph is constructed based on the work-load and communication of MoC.

For 2-D simulations, the application of graph partitioning methods is clear: abstract the 2-D mesh into a graph for partitioning. However, for 3-D, there are additional concerns. MPACT's primary 3-D transport method is the 2D-1D method, in which the MoC is used in the radial directions, and a lower-order solver couples axial planes [25]. For 2D-1D simulations, MPACT currently restricts spatial domains to be aligned in both the radial and axial directions; this is due to implementation, and is not a general requirement of the methods.

To comply with MPACT's restrictions on 3-D spatial domains, the current approach is to axially average module weights (numbers of cells), perform a 2-D graph partitioning on a single plane, and apply the resulting partitioning to all axial planes. This approach restricts the number of spatial domains to be an integer multiple of the number of planes. This axially and radially aligned scheme is expected to work well in many cases, since reactor cores do not typically vary significantly in the axial direction. However, for some designs, this may not be true, and planes near the top or bottom of the core may have significantly fewer cells; in these cases, high load imbalance is to be expected.

It is possible to change MPACT's implementation to lift these alignment restrictions, but that is

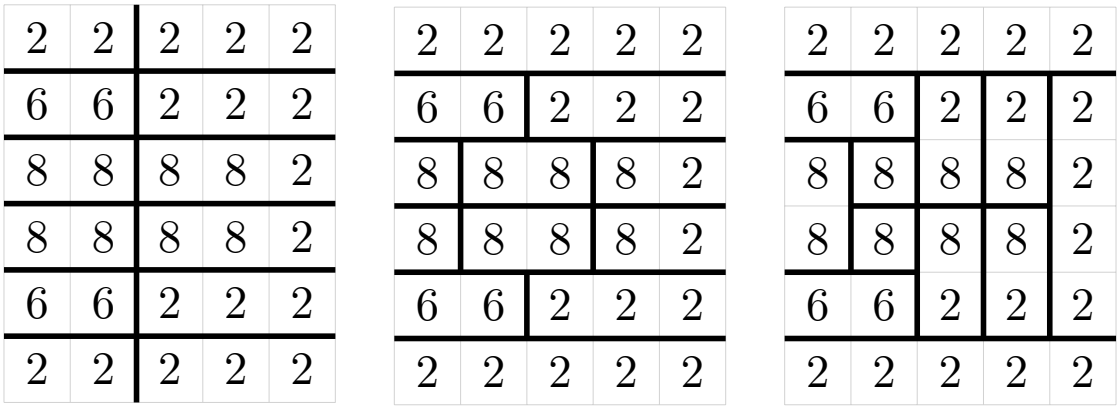


Figure 4.5: Sample decompositions for (a) axially aligned, (b) radially aligned, and (c) generalized decomposition strategies. Rows represent axial planes. Numbers are the vertex weights.

beyond the scope of this work. If spatial domains were aligned in only the radial direction, there may be some benefit to load-balance. In this scheme, each axial plane can be assigned an appropriate number of processes, and a separate 2-D decomposition can be performed for each plane. This also lifts restrictions on the number of domains; the number of domains must only be greater than or equal to the number of planes.

If all alignment restrictions were lifted on MPACT's spatial domains, then a direct partitioning of the 3-D core can be performed by abstracting the entire core to a graph. This scheme provides the most freedom and would be expected to give the most balanced decompositions. In MPACT's 2D-1D solver, the amount of data communicated radially is significantly larger than that communicated axially, so it may be advantageous to assign the edges connecting the neighboring modules in the axial direction lower weights than those in radial directions. By doing so, the overall communication would be expected to be decreased.

Figure 4.5 shows a comparison of the three hypothetical decomposition schemes, for the purposes of illustration. Looking at the maximum-to-minimum ratio (MMR), as an indicator of load-balance, the strategies have clear differences. The MMR for the axially aligned, radially aligned, and generalized strategies in this example are 4.00, 2.67, and 2.00, respectively. However, it is important to note that the largest domain in each case has the same weight (16), so while the different schemes give different balances, the overall run-times are not expected to be different.

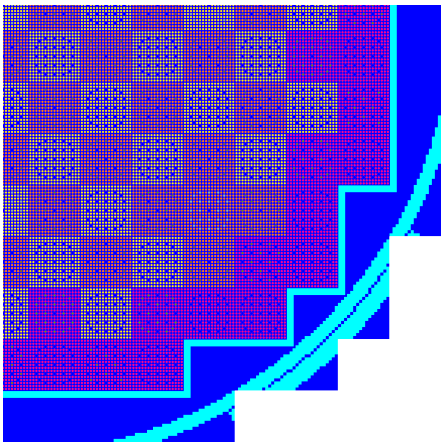
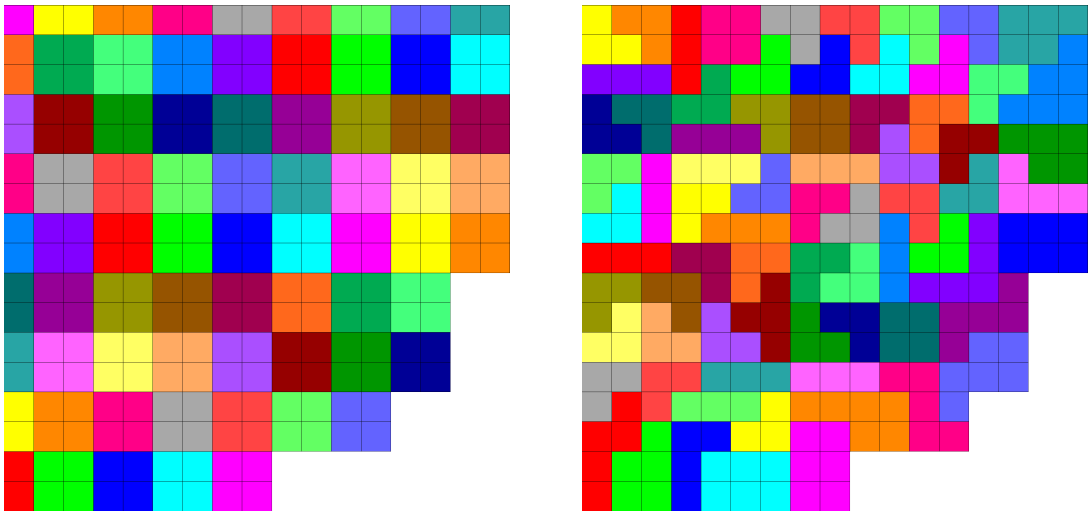


Figure 4.6: VERA progression problem 5a-2d core configuration.

4.5 Results

4.5.1 2-D Results

Results were generated for the planar 2-D version of VERA progression problem 5a [26]. This problem is a quarter core with reflector, barrel, and neutron pad regions surrounding several fuel assemblies, as shown in Fig. 4.6. In the model, there are 257 ray-tracing modules in total, which provides the upper bound for the number of domains. Each subdomain is assigned to a single processor, with a maximum of 36 processors (subdomains) per computational node. Each of the graph decomposition methods was applied to the geometry of this problem, and MPACT was run for each case without applying refinement methods. The assembly-based decomposition was run for all possible numbers of subdomains: 1, 4, 16, 73, and 257. The possible numbers of subdomains are limited by powers of 4 (8 in 3-D), until subdomains would be located entirely outside the core shape. It is also possible to decompose into the fuel assemblies or ray-tracing modules, in this case 73 and 257 respectively. Example decompositions for 73 subdomains are shown in Fig. 4.7.



(a) Assembly-based decomposition method. (b) REB partitioning method.

Figure 4.7: Example decompositions for 73 subdomains for VERA progression problem 5a-2d. Each color represents a different subdomain.

4.5.1.1 Load-Balance

In MPACT, each spatial subdomain is simulated concurrently. After each iteration, parallel boundary conditions are communicated and updated in parallel. The time for solve routines is measured for each subdomain, as is the MoC communication time. However, the communication time includes time spent waiting for other subdomains to finish computation; that is, blocking communication.

This parallel iteration scheme means that the wall-time of each iteration is controlled by the subdomain with the longest run-time; this is expected to be the subdomain with the largest number of cells. However, the wall-time is not the only important consideration; it is important to consider how well the computational resources are used. A measure for how well computational resources are used is the parallel efficiency, as defined by

$$E \equiv \frac{T_s}{N \cdot T}, \tag{4.4}$$

where T_s is the time in serial, and T is the time in parallel with N processes.

If runtime is highly correlated with the largest number of cells in a subdomain, then this efficiency is expected to be related to the largest *fraction* of cells in a subdomain. Specifically, the parallel efficiency is expected to be inversely proportional to the ratio of the largest fraction of cells to the optimal fraction of cells. In an ideally balanced decomposition, each subdomain would have $1/N$ cell fraction. By comparing the maximum cell fraction to this optimal value, one can estimate how much longer the simulation will take compared to an ideal decomposition, neglecting serial code sections and overhead.

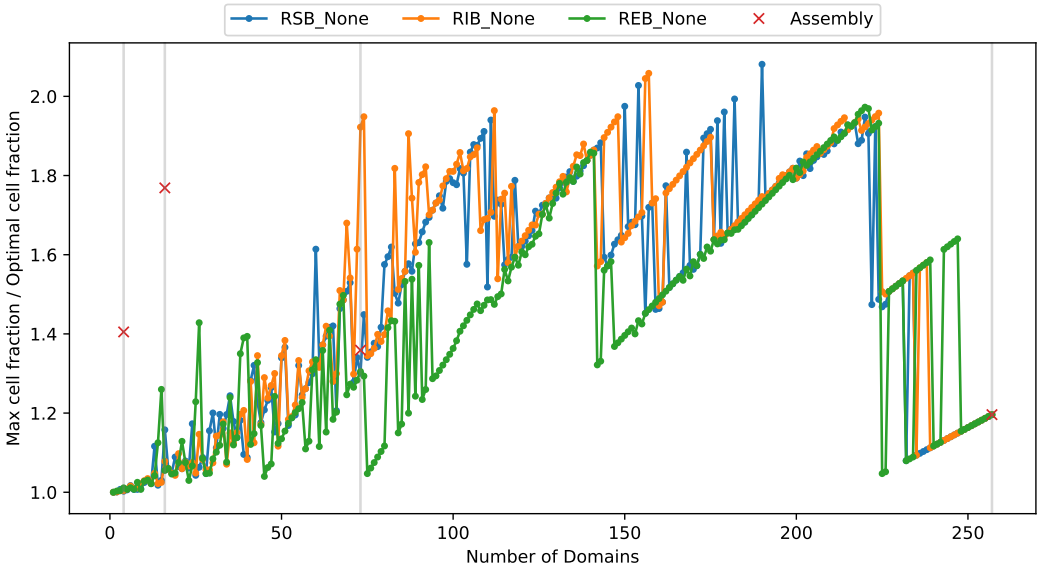


Figure 4.8: Ratio of largest fraction of cells to the optimal fraction of cells as a function of number of subdomains for each partitioning method without refinement.

Higher parallel efficiency indicates better utilization of the available computational resources, and lower runtime. The expectation is that higher parallel efficiency can be obtained by having the largest cell fraction closer to the optimal cell fraction. As seen in Fig. 4.8, the REB method is expected to have slightly better parallel efficiency than the other methods for many cases. It is also expected that for a low number of subdomains, the assembly-based decomposition will result in significantly lower parallel efficiency. However, for large numbers of subdomains, the assembly-based decomposition is expected to result in comparable parallel efficiency to the graph partitioning methods.

4.5.1.2 Communication

In MPACT, parallel boundary conditions are communicated concurrently. However, the measurement of communication time includes any time spent waiting for other subdomains to complete their calculations. Generally, the time spent sending or receiving parallel boundary conditions is relatively small. This time is expected to increase with the weight of edges cut by parallel boundaries.

Figure 4.9 shows that the number of edges cut increases as the number of domains increases. This indicates that time sending and receiving parallel boundary conditions is expected to increase with the number of subdomains. As the number of subdomains increases, the time spent sending and receiving parallel boundary conditions is expected to increase as more data is communicated. However, the time difference between the slowest and fastest subdomains decreases, so the overall communication time measurement is expected to decrease. Because subdomains in the assembly-

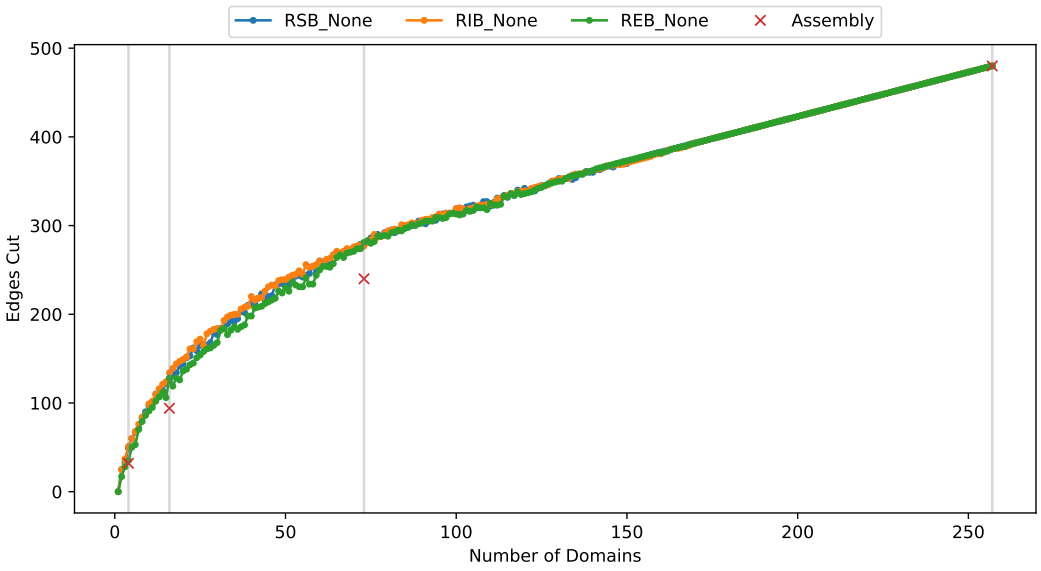


Figure 4.9: Number of edges cut as a fraction of number of domains for each partitioning method without refinement.

based decomposition are rectangular, the number of edges cut is typically lower than in the graph partitioning methods.

4.5.1.3 MPACT Results

As expected, the total and MoC run-times are highly correlated with the largest fraction of cells in any subdomain, as shown in Fig. 4.10. Both total and MoC run-times are very highly correlated with the largest fraction of cells in a subdomain, indicating that this metric can be used to estimate the relative run-times of decompositions. The assembly-based decomposition method was not used in this correlation, as there are only a few data points available.

Utilization of computational resources is also an important aspect for a high performance simulation code; this can be measured by the parallel efficiency. The total parallel efficiency is shown in Fig. 4.11 for each decomposition method. As the core becomes more decomposed, the parallel efficiency drops off rapidly, approaching around 20%. Generally, the graph partitioning methods result in similar parallel efficiency, though the REB method appears to give very slightly higher efficiency in many cases. The assembly-based decomposition method has significantly lower parallel efficiency when there are few subdomains. However, for the moderately decomposed problem (73 subdomains) the assembly-based decomposition method results in significantly higher parallel efficiency. This was not initially expected, as the largest subdomain has a cell fraction similar to that of the graph partitioning methods.

As parallel boundary conditions have a more significant effect on the solution within each subdo-

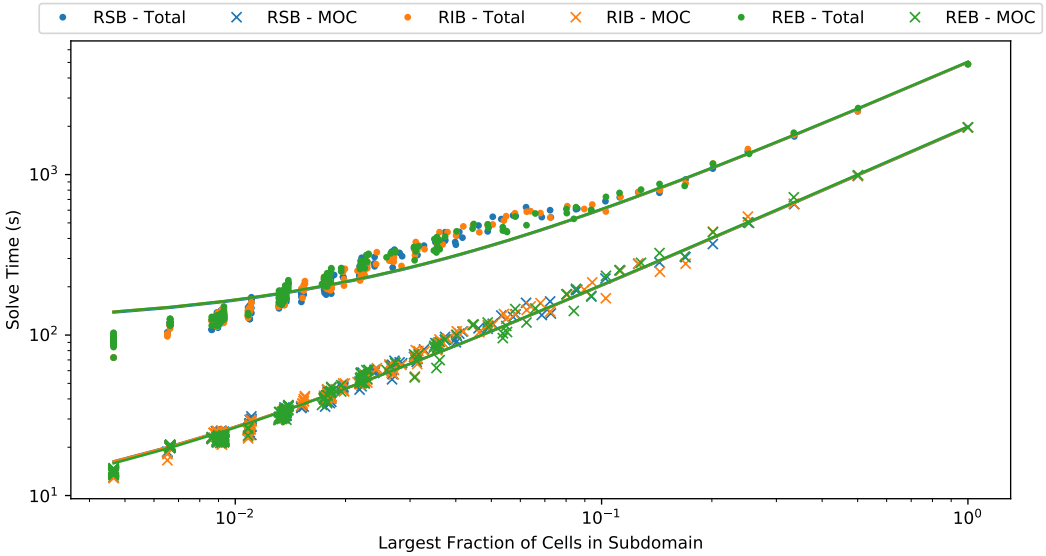


Figure 4.10: Correlation of total and MoC run-times to the largest fraction of cells in a subdomain for each partitioning method.

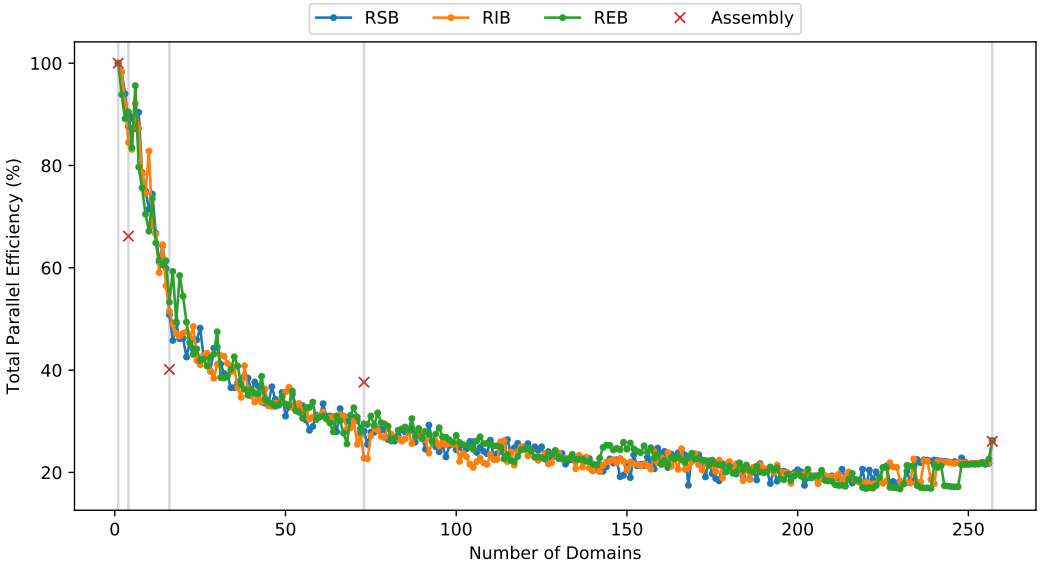


Figure 4.11: The total parallel efficiency for each partitioning method as a function of the number of domains.

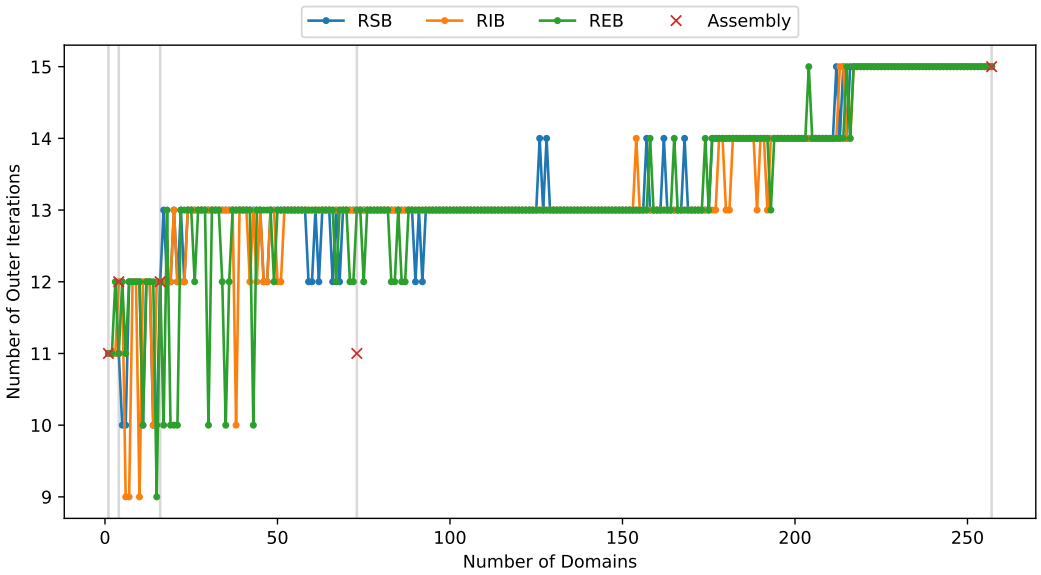


Figure 4.12: The number of iterations used by each decomposition method as a function of the number of subdomains.

main, the convergence rate decreases. This occurs as subdomains become smaller (geometrically), or as they become more "jagged." These jagged parallel boundaries cause re-entrant rays, in which a single ray in the MoC will re-enter the subdomain after leaving. These re-entrant rays will *not* occur in the assembly-based decomposition, because subdomains are forced to be rectangular. This can be observed in Fig. 4.7. The number of MoC iterations required for convergence in each case is shown in Fig. 4.12. There is not a significant increase in the number of outer iterations as the subdomains become smaller, this is consistent with previous results in parallel accelerated transport calculations [27–29]. The increase in number of iterations would be expected to be much more significant in an unaccelerated transport calculation. It may be possible to consider spectral information, such as subdomain optical thickness or scattering ratios, during decomposition to allow for a smaller increase in required iterations; however, this will only have an effect at moderately decomposed problems.

By examining the parallel efficiency of *runtime per iteration*, these spectral effects and the scaling of the solvers in parallel can be determined. As shown in Fig. 4.13, the total parallel efficiency per iteration decreases as the core becomes more decomposed, limiting toward 25%. If only the MoC solver time is being examined, the parallel efficiency per iteration decreases at a much slower rate, as shown in Fig. 4.14. This indicates that the MoC solver in MPACT is highly efficient in parallel, and that other components of MPACT are the bottleneck in parallel simulations. Furthermore, for both total and MoC run-times, the graph partitioning methods give comparable parallel efficiencies. The assembly-based decomposition method still results in lower efficiency

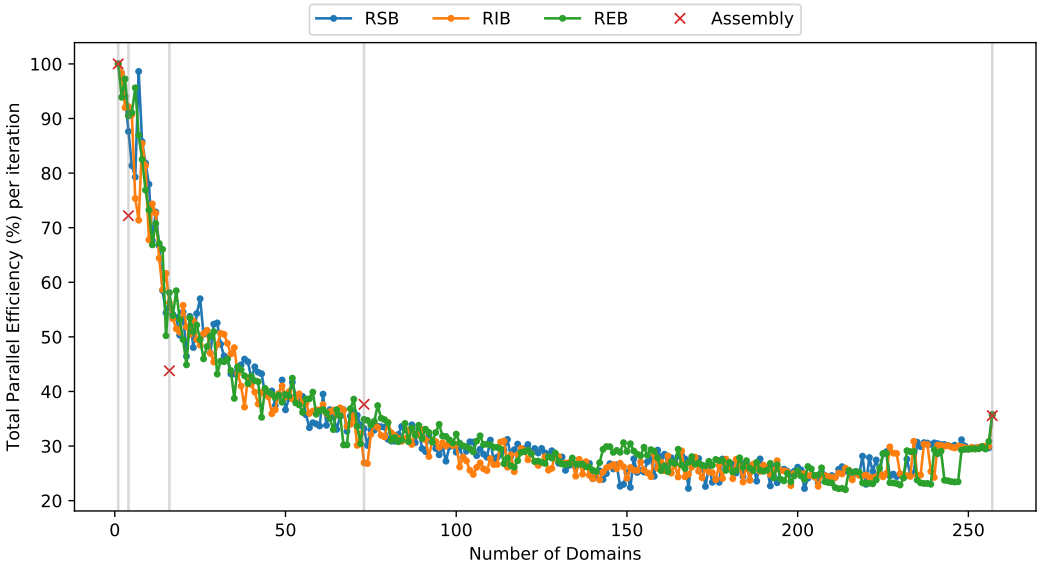


Figure 4.13: The total parallel efficiency per iteration for each partitioning method as a function of the number of domains.

when using few subdomains, but for high numbers of subdomains, it is comparable with the graph partitioning methods.

Finally, the ratio of the optimal cell fraction to the maximum cell fraction is expected to be proportional to the parallel efficiency per iteration of the MoC solver. As shown in Fig. 4.15, the parallel efficiency is correlated with the the ratio of optimal-to-maximum cell fractions, though it is not correlated as strongly as the runtime with the maximum cell fraction. This indicates that this ratio can be used to estimate the parallel efficiency of the MoC solver for a decomposition. Even isolating the effect due to increased numbers of iterations, there is a significant spread in parallel efficiency as a function of the optimal to maximum cell fraction ratio; this can be explained by the fact that significantly different numbers of domains (processes) can result in similar fractions. This is clearly observed in Fig. 4.8.

4.5.1.4 CMFD Acceleration

In MPACT, the two main solvers contributing to runtime of a steady-state eigenvalue calculation are the MoC solver and CMFD acceleration, each of which is parallelized using the same computational resources. From Fig. 4.16, the MoC and CMFD solvers take similar amounts of time in serial; however, as more subdomains are used, the fractional runtime of CMFD increases to almost 70% of the total. This may be partially alleviated by using a multigrid method [30] to more efficiently solve the CMFD system. This indicates that the parallel efficiency is quite low for MPACT's CMFD linear system solvers, which heavily leverage PETSc [24] for parallelism.

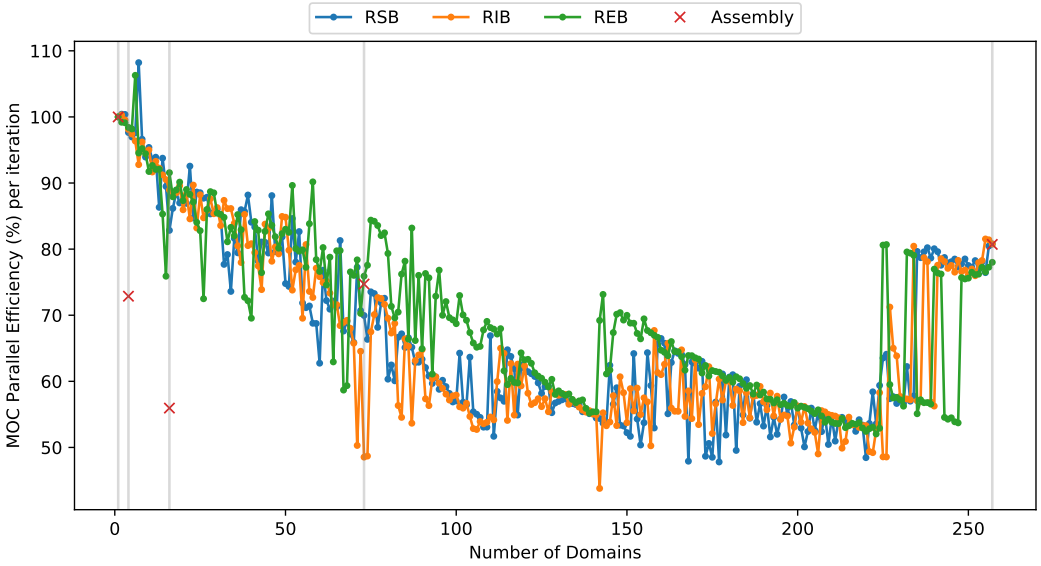


Figure 4.14: The MoC parallel efficiency per iteration for each partitioning method as a function of the number of domains.

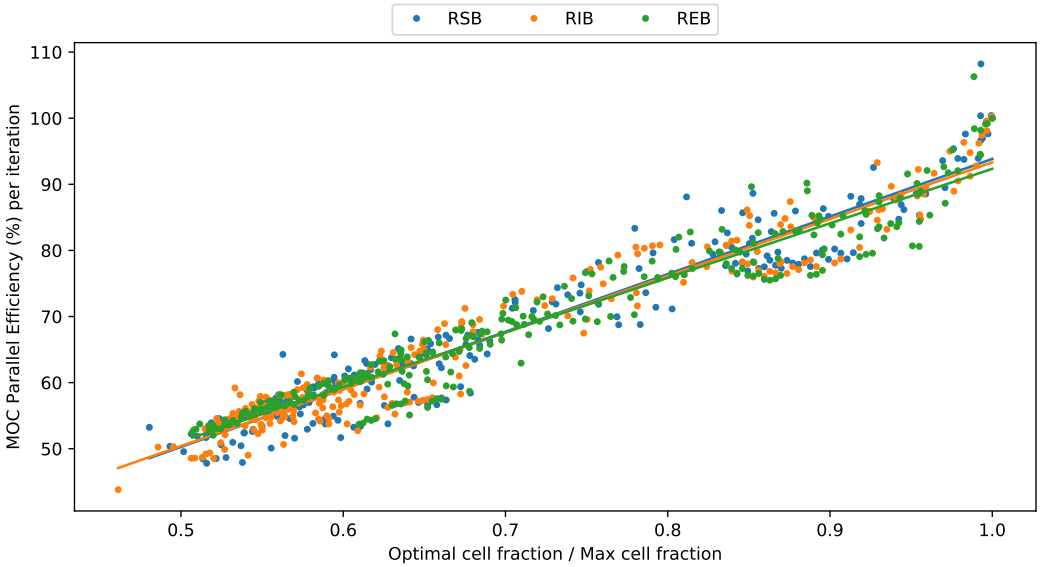


Figure 4.15: Correlation of the MoC parallel efficiency per iteration and the ratio of optimal and maximum cell fractions for each of the partitioning methods.

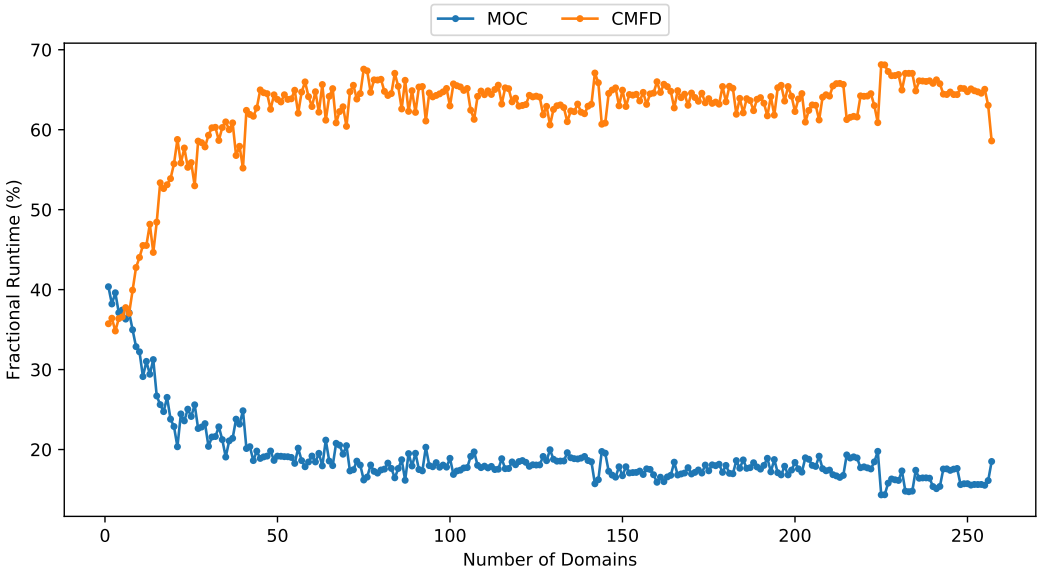


Figure 4.16: Fractional runtime of the MoC solver and CMFD acceleration method in MPACT for varying number of domains with the REB partitioning method.

Similar as the number of outer transport iterations, as subdomains become smaller, it is expected the CMFD linear system will require more iterations for convergence; this is shown for the REB partitioning method in Fig. 4.17. This is because the pre-conditioner used in MPACT is less efficient with more domains. However, by considering the parallel efficiency per inner iteration, these convergence effects can be eliminated, and the parallel scaling of the linear system solvers can be examined. Figure 4.17 shows that the parallel efficiency of the linear system solvers used in MPACT for CMFD calculations is quite low, limiting to around 30%. By using a linear system solver that has better parallel scaling [31], the overall parallel efficiency of MPACT may be increased. However, these results also indicate that the parallel efficiency is, in part, lowered by spectral effects, that will not be eliminated by a more efficient parallel linear solver.

4.5.2 3-D Results

As shown in Section 4.5.1, decomposition metrics can be used to estimate the runtime and parallel efficiency without needing to run the simulations. There are different approaches to decomposition in 3-D; these are discussed in more detail in Section 4.4. Decompositions were performed without refinement for three different 3-D decomposition schemes: axially and radially aligned (ARA), radially aligned (RA), and unrestricted (UR). Given fewer restrictions, the resulting decompositions were expected to be more balanced. Decompositions were performed on VERA progression problem 5a-0 in 3-D [26] with 58 axial planes, but the simulations for this problem were not run.

The ratio of maximum cell fraction to optimal cell fraction can easily be converted to the

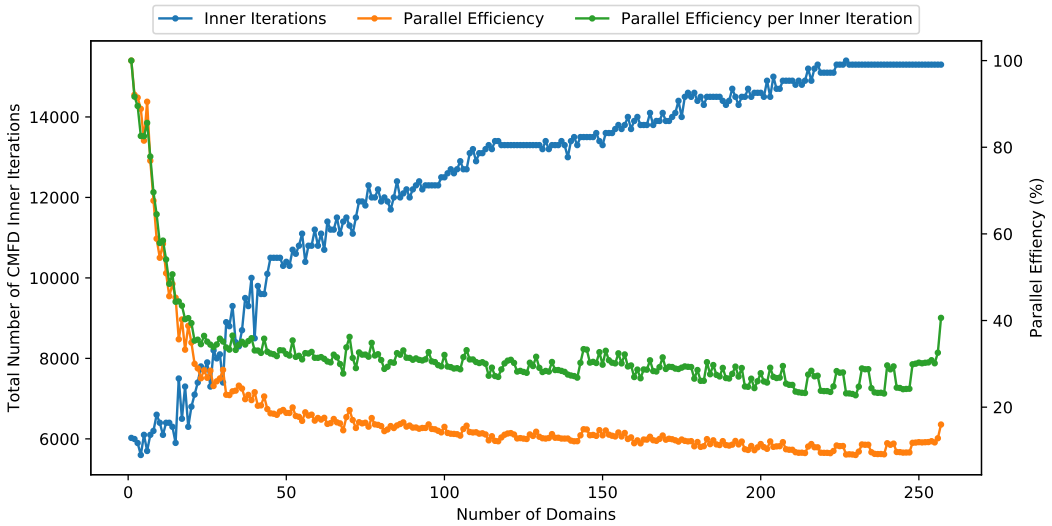


Figure 4.17: The total number of CMFD inner iterations, parallel efficiency, and parallel efficiency per inner iteration for varying number of domains with the REB partitioning method.

maximum cell fraction by dividing by N. For brevity, only this load balance metric is shown herein. The resulting decompositions from the ARA approach are very similar to those in the 2-D case. Just as in the 2-D case, the maximum-to-optimal cell fraction ratio is lowest for the REB method as compared to the other partitioning methods for many cases, as seen in Fig. 4.18. This indicates that, barring any differences in the number of iterations, the REB method is expected to have slightly higher parallel efficiencies.

In the RA scheme, a separate decomposition is performed for each axial plane, with an appropriate number of subdomains based on the number of cells in the plane. Unlike in the 2-D case, the REB method seems to perform significantly worse than the other two methods for low numbers of subdomains. For highly decomposed cores, the REB method seems to perform slightly better than the other partitioning methods. Additionally, by comparing the magnitude of the ratios in Fig. 4.19 and Fig. 4.18, it is clear that the RA approach typically has less imbalance.

Finally, the UR approach decomposes the 3-D core by directly abstracting the entire core into a graph. The REB method is significantly more imbalanced than other methods for lower numbers of subdomains; however, for highly decomposed cores, the REB method outperforms the other methods, as shown in Fig. 4.20. Additionally, for highly decomposed problems, both the RSB and RIB methods seem to have worse balance when using the UR scheme compared to the RA scheme.

The approach currently used in MPACT is the axially and radially aligned 3-D decomposition scheme. If other approaches were to be used, the implementation of parallel communication in the 2D-1D method would need to be made more general. To justify these changes, the less restricted approaches would need to offer significant advantages over the current approach. Figures 4.21

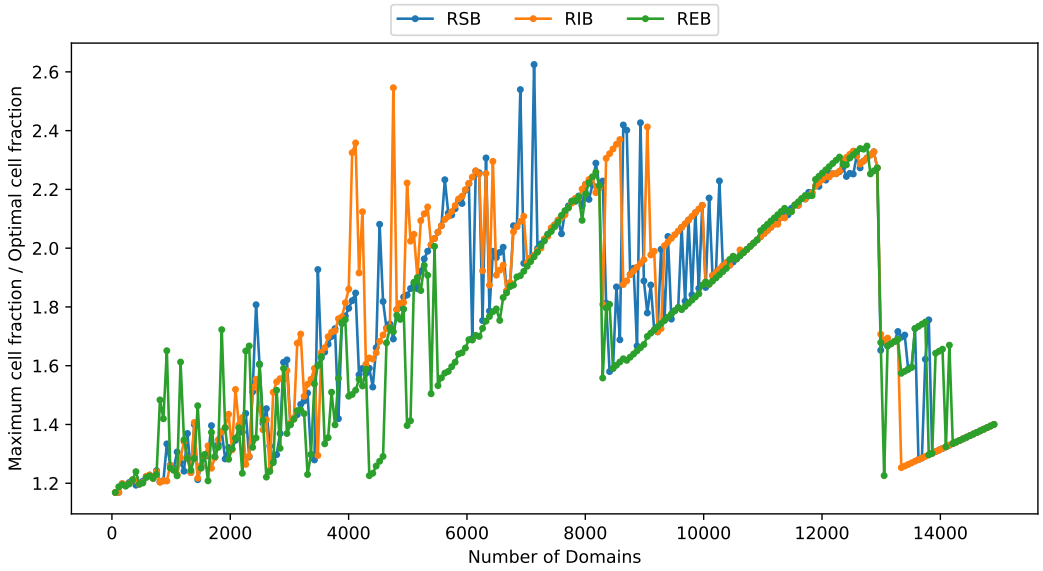


Figure 4.18: Maximum-to-optimal cell fraction ratio for each partitioning method as a function of number of domains in the axially and radially aligned (ARA) scheme.

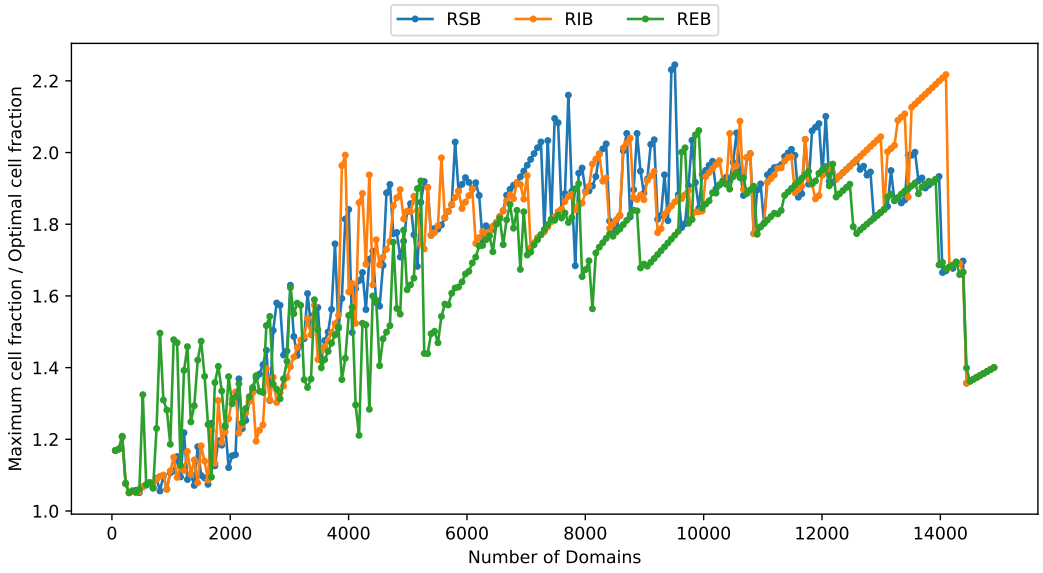


Figure 4.19: Maximum-to-optimal cell fraction ratio for each partitioning method as a function of number of domains in the radially aligned (RA) scheme.

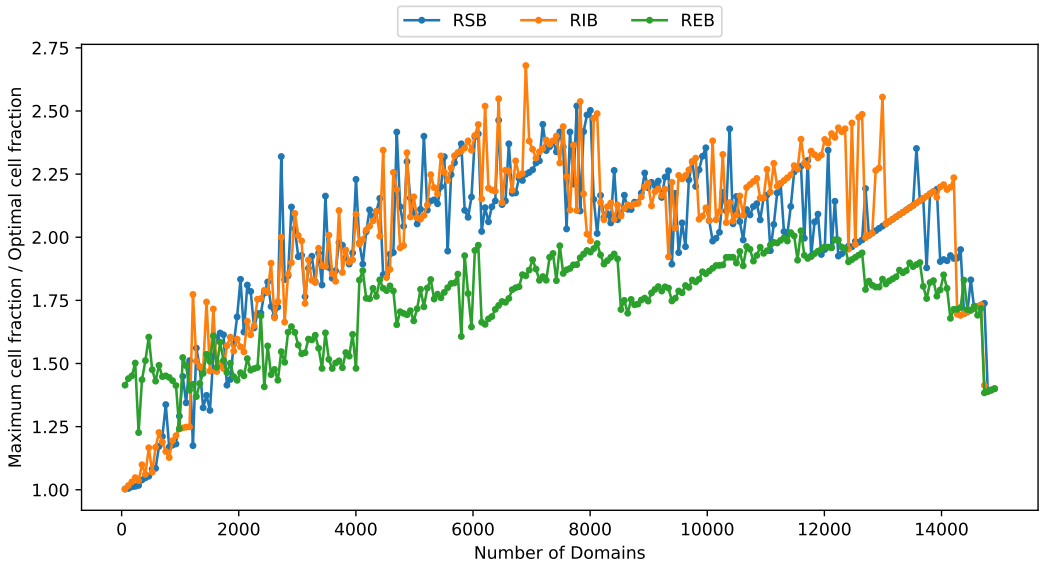


Figure 4.20: Maximum-to-optimal cell fraction ratio for each partitioning method as a function of number of domains in the unrestricted (UR) scheme.

to 4.23 examine the maximum cell fraction of each scheme relative to the current scheme. The smaller the relative maximum cell fraction, the lower the expected runtime will be.

For most applications, reactor cores in MPACT are not highly decomposed. At maximum, on the order of 30 subdomains per axial plane are unrestricted. In this context, modestly decomposed cores are defined as those with fewer than 2,000 subdomains; otherwise, the core is considered highly decomposed. For the RSB and RIB methods, the RA approach is expected to give 10% better performance than the ARA approach on average. For highly decomposed cases, these partitioning methods are only expected to give an average of 2 – 3% better performance. On average, the UR approach is expected to give *worse* performance by more than 10% using these partitioning methods. However, for the REB partitioning method, the RA approach is only expected to give 3% better performance on average. Typically, the UR approach is still expected to result in worse performance. This may be a result of the problem examined in this work which had axial planes that were fairly well balanced. By giving the graph partitioning methods more degrees of freedom the heuristic graph partitioning methods perform worse overall. Multi-level partitioning methods reduce the degrees of freedom during coarsening and may be more appropriate in these cases.

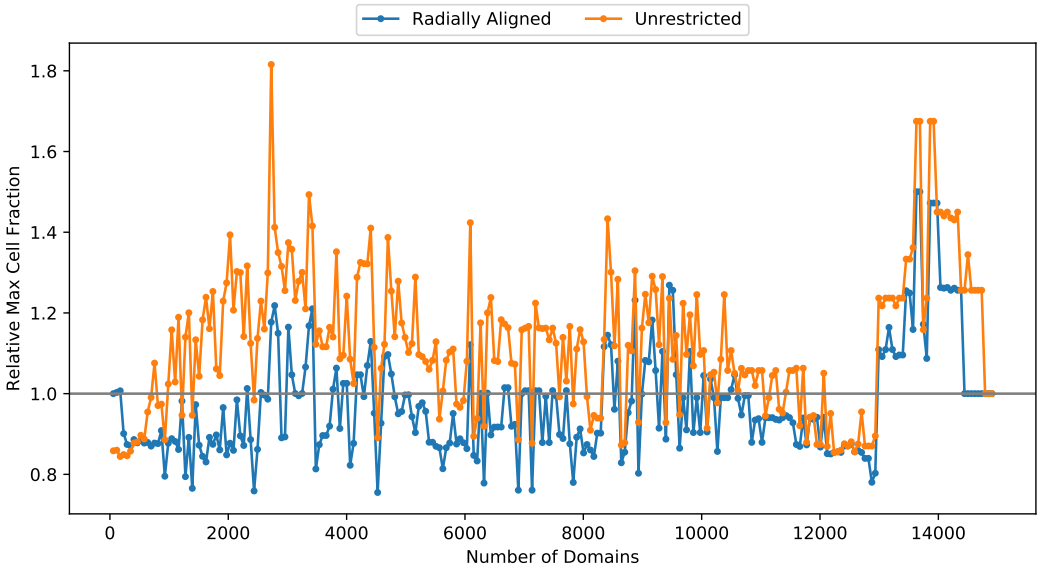


Figure 4.21: Maximum cell fraction relative to the axially and radially aligned (ARA) approach for the RSB partitioning method.

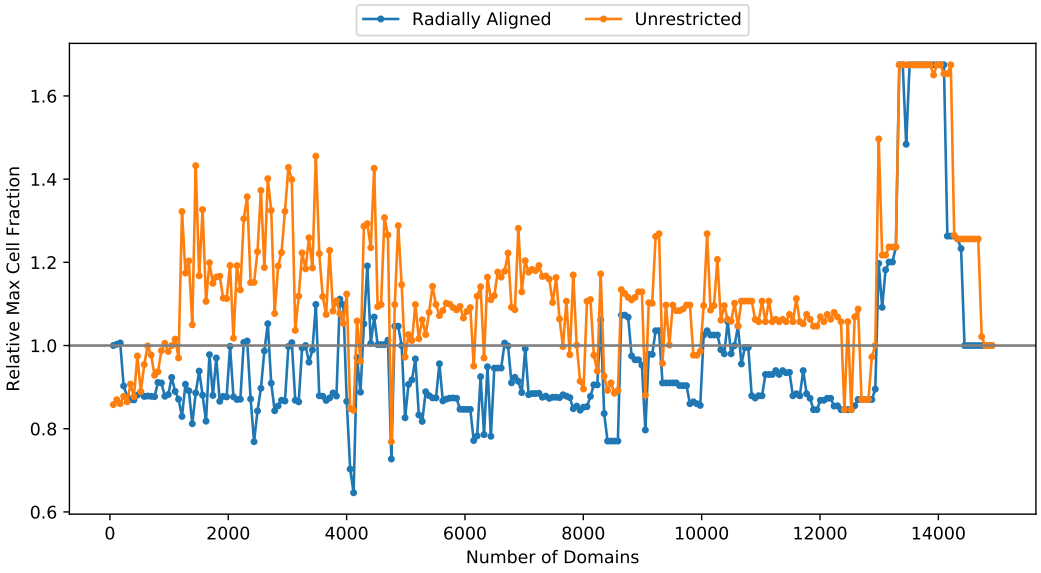


Figure 4.22: Maximum cell fraction relative to the axially and radially aligned (ARA) approach for the RIB partitioning method.

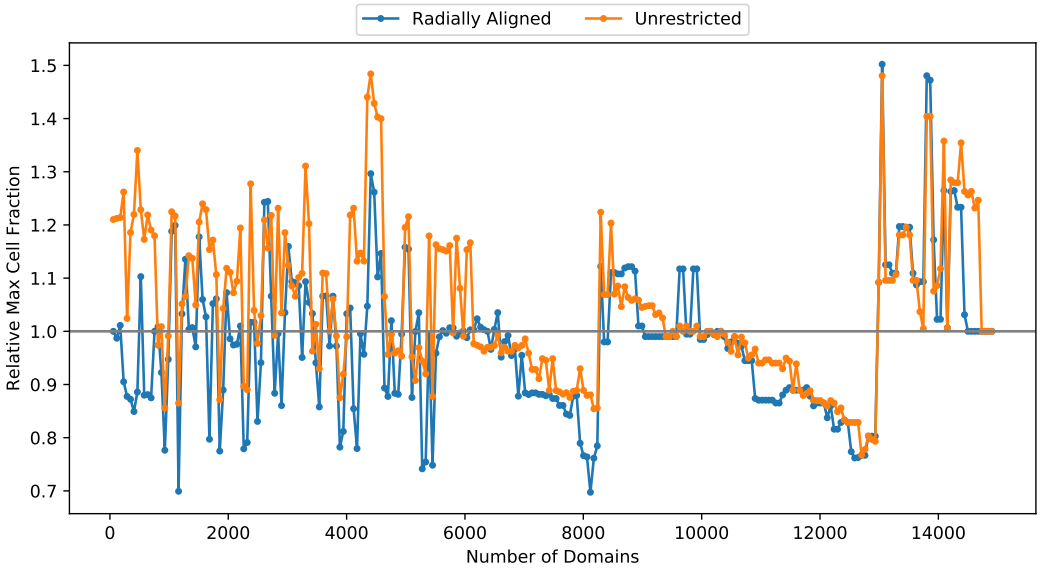


Figure 4.23: Maximum cell fraction relative to the axially and radially aligned (ARA) approach for the REB partitioning method.

4.6 Partition Refinement

4.6.1 Partition Refinement Methods

The Kernighan-Lin algorithm [32] is often described as one of the earliest developed graph partitioning algorithms; however, the algorithm does not actually create a partitioning of the graph, it improves, or refines, the quality by reducing the number of edges cut between existing partitions [6]. Therefore, in this work, this method and a modified version of it are called *refinement methods*.

As suggested by Pothen [15], the RSB method or other partitioning methods can create a high quality initial partitioning to use in the Kernighan-Lin algorithm. Significant improvements have been made to the efficiency of the original Kernighan-Lin algorithm [33]; however, the graphs of concern in this work are relatively small, and graph decomposition time is negligible when compared to the overall simulation runtime. Two partition refinement methods were examined: the Kernighan-Lin algorithm [32] and a modification to the Kernighan-Lin algorithm which takes some geometric information of the graph into account [5].

The investigated partition refinement algorithms reduce the weight of edges cut between two partitions by swapping vertex pairs between the partitions iteratively. The original Kernighan-Lin algorithm operates entirely on the connectivity of the graph, while the modified Spatial Kernighan-Lin algorithm uses both connectivity and geometric information from the graph.

For each vertex in the graph, D is defined as

$$D_i \equiv E_{E,i} - E_{I,i}, \tag{4.5}$$

where $E_{E,i}$ is the sum of edge weights from vertex i connecting with vertices outside the partition containing vertex i, and $E_{I,i}$ is the sum of edge weights from vertex i connecting with vertices within the partition containing vertex i. The reduction in communication or "gain," from swapping a pair of vertices (a, b) is defined as

$$g_{(a,b)} \equiv D_a + D_b - 2c_{a,b}. \tag{4.6}$$

The Kernighan-Lin algorithm, given in Algorithm 7, is a greedy algorithm, in that it will swap a pair (a, b) with maximal g at the current step. The idea is that by doing this iteratively, the algorithm will lead to a minimized cut-size; in reality, the algorithm will often get stuck in local minima that do not have a global minimized cut-size. Additionally, there may be multiple pairs (a, b) with the same maximal gain value: the algorithm will only consider one of these pairs.

The Spatial Kernighan-Lin algorithm, given in Algorithm 8, is an adaptation of the Kernighan-Lin algorithm that accounts for the multiple pairs with maximal gain. This algorithm was developed as part of this work, in an attempt to improve the original Kernighan-Lin algorithm for this specific application. This algorithm prioritizes vertex pairs which are geometrically distant from one another. The idea behind this modification is that to minimize the edge-cut, the cut should be as straight as possible. By prioritizing distant vertex pairs, the bisector is typically "straightened out"; this process can be likened to pulling on the ends of a string in order to straighten it.

4.6.2 Partition Refinement Results

In Section 4.6.1, refinement methods are introduced as a method for further reducing communication. Figure 4.24 shows that both refinement methods are able to slightly reduce the number of edges cut compared to the cases without refinement for RSB. Similar trends are observed for the other partitioning methods. While these refinement methods offer a slight reduction in communication, the parallel efficiency due to load imbalance may be negatively affected by applying refinement, as shown in Fig. 4.25. Communication is not expected to have as significant of an effect as load imbalance, so for the simulations in MPACT, partitioning methods were used without refinement.

Algorithm 7 Kernighan-Lin Algorithm, with input graph $G(V, E)$, and vertex sets A and B within the graph.

procedure KERNIGHAN-LIN($G(V, E), A, B$)
  $g_m = 1$
  while $g_m > 0$ **do**
    $W_A = \sum_{i \in A} w_i$
    $W_B = \sum_{i \in B} w_i$
    Compute $D \; \forall \; V$ (Equation (4.5))
    Let a_v, b_v, g_v be empty sets
    for $n = 1$ to $N/2$ **do**
      Find unmarked pair (a, b) such that:
        1. $a \in A$ and $b \in B$
        2. g is maximized (Equation (4.6))
      $\widehat{W}_A = W_A + w_b - w_a$
      $\widehat{W}_B = W_B + w_a - w_b$
      if MAX(W_A, W_B) $\geq$ MAX($\widehat{W}_A, \widehat{W}_B$) **then**
        Append a to a_v, b to b_v, and g to g_v
        Update D values as if a, b have been swapped
        $W_A = \widehat{W}_A$
        $W_A = \widehat{W}_B$
      else
        End search
      end if
    end for
    Find k maximizing $g_m = \sum_{i=1}^{k} g_v(i)$
    if $g_m > 0$ **then**
      Exchange vertices in $a_v(1 : k)$ and $b_v(1 : k)$
    end if
  end while
end procedure

Algorithm 8 Spatial Kernighan-Lin Algorithm, with input graph $G(V, E)$, and vertex sets A and B within the graph.

procedure SPATIAL KERNIGHAN-LIN($G(V, E), A, B$)
  $g_m = 1$
  while $g_m > 0$ **do**
    $W_A = \sum_{i \in A} w_i$
    $W_B = \sum_{i \in B} w_i$
    Compute $D \; \forall \; V$ (Equation (4.5))
    Let a_v, b_v, g_v be empty sets
    for $n = 1$ to $N/2$ **do**
      Allow (f_a, f_b) to be sets from A, B satisfying:

1. $a \in A$, $b \in B$
2. g is maximized (Equation (4.6))
3. a and b are on the boundary between A and B

      Find pair (f'_a, f'_b) such that distance is maximized
      if No pair found **then**
        Search using standard Kernighan-Lin rules
      end if
      $\widehat{W}_A = W_A + w_b - w_a$
      $\widehat{W}_B = W_B + w_a - w_b$
      if MAX(W_A, W_B) $\geq$ MAX($\widehat{W}_A, \widehat{W}_B$) **then**
        Append a to a_v, b to b_v, and g to g_v
        Update D values as if a, b have been swapped
        $W_A = \widehat{W}_A$
        $W_A = \widehat{W}_B$
      else
        End search
      end if
    end for
    Find k maximizing $g_m = \sum_{i=1}^{k} g_v(i)$
    if $g_m > 0$ **then**
      Exchange vertices in $a_v(1:k)$ and $b_v(1:k)$
    end if
  end while
end procedure

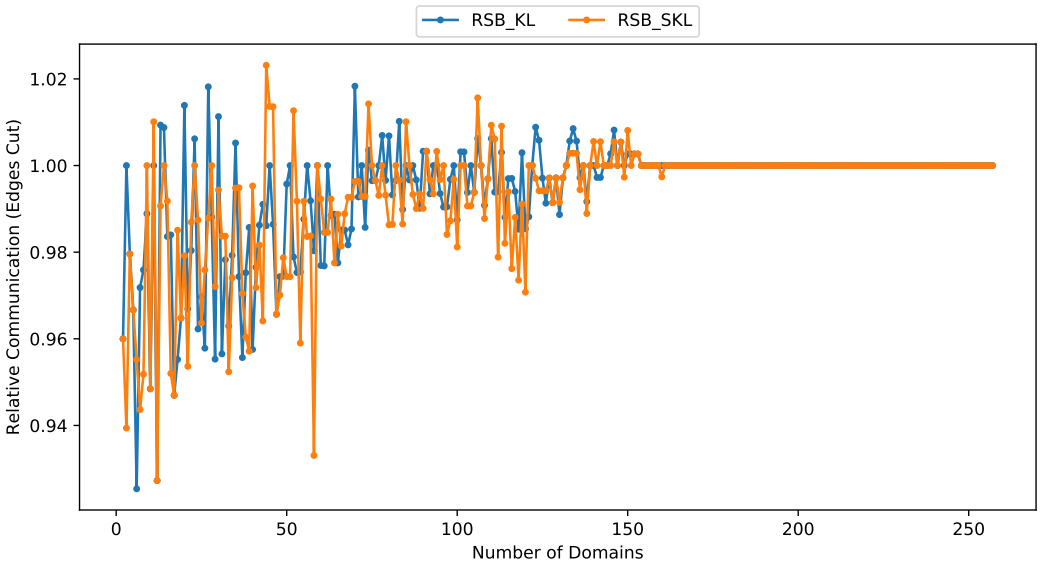


Figure 4.24: Communication relative to the RSB method without refinement as a function of number of domains for each refinement method using the RSB partitioning method.

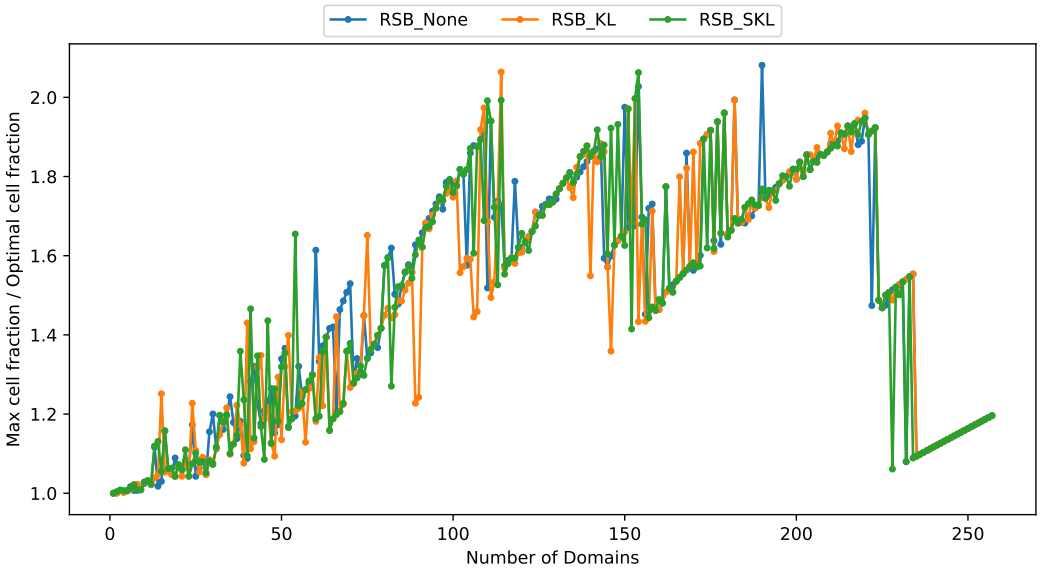


Figure 4.25: Ratio of maximum cell fraction to optimal cell fraction for the RSB partitioning method with each refinement method.

4.7 Conclusions

Spatial decomposition is a useful technique for reducing the runtime of simulations and is necessary to run whole-core high fidelity reactor calculations. Using graph partitioning methods to decompose the spatial domain of a core has significant advantages when compared to previous decomposition methods. Graph partitioning allows for the usage of an arbitrary number of spatial subdomains, and it generalizes to different module geometries such as a hexagonal lattice. Graph partitioning methods generally provide high quality decompositions that increase parallel efficiency. These automated spatial decomposition methods improve code usability and flexibility by allowing users to easily fit simulations to any number of processors.

However, for highly decomposed cores, the convergence rate decreases due to jagged subdomain boundaries. This caused graph partitioning methods to significantly reduce runtime for problems that were not highly decomposed but actually increase runtime for highly decomposed problems. Additionally, the graph partitioning methods allow for an arbitrary number of domains to be used, unlike the previous method, giving the user significantly more options. This is because the assembly-based decomposition method used rectangular (non-jagged) subdomains. This indicates that it may be advantageous to create a high quality decomposition method that enforces rectangular subdomains. However, this approach will not generalize to other lattice types, such as hexagonal lattices.

In the current MPACT implementation, there is no significant difference in run-times when using any of the three partitioning methods discussed, although REB typically results in slightly lower MoC run-times. The 2-D results indicate that the maximum fraction of cells in a subdomain is highly correlated with the runtime of the simulation. Furthermore, 2-D results indicate that the parallel efficiency of MoC is highly correlated with the ratio of the optimal cell fraction per subdomain to the maximum cell fraction in a subdomain.

In 3-D, MPACT currently requires spatial domains to be axially and radially aligned, for the 2D/1D method. If these restrictions are lifted, then other approaches can be used to perform the 3-D spatial decomposition. Three 3-D decomposition approaches were investigated in this work: radial and axial aligned subdomains, radially aligned subdomains, and an approach with no alignment restrictions. The radially aligned approach is expected to out perform the current approach by an average of 10% for typical cases. However, the unrestricted approach is actually expected to perform worse than the current approach on average because the problem is well balanced axially. This analysis was performed on a 3-D core, which was relatively homogeneous in the axial direction; if a core design were to be more axially heterogeneous, then a more significant increase in performance might be expected.

The parallel efficiency is a measure of how well computational resources are utilized in parallel

applications. MPACT's overall parallel efficiency decreases rapidly as more domains are used due to two factors: increased number of iterations, and the inefficiency of the parallel CMFD solve. For the 2-D VERA progression problem 5a, the overall parallel efficiency of MPACT dropped to nearly 20%. The parallel efficiency of only the MoC computations in MPACT drops to between 40–60%, while the parallel efficiency of the CMFD computation drops to $<20\%$. This causes the CMFD computation to dominate runtime in spatially decomposed cases, and motivates the implementation of a more efficient parallel linear system operator in MPACT rather than using a third-party library.

The graph-based partitioning methods provide significant benefits in the quality of spatial decomposition (compared to the previous method), and user experience. For a realistic 2-D quarter core calculation with a typical number of subdomains, the graph based methods were able to increase total parallel efficiency by more than 20%, and MoC efficiency by 35%. The graph-based methods also allow for the user to use any number of subdomains, rather than a small subset of choices with the previous method. The methods also require no additional input from the user.

Bibliography

[1] J.R. Askew. *A Characteristics Formulation of the Neutron Transport Equation in Complicated Geometries*. 1972.

[2] MPACT Team. *MPACT Theory Manual v2.2.0*. Tech. rep. Consortium for Advanced Simulation of Light Water Reactors, 2016.

[3] Andrew P Fitzgerald et al. "Spatial decomposition of structured grids for nuclear reactor simulations". In: *Annals of Nuclear Energy* 132 (2019), pp. 686–701. ISSN: 0306-4549. DOI: 10.1016/j.anucene.2019.06.054.

[4] Geoffrey Alexander Gunow. "Full Core 3D Neutron Transport Simulation Using the Method of Characteristics with Linear Sources". Doctoral. Massachusetts Institute of Technology, 2018.

[5] Andrew Fitzgerald et al. "Automated Decomposition of a Structured Grid". In: *Transactions of the American Nuclear Society*. Vol. 117. Washington D.C., 2017, pp. 731–734.

[6] Ulrich Elsner. *Graph partitioning A Survey*. Tech. rep. Technische Universitat Chemnitz, 1997, p. 52. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.54.2060.

[7] Y.F. Yao and B.E. Richards. "Parallel CFD Computation on Unstructured Grids". In: *Parallel Computational Fluid Dynamics*. Elsevier, 1998, pp. 289–296. DOI: 10.1016/B978-044482849-1/50035-X.

[8] George Karypis and Vipin Kumar. "A fast and high quality multilevel scheme for partitioning irregular graphs". In: *SIAM Journal on Scientific Computing* 20.1 (1998), pp. 359–392.

[9] E. G. Boman et al. "The Zoltan and Isorropia Parallel Toolkits for Combinatorial Scientific Computing: Partitioning, Ordering, and Coloring". In: *Scientific Programming* 20.2 (2012), pp. 129–150.

[10] Abel Marin-Lafleche, Micheal A. Smith, and Changho Lee. "Proteus-MOC: A 3D deterministic solver incorporating 2D method of characteristics". In: *M&C 2013*. Vol. 4. 2013, pp. 2759–2770. ISBN: 9781627486439. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-84883349083{\&}partnerID=tZOtx3y1.

[11] Sebastian Schunert et al. "A flexible nonlinear diffusion acceleration method for the SN transport equations discretized with discontinuous finite elements". In: *Journal of Computational Physics* 338 (2017), pp. 107–136. ISSN: 10902716. DOI: 10.1016/j.jcp.2017.01.070.

[12] Shinya Kosaka and Etsuro Saji. "Transport theory calculation for a heterogeneous multi-assembly problem by characteristics method with direct neutron path linking technique". In: *Journal of Nuclear Science and Technology* 37.12 (2000), pp. 1015–1023. ISSN: 00223131. DOI: 10.1080/18811248.2000.9714987.

[13] Shane Stimpson and Benjamin Collins. "Flexible Spatial Partitions in MPACT Through Module-Based Data Passing". In: *Transactions of the American Nuclear Society*. Vol. 116. San Francisco, 2017, pp. 654–657.

[14] G. M. Morton. *A computer Oriented Geodetic Data Base; and a New Technique in File Sequencing*. Tech. rep. Ottawa, Canada: IBM Ltd., 1966.

[15] Alex Pothen, Horst D. Simon, and Kang-Pu Paul Liu. "Partitioning Sparse Matrices with Eigenvectors of Graphs". In: *SIAM Journal on Matrix Analysis and Applications* 11.3 (1989), pp. 430–452.

[16] H. D. Simon. "Partitioning of unstructured problems for parallel processing". In: *Computing Systems in Engineering* 2.2-3 (1991), pp. 135–148. ISSN: 09560521. DOI: 10.1016/0956-0521(91)90014-V.

[17] Daniel A. Spielman and Shang Hua Teng. "Spectral partitioning works: Planar graphs and finite element meshes". In: *Linear Algebra and Its Applications* 421.2-3 SPEC. ISS. (2007), pp. 284–305. ISSN: 00243795. DOI: 10.1016/j.laa.2006.07.020.

[18] Shang Hsien Hsieh, Glaucio H. Paulino, and John F. Abel. "Recursive spectral algorithms for automatic domain partitioning in parallel finite element analysis". In: *Computer Methods in Applied Mechanics and Engineering* 121.1-4 (1995), pp. 137–162. ISSN: 00457825. DOI: 10.1016/0045-7825(94)00704-Q.

[19] Miroslav Fiedler. "Algebraic Connectivity of Graphs". In: *Czechoslovak Mathematical Journal* 23.2 (1973), pp. 298–305.

[20] N. Floros et al. “Comparative efficiencies of domain decompositions”. In: *Parallel Computing* 21.11 (1995), pp. 1823–1835. ISSN: 01678191. DOI: 10.1016/0167-8191(95)00031-7.

[21] Charbel Farhat. “A simple and efficient automatic fem domain decomposer”. In: *Computers and Structures* 28.5 (1988), pp. 579–602. ISSN: 00457949. DOI: 10.1016/0045-7949(88)90004-1.

[22] M A Nasra and D T Nguyen. “An algorithm for domain decomposition in finite element analysis”. In: *Computers & Structures* 39.3/4 (1991), pp. 277–289.

[23] Kord S. Smith. “Nodal method storage reduction by nonlinear iteration”. In: *Transactions of the American Nuclear Society*. Vol. 44. 1983, pp. 265–266.

[24] S. Balay et al. *PETSc Users Manual Revision 3.10*. Tech. rep. Argonne, IL (United States): Argonne National Laboratory (ANL), 2018.

[25] Benjamin Collins 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. ISSN: 10902716. DOI: 10.1016/j.jcp.2016.08.022.

[26] A. T. Godfrey. *VERA Core Physics Benchmark Progression Problem Specifications*. Tech. rep. 4. 2014, pp. 1–173. URL: https://www.casl.gov/sites/default/files/docs/CASL-U-2012-0131-004.pdf.

[27] B.W. Kelley and E.W. Larsen. “CMFD Acceleration of Spatial Domain-Decomposed Neutron Transport Problems”. In: *Physor*. Knoxville, 2012, pp. 1–13.

[28] Shane G Stimpson et al. “Boundary Acceleration Techniques for CMFD-Accelerated 2D-MOC”. In: Kyoto, 2014, pp. 1–11.

[29] Brendan Kochunas et al. “Application of the SDD-Cmfd Acceleration Technique To Parallel 3-D Method of Characteristics Transport”. In: *Physor*. Kyoto, 2014.

[30] Ben C. Yee. “A Multilevel in Space and Energy Solver for Multigroup Diffusion and Coarse Mesh Finite Difference Eigenvalue Problems”. Doctoral. University of Michigan, 2018.

[31] Chen Hao et al. “3D whole-core neutron transport simulation using 2D/1D method via multi-level generalized equivalence theory based CMFD acceleration”. In: *Annals of Nuclear Energy* 122 (2018), pp. 79–90. ISSN: 18732100. DOI: 10.1016/j.anucene.2018.08.014.

[32] B.W. Kernighan and S. Lin. “An efficient heuristic for partitioning graphs”. In: *Bell Systems Technical Journal* 49 (1970), pp. 291–308.

[33] C. M. Fiduccia and R. M. Mattheyses. *A Linear-Time Heuristic for Improving Network Partitions*. Tech. rep. Schenectady, NY: General Electric Research and Development Center, 1982, pp. 175–181.

CHAPTER 5

Improved Linear Source Formulation for Multi-physics and 2D/1D Applications

The studies performed in this thesis have made extensive use of the linear-source approximation (LSA) introduced by Ferrer and Rhodes [1]. Through the use of the approximation, as presented in the original work, instabilities (due to implementation) and inefficiencies were found. This chapter aims to present two improvements made to this approximation that are a focus of this research: improved exponential tabulation [2], and an improved formulation for multi-physics and 2D/1D applications [3].

5.1 Exponential Tabulation

The original moment-based LSMoC formulation [1], detailed in Section 3.3, uses several exponential functions:

$$F_1(\tau_m^g) \equiv 1 - e^{\tau_m^g}, \tag{5.1a}$$

$$F_2(\tau_m^g) \equiv 2\left[\tau_m^g - F_1(\tau_m^g)\right] - \tau_{mki}^g F_1(\tau_m^g), \tag{5.1b}$$

$$G_1(\tau_{mki}^g) \equiv 1 + \frac{\tau_{mki}^g}{2} - \left(1 + \frac{1}{\tau_{mki}^g}\right) F_1(\tau_{mki}^g), \tag{5.1c}$$

$$G_2(\tau_{mki}^g) \equiv \frac{2}{3}\tau_{mki}^g - \left(1 + \frac{2}{\tau_{mki}^g}\right) G_1(\tau_{mki}^g), \tag{5.1d}$$

and

$$H(\tau_{mki}^g) \equiv \frac{\tau_{mki}^g}{2} - G_1(\tau_{mki}^g), \tag{5.1e}$$

where τ_{mki}^g is the total optical thickness of a segment, and τ_m^g is the variable optical thickness at any point along the track-segment. Although the functions $F_1(\tau_m^g)$, and $F_2(\tau_m^g)$ are functions of variable τ_m^g, in the implementation of this method in the code they are only ever evaluated over the total segment optical thickness, τ_{mki}^g. The functions, $G_1(\tau_{mki}^g)$, $G_2(\tau_{mki}^g)$, and $H(\tau_{mki}^g)$ all require special

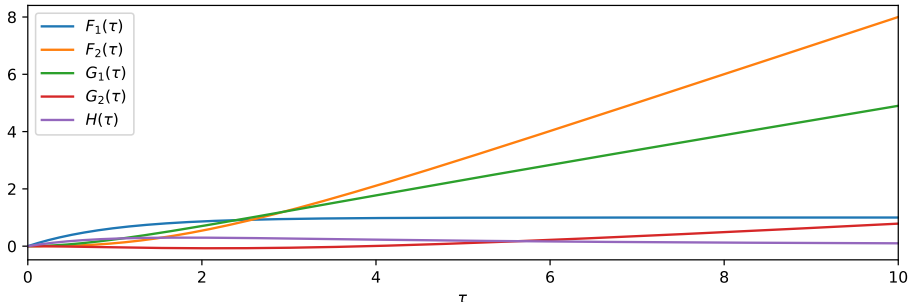


Figure 5.1: The exponential functions, Eqs. (5.1), from 0 to 10.

treatment around $\tau^g_{mki} = 0$, in this work this is handled via Taylor expansion about $\tau^g_{mki} = 0$.

These functions all involve an exponential, $e^{-\tau^g_{mki}}$; although this does not present a problem mathematically, the exponential function is a transcendental function that tends to be slow when evaluated analytically on computers. A single exponential evaluation may take more than 10 floating point operations (FLOPs). For transport codes to be efficient, this presents a challenge. Previous works have demonstrated that tabulation of the exponential and interpolation in the table can provide significant run-time reduction [4], reducing exponential calculation times by up to a factor of 3. The original formulation [1] suggested the use of tabulation for each the functions Eqs. (5.1) (though no details were provided).

5.1.1 First Approach: Improved Accuracy

With the implementation of LSA into MPACT, stability issues were observed in problems with very small transport cross sections, such as the fuel-clad gap in LWRs. These stability issues were only observed when exponential function interpolation was used, and the simplest method for addressing the issue was to increase the accuracy of the exponential evaluation. It was discovered that the root of the problem was the $F_2(\tau^g_{mki})$ function in the transmission equation

$$\psi^{g,\text{out}}_{mki} = \psi^{g,\text{in}}_{mki} + \left(\frac{\overline{q}^g_{mki}}{\Sigma^g_{t,i}} - \psi^{g,\text{in}}_{mki} \right) F_1(\tau^g_{mki}) + \frac{\widehat{q}^g_{mi}}{2\left(\Sigma^g_{t,i}\right)^2} F_2(\tau^g_{mki}). \tag{5.2}$$

In MPACT, source terms are actually computed and stored as $q/\Sigma^g_{t,i}$; however the $F_2(\tau^g_{mki})$ term has an additional inverse $\Sigma^g_{t,i}$. In problems with near-void regions, where $\Sigma^g_{t,i}$ is small, any error in $F_2(\tau^g_{mki})$ interpolation will be magnified. The simplest approach is to increase the accuracy of the exponential interpolation to account for the lowest expected cross sections; in the test problem this was on the order of 10^{-5}. Thus, the expectation is that an interpolation 5 orders of magnitude more accurate than previous accuracy would be sufficient. Previous work [4] indicated that for FSMoC calculations a max interpolation error of 10^{-7} was sufficient, i.e. for LSMoC the max error would

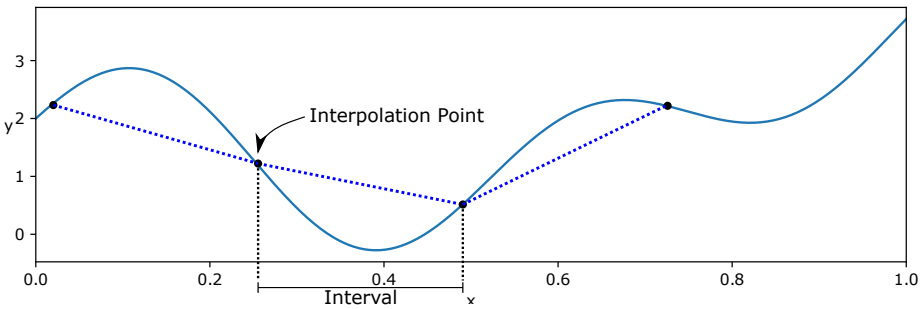


Figure 5.2: Example of linear interpolation with uniform interval widths and interpolation points on the edge of the domain.

need to be approximately 10^{-12}.

Creating interpolation tables with the necessary accuracy for these problems requires some care. The original investigation of exponential interpolation for transport calculations done by Yamamoto et al. [4] indicated two methods of controlling interpolation accuracy. One method to increase interpolation accuracy is to increase the number of intervals (decreased interval width); however, this increases the memory required to store the table. Alternatively, higher order polynomials can be used in the interpolation, that overall tends to reduce memory at the expense of increased run-times.

In this investigation, two additional methods for controlling accuracy were investigated: interpolation node choice, and non-uniform interval widths.

5.1.1.1 Interpolation Points

Within each interval of an interpolation table, the function is computed at interpolation points and an approximation of the function is made as the polynomial passing through these points. The placement of these points within each interval can greatly affect the accuracy of the approximation. Previous works [4, 5] used evenly-spaced interpolation points within each interval; however, this does not minimize the error in the approximation. An example is shown in Fig. 5.2.

For $P_n(x)$, a polynomial of order n, approximating a function, $f(x)$, on an arbitrary interval $[a, b]$, the maximum error, ϵ, within an interval is given by

$$\epsilon = \frac{1}{(n+1)!} \left(\max_{\xi \in [a,b]} \left| f^{(n+1)}(\xi) \right| \right) \left(\max_{x \in [a,b]} \left| \prod_{j=1}^{n+1} (x - x_j) \right| \right), \tag{5.3}$$

for some value $\xi \in [a, b]$, where $f^{(n+1)}$ is the $n+1$-th derivative of $f(x)$. The choice of interpolation points will only affect the last term enclosed in parentheses.

The Chebyshev points [6] are a set of values in $[a, b]$ that minimize $\max_{x \in [a,b]} \left| \prod_{j=1}^{n+1} (x - x_j) \right|$, and

Table 5.1: Maximum error in $F_1(\tau)$ for interval width Δ.

Polynomial Order	Uniform points	Chebyshev points
1	$\frac{\Delta^2}{8}$	$\frac{\Delta^2}{16}$
2	$\frac{\Delta^3}{72\sqrt{3}}$	$\frac{\Delta^3}{192}$
3	$\frac{\Delta^4}{1536}$	$\frac{\Delta^4}{3072}$

are given by

$$x_k = \frac{1}{2}\left[(a+b) + (b-1)\cos\left(\frac{2k-1}{2(n+1)}\pi\right)\right], \forall k \in \{1, 2, ..., n, n+1\}. \tag{5.4}$$

By using the Chebyshev points, the maximum interpolation error, ϵ, can be simplified to

$$\epsilon = \frac{1}{2^n(n+1)!}\left(\frac{b-a}{2}\right)^{n+1} \max_{\xi\in[a,b]} \left|f^{(n+1)}(\xi)\right|. \tag{5.5}$$

Because the Chebyshev points do not include the end-points of the interval, there is additional cost in setting up the interpolation table, but this is negligible to typical MoC calculation times. An interpolation table using Chebyshev points *reduces error at no run-time cost* compared to a table using evenly spaced points. For this reason, Chebyshev points will be assumed for the remainder of this section. Table 5.1 shows maximum errors for $F_1(\tau_m^g)$ interpolation for uniformly spaced points and Chebyshev points for an interval width $\Delta = b - a$.

5.1.1.2 Interval Width

The conventional approach for interpolation tables has been to use a constant interval width, Δ, for all intervals in the domain. This interval width is then used to control the error of the table. Equation (5.3) shows that the interval bounds affect the interpolation error through the derivative term. The second and higher order derivatives of each of the exponential functions (Eqs. (5.1)) approach zero as τ_{mki}^g approaches infinity. This indicates that the interpolation error typically decreases as the optical thickness increases in the conventional approach.

However, it is possible to maintain the same maximum error over each interval if a variable interval width, Δ_i, is used, where i indicates the interval index. By using a variable interval width, a table can use fewer intervals while maintaining the same maximum error. However, since the widths are no longer constant, there is no longer a simple/direct conversion from τ_{mki}^g to i. Although there may be better ways to address this, for this work the smallest interval is used to divide the domain into a map that points to the correct interval for that range of values.

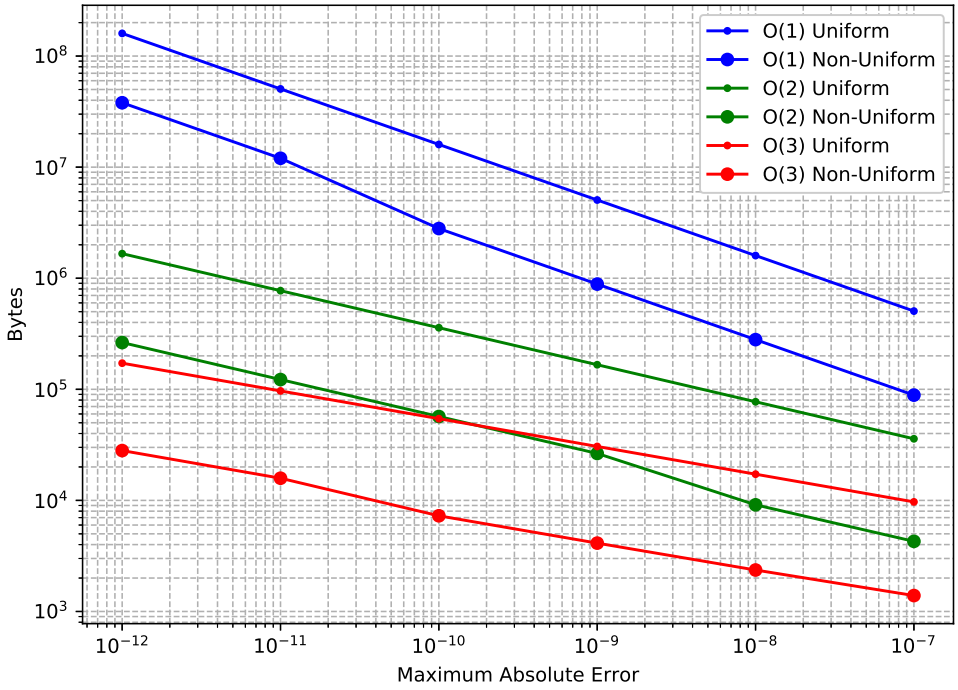


Figure 5.3: The memory usage of a single interpolation table (polar independent) for $F_1(\tau^g_{mki})$ is shown as a function of the maximum error for different interpolation orders and tabulation methods.

It was found that the use of non-uniform interval widths allowed for significantly fewer total intervals, reducing the memory usage, but it incurs overhead for the additional mapping to index. Figure 5.3 shows the memory usage for a polar-independent interpolation table for $F_1(\tau^g_{mki})$. Using a non-uniform table typically decreases the memory usage by nearly an order of magnitude. For polar-dependent tables, the memory usage is multiplied by the largest inverse sine of the polar angle, and the number of polar angles.

Because all the functions of Eqs. (5.1) are related, it is possible to tabulate only a single function and compute the others from the resulting interpolated value. However, if $F_1(\tau^g_{mki})$ is known accurately, it is not possible to compute $H(\tau^g_{mki})$ for very small τ^g_{mki} due to round-off errors. Instead one can tabulate $H(\tau^g_{mki})$ and use the result to compute the other functions.

Results for this approach are detailed in Section 5.1.3. In cases with near-void regions, the function tables are required to be very accurate. This has an adverse affect on performance, and is not a robust solution as if cross sections change the required accuracy might also change. In the next section, an alternative approach will be provided.

5.1.2 Function Modification

A second approach for addressing the instability due to inaccurate function interpolation was taken: change the functions. Recall that the highly accurate interpolation tables of Section 5.1.1 were necessary due to the inverse total (or transport) cross section on the $F_2(\tau^g_{mki})$ term of Eq. (5.2). Manipulating this term, we define a new function

$$\frac{F_2(\tau^g_{mki})}{\Sigma^g_{t,i}} = t_m \widehat{F}_2(\tau^g_{mki}), \tag{5.6}$$

where

$$\widehat{F}_2(\tau^g_{mki}) \equiv 2\left(1 - \frac{F_1(\tau^g_{mki})}{\tau^g_{mki}}\right) - F_1(\tau^g_{mki}). \tag{5.7}$$

The $\widehat{F}_2(\tau^g_{mki})$ function can be tabulated in place of $F_2(\tau^g_{mki})$; however, this does require an additional multiplication by the segment length t_{mki}. But, by using the modified $\widehat{F}_2(\tau^g_{mki})$ function, the underlying cause of the numerical instability of the LSMoC transport sweep with near-void regions is addressed, and the more accurate interpolation (e.g. larger interpolation tables) is no longer necessary. This modification of the function is not presented in the original work [1], but a similar modification is made in the implementation of the method in CASMO5 [7].

Although one approach is to tabulate the $\widehat{F}_2(\tau^g_{mki})$ function in place of the $F_2(\tau^g_{mki})$ function, and multiply by t_{mki} during evaluation, it is possible to tabulate a single function and compute the others. As stated in the previous section, the $H(\tau^g_{mki})$ function could be tabulated again; however, tabulating an intermediate function,

$$E_1(\tau^g_{mki}) \equiv \frac{1 - e^{-\tau^g_{mki}}}{\tau^g_{mki}}, \tag{5.8}$$

uses the same number of operations to compute the other functions. This $E_1(\tau^g_{mki})$ function also has smaller derivative terms around $\tau^g_{mki} = 0$, allowing for larger interval widths with no loss in accuracy. Therefore, it is expected that $E_1(\tau^g_{mki})$ tabulation is more efficient than tabulating $H(\tau^g_{mki})$.

5.1.3 Results

The results of this section were generated in serial on a Linux system with an Intel Xeon E3-1241 v3 (3.50 GHz) processors with 8 MB L3 cache. Results were generated for a 2-D MoC transport calculation. In 2-D it is sometimes convenient to store separate interpolation tables for each polar angle. Both polar dependent and independent tables were tested. In both of the results sections, results were generated using tables for the three functions ($F_1(\tau^g_{mki})$, $F_2(\tau^g_{mki})$, $H(\tau^g_{mki})$), or a single

function (which is then used to compute the others).

5.1.3.1 Results using More Accurate Tables

Table 5.2 shows results for the different interpolation methods in a 2-D MoC calculation. The use of non-uniform intervals significantly reduced the memory usage, but the run-times generally increased. This is likely due to the overhead from the additional index mapping operation; it is possible that a more efficient index mapping could change these results.

The linear tables were significantly slower than higher-order interpolations, and the non-uniform linear tables were actually slower than just using the builtin transcendental functions. The poor performance is due to the main memory accesses caused by the table size exceeding the largest (L3) cache size. The polar independent tables were faster only in the case of the linear tables, due primarily to the reduction in memory. In all other cases the polar-dependent tables were able to more efficiently approximate the functions. Using either second or third order tables gave reasonably low run-times; compared to the analytic evaluations the order 2 uniform polar dependent table led to ~ 3.5x speed-up in the exponential time, and ~ 1.9x speed-up in the overall MoC run-time. The results for both second and third order interpolation tables did not vary significantly, the second order was slightly faster for uniform tables.

However, if a single function is tabulated and used to compute the others, the memory footprint of the interpolation tables can be significantly reduced. Moreover, the memory accesses are reduced in favor of floating point operations which increases the computational intensity and temporal cache locality of the implementation. Generally, this is a favorable strategy for improving code performance as a memory access is expected to take *at minimum* the same time as a single FLOP, with cache-misses increasing that time. Indeed, Table 5.2 shows that tabulating the single function reduced run-time in all cases.

5.1.3.2 Results using Modified Function

Results were generated for the same cases as the more accurate table, but the interpolation accuracy was 10^{-7} rather than 10^{-12}, as the function modification addressed the cause of the instability. The results are summarized in Table 5.3. In one approach the three functions are tabulated, in the other $E_1(\tau^g_{mki})$ is tabulated and used to compute the three functions. Previous results showed that the polar-independent and non-uniform interval width tables were slower than uniform polar-dependent tables; here the results are only shown for the uniform polar-dependent interpolation tables.

The interpolation accuracy is not required to be as high as in the previous cases, thus the tables are significantly smaller in memory size. The linear tables no longer exceed the L3 cache size and are not prohibitively slow to use. In fact, the linear tables are the fastest option for both 3-table and

Table 5.2: Results for different exponential function evaluation methods. Maximum interpolation error of 10^{-12}. "w/ pol" indicates a table with polar dependence, and "w/o pol" indicates a table without polar dependence.

Method	Table Type	Order	Exponential Time (s)		MoC Time (s)		Intervals (Memory)	
			w/ pol	w/o pol	w/ pol	w/o pol	w/ pol	w/o pol
Analytic	N/A	N/A	432.5		654.7		N/A	
3-Functions	Uniform	1	382.6	360.3	622.5	591.0	6.00E7 (8.04 GB)	1.00E7 (458 MB)
		2	121.3	151.7	343.2	372.1	416080 (85.7 MB)	69360 (4.8 MB)
		3	124.1	169.0	347.5	388.7	32280 (9079 KB)	5400 (506 KB)
	Non-Uniform	1	592.9	559.2	825.8	800.6	4.07E6 (878 MB)	840658 (76.6 MB)
		2	144.6	178.5	364.3	397.8	45265 (10.9 MB)	8020 (835 KB)
		3	139.6	199.8	358.5	420.0	4491 (1.4 MB)	796 (85.2 KB)
1-Function	Uniform	1	296.2	259.3	523.9	484.5	6.00E7 (2.68 GB)	1.00E7 (153 MB)
		2	91.4	123.2	310.2	345.7	416080 (28.6 MB)	69360 (1.59 MB)
		3	87.2	133.2	305.4	353.6	32280 (2.96 MB)	5400 (169 KB)
	Non-Uniform	1	474.6	424.0	714.6	653.9	4.07E6 (415 MB)	722899 (49.2 MB)
		2	105.8	146.4	326.6	367.7	41444 (4.4 MB)	7338 (443 KB)
		3	98.9	153.3	318.8	373.2	4152 (515 KB)	735 (33.5 KB)

single-table interpolation methods. Interpolating $E_1(\tau^g_{mki})$ and computing the other functions from the result was the fastest approach overall. As before, this is expected because this method favors FLOPs over memory accesses.

This approach also significantly reduces the number of intervals necessary (by about 3 orders of magnitude). Although the higher-order interpolation ended up not being more efficient than the linear interpolation tables, the number of intervals is extremely small. These methods may be useful on different architectures (such as GPGPUs[1]) where memory is more limited than on a single-threaded CPU calculation.

5.1.4 Conclusions

This study focused on the investigation of efficient approximation of the exponential functions in the LSMoC. Methods for efficiently approximating the exponential function for FSMoC have previously been investigated. However, these methods, applied without modification to the LSMoC can lead to numerical instability in problems with near-void regions. One approach to deal with these instabilities is to improve the accuracy of the interpolation tables based on the lowest cross section in the problem. While this approach can be made efficient, if cross sections become too small the tables may become exceedingly large and significantly hamper performance.

[1]this statement was not confirmed as part of this study, and is speculation based on knowledge of architecture constraints.

Table 5.3: Results using the modified function, with maximum interpolation error of 10^{-7}. Memory is under 1 MB in all cases.

Method	Order	Exp. Time (s)	MoC Time (s)	# Intervals
3-Functions	1	88.4	307.9	4774
	2	99.7	316.8	225
	3	116.5	333.1	46
$E_1(\tau_m^g)$ Only	1	80.5	299.0	2739
	2	87.6	307.3	158
	3	99.0	317.7	38

This investigation indicated that the cause of the instability was the $F_2(\tau_{mki}^g)$ function in the transmission equation. By instead manipulating the term including this function, the cause of the instabilities can be addressed. This approach no longer requires excessively large interpolation tables; however, some key-findings from the first approach can be applied to make this approach faster. First, the use of Chebyshev points significantly increases interpolation accuracy for the same number of intervals as uniformly spaced points, at no run-time cost. Additionally, by interpolating a single function and using the result to compute the other functions performance can be significantly improved.

5.2 Improved Linear Source Formulation for Multi-physics and 2D/1D Applications

As stated in Section 3.3, the original derivation of the moment-based linear-source approximation (LSA) from Ferrer and Rhodes [1] faced inefficiencies in problems with non-constant cross sections. Multiphysics applications were not of interest to the original authors [1]. The inefficiencies arose from the use of pre-computed coefficients, Eq. (3.53), that must be re-computed if the cross sections change. The computation of these coefficients was approximately 10-20% the run-time cost of a MoC sweep. In multiphysics calculations, such as with T/H feedback, cross sections typically change every iteration, leading to significant overhead from re-computing these terms. This section presents an equivalent formulation, that eliminates the need to re-compute these terms if cross sections change, without additional operations.

5.2.1 Derivation

The notation used in this section is described in detail in Chapter 3. The derivation of this improved formulation begins with the same initial steps as the LSMoC derivation detailed in Section 3.3.2. Several of these equations are repeated in this section for clarity. The source is assumed to have a spatially linear shape

$$q^g_{mi}(\boldsymbol{x}) \approx q^g_{mi} + \boldsymbol{x} \cdot \underline{\widehat{\boldsymbol{q}}}^g_{mi}, \tag{5.9}$$

and the angular flux moments can be similarly expanded

$$\phi^{\ell,g}_{i,n}(\boldsymbol{x}) = \overline{\phi}^{\ell,g}_{i,n} + \boldsymbol{x} \cdot \underline{\widehat{\boldsymbol{\phi}}}^{\ell,g}_{i,n}. \tag{5.10}$$

The source can be computed from the flux moments using Eq. (3.36).

Computing the source requires the flux to be evaluated during the transport sweep. The angular flux moments are given by Eqs. (3.41), and are repeated here in Eq. (5.11a). The spatially flat angular flux moments are defined by integrating the flux multiplied by a spherical harmonics moment function over space and directions,

$$\overline{\phi}^{\ell,g}_{i,n} \equiv \left\langle R^n_\ell(\widehat{\boldsymbol{\Omega}})\psi^g \right\rangle_i = \frac{4\pi}{V_i} \sum_m w_m R^n_\ell(\widehat{\boldsymbol{\Omega}}_m) \sum_k \delta A_{mki} t_{mki} \langle \psi^g \rangle_{mki}. \tag{5.11a}$$

To determine the spatial expansion coefficients of the flux moments, Eq. (5.10) is operated on by $\left\langle R^n_\ell(\widehat{\boldsymbol{\Omega}})\boldsymbol{x}(\cdot) \right\rangle_i$. As before, this should be directly proportional to the angular flux operated on by $\left\langle R^n_\ell(\widehat{\boldsymbol{\Omega}})\boldsymbol{x}\psi^g \right\rangle_i$. From this a system of equations is found

$$\boldsymbol{M}_i \underline{\widehat{\boldsymbol{\phi}}}^{\ell,g}_{i,n} = \left\langle R^n_\ell(\widehat{\boldsymbol{\Omega}})\boldsymbol{x}\psi^g \right\rangle_i, \tag{5.11b}$$

where

$$\boldsymbol{M}_i \equiv \left\langle \boldsymbol{x}^T \boldsymbol{x} \right\rangle_i. \tag{5.11c}$$

The spatial-angular flux moments, $\left\langle R^n_\ell(\widehat{\boldsymbol{\Omega}})\boldsymbol{x}\psi^g \right\rangle_i$, are then defined as

$$\left\langle R^n_\ell(\widehat{\boldsymbol{\Omega}})\boldsymbol{x}\psi^g \right\rangle_i = \frac{4\pi}{V_i} \sum_m w_m R^n_\ell(\widehat{\boldsymbol{\Omega}}_m) \sum_k \delta A_{mki} t_{mki} \left(\boldsymbol{x}^{\text{in}}_{mki} \langle \psi^g \rangle_{mki} + \widehat{\boldsymbol{\Omega}}_m \langle t_m \psi^g \rangle_{mki} / \xi_{mi} \right). \tag{5.11d}$$

With the assumed source shape, Eq. (5.9), the characteristic transport equation becomes

$$\left[\frac{\mathrm{d}}{\mathrm{d}t_m} + \Sigma^g_{t,i} \right] \psi^g_{mki}(s) = \overline{q}^g_{mki} + \widehat{q}^g_{mi} \left(t_m - \frac{t_{mki}}{2} \right), \tag{5.12a}$$

where

$$\overline{q}^g_{mki} \equiv \frac{1}{4\pi}\left[q^g_{mi} + \boldsymbol{x}^{\mathrm{c}}_{mki} \cdot \underline{\widehat{\boldsymbol{q}}}^g_{mi}\right], \tag{5.12b}$$

$$\widehat{q}^g_{mi} \equiv \frac{1}{4\pi}\left[\frac{\widehat{\boldsymbol{\Omega}}_m \cdot \underline{\widehat{\boldsymbol{q}}}^g_{mi}}{\xi_{mi}}\right], \tag{5.12c}$$

and $\boldsymbol{x}^{\mathrm{c}}_{mki}$ is the centroid of the track-segment in local-coordinates. Substituting this assumed source shape (linear) into the generic MoC solution, given by Eq. (3.9), the angular flux along a track-segment is found to be

$$\psi^g_{mki}(s) = \psi^{g,\mathrm{in}}_{mki} + \left(\frac{\overline{q}^g_{mki}}{\Sigma^g_{t,i}} - \psi^{g,\mathrm{in}}_{mki}\right) F_1(\tau^g_m) + \frac{\widehat{q}^g_{mi}}{2(\Sigma^g_{t,i})^2} F_2(\tau^g_m), \tag{5.13a}$$

where

$$F_1(\tau^g_m) \equiv 1 - \exp(-\tau^g_m), \tag{5.13b}$$

and

$$F_2(\tau^g_m) \equiv 2[\tau^g_m - F_1(\tau^g_m)] - \tau^g_{mki} F_1(\tau^g_m). \tag{5.13c}$$

To evaluate the moments defined in Eqs. (5.11), the track-average flux and first spatial moment of the angular flux must be determined. The track-average flux is determined as it was before, by operating on Eq. (5.12a) by $t_{mki}\langle(\cdot)\rangle_{mki}$, yielding

$$\langle\psi^g\rangle_{mki} = \frac{\overline{q}^g_{mki}}{\Sigma^g_{t,i}} + \frac{\Delta\psi^g_{mki}}{\tau^g_{mki}}. \tag{5.14a}$$

At this point the two derivations diverge from one another. In the previous formulation, the integral in $\langle t_m\psi^g\rangle_{mki}$ is *explicitly* evaluated by substituting in the solution of the angular flux along the track (Eqs. (3.43)). Here, the moment will be found implicitly by operating on Eq. (5.12a) by $t_{mki}\langle t_m(\cdot)\rangle_{mki}$, just as was done for the 0th moment. This results in

$$\langle t_m\psi^g\rangle_{mki} = \frac{\langle\psi^g\rangle_{mki} - \psi^{g,\mathrm{out}}_{mki}}{\Sigma^g_{t,i}} + \frac{t_{mki}}{2}\left[\frac{\overline{q}^g_{mki}}{\Sigma^g_{t,i}} + \frac{\widehat{q}^g_{mi}}{\Sigma^g_{t,i}}\frac{t_{mki}}{6}\right]. \tag{5.14b}$$

Unlike Eq. (3.44b), there a not any new exponential functions introduced in this form.

Previously, the average track flux, $\langle\psi^g\rangle_{mki}$, was expanded to find a simpler final form. This is the case here as well; however, the out-going flux, $\psi^{g,\mathrm{out}}_{mki}$, will not be expanded, as this must necessarily be computed during transmission. Eqs. (5.11) are then simplified into

$$\overline{\phi}^{\ell,g}_{i,n} = \frac{4\pi}{V_i\Sigma^g_{t,i}} \sum_m w_m R^n_\ell(\widehat{\boldsymbol{\Omega}}) \sum_k \delta A_{mki}\left(t_{mki}\overline{q}^g_{mki} + \Delta\psi^g_{mki}\right), \tag{5.15a}$$

and

$$\left\langle R_\ell^n(\widehat{\boldsymbol{\Omega}})\boldsymbol{x}\psi^g\right\rangle_i = \frac{1}{V_i}\sum_m w_m R_\ell^n(\widehat{\boldsymbol{\Omega}})\sum_k \delta A_{mki} t_{mki}\left[\boldsymbol{x}_{mki}^{\mathrm{c}} q_i^g + \left(\boldsymbol{x}_{mki}^{\mathrm{c}}(\boldsymbol{x}_{mki}^{\mathrm{c}})^T + \frac{s_{mki}^2}{12}\widehat{\boldsymbol{\Omega}}_m\widehat{\boldsymbol{\Omega}}_m^T\right)\underline{\widehat{\boldsymbol{q}}}_i^g\right] + \frac{4\pi}{V_i\Sigma_{t,i}^g}\sum_m w_m R_\ell^n(\widehat{\boldsymbol{\Omega}})\sum_k \delta A_{mki}\left[\boldsymbol{x}_{mki}^{\mathrm{in}}\Delta\psi_{mki}^g + \widehat{\boldsymbol{\Omega}}_m s_{mki}\left(\frac{\Delta\psi_{mki}^g}{\tau_{mki}^g} - \psi_{mki}^{g,\mathrm{out}} + \frac{\overline{q}_{mki}^g}{\Sigma_{t,i}^g}\right)\right]. \tag{5.15b}$$

Equation (5.15b) contains $\Delta\psi_{mki}^g/\tau_{mki}^g$ terms, rather than only $\Delta\psi_{mki}^g$ terms. For numerical stability, it is beneficial to compute this quantity directly, rather than $\Delta\psi_{mki}^g$ and performing division. This can be found by evaluating the transmission equation, Eq. (5.13a), giving

$$\frac{\Delta\psi_{mki}^g}{\tau_{mki}^g} = \left(\psi_{mki}^{g,\mathrm{in}} - \frac{\overline{q}_{mki}^g}{\Sigma_{t,i}^g}\right)E_1(\tau_{mki}^g) - \frac{t_{mki}}{2}\frac{\widehat{q}_{mi}^g}{\Sigma_{t,i}^g}T_2(\tau_{mki}^g), \tag{5.16a}$$

where

$$E_1(\tau_{mki}^g) \equiv \frac{F_1(\tau_{mki}^g)}{\tau_{mki}^g}, \tag{5.16b}$$

$$T_2(\tau_{mki}^g) \equiv 2E_2(\tau_{mki}^g) - E_1(\tau_{mki}^g), \tag{5.16c}$$

and

$$E_2(\tau_{mki}^g) \equiv \frac{1 - E_1(\tau_{mki}^g)}{\tau_{mki}^g}. \tag{5.16d}$$

Here, $E_2(\tau_{mki}^g)$ is defined as an intermediate function which has smaller derivative terms, meaning higher accuracy with fewer interpolation intervals (see Section 5.1). Following the conclusions of Section 5.1, only the $E_2(\tau_{mki}^g)$ function needs to be tabulated, and then is used to compute the other two exponential functions, $E_1(\tau_{mki}^g)$, and $T_2(\tau_{mki}^g)$. The outgoing flux can then be evaluated as

$$\psi_{mki}^{g,\mathrm{out}} = \psi_{mki}^{g,\mathrm{in}} - \tau_{mki}^g\frac{\Delta\psi_{mki}^g}{\tau_{mki}^g}. \tag{5.17}$$

5.2.2 Particle Conservation

As described in Section 3.3.3, particle conservation puts the constraint of using direction-dependent renormalization and centroids, in addition to directional quadrature constraints. If these constraints are satisfied, Eqs. (5.15) can be simplified into

$$\overline{\phi}_{i,n}^{\ell,g} = \sum_m w_m R_\ell^n(\widehat{\boldsymbol{\Omega}})\frac{q_i^g}{\Sigma_{t,i}^g} + \frac{4\pi}{V_i\Sigma_{t,i}^g}\sum_m w_m R_\ell^n(\widehat{\boldsymbol{\Omega}})\sum_k \delta A_{mki}\Delta\psi_{mki}^g, \tag{5.18a}$$

and

$$\left\langle R_\ell^n(\widehat{\boldsymbol{\Omega}})\boldsymbol{x}\psi^g\right\rangle_i = \sum_m w_m R_\ell^n(\widehat{\boldsymbol{\Omega}})\boldsymbol{M}_{mi}\frac{\widehat{\boldsymbol{q}}_i^g}{\Sigma_{t,i}^g} + \frac{4\pi}{V_i\Sigma_{t,i}^g}\sum_m w_m R_\ell^n(\widehat{\boldsymbol{\Omega}})\sum_k \delta A_{mki}\left[\boldsymbol{x}_{mki}^{\text{in}}\Delta\psi_{mki}^g + \widehat{\boldsymbol{\Omega}}_m s_{mki}\left(\frac{\Delta\psi_{mki}^g}{\tau_{mki}^g} - \psi_{mki}^{g,\text{out}} + \frac{\overline{q}_{mki}^g}{\Sigma_{t,i}^g}\right)\right], \tag{5.18b}$$

where

$$\boldsymbol{M}_{mi} \equiv \frac{1}{V_i}\sum_k \delta A_{mki}\left[\boldsymbol{x}_{mki}^{\text{c}}(\boldsymbol{x}_{mki}^{\text{c}})^T + \frac{s_{mki}^2}{12}\widehat{\boldsymbol{\Omega}}_m\widehat{\boldsymbol{\Omega}}_m^T\right]. \tag{5.18c}$$

Although the above form seems to be more computationally efficient, it would be negligent to not include a more mathematically elegant form of Eq. (5.15b). The terms in the second summation may be rewritten as

$$\left\langle R_\ell^n(\widehat{\boldsymbol{\Omega}})\boldsymbol{x}\psi^g\right\rangle_i = \sum_m w_m R_\ell^n(\widehat{\boldsymbol{\Omega}})\boldsymbol{M}_{mi}\frac{\widehat{\boldsymbol{q}}_i^g}{\Sigma_{t,i}^g} + \frac{4\pi}{V_i\Sigma_{t,i}^g}\sum_m w_m R_\ell^n(\widehat{\boldsymbol{\Omega}})\sum_k \delta A_{mki}\left[\boldsymbol{x}_{mki}^{\text{in}}\left(\psi_{mki}^{g,\text{in}} - \overline{\psi}_{mki}^g\right) - \boldsymbol{x}_{mki}^{\text{out}}\left(\psi_{mki}^{g,\text{out}} - \overline{\psi}_{mki}^g\right)\right], \tag{5.19a}$$

where $\overline{\psi}_{mki}^g$ is the average track flux,

$$\overline{\psi}_{mki}^g \equiv \langle\psi^g\rangle_{mki}. \tag{5.19b}$$

In this form, the interior summation over track-segments becomes a spatially weighted average of the incident and outlet flux differences from the average flux.

5.2.3 Isotropic Simplifications

If the so called linear-isotropic-flat-anisotropic (LIFA) scheme is used, the spatially flat moment equations do not change, however, Eq. (5.15b) can be simplified to

$$\left\langle R_\ell^n(\widehat{\boldsymbol{\Omega}})\boldsymbol{x}\psi^g\right\rangle_i = \boldsymbol{M}_i\frac{\widehat{\boldsymbol{q}}_i^g}{\Sigma_{t,i}^g} + \frac{4\pi}{V_i\Sigma_{t,i}^g}\sum_m w_m \sum_k \delta A_{mki}\left[\boldsymbol{x}_{mki}^{\text{in}}\Delta\psi_{mki}^g + \widehat{\boldsymbol{\Omega}}_m s_{mki}\left(\frac{\Delta\psi_{mki}^g}{\tau_{mki}^g} - \psi_{mki}^{g,\text{out}} + \frac{\overline{q}_{mki}^g}{\Sigma_{t,i}^g}\right)\right]. \tag{5.20}$$

If isotropic scattering is used, the the scalar flux moments can be found as

$$\begin{aligned}\langle \boldsymbol{x}\psi^g \rangle_i = \boldsymbol{M}_i \frac{\widehat{\boldsymbol{q}}_i^g}{\Sigma_{t,i}^g} \\ + \frac{4\pi}{V_i \Sigma_{t,i}^g} \sum_m w_m \sum_k \delta A_{mki} \left[\boldsymbol{x}_{mki}^{\text{in}} \Delta\psi_{mki}^g + \widehat{\boldsymbol{\Omega}}_m s_{mki} \left(\frac{\Delta\psi_{mki}^g}{\tau_{mki}^g} - \psi_{mki}^{g,\text{out}} \right) \right].\end{aligned} \tag{5.21}$$

5.3 Results

This section highlights several of the key results found using the LSA solver implemented in MPACT. Only results using 2-D transport or the 2D/1D method in MPACT are shown, as 3-D transport results will be discussed in Chapter 6. Isotopic depletion is performed using the ORIGEN code, and T/H feedback is performed using COBRA-TF (CTF). Although Section 5.2 provides the derivation of a spatially linear anisotropic source, the initial implementation in MPACT was limited to the isotropic scattering approximation. With the use of the TCP0 approximation, problems of interest are able to be simulated with acceptable levels of error. However, one of our recent works [8] has implemented the LIFA approximation in MPACT, but these results are not presented here.

5.3.1 C5G7 Benchmark

The 2-D C5G7 benchmark [9] is a neutronics benchmark used by various codes to help verify the correctness of the solvers used. A 7-group cross section library of macroscopic cross sections is provided by the benchmark. The benchmark consists of two uranium oxide (UO_2) assemblies and two mixed oxide (MOX) assemblies surrounded by a large radial reflector, as shown in Fig. 5.4. Reflective boundary conditions are used on the north and west faces, while vacuum conditions are used on the south and east faces. The benchmark provides reference MCNP calculation results, which can be used to determine the accuracy of the methods used to solve the benchmark cases.

This problem was studied in order to verify that the implementation of the LSA derived in this chapter yields similar results to those found by Ferrer and Rhodes [1]. Similar meshing, ray, and direction parameters were used in this study as were used in the original study [1]; some parameters could not be replicated exactly due to differences in the MPACT and CASMO5 codes. All cases used a uniform ray-spacing of 0.05 cm, and directional quadrature using 128 azimuthal angles and 6 polar angles in 4π. Each case was run using odCMFD [10].

The fine pin cell mesh used 5 rings in the fuel region, 10 in the moderator region, and 16 azimuthal divisions in all regions. The fine reflector cell mesh was divided into 0.105×0.105

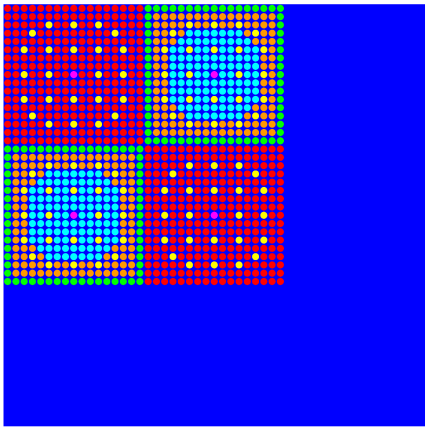


Figure 5.4: Geometry and materials of the 2-D C5G7 benchmark.

cm^2 cells. The coarse pin cell mesh used a single fuel ring with four azimuthal divisions, and no moderator rings, using 8 azimuthal divisions in the surrounding moderator. The coarse reflector cell mesh was divided into 0.42×0.42 cm^2 cells.

Three cases were run in this study: fine mesh with FSA, coarse mesh with FSA, and coarse mesh with LSA. To be consistent with the Ferrer and Rhodes [1] results, it is expected that the LSA on the coarse mesh should reduce run-time while maintaining accuracy compared to the finely mesh FSA calculation. Additionally, the coarse mesh FSA calculation should have significantly worse accuracy, thus justifying the use of the LSA.

Table 5.4 shows very clearly that the LSA on the coarse mesh is more accurate than the FSA on the fine mesh. While FSA on the fine mesh provides reasonable accuracy, the FSA on the coarse mesh shows significant errors compared to the MCNP reference. In terms of accuracy, the results agree very closely with those reported for the original derivation of the moment-based LSA. The reference pin-powers are visualized in Fig. 5.5, and the differences are shown in Figs. 5.6 to 5.8. The largest pin-power differences for the fine mesh with flat source (FS) and coarse mesh with LS are in the center assembly where pin-powers are higher. Interestingly, the largest pin-power differences for the coarse mesh with FS case are in the pins bordering the radial reflector region; this indicates that the mesh in the reflector was not sufficient. Errors in the interior of the assemblies are still larger for this case than the others.

Table 5.5 indicates that the LSA on the coarse mesh significantly out-performs the FSA on the fine mesh; MoC run-time is reduced by approximately 50%, and memory usage is reduced by 45%. However, this also shows that for this problem the LSA solver takes just under 2 times the time of the FSA on the same mesh (coarse). It is also shown, that there is very little memory overhead

from using the LSA on the same mesh as the FSA. The number of iterations is quite different of that reported by Ferrer and Rhodes [1]; however, the iteration scheme in CASMO5 is different than that of MPACT [11]. In MPACT, each iteration loops over all energy groups a single time, but in CASMO5 the thermal energy groups are iterated over multiple times.

Table 5.4: Accuracy comparisons for the C5G7 2-D benchmark calculation using different source approximations and meshes. In addition to eigenvalue difference, several metrics for comparing pin-powers are provided. The abbreviations are as follows: MPP+ = difference in maximum pin-power, MPP- = difference in minimum pin-power, AVE = average pin-power error, RMS = root-mean-square pin-power error, MRE = mean relative error. The number of pins with power within 68% and 95% of the MCNP reference results are also listed.

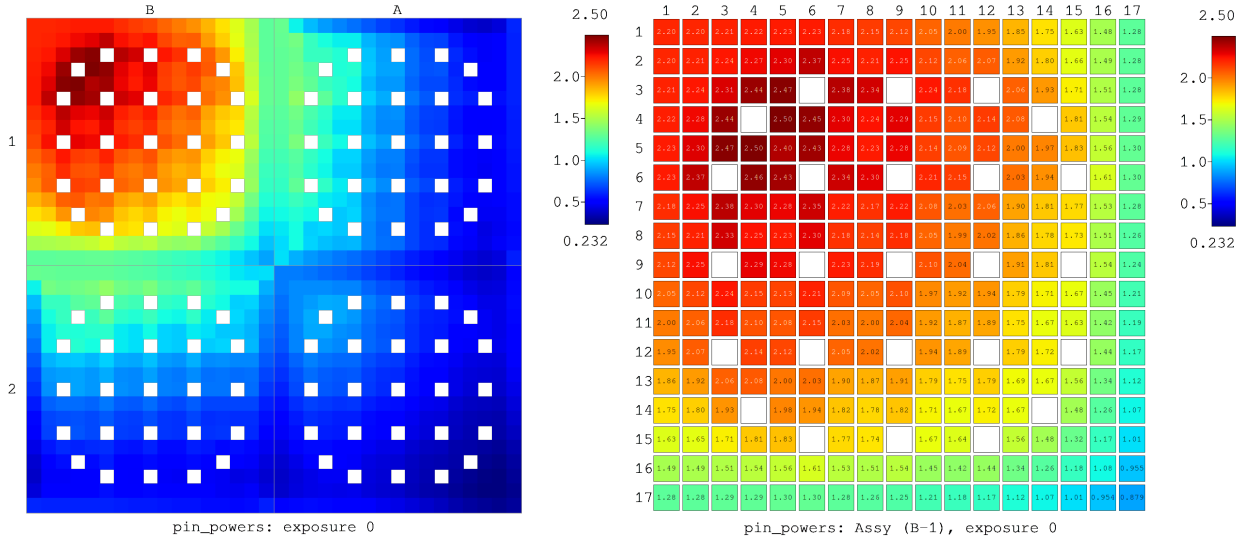
			Pin Powers (%)						# Pins within	
Solver	Mesh	Δk_{eff} (pcm)	MPP+	MPP-	MPE	AVE	RMS	MRE	68%	95%
FSA	Fine	8	-0.13	0.89	0.98	0.18	0.24	0.15	494	763
FSA	Coarse	65	-0.76	3.74	4.23	0.78	1.16	0.61	142	217
LSA	Coarse	2	0.02	0.30	0.66	0.12	0.16	0.10	663	901

Table 5.5: Performance comparisons for the C5G7 2-D benchmark calculation using different source approximations and meshes.

			Run-time (s)		
Solver	Mesh	Iterations	MoC	MoC / iter	Memory (MB)
FSA	Fine	13	435	33	409
FSA	Coarse	11	89	8	213
LSA	Coarse	13	209	16	223

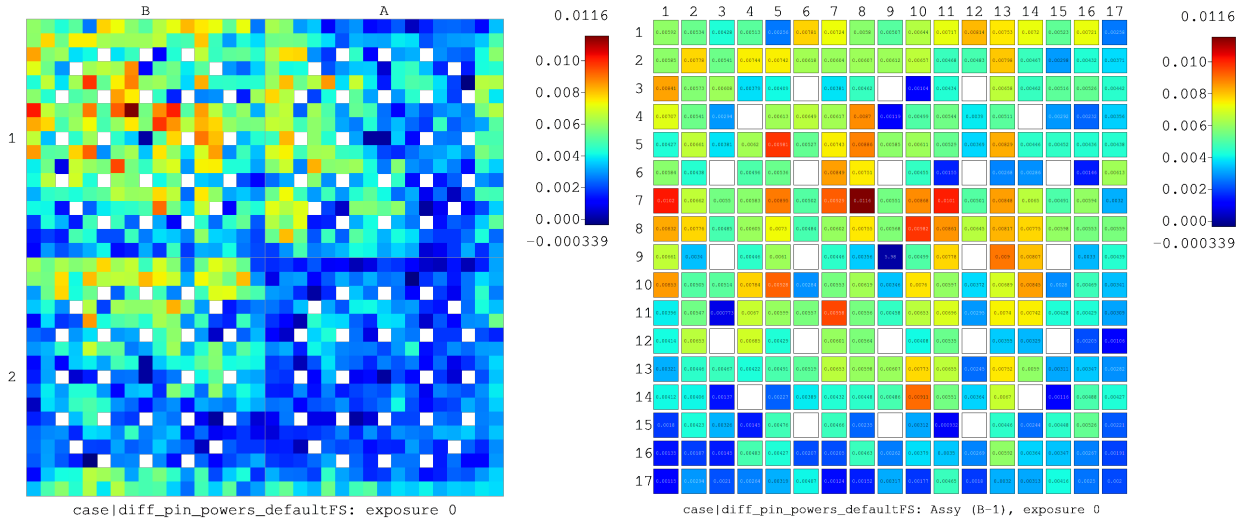
5.3.2 Typical Pin Cell Depletion

In order to evaluate the benefits of this new formulation, the first multiphysics case studied was a typical UO_2 fuel cell, as specified by Virtual Environment for Reactor Analysis (VERA) progression problem 1A [12]. Isotopic depletion calculations were run up to 70 gigawatt-days per metric ton heavy metal (GWDMT) ($\sim$1820 effective full power days (EFPD)) at hot full power (HFP) conditions. Using the current default meshing parameters in MPACT as a starting point, various mesh parameters were coarsened to study their affect when using the LSMoC. These cases were compared against a reference case, which was very finely meshed with fine ray-spacing (0.001 cm), and a Tabuchi-Yamamoto [13] quadrature using 128 azimuthal angles and 4 polar angles over 4π All other cases were run using a Tabuchi-Yamamoto quadrature with 64 azimuthal angles and 4



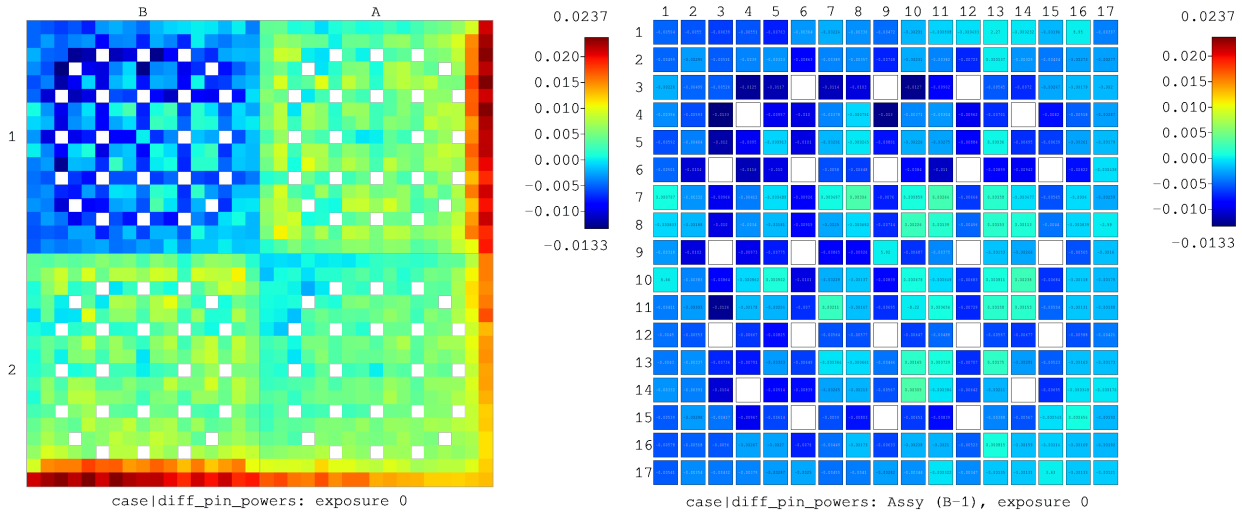
(a) core (b) center assembly

Figure 5.5: Reference pin powers shown for the core and center assembly.



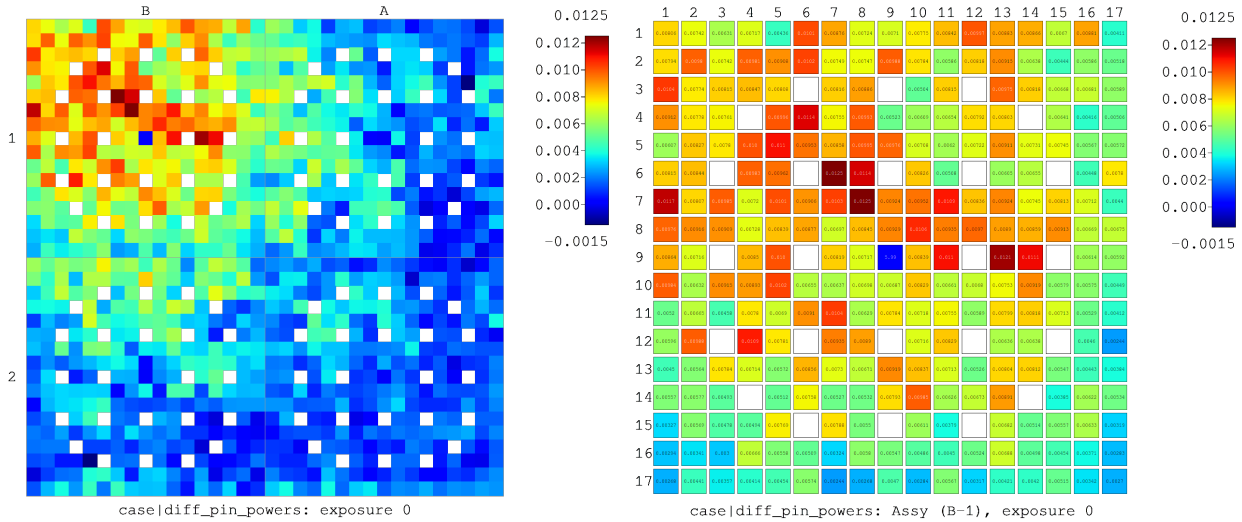
(a) core (b) center assembly

Figure 5.6: Pin power differences from reference shown for the core and center assembly for the fine mesh using the FSA solver.



(a) core (b) center assembly

Figure 5.7: Pin power differences from reference shown for the core and center assembly for the coarse mesh using the FSA solver.



(a) core (b) center assembly

Figure 5.8: Pin power differences from reference shown for the core and center assembly for the coarse mesh using the LSA solver.

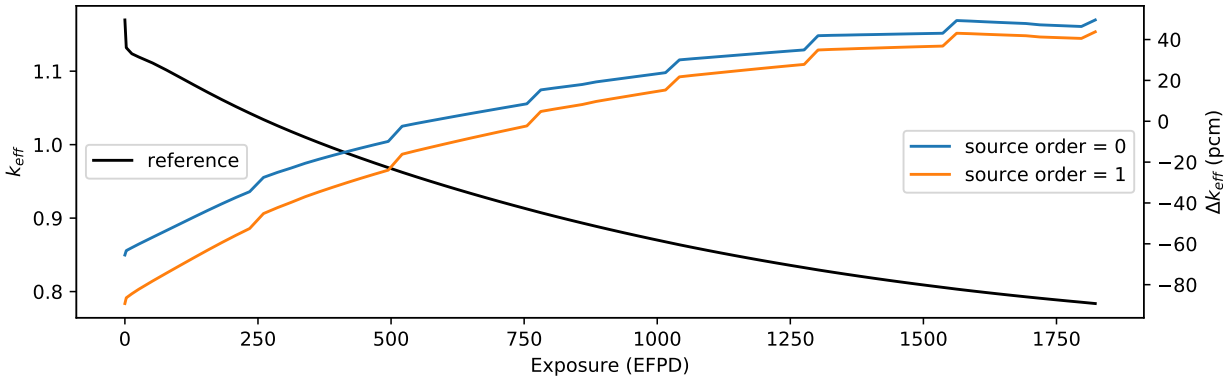


Figure 5.9: Reference eigenvalues and differences for the default mesh pin cell case with isotopic depletion.

polar angles, with a uniform ray-spacing of 0.05 cm. This case used the TCP0 approximation to scattering.

Default Mesh MPACT's current default meshing parameters were used as a starting point. Flat and linear source calculations were run on this mesh, to set a baseline for "acceptable" levels of error in the eigenvalue. Compared to the reference case, the LS calculation had a larger maximum error in eigenvalue of 89.2 per cent mille (pcm), whereas the FS calculation had a maximum error of 65.5 pcm. However, the average errors over the depletion were approximately the same: 31.6 and 33.7 pcm, for the FS and LS calculations, respectively. The eigenvalue differences over the depletion calculation are shown in Fig. 5.9. The goal in this mesh refinement study is to determine acceptable meshing parameters without worse maximum or average eigenvalue errors.

Fuel Radius During isotopic depletion, it is important to accurately capture the radial distribution of Plutonium, due to self-shielding effects. Plutonium is primarily produced in the outer rim of the pin (this is well known as the rim-effect). The default mesh uses three equal-volume rings in the fuel region; however, we expect that two rings will be sufficient, when using the LSA, if the extra ring is placed in such a way that it captures this rim-effect. Experimental studies of the rim-effect [14] have found that there is a sharp rise in Plutonium at about 80-90% of the outer radius of the fuel.

A series of calculations were run using two fuel rings with varying inner radius. The eigenvalue and Plutonium comparisons are shown in Figs. 5.10 and 5.11. Comparing eigenvalues, a fractional radius between 0.825 and 0.875 seems to have little effect on the mean or worst-case eigenvalue difference. By comparing the concentration of Pu-239, these radii are again shown to be the most accurate; a fractional radius of 0.875 was chosen to be sufficient, and used in the remainder of the studies.

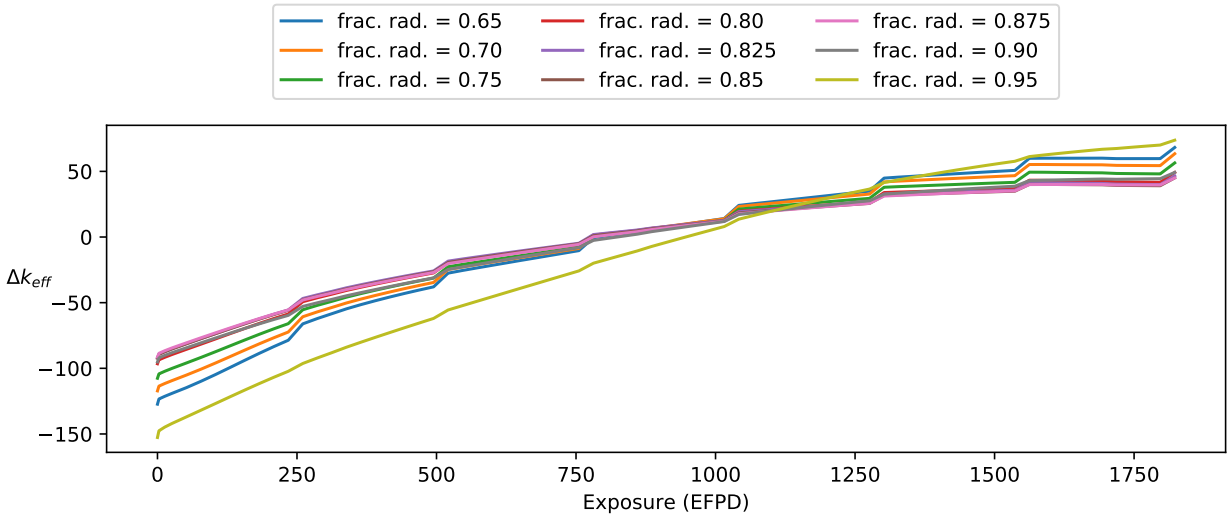


Figure 5.10: Eigenvalue comparisons for problem 1A using various inner fuel radii.

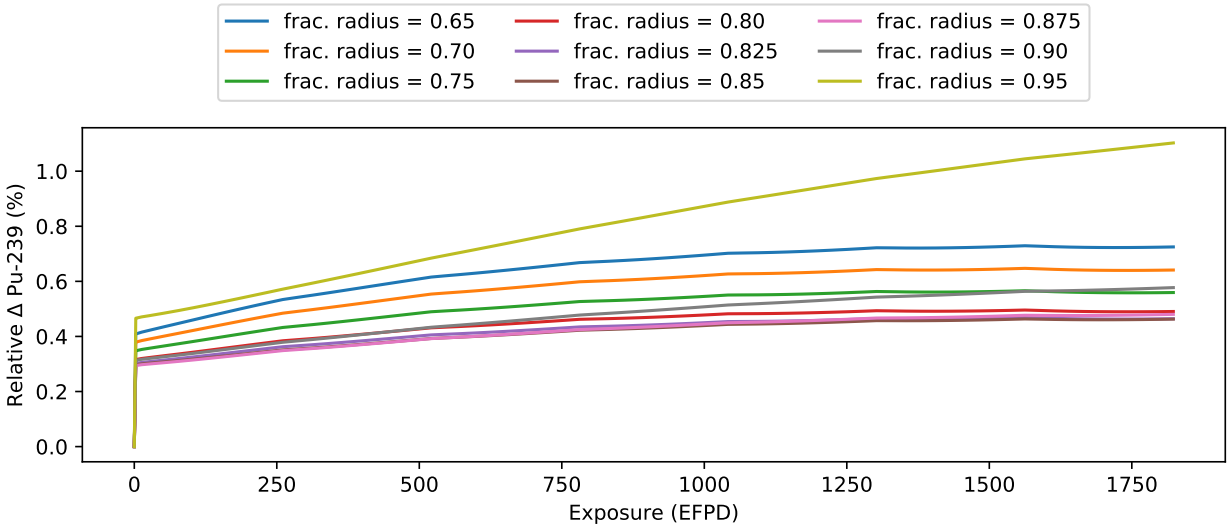
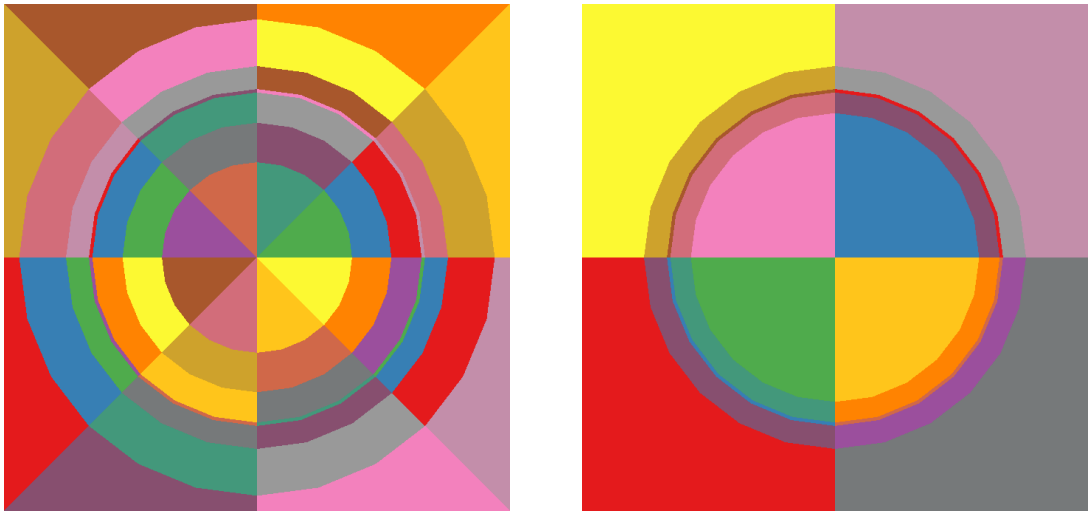


Figure 5.11: Pu-239 concentration comparisons for problem 1A using various inner fuel radii.

Moderator Ring and Azimuthal Divisions The remaining mesh parameters are the azimuthal divisions in the fuel, clad, gap, and moderator regions, and the presence of an additional surrounding ring of moderator. While it may be sufficient to use a single azimuthal division in some regions in larger cases, due to symmetry of this single pin case, using a single azimuthal region would cause the linear components of the source to be zero (i.e. it is equivalent to the FSA). It was found that the coarsening of these parameters has an insignificant affect on the resulting eigenvalue when using the LSA, with less than 1 pcm difference over the entire depletion. Thus, four azimuthal divisions in each material region, and no additional surrounding moderator ring was found to be sufficiently accurate in this case. The coarse default and coarse meshes are displayed in Fig. 5.12.



(a) default mesh (b) coarse mesh

Figure 5.12: VERA problem 1A (a) default and (b) coarse meshes.

5.3.3 VERA Problem 2A and 2P Depletion

A mesh study similar to that done in Section 5.3.2 was performed on modified VERA problems 2A and 2P [12] with isotopic depletion up to 70 GWDMT. This was done to verify that the meshing parameters previously found were valid for larger problems, and determine if even coarser parameters were sufficient. First, problem 2A was studied; the meshing parameters of the fuel cell and guide-tube cells were studied independently. Both problems were simulated using the TCP0 scattering approximation.

5.3.3.1 Problem 2A

2A - Default Mesh For this problem, a reference calculation was carried out on the lattice with a very fine mesh. This reference calclation also used a finer directional quadrature of 128 azimuthal

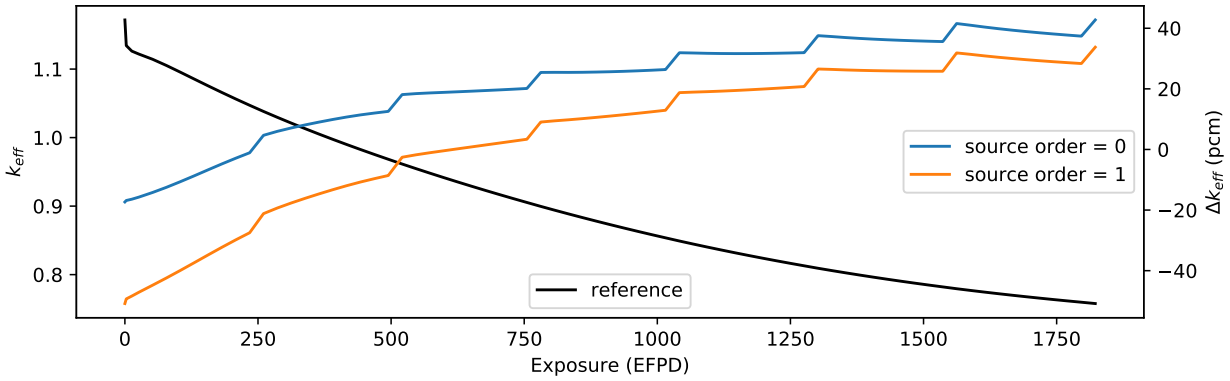


Figure 5.13: VERA Problem 2A default mesh eigenvalue comparison.

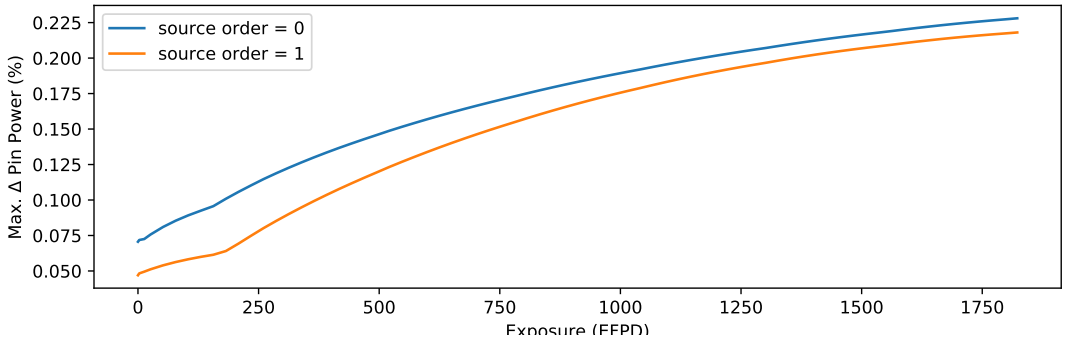


Figure 5.14: VERA Problem 2A default mesh pin power comparison.

angles and 4 polar angles, with a uniform ray-spacing of 0.001 cm. Again, the MPACT's current default mesh was used a starting point for the mesh study; the results of the FS and LS calculations on this default mesh are used as a baseline. Eigenvalue and pin-power comparisons are provided in Figs. 5.13 and 5.14.

The trend in eigenvalue is similar to that found for problem 1A with the default mesh; the LS calculation has a higher absolute eigenvalue difference of 51 pcm compared to the FS calculation with 43 pcm. However, the average eigenvalue differences are similar at 24 and 21 pcm for the FS and LS calculations, respectively. Additionally, Fig. 5.14 shows that maximum pin power differences are consistently lower for the linear source calculation. This indicates that local effects are predicted better by the linear source, and there is some cancellation of errors occurring in the eigenvalue.

2A - Fuel Radius and Azimuthal Divisions For this problem, only the fractional radius found in Section 5.3.2 is tested against the current default mesh. There was no significant affect in eigenvalue or pin powers by using the 2-ring model with an inner radius fraction of 0.875. Even at 0 EFPD

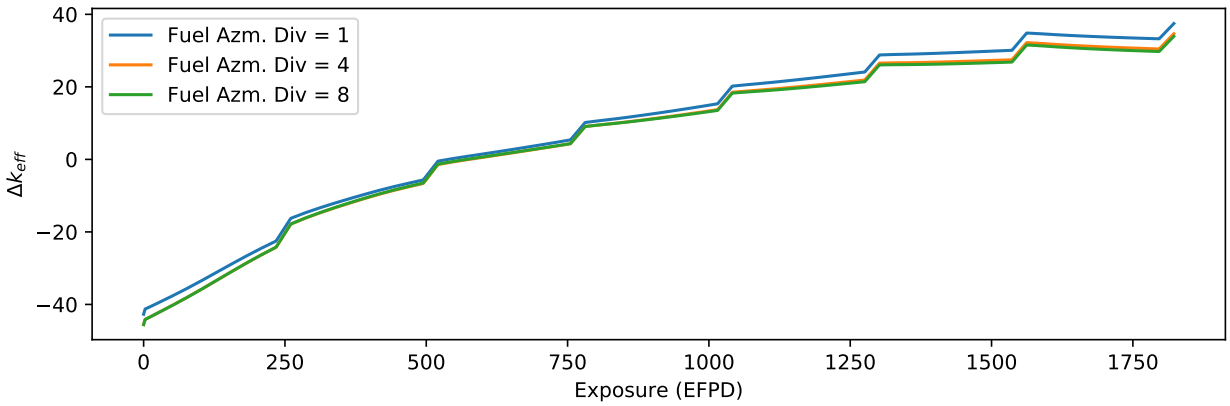


Figure 5.15: VERA Problem 2A eigenvalue comparison for varied fuel azimuthal divisions.

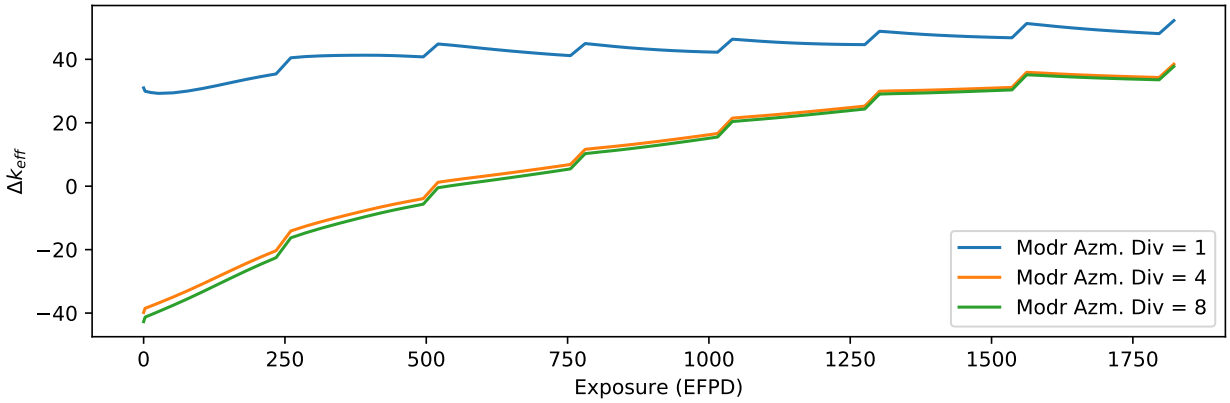


Figure 5.16: VERA Problem 2A eigenvalue comparison for varied number of surrounding moderator azimuthal divisions in the fuel cells.

(before Plutonium production), the eigenvalue difference can be large for the 2-ring models; this is hypothesized to be due to self-shielding affects.

Next, the varied azimuthal divisions in the fuel were tested; this time testing a single azimuthal division. Figure 5.15 shows that there is a noticeable affect moving to a single azimuthal division; however, the mean difference is changed by less than 1 pcm, and the max difference is actually improved by 3 pcm. We acknowledge that this improvement is likely due to cancellation of errors. Regardless, the net effect of this change is small.

2A - Moderator Azimuthal Divisions The azimuthal divisions of the surrounding moderator seems to have a more significant effect on the accuracy of the calculation. As shown by Fig. 5.16, using 4 azimuthal divisions seems to be sufficient; but moving to a single division is not feasible. A single division more than doubles the mean eigenvalue difference, and shows a consistent positive bias.

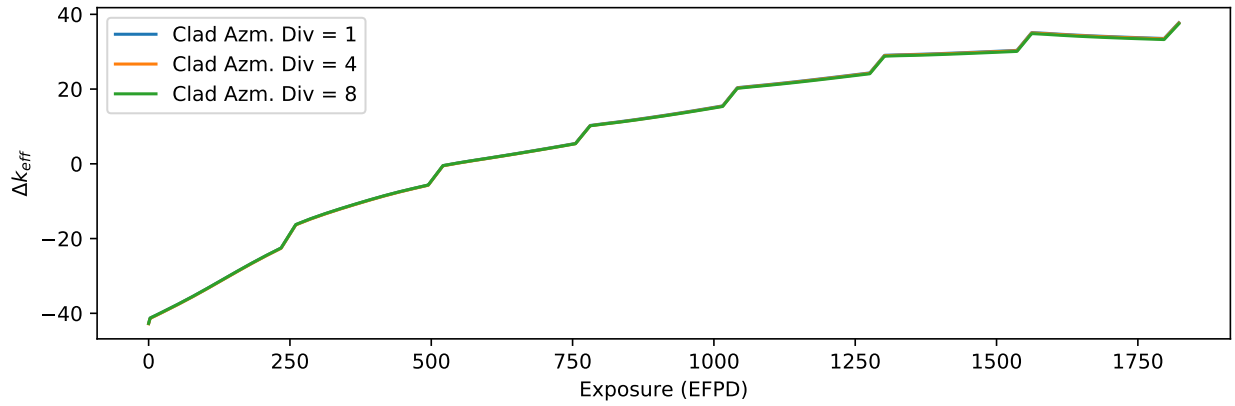


Figure 5.17: VERA Problem 2A eigenvalue comparison for varied number of the cladding/gap azimuthal divisions in the fuel cells.

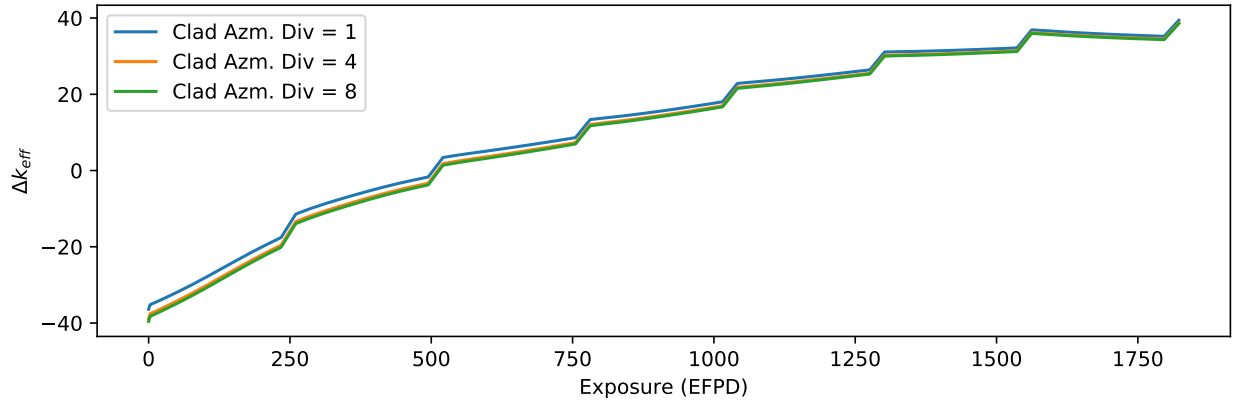


Figure 5.18: VERA Problem 2A eigenvalue comparison for varied number of inner moderator azimuthal divisions in the guide-tube cells.

2A - Cladding and Gap Azimuthal Divisions The azimuthal divisions of the cladding and gap regions do not seem to play a significant role. The difference is not even visually apparent in a graph of the eigenvalue, as seen in Fig. 5.17.

2A - Guide-Tube Azimuthal Divisions In the 2A lattice, guide-tube pins are present; each guide-tube consists of three regions: inner moderator, cladding, and outer moderator. The effect of different azimuthal divisions in each of these regions was tested; the resulting eigenvalues are shown in Figs. 5.18 to 5.20. A single azimuthal division in each of these regions seems to be sufficient, with no significant affect on eigenvalue accuracy. However, 4 azimuthal divisions in the surrounding moderator was selected for moving forward, due to compatibility issues with CTF.

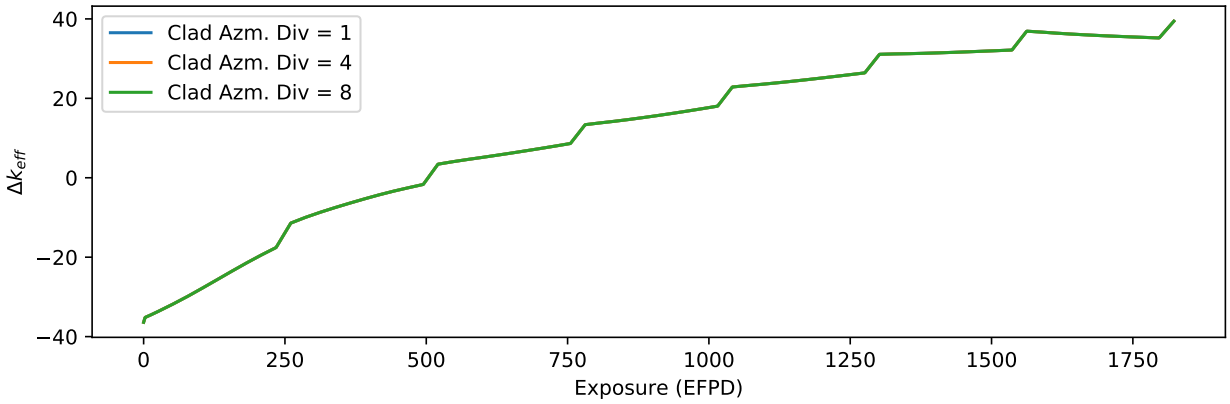


Figure 5.19: VERA Problem 2A eigenvalue comparison for varied number of cladding azimuthal divisions in the guide-tube cells.

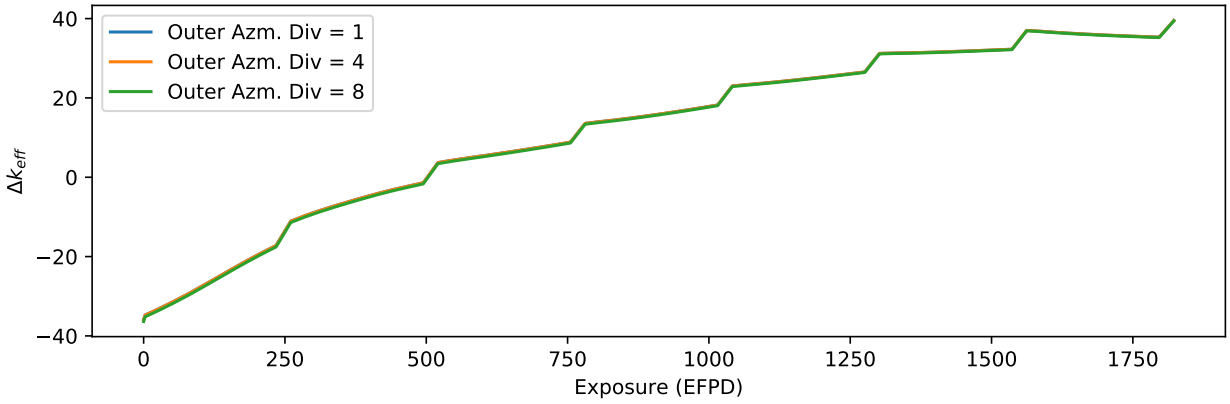


Figure 5.20: VERA Problem 2A eigenvalue comparison for varied number of outer moderator azimuthal divisions in the guide-tube cells.

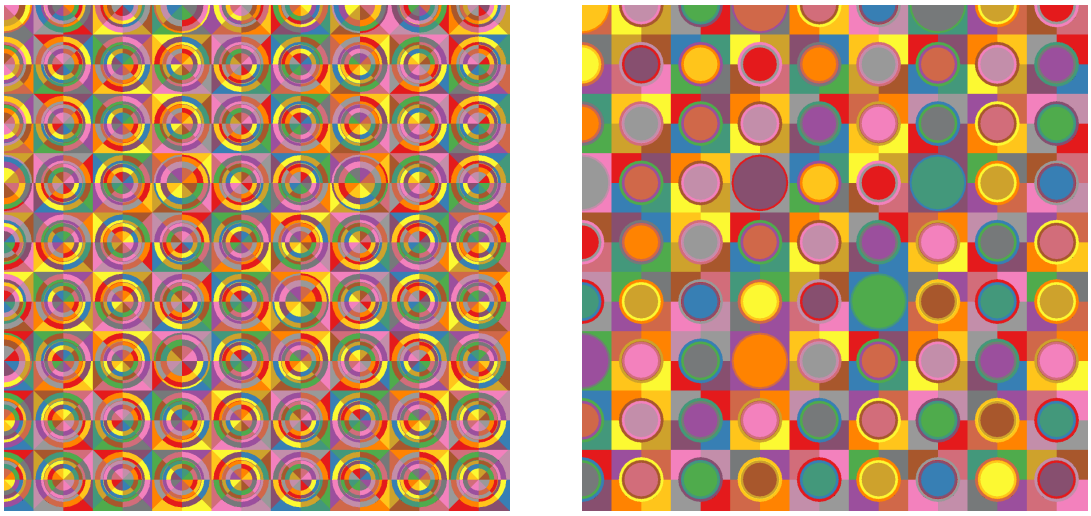
2A - Coarse Mesh Summary Coarse mesh parameters were found using this isotopic depletion calculation on VERA problem 2A. The fuel cell

- has 2 fuel rings, with the inner radius being 0.875 fraction of the outer,
- no additional moderator ring in the surrounding moderator,
- a single azimuthal division in the fuel, clad, and gap regions,
- and four azimuthal divisions in the surrounding moderator.

The guide-tube cell

- has no additional moderator ring in the surrounding moderator,
- has a single azimuthal division for the inner moderator, and cladding,
- and four azimuthal divisions for the surrounding moderator.

The lattice meshes are shown in Fig. 5.21.



(a) default mesh (b) coarse mesh

Figure 5.21: VERA problem 2A (a) default and (b) coarse meshes.

Eigenvalue comparisons are shown in Fig. 5.22, and pin power comparisons are made in Fig. 5.23. While it is visually apparent that the accuracy is made slightly worse than the default mesh LS calculation, the entire goal of the this mesh study was to maintain error with the FS calculation. This goal has been met, while reducing the number of cells from 3946 to 579.

5.3.3.2 Problem 2P

2P - Applying Coarse Mesh Parameters VERA problem 2P contains several gadolinia-enriched fuel rods; as the lattice is depleted, these fuel rods "burn" the gadolinia. The gadolinia is primarily burned away started at the outer radius and moving inwards as time progresses. This inward burning makes the pins difficult to model, and requires many radial divisions for accurate calculations; it is

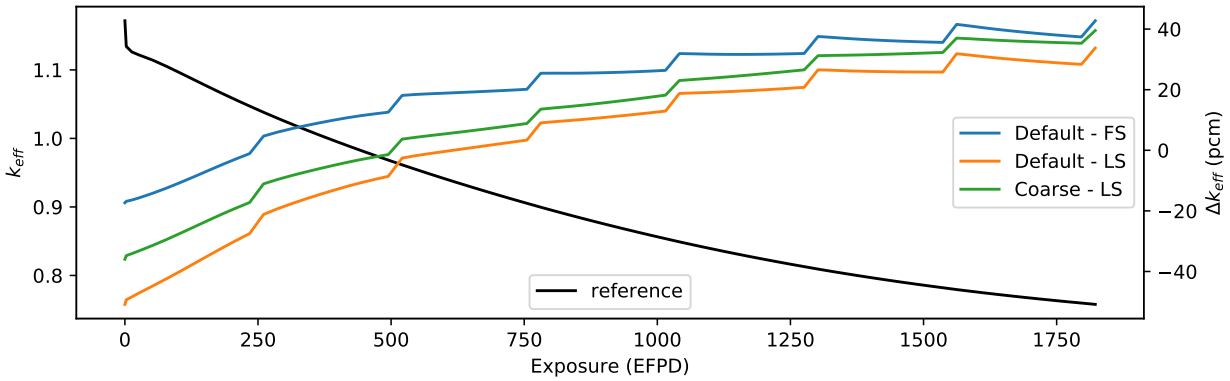


Figure 5.22: VERA Problem 2A coarse mesh eigenvalue comparison.

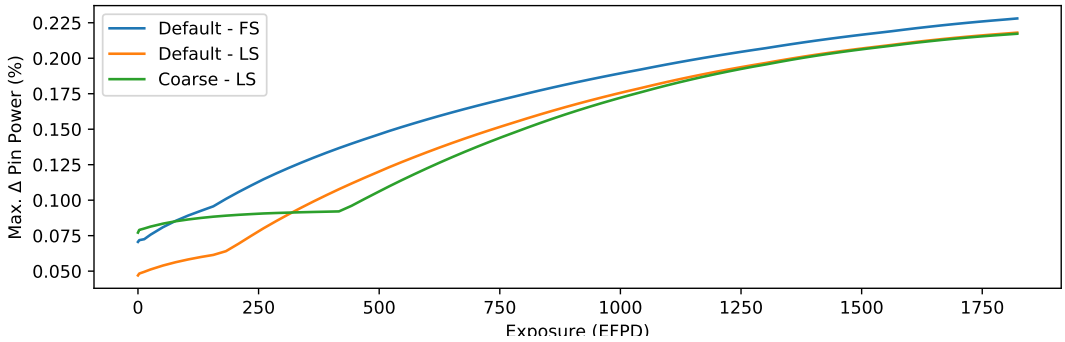


Figure 5.23: VERA Problem 2A coarse mesh pin power comparison.

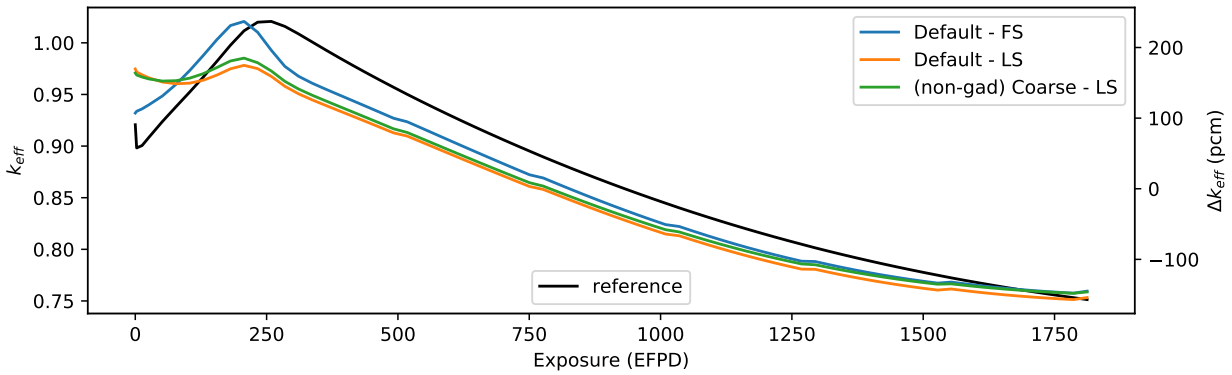


Figure 5.24: Eigenvalue comparisons for VERA problem 2P with default mesh parameters and coarse mesh parameters for the fuel and guide-tube cells.

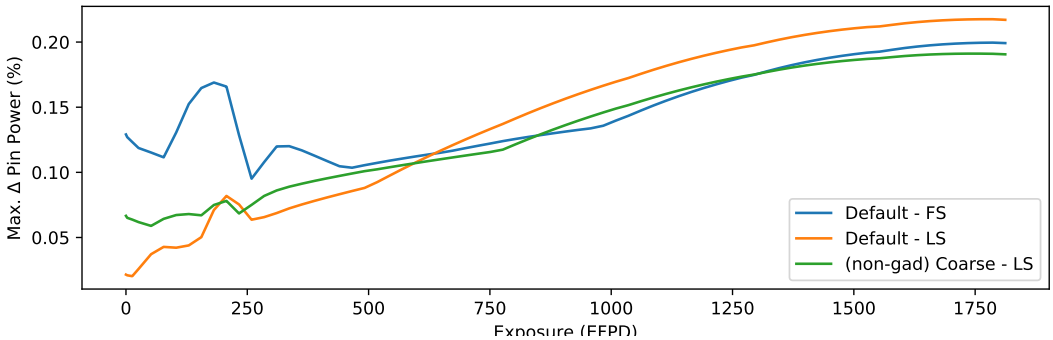


Figure 5.25: Pin power comparisons for VERA problem 2P with default mesh parameters and coarse mesh parameters for the fuel and guide-tube cells.

effectively a "moving rim effect". Additionally, because the presence of these small radial regions, a finer ray-spacing (0.01 cm) is necessary.

To evaluate if the previously determined coarse mesh parameters are sufficient in this problem, they are applied to the regular fuel and guide-tube cells, while the gadolinia rods use the default meshing parameters. Figures 5.24 and 5.25 show that applying the coarse mesh parameters to the fuel and guide-tube cells does not significantly worsen the eigenvalue or pin power results. It is worth observing that errors in this cases are significantly higher than in the problem 2A; this is likely due to the complicated radial dependence of the gadolinia rods during depletion.

2P - Azimuthal Divisions It is not expected that the radial divisions in the gadolinia rods can be significantly coarsened. However, this is not true for the azimuthal divisions. We apply the same azimuthal divisions as we did for the fuel: 1 in the fuel, clad, and gap, and 4 in the surrounding moderator. Figure 5.26 shows the eigenvalue results are similar at all but the initial state to the previous coarse mesh parameters. Even at the initial state, the error is between the flat and linear

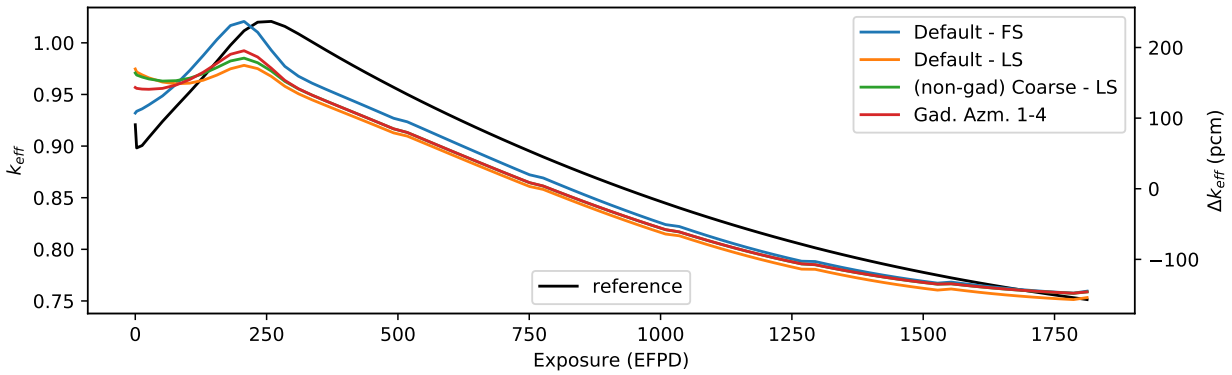


Figure 5.26: Eigenvalue comparisons for VERA problem 2P for the default mesh, previous coarse mesh, and gadolinia rods with fewer azimuthal divisions.

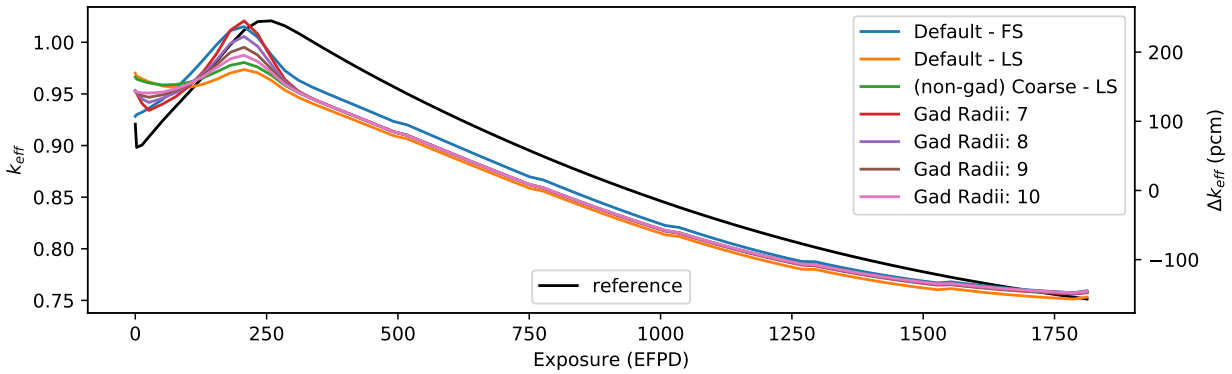


Figure 5.27: Eigenvalue comparisons for VERA problem 2P for varied number of radii in gadolinia rods.

source calculations on the default mesh. Thus, it is "acceptable" to coarsen the azimuthal divisions in the gadolinia pins.

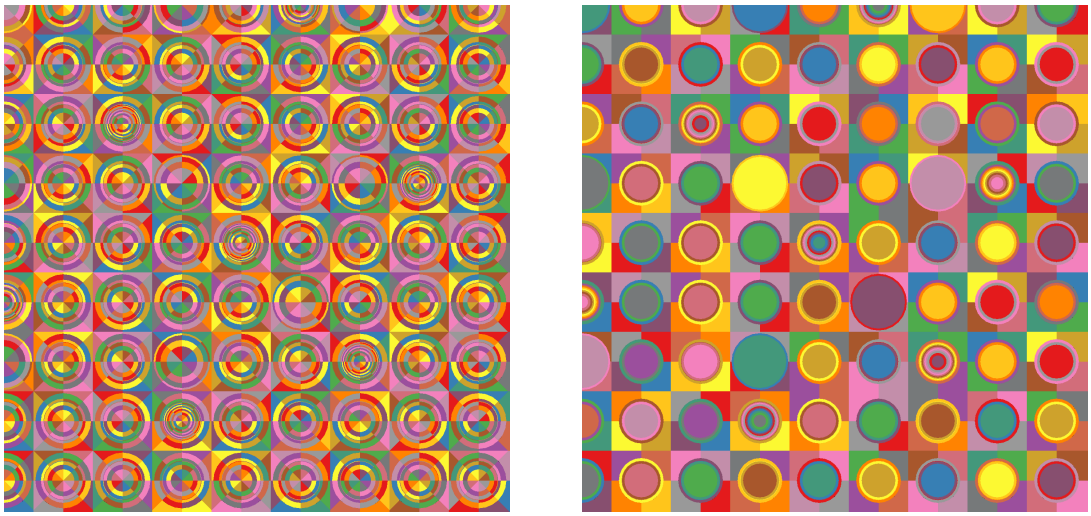
2P - Radial Divisions While it is unexpected that significant radial coarsening can occur in these pins, it is best to test this assumption. Figure 5.27 shows the eigenvalue errors for several different numbers of equal volume radii in the gadolinia fuel rods. It is clear that the number of radii can significantly affect the accuracy. For 8 radial divisions, the error is higher near the reactivity peak, it is still between the flat and linear source calculation errors on the default mesh. The goal of these mesh reduction studies was not to preserve the accuracy of the linear source on the default mesh, but to nearly match the accuracy of the flat source calculation on the default mesh. For this goal, 8 radial divisions seems to be sufficient.

2P - Coarse Mesh Summary The coarse mesh parameters for the fuel and guide-tube cells previously found were also found to be sufficient for VERA problem 2P. For the gadolinia rods, it

was found to be sufficient to

- have no additional moderator ring,
- use 8 equal volume fuel radial fuel divisions,
- a single azimuthal division in the fuel, clad, and gap regions,
- and four azimuthal divisions in the surrounding moderator.

Eigenvalue comparisons are shown in Fig. 5.29, and pin power comparisons are made in Fig. 5.30. Using these coarse mesh parameters reduces the number of cells from 4282 to 621. The lattice meshes are visualized in Fig. 5.28.



(a) default mesh (b) coarse mesh

Figure 5.28: VERA problem 2P (a) default and (b) coarse meshes.

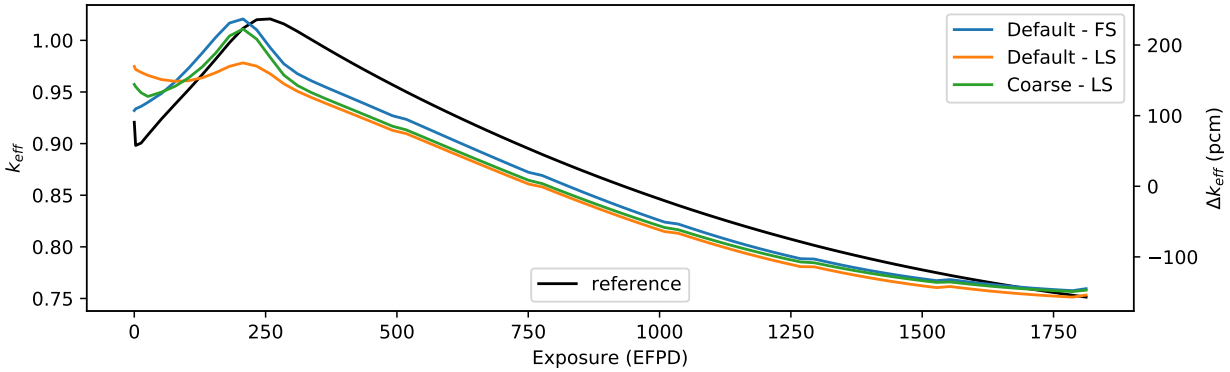


Figure 5.29: VERA Problem 2P coarse mesh eigenvalue comparison.

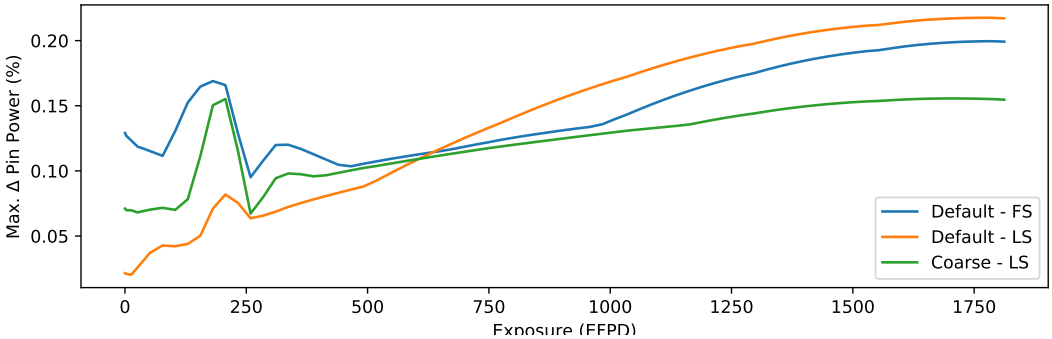


Figure 5.30: VERA Problem 2P coarse mesh pin power comparison.

5.3.3.3 Performance

Several performance metrics for the 2A and 2P lattice depletion calculations are listed in Table 5.6. For these two cases, compared to the default FSA calculation, the coarse mesh LSA calculation reduced total run-time by 17% and 11%. Total memory requirements were reduced by 14% and 9.3%, and also shows there is very little memory overhead from the use of the LSA. Interestingly, the MoC calculations seem to be slowed in these cases with run-times increased by 7% and 6%. The majority of the time saved comes from the reduction in the depletion calculation time; this time is reduced because with the coarse mesh, there are fewer material regions. The material region reduction comes only from the elimination of radial rings, not the azimuthal coarsening.

Table 5.6: VERA Problems 2A and 2P: Performance metrics.

			Time (s)					
Problem	Solver	Mesh	Total	MoC	calcMacro	CMFD	Depl	Memory (MB)
2A	FSA	Default	665	92	76	102	328	232
	FSA	Coarse	483	55	53	98	227	196
	LSA	Default	845	219	79	108	344	255
	LSA	Coarse	553	98	56	104	238	200
2P	FSA	Default	1370	548	101	202	415	558
	FSA	Coarse	1070	401	79	196	313	504
	LSA	Default	1834	965	103	200	428	591
	LSA	Coarse	1217	579	74	190	294	506

5.3.4 2-D Lattices

To evaluate the effect on accuracy these these coarse mesh parameters result in, they were applied to the VERA problem 2 lattice series. This series contains 17 common lattice configurations, and it is

expected that if the meshing parameters are sufficient for these cases, they will be sufficient in most applications. The eigenvalue and pin-power results compared to a very finely meshed reference case are shown in Table 5.7 and Table 5.8, respectively.

On average, the LS on the coarse mesh, using the parameters found in the previous sections, had accuracy comparable to the FS on the default mesh. Additionally, the worst case errors were lower for the LS on the coarse mesh. These results also show that the FS on the coarse mesh is not sufficient, with significantly worse eigenvalue and pin-power comparisons.

The run-time per iteration and number of iterations are listed in Table 5.9 and Table 5.10, respectively. In most cases, the number of iterations is approximately the same; however, the LSA solver often takes an additional iteration or two to converge for the same problem. As discussed in Section 5.3.1 and iteration scheme which performs more thermal-group sweeps, as is done in CASMO5, may help to improve performance of both solvers [11]. In terms of computational performance, the LSA solver reduces total run-time per iteration by 7% on average.

Table 5.7: VERA Problem 2: Eigenvalue errors for each mesh and source approximation.

	Δk_{eff} (pcm)			
Case	FS Default	LS Default	FS Coarse	LS Coarse
A	-17	-51	44	-36
B	-15	-49	46	-34
C	-17	-51	44	-36
D	-25	-59	39	-41
E	-49	-40	-52	-27
F	-74	-32	-120	-26
G	-98	-45	-210	-43
H	-78	17	-206	18
I	2	-43	82	-24
J	-74	-32	-119	-26
K	-61	-26	-97	-19
L	60	37	105	52
M	74	54	124	71
N	-1	31	-45	43
O	-21	9	-109	7
P	-115	-42	-311	-64
Q	-12	-48	54	-30
AVG	47	39	106	35
MAX	115	59	311	71

Table 5.8: VERA Problem 2: Maximum pin-power errors for each mesh and source approximation.

	Max Pin Power Difference (%)			
Case	FS Default	LS Default	FS Coarse	LS Coarse
A	0.07	0.05	0.24	0.08
B	0.07	0.05	0.24	0.08
C	0.07	0.05	0.24	0.08
D	0.07	0.05	0.24	0.08
E	0.12	0.05	0.37	0.08
F	0.15	0.05	0.51	0.08
G	0.19	0.09	0.59	0.10
H	0.25	0.15	0.58	0.15
I	0.09	0.05	0.29	0.08
J	0.10	0.05	0.34	0.08
K	0.17	0.06	0.55	0.08
L	0.13	0.07	0.35	0.12
M	0.11	0.06	0.26	0.11
N	0.18	0.05	0.51	0.11
O	0.15	0.06	0.43	0.08
P	0.17	0.08	0.69	0.14
Q	0.08	0.05	0.28	0.08
AVG	0.13	0.06	0.39	0.09
MAX	0.25	0.15	0.69	0.15

Table 5.9: VERA Problem 2: Number of iterations.

	Iterations			
Case	FS Default	LS Default	FS Coarse	LS Coarse
A	10	10	9	10
B	10	10	9	10
C	10	10	9	10
D	10	10	9	10
E	12	12	11	12
F	12	13	11	13
G	12	12	11	12
H	12	12	11	12
I	10	11	10	11
J	12	13	11	13
K	12	13	11	13
L	8	10	8	10
M	8	10	8	10
N	13	13	12	13
O	12	12	11	12
P	12	12	11	12
Q	10	10	9	11

Table 5.10: VERA Problem 2: Run-time per iteration.

Case	Run-time per iteration (s)			
	FS Default	LS Default	FS Coarse	LS Coarse
A	0.56	0.76	0.46	0.51
B	0.56	0.77	0.46	0.52
C	0.56	0.77	0.46	0.51
D	0.56	0.77	0.45	0.51
E	0.53	0.74	0.43	0.52
F	0.54	0.76	0.44	0.51
G	0.57	0.78	0.49	0.55
H	0.56	0.79	0.47	0.55
I	0.56	0.77	0.44	0.50
J	0.54	0.75	0.44	0.51
K	0.54	0.75	0.45	0.51
L	1.57	2.07	1.20	1.43
M	1.58	2.10	1.19	1.44
N	1.46	2.09	1.16	1.49
O	0.56	0.78	0.45	0.51
P	0.56	0.78	0.44	0.51
Q	0.63	0.99	0.48	0.52

htbp

Table 5.11: VERA Problem 2: Mesh and ray-segment metrics.

Case	# Source Regions			# Ray-Segments		
	Default	Coarse	Reduction	Default	Coarse	Reduction
A	3946	595	85%	715699	402620	44%
B	3946	595	85%	715699	402620	44%
C	3946	595	85%	715699	402620	44%
D	3946	595	85%	715699	402620	44%
E	4102	851	79%	728469	428444	41%
F	4258	1107	74%	741268	454290	39%
G	4114	963	77%	733619	446641	39%
H	4114	963	77%	733204	446226	39%
I	3951	607	85%	715923	403726	44%
J	4263	1119	74%	741492	455396	39%
K	4258	1107	74%	741268	454290	39%
L	4106	675	84%	3632023	2097604	42%
M	4202	723	83%	3695367	2166086	41%
N	4414	1125	75%	3792640	2367134	38%
O	4114	613	85%	730899	415100	43%
P	4282	637	85%	745991	427608	43%
Q	3946	595	85%	715699	402620	44%

5.3.4.1 Coarse Rays

In each of the these lattice calculations, the MoC calculation takes up the majority of the runtime. The MoC runtime is directly proportional to the number of track-segments that are generated during the ray-tracing. This number is reduced as the mesh becomes coarser; however, due to material and geometric limitations, the mesh can only be coarsened so much. Another parameter that affects the number of track-segments is the ray-spacing, or how far apart each ray is. The rays can be spaced more coarsely, but this may cause some regions to not be integrated as accurately, thus degrading overall accuracy.

In this section, the use of coarser ray-spacing is investigated for the VERA problem 2 series. Results were generated using a ray-spacing of 0.1 cm and 0.2 cm for the default mesh with the FS solver, and for the coarse spatial mesh with LS solver. Eigenvalue and pin power comparisons are shown in Table 5.12 and Table 5.13, respectively. In most of the cases, it is possible to use a reasonably coarser ray-spacing (0.10 cm) without incurring significant error. The original hypothesis of this work was that using a coarser mesh would allow for the use of coarser rays; however, Tables 5.12 and 5.13 show that this is not the case. The determining factor for whether or not coarser rays can be used is the presence of fine material discretizations in some rods. However, Table 5.15 shows that the use of a coarse mesh and coarse rays reduces the number of ray-segments (from the default parameters) by a much more significant fraction than simply using a coarser spatial mesh; meaning the expected run-time will be considerably lower.

5.3.5 Assembly with Feedback

VERA problem 6 is a single 3-D PWR assembly with T/H feedback [12], as visualized in Fig. 5.31. COBRA-TF (CTF) [15] is a T/H code that can be coupled to the neutronics solver in MPACT to perform multiphysics calculations. The coarse meshing parameters from the previous sections were applied to this problem; these results will help to determine whether or not the coarse mesh parameters are also sufficient when T/H feedback is present. As a 3-D assembly, this problem has lower and upper plates, and nozzles. These elements were models as rectilinear grids of 0.42×0.42 cm^2 sized cells.

This problem was run with the default and coarse meshes using the FSA and LSA solvers. A parallel spatial decomposition was applied into 58 sub-domains, each consisting of a single axial plane. Each case was compared against a very finely meshed case run using the LSA solver. Each case used a Tabuchi-Yamamoto quadrature with 64 azimuthal angles and 4 polar angles, with a uniform ray-spacing of 0.05 cm. An additional case was run using the coarse mesh, LSA solver, and 0.1 cm ray-spacing.

Results are summarized in Table 5.16, and performance metrics are listed in Table 5.17. It is

Table 5.12: VERA Problem 2: Eigenvalue errors with coarse-rays.

Case	Δk_{eff} (pcm)			
	FS (0.1 cm)	FS (0.2 cm)	LS (0.1 cm)	LS (0.2cm)
A	42	34	21	18
B	41	33	19	16
C	42	34	21	18
D	45	38	23	20
E	4	-59	25	-29
F	-22	-32	26	28
G	27	73	98	162
H	-37	71	59	180
I	54	45	25	22
J	-22	-31	26	28
K	-5	-15	37	38
L	-30	-171	70	-40
M	-140	-220	-43	-15
N	-95	-227	23	-103
O	4	-36	27	-21
P	-7	-35	39	-7
Q	43	35	22	19
AVG	39	70	36	45
MAX	140	227	98	180

Table 5.13: VERA Problem 2: Maximum pin-power errors with coarse-rays.

Case	Δk_{eff} (pcm)			
	FS (0.1 cm)	FS (0.2 cm)	LS (0.1 cm)	LS (0.2cm)
A	0.15	0.45	0.10	0.33
B	0.15	0.45	0.10	0.33
C	0.15	0.45	0.10	0.33
D	0.15	0.45	0.10	0.33
E	0.22	0.28	0.09	0.28
F	0.19	0.31	0.09	0.42
G	0.31	0.49	0.24	0.44
H	0.35	0.52	0.24	0.51
I	0.16	0.47	0.10	0.32
J	0.19	0.31	0.08	0.42
K	0.20	0.33	0.10	0.43
L	0.79	0.90	0.80	1.30
M	1.08	0.94	1.11	1.20
N	0.91	0.85	0.98	0.98
O	0.25	0.62	0.14	0.43
P	0.30	0.61	0.20	0.40
Q	0.15	0.46	0.11	0.32
AVG	0.34	0.52	0.28	0.52
MAX	1.08	0.94	1.11	1.30

Table 5.14: VERA Problem 2: Run-time per iteration with coarse rays.

	Run-time per iteration (s)			
Case	FS (0.1 cm)	FS (0.2 cm)	LS (0.1 cm)	LS (0.2cm)
A	0.46	0.41	0.43	0.37
B	0.47	0.42	0.41	0.38
C	0.46	0.41	0.42	0.37
D	0.47	0.41	0.42	0.37
E	0.43	0.38	0.41	0.40
F	0.49	0.39	0.42	0.36
G	0.47	0.42	0.44	0.40
H	0.47	0.43	0.45	0.40
I	0.46	0.40	0.41	0.37
J	0.45	0.40	0.42	0.42
K	0.49	0.43	0.45	0.37
L	0.49	0.45	0.46	0.38
M	0.47	0.43	0.41	0.35
N	0.45	0.39	0.43	0.38
O	0.46	0.41	0.42	0.37
P	0.46	0.43	0.46	0.40
Q	0.61	0.44	0.43	0.37

Table 5.15: VERA Problem 2: Number of ray-segments with coarse rays. Also shows reduction in number of segments from the default mesh with default ray parameters.

	# Segments (% Reduction from default)			
Case	Default (0.1 cm)	Default (0.2 cm)	Coarse (0.1 cm)	Coarse (0.2cm)
A	371468 (48%)	187746 (74%)	208804 (71%)	105532 (85%)
B	371468 (48%)	187746 (74%)	208804 (71%)	105532 (85%)
C	371468 (48%)	187746 (74%)	208804 (71%)	105532 (85%)
D	371468 (48%)	187746 (74%)	208804 (71%)	105532 (85%)
E	378076 (48%)	191152 (74%)	222210 (69%)	112348 (85%)
F	384682 (48%)	194434 (74%)	235618 (68%)	119092 (84%)
G	380680 (48%)	192394 (74%)	231616 (68%)	117052 (84%)
H	380484 (48%)	192300 (74%)	231420 (68%)	116958 (84%)
I	371564 (48%)	187786 (74%)	209360 (71%)	105804 (85%)
J	384778 (48%)	194474 (74%)	236174 (68%)	119364 (84%)
K	384682 (48%)	194434 (74%)	235618 (68%)	119092 (84%)
L	382584 (89%)	193342 (95%)	220850 (94%)	111590 (97%)
M	389264 (89%)	196682 (95%)	228044 (94%)	115230 (97%)
N	399506 (89%)	201840 (95%)	249250 (93%)	125894 (97%)
O	379328 (48%)	191718 (74%)	215300 (71%)	108824 (85%)
P	387276 (48%)	195650 (74%)	221776 (70%)	112124 (85%)
Q	371468 (48%)	187746 (74%)	208804 (71%)	105532 (85%)

quite clear that the LSA solver on the coarse mesh is at least as accurate, for this problem, as the FSA on the default mesh. Additionally, the FSA on the coarse mesh performs significantly worse in every metric of Table 5.16. Finally, it was shown that for this problem the use of coarser ray-spacing was possible and had very little affect on the results. Problem 6 has no inserted control rods or other finely meshed elements, and this is likely why using coarsely spaced rays is possible in this case.

Table 5.17 might seem to imply that the LSA solvers outperformed the FSA solvers on the same mesh. This is not the case, as the LSA cases required fewer iterations to converge. The iterations may differ due to non-uniform iterative convergence [2] that has previously been observed in problems with CMFD acceleration and feedback [16]. If the MoC run-time per iteration is compared, the LSA solver takes 79% longer than the FSA solver on the default mesh, but only 35% longer on the coarse mesh.

Because CTF takes far longer in the first several iterations, it is not fair to compare total run-time per iteration. Rather, we can look at the time that the FSA solver cases took to complete the same number of iterations the LSA solver cases needed. The times to complete 9 iterations are shown in Table 5.17. These results still show that the use of the LSA on the coarse mesh is advantageous compared to the default mesh with FSA. The coarse mesh LSA case reduced the run-time for 9

[2] due to complex eigenvalues in the spectral radius

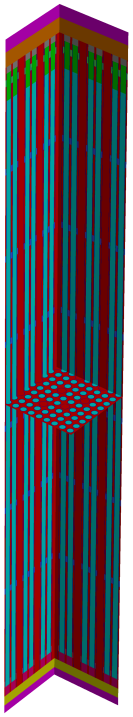


Figure 5.31: VERA Problem 6: Geometry.

iterations by 4.8%, but reduced MoC run-time per iteration by 9.3%. By also using coarse rays the run-time is reduced by 8.4% for total time, and 72.7% for MoC time per iteration.

The reduction in memory usage from using a coarser mesh is also an important aspect of the LSA solvers. The coarse mesh LSA case reduced total memory usage, compared to the default mesh FSA case, by 15.5%, and maximum process memory usage by 4.2%. By using coarser ray-spacing, the reduction in total memory is significantly higher at 33.6%, and 20.8% reduction in maximum process memory.

Figure 5.32 shows the radial shape of the pin-power for two axial planes of the references solution. These planes were chosen as the highest power plane, and the plane with the highest pin-power error in the calculation using the default mesh with FSA solver. The planes with the largest pin-power errors for each of the cases run are shown in Fig. 5.33. The pin-power errors in the case using the LSA with coarse mesh are shown to be lower than that using the FSA with default mesh.

The axial power shape can be found by taking the mean radial pin-power for each axial plane. This reference axial power shape, and the errors in it for each of the calculations run are shown in Fig. 5.34. Though the errors might seem low ($<$0.5% in the worst case), it should be remembered that these are the radially averaged power and thus a 0.5% error means the entire plane is mispredicted

by an average of 0.5%. It is also clear that the using the coarse mesh significantly worsens accuracy when using the FSA. However, the LSA calculation errors do not change significantly from changing from default to coarse mesh parameters. Additionally, the LSA calculations improve accuracy considerably compared to the FSA, with errors $<0.1\%$.

Table 5.16: Results for VERA problem 6 simulations using different source approximations and mesh parameters. MPPE is the maximum pin-power error, and MPT is the maximum pin temperature.

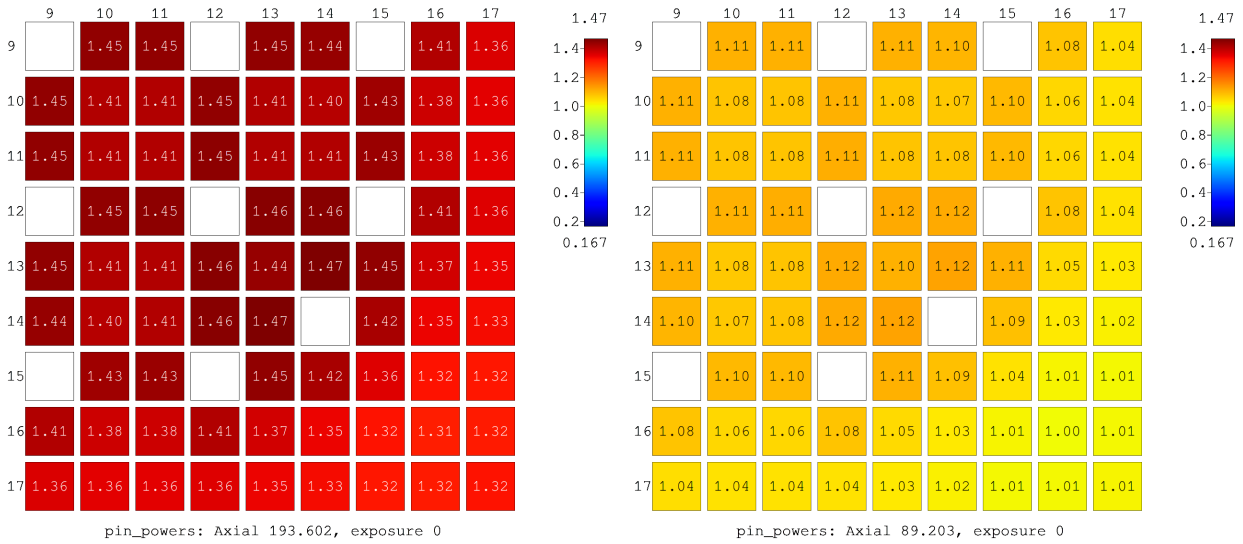
Solver	Mesh (Ray-Spacing)	Δk_{eff} (pcm)	MPPE (%)	Δ MPT (K)
FSA	Default (0.05 cm)	34	0.31	-0.9
LSA	Default (0.05 cm)	-7	0.08	-0.3
FSA	Coarse (0.05 cm)	109	0.85	-3.5
LSA	Coarse (0.05 cm)	13	0.09	0.0
LSA	Coarse (0.10 cm)	12	0.17	0.1

Table 5.17: Performance metrics for VERA problem 6 using different source approximations and mesh parameters.

			Run-time (s)			Memory (MB)	
Solver	Mesh (Ray-Spacing)	Iterations	Total	9 Iter.	MoC	Total	Max.
FSA	Default (0.05 cm)	11	391	393	16.1	12438	264
LSA	Default (0.05 cm)	9	390	399	23.6	13156	272
FSA	Coarse (0.05 cm)	11	379	379	10.8	10381	248
LSA	Coarse (0.05 cm)	9	366	374	14.6	10506	253
LSA	Coarse (0.10 cm)	9	354	360	4.4	7887	209

5.3.6 Core Depletion with Feedback

The VERA progression problem 9 is based on the Watts Bar reactor and models a typical 18-month power cycle, though with simplified power changes [12, 17]. Simplified changes in power are used such that transient calculations are not necessary. The power levels, inlet coolant temperature, and bank D position over the cycle depletion and are shown in Fig. 5.35. The radial materials are shown in Fig. 5.36. This problem models a 3-D quarter core with isotopic depletion and T/H feedback, and performs a critical boron search at each state. As with Section 5.3.5, this calculations performed in this section use the 2D/1D method [17, 18]. The Virtual Environment for Reactor Analysis Core Simulator (VERA-CS) has previously been used to simulate problem 9 using the traditional FSA on the default mesh in MPACT, and compared well against Watts Bar plant data [17]. In this section, the use of the coarse mesh with LSA solver will be compared against the previous results.



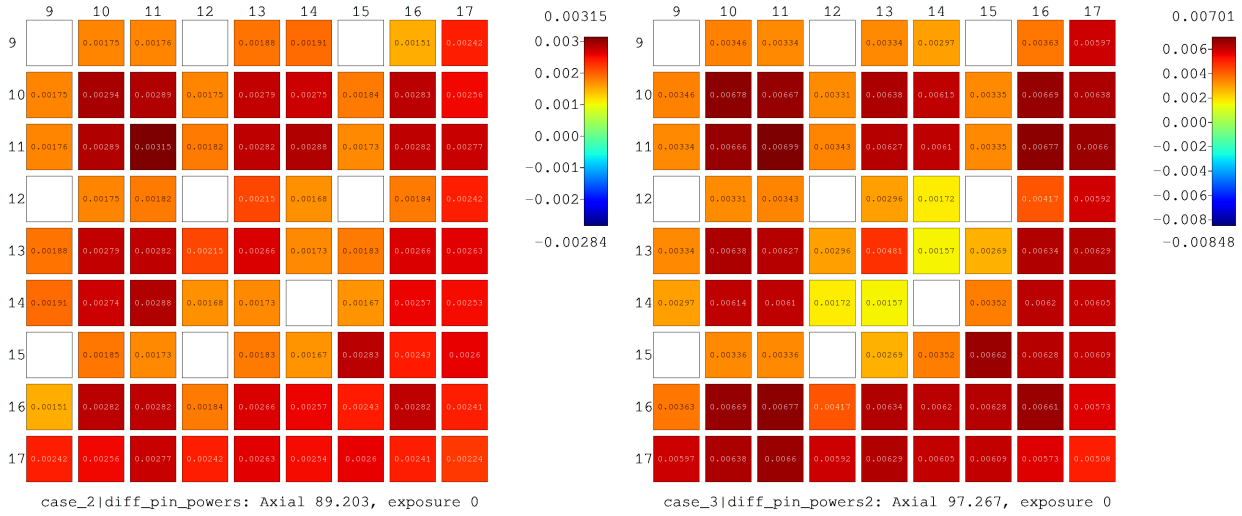
(a) Plane (193.6 cm) (b) Plane (89.2 cm)

Figure 5.32: Reference pin-powers from two planes of VERA problem 6.

The simulation used the graph partitioning spatial decomposition methods described in Chapter 4 with 692 spatial domains. A Tabuchi-Yamamoto quadrature set with 32 azimuthal and 4 polar angles over 4π was used with 0.05 cm ray-spacing. The TCP0 scattering approximation was also used in this case. These parameters are the same as those which were previously shown to yield results which agree closely with plant data [17]. The default mesh contains a total of 51513673 source-regions, while the coarse mesh contains a total of 14932511 source-regions. The initial coarse mesh results were generated using a fine reflector region, with coarse fuel rod meshes; this mesh had a total of 19743263 source-regions. These initial coarse mesh results were generated due to a mistake by the author, but these show close agreement in terms of accuracy with the actual coarse mesh results.

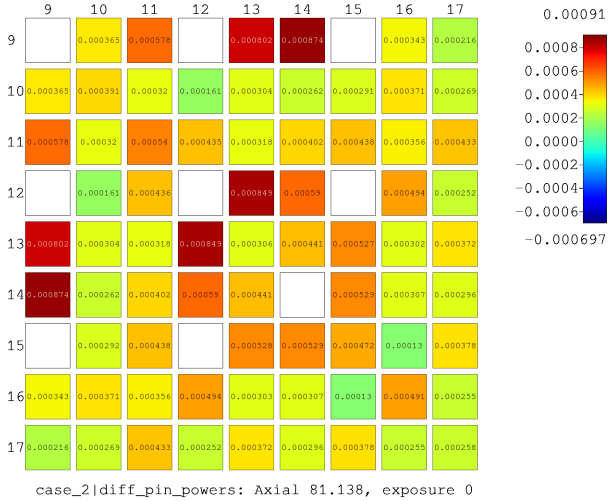
Transverse leakage splitting, sometimes used to help iteration stability (Section 2.4.3.2), was disabled for these simulations. In this problem, transverse leakage splitting was confirmed not to be necessary for convergence, and generally has negative impacts on accuracy. Additionally, the use of transverse leakage splitting caused instability when using the LSA. In MPACT, the splitting factor is determined by the average source in a region; however, when this is applied to the linear source, it guarantees that half the region will still have a negative source.

Figure 5.37 shows the reference critical boron levels and the errors of the coarse mesh solutions over the cycle. The FSA coarse mesh solution tends to over-predict the boron levels, while the LSA coarse mesh solution tends to under-predict. The maximum boron difference is about 3 ppm for the FSA solution, and about 1.75 for the LSA solution. Each calculation finds that the core becomes sub-critical at the same state; for the last five sub-critical states the maximum eigenvalues are within



(a) Default - FS

(b) Coarse - FS



(c) Coarse - LS

Figure 5.33: VERA problem 6, planes with largest pin power errors for different meshes and source approximations.

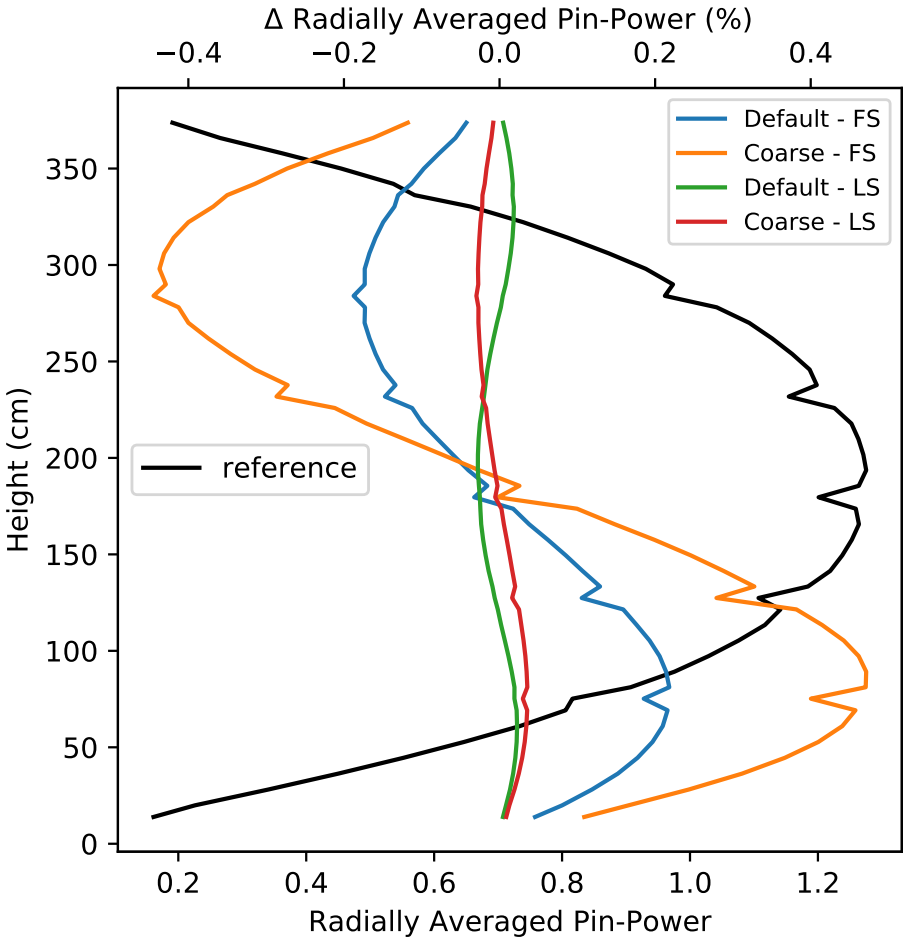


Figure 5.34: VERA problem 6: radially averaged pin-power errors for each axial plane using different meshes and source approximations. The reference axial power shape is also given.

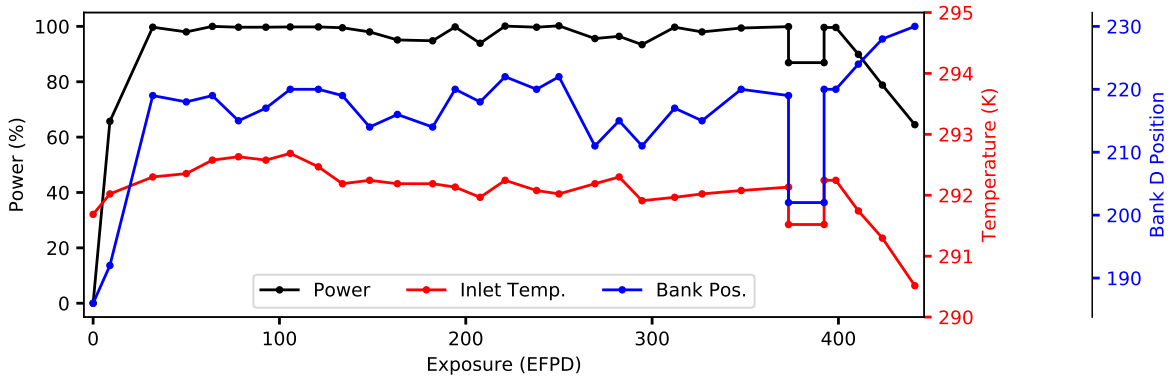


Figure 5.35: VERA problem 9 power, inlet temperature, and bank D position over the 18-month cycle.

10 pcm of the reference.

The maximum pin-power errors are shown in Fig. 5.38. The LSA shows clear benefits over the FSA, with the maximum pin-power difference being lower for all but two states where errors are similar. Additionally, the maximum error is within 1% using the LSA solver, but reaches over 1.2% for the FSA solver. Figure 5.39 shows the maximum pin temperature, and the errors for the coarse mesh cases. The errors for each of the coarse mesh solutions follows similar trends, but the FSA results seem to show a slight negative bias. Generally, the temperatures are within 2 K of the reference maximum.

Detailed profiles of the pin-powers are provided for the beginning, middle, and end of cycles in Figs. 5.40 to 5.42, respectively. A radial profile is shown for the mid-plane, and an axial profile is shown for the assembly which was hottest at the beginning of the cycle. The pin-power difference profiles for the coarse mesh solutions are shown in Figs. 5.43 to 5.45 for the FSA solver, and Figs. 5.46 to 5.48 for the LSA solver. At the beginning of the cycle, the FSA solver tends to over-predict power near the center of the core and under predict near the peripherals, while the LSA solver shows the opposite trend. These figures, along with Fig. 5.49, also show that using the radial LSA is able to better capture the power-shape axially.

Several performance metrics are listed in Table 5.18. For the calculations with 696 spatial domains, the coarse LSA uses 20% more core-hours than the default FSA calculation. However, the use of the coarse mesh saves a considerable amount of memory: 249 GB. Each of the coarse mesh cases required nearly 10% more iterations, likely because of iterative inefficiency from the spatial decomposition (see Chapter 4), which is different for the different meshes. In fact, even the FSA coarse mesh calculation was slightly slower than the FSA calculation on the default mesh even though the number of mesh elements had been significantly reduced.

These results have shown that the mesh that was found in the previous mesh-convergence study

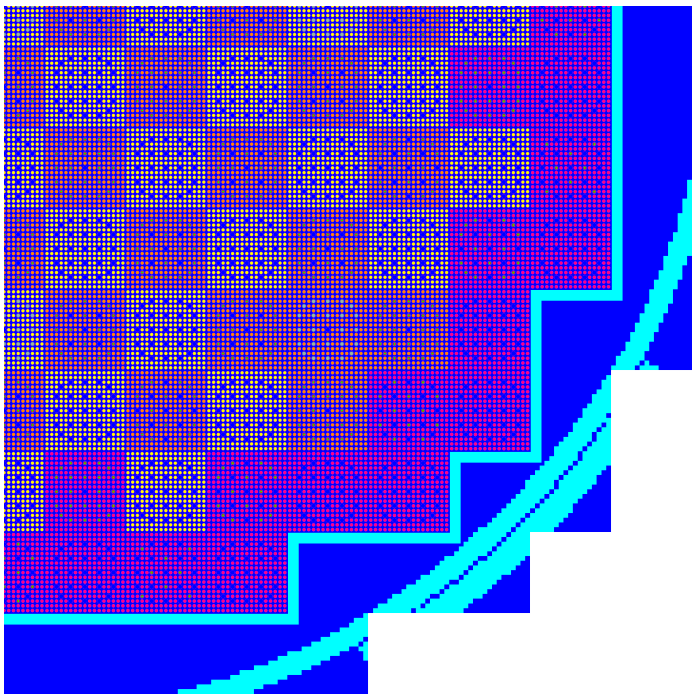


Figure 5.36: VERA problem 9: Visualization of the radial core materials.

Table 5.18: VERA problem 9: Performance metrics.

		Iterations				Memory (GB)		
Solver	Mesh	Outer	CTF	Total Time (hrs)	Core-Hours	Average	Max.	Total
FSA	Default	895	58180	57.4	39954	2.67	3.21	1858
FSA	Coarse[1]	969	58788	58.5	40681	2.16	3.00	1505
LSA	Coarse[1]	958	59167	68.7	47835	2.31	3.19	1609
LSA[3]	Coarse	963	59397	76.7	35607	2.79	3.25	1295

is sufficient for this quarter-core cycle depletion with T/H feedback. This, along with the results of previous sections, have shown that the coarse mesh parameters are sufficient for a large range of problems, and should be considered as new default parameters when using the LSA in MPACT. The use of the coarse mesh with the LSA was able to reduce the number of core-hours by 10.8%, and total memory usage by 30.3%. This represents significant advantage of the LSA even in simulations with multiple physics.

The results presented in this section did not use the recently added capability for radial power, temperature, and burnup between MPACT and CTF. In initial testing with the coarse mesh parameters, it appears that the capability assumed that each radial division in the fuel was equal volume, which is not the case for the coarse mesh parameters developed in this work. Further testing on the coarse mesh with LSA and radial coupling will be required once this assumption is lifted from the new capability.

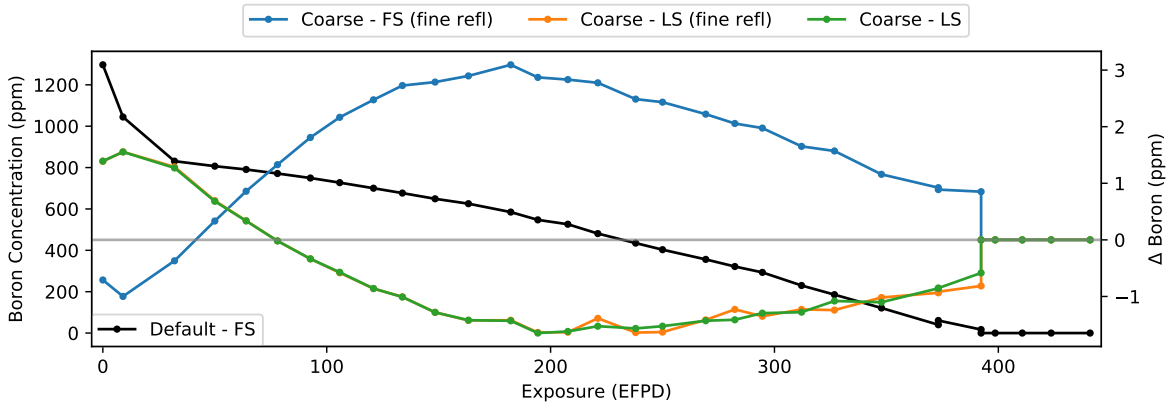


Figure 5.37: VERA problem 9: Critical boron levels for the FSA solver on the default mesh, and errors for the coarse mesh solutions.

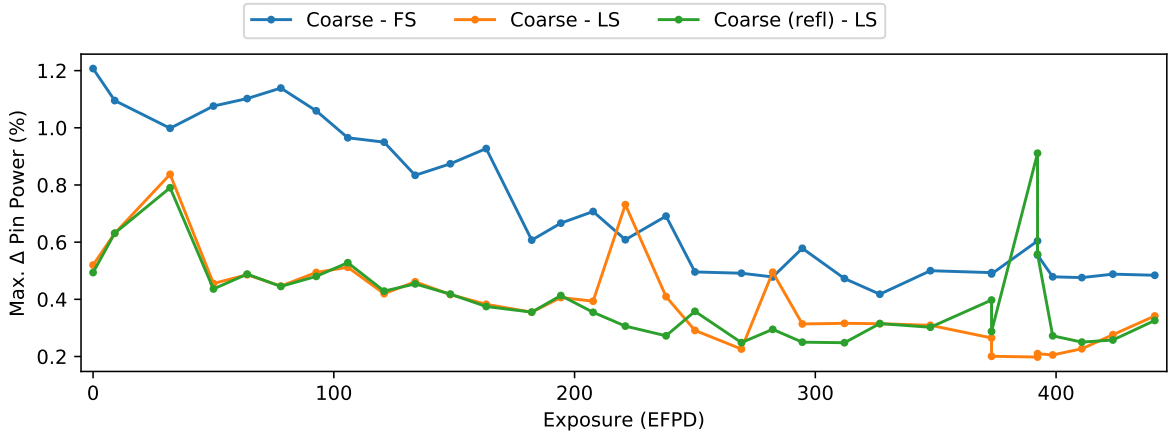


Figure 5.38: VERA problem 9: Maximum pin-power errors (compared to default mesh with FSA solver) for coarse mesh solutions.

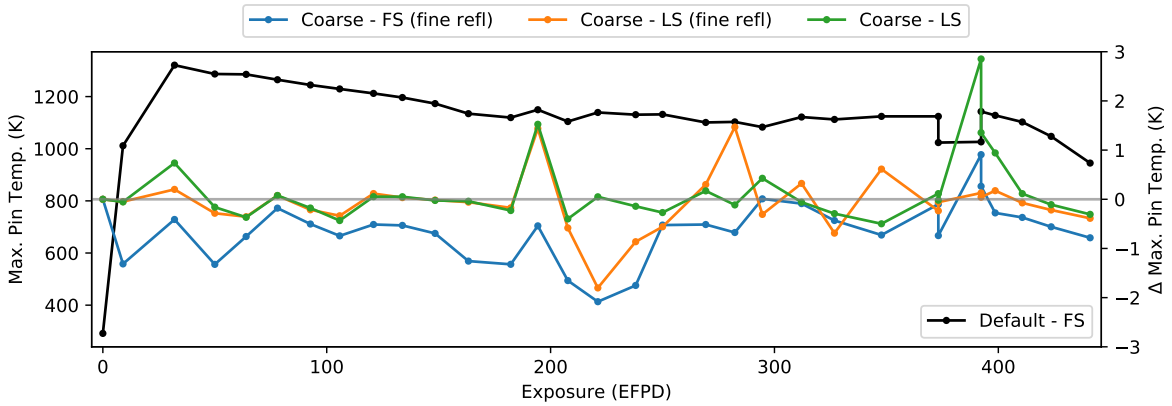
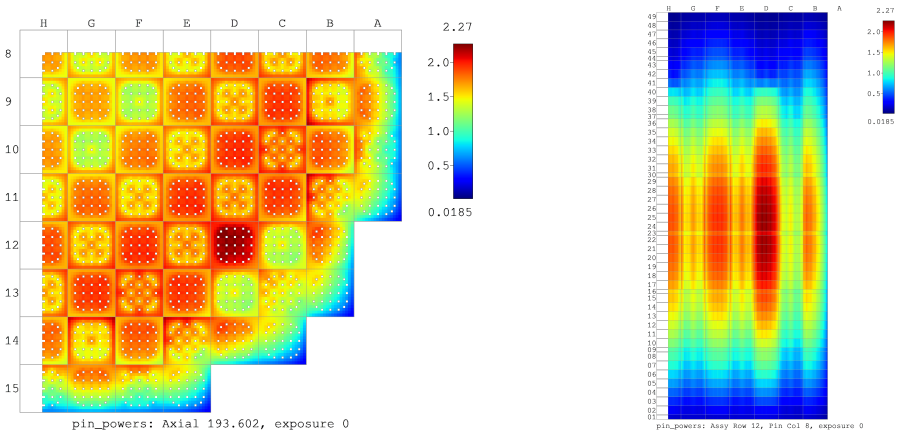
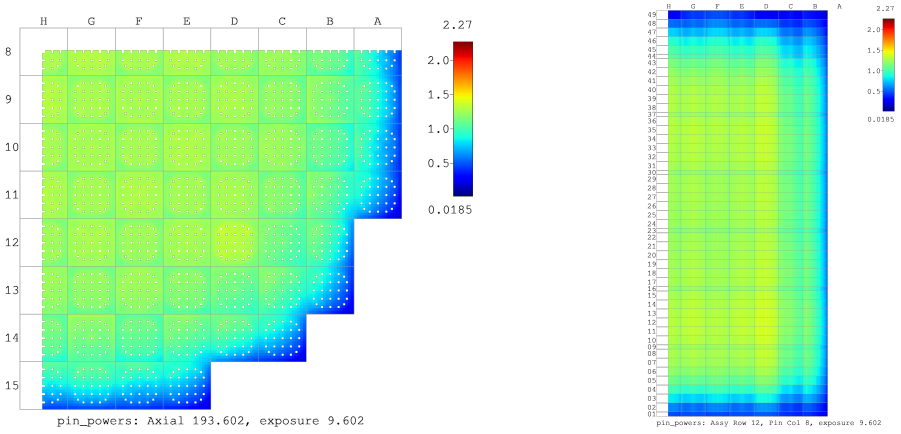


Figure 5.39: VERA problem 9: Maximum pin temperature for the FSA solver on the default mesh, and errors for the coarse mesh solutions.



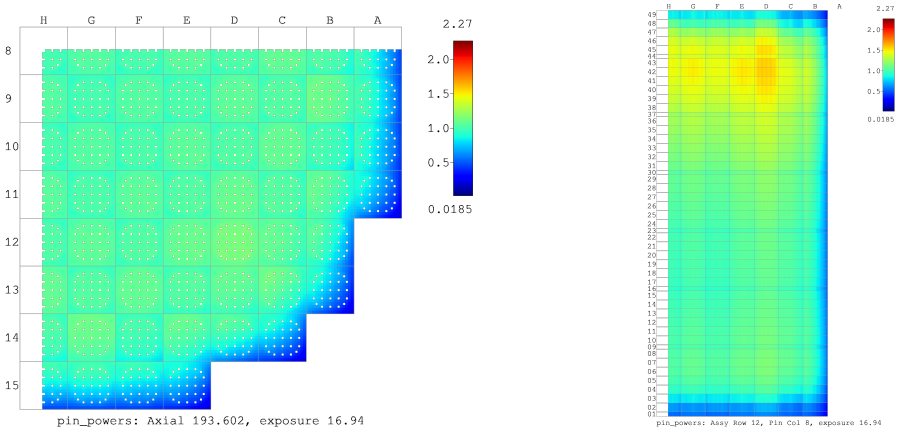
(a) Radial (b) Axial

Figure 5.40: VERA Problem 9: (Reference) Default mesh with FSA solver pin-powers for the beginning of the cycle.



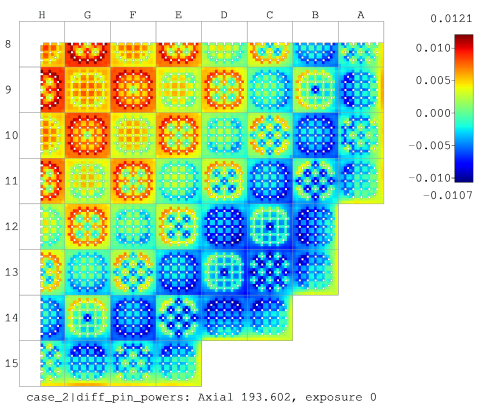
(a) Radial (b) Axial

Figure 5.41: VERA Problem 9: (Reference) Default mesh with FSA solver pin-powers for a state in the middle of the cycle.

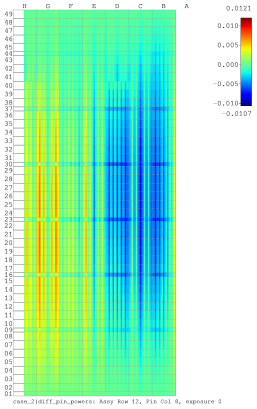


(a) Radial (b) Axial

Figure 5.42: VERA Problem 9: (Reference) Default mesh with FSA solver pin-powers for the end of the cycle.

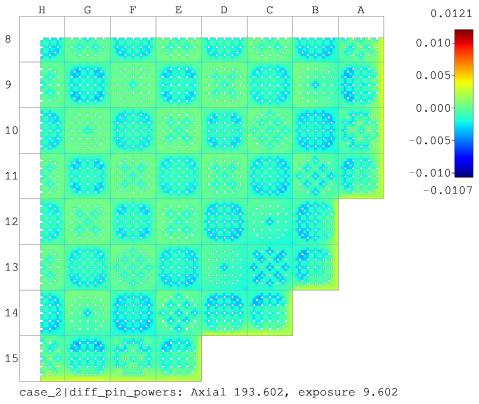


(a) Radial

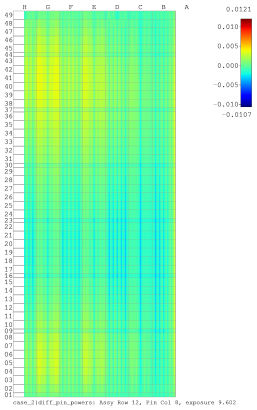


(b) Axial

Figure 5.43: VERA Problem 9: Coarse mesh with FSA solver pin-power errors for the beginning of the cycle.

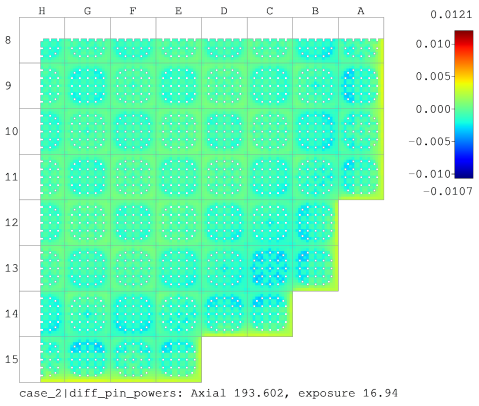


(a) Radial

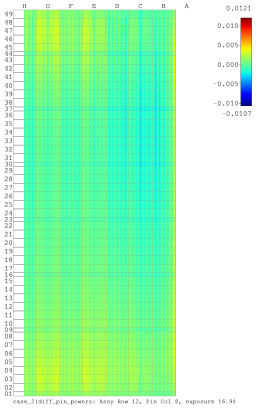


(b) Axial

Figure 5.44: VERA Problem 9: Coarse mesh with FSA solver pin-power errors for a state in the middle of the cycle.

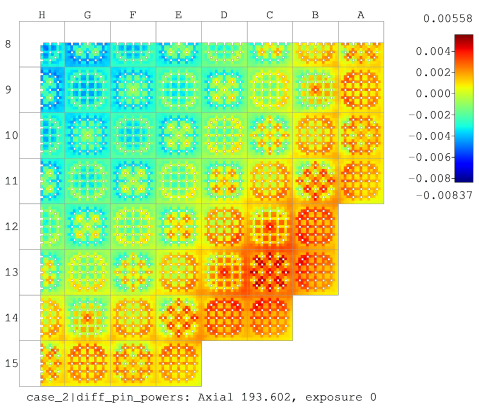


(a) Radial

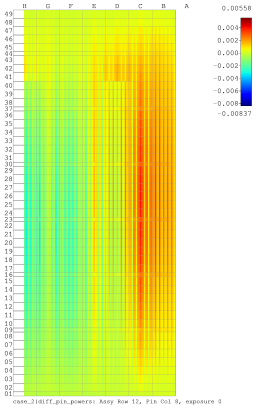


(b) Axial

Figure 5.45: VERA Problem 9: Coarse mesh with FSA solver pin-power errors for the end of the cycle.

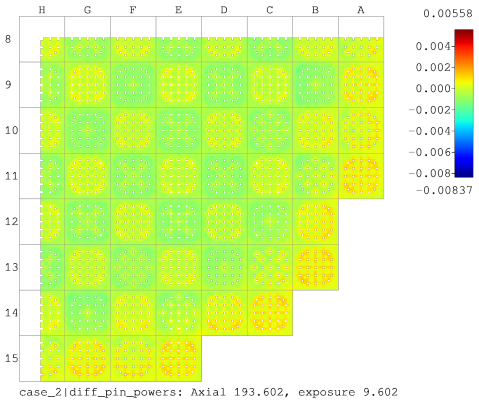


(a) Radial

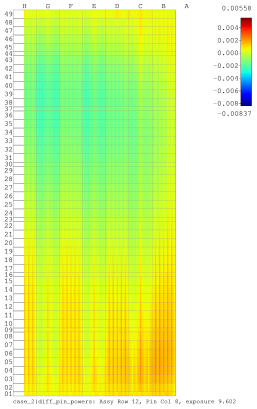


(b) Axial

Figure 5.46: VERA Problem 9: Coarse mesh with LSA solver pin-power errors for the beginning of the cycle.

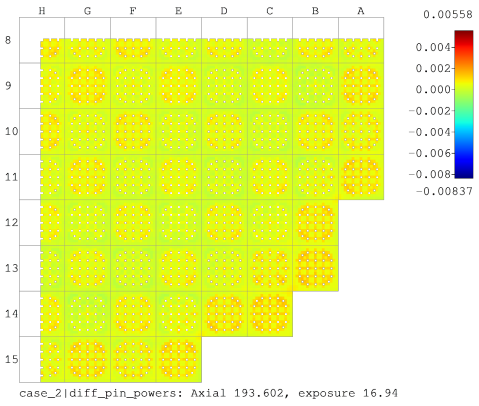


(a) Radial

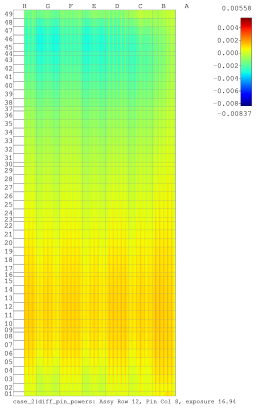


(b) Axial

Figure 5.47: VERA Problem 9: Coarse mesh with LSA solver pin-power errors for a state in the middle of the cycle.



(a) Radial



(b) Axial

Figure 5.48: VERA Problem 9: Coarse mesh with LSA solver pin-power errors for the end of the cycle.

5.4 Conclusions

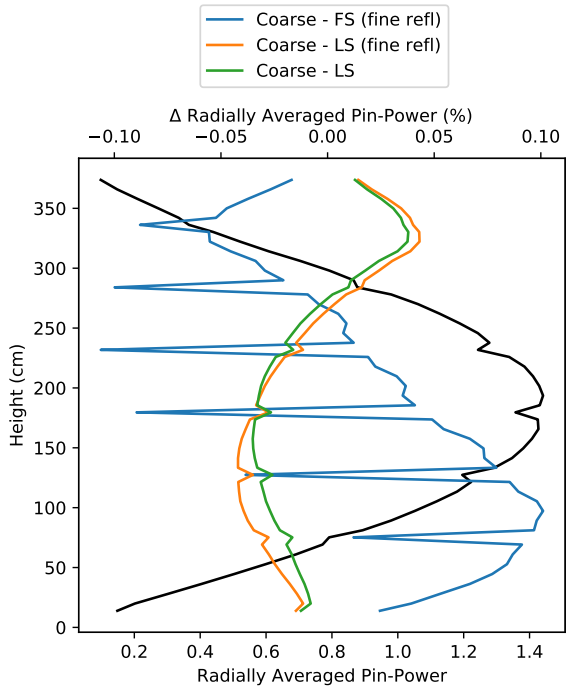
In this chapter, techniques for more stable and efficient exponential calculations for the LSMoC, and a new formulation of the LSMoC that is more efficient in calculations with non-constant cross-sections, were presented. Previously, due to the tabulation error, the exponential functions in LSMoC could cause problems to become unstable when they contained near-void regions, such as the fuel-clad gap. This could be solved by using more accurate tables, but at the expense of performance. Instead, a different function, that is not scaled by the inverse transport cross section in the transmission calculation, can be tabulated and used to calculate the other exponential functions. This solved the issue which caused the instability and reduced the cost of exponential computations in LSMoC calculations by 10% and total MoC time by 3%.

The improved formulation is equivalent to the previous formulation, but is found in such a way that the group-dependent coefficients, $\boldsymbol{C}_i^g$, can be eliminated. This reduces memory usage, as the $\boldsymbol{C}_i^g$ terms are no longer need to be stored, which was at least as expensive as storing an additional current term on the fine mesh. Additionally, the $\boldsymbol{C}_i^g$ terms no longer need to be computed (previous results have indicated this takes around 15-20% of the time a transport sweep take [3]). This is particularly advantageous in cases with non-constant cross sections, such as multi-physics calculations. In calculations with T/H feedback, the cross sections can change each iteration, meaning the new formulation removes a 15-20% MoC overhead.

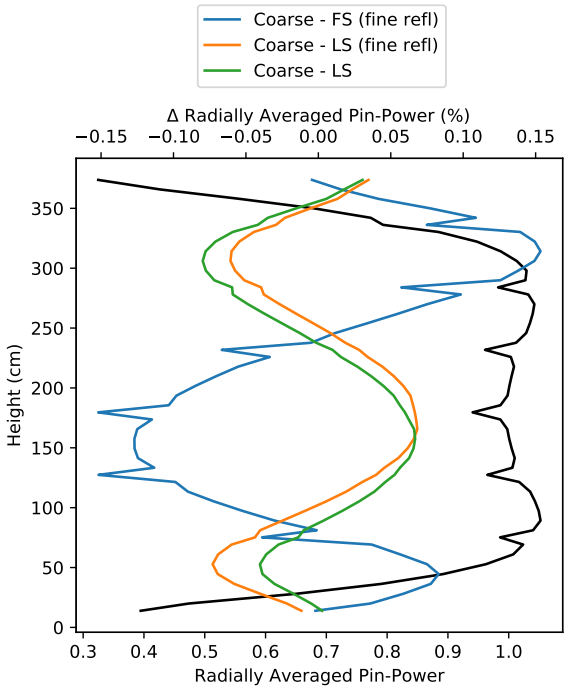
The improved LSMoC formulation was used in calculations to generate results for a variety of steady-state single and multi-physics calculations. These cases were used to determine possible default meshing parameters for MPACT. The determined coarse meshing parameters were shown to be sufficient for a 3-D quarter core cycle depletion with T/H feedback. It was shown that the LSA allowed for significant reduction in the number of source regions, memory usage, and reduced overall run-time in many of these cases.

While it is known the use of the LSA allows for a coarser mesh while maintaining accuracy and typically reducing run-times, this work has led to two additional conclusions. First, the maintained accuracy and reduced run-times seem to hold even in the presence of additional physics such as isotropic depletion, and T/H feedback. Additionally, it was hypothesized that the use of a coarser mesh may allow for coarser ray-spacing. However, this work has demonstrated that this is not the case; the determining factor for whether or not coarse rays are possible, without negatively impacting accuracy, is the presence of fine material discretizations such as control rods, or other strong absorbing elements. In reactors, these problematic elements only make up a small fraction of the total number of pins, and is is expected, that the additional detail provided by using finer ray-spacing is need only locally within the elements. This motivates the use of a ray-tracing technique that allows for finer *effective* ray-spacing to be used in these elements, while coarser

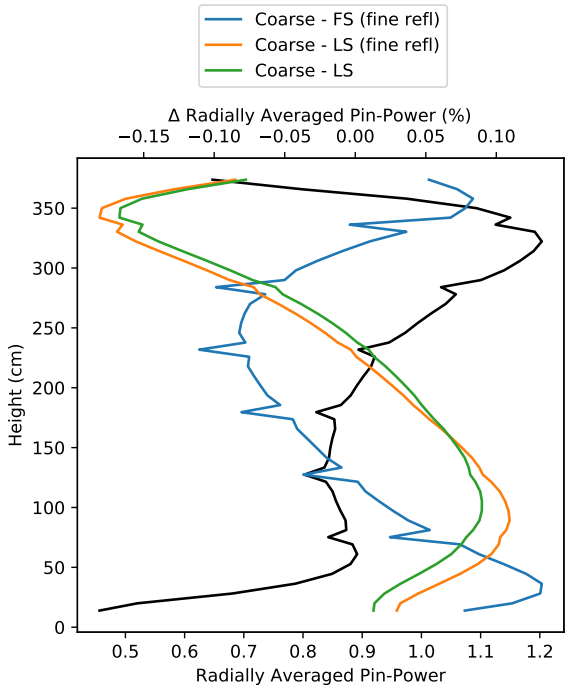
ray-spacing is used elsewhere. The macroband method seems to fit this bill for 2-D calculations, but there exists no similar ray-tracing technique for 3-D calculations. Chapter 6 will provide details on the extension of the macroband method for 3-D calculations.



(a) Beginning of cycle



(b) Middle of cycle



(c) End of cycle

Figure 5.49: VERA problem 9: Radially averaged pin-power at beginning, middle, and end of the cycle for the reference solution. Errors are shown for the coarse mesh solutions.

Bibliography

[1] Rodolfo M. Ferrer and Joel D. Rhodes. "A Linear Source Approximation Scheme for the Method of Characteristics". In: *Nuclear Science and Engineering* 182.2 (2016), pp. 151–165. ISSN: 0029-5639. DOI: 10.13182/NSE15-6.

[2] Andrew Fitzgerald and Brendan Kochunas. "Fast Exponential Function Approximations for the Method of Characteristics with Linear Source". In: *Transactions of the American Nuclear Society*. Orlando, USA, 2018, pp. 645–648.

[3] Andrew P Fitzgerald, Brendan Kochunas, and Thomas Downar. "Improved Formulation of the Method of Characteristics with Linear Source for 2D/1D and Multiphysics Calculations". In: *M&C*. Portland, OR, 2019, pp. 1093–1103.

[4] Akio Yamamoto, Yasunori Kitamura, and Yoshihiro Yamane. "Computational efficiencies of approximated exponential functions for transport calculations of the characteristics method". In: *Annals of Nuclear Energy* 31.9 (2004), pp. 1027–1037. DOI: 10.1016/j.anucene.2004.01.003.

[5] Brendan Matthew Kochunas. "A Hybrid Parallel Algorithm for the 3-D Method of Characteristics Solution of the Boltzmann Transport Equation on High Performance Compute Clusters". Doctoral. University of Michigan, 2013, pp. 1–194.

[6] Gilbert W. Stewart. "20. Lecture 20". In: *Afternotes on Numerical Analysis*. 1996, pp. 147–153. DOI: 10.1137/1.9781611971491.ch20.

[7] Rodolfo M. Ferrer. *Personal Communications*. 2019.

[8] Nicholas F Herring et al. "IMPROVED ANISOTROPIC LINEAR SOURCE FORMULATION FOR MULTIPHYSICS PROBLEMS". In: *PHYSOR (submitted)*. Cambridge, 2020.

[9] M. A. Smith, E. E. Lewis, and Byung Chan Na. "Benchmark on Deterministic 2-D MOX Fuel Assembly Transport Calculations without Spatial Homogenization". In: *Progress in Nuclear Energy* 48.5 (2006), pp. 383–393. ISSN: 01491970. DOI: 10.1016/j.pnucene.2006.01.002.

[10] Ang Zhu 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. ISSN: 18732100. DOI: 10.1016/j.anucene.2016.05.004.

[11] Rodolfo Ferrer. *Personal Communications*. 2018.

[12] A. T. Godfrey. *VERA Core Physics Benchmark Progression Problem Specifications*. Tech. rep. 4. 2014, pp. 1–173. URL: https://www.casl.gov/sites/default/files/docs/CASL-U-2012-0131-004.pdf.

[13] Akio Yamamoto et al. “Non-Equidistant Ray Tracing for the Method of Characteristics”. In: *M&C 2005* (2005), pp. 1–10.

[14] K Lassmann et al. “The radial distribution of plutonium in high burnup UO2 fuels”. In: *Journal of Nuclear Materials* 208.3 (1994), pp. 223–231. ISSN: 00223115. DOI: 10.1016/0022-3115(94)90331-X.

[15] Maria N Avramova and Robert K Salko. “CTF - A Thermal-Hydraulic Subchannel Code for LWRs Transient Analyses User’s Manual, Revision 0”. In: (2015).

[16] Brendan Kochunas, Andrew Fitzgerald, and Edward 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. ISSN: 10902716. DOI: 10.1016/j.jcp.2017.05.028.

[17] Brendan Kochunas et al. “VERA Core Simulator Methodology for Pressurized Water Reactor Cycle Depletion”. In: *Nuclear Science and Engineering* 185.1 (2017), pp. 217–231. ISSN: 0029-5639. DOI: 10.13182/NSE16-39.

[18] Benjamin Collins 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. ISSN: 10902716. DOI: 10.1016/j.jcp.2016.08.022.

CHAPTER 6

MacroRay Three-Dimensional Ray-tracing Technique

The primary motivation of the three-dimensional macroray ray-tracing technique is to reduce the number of tracks generated for 3-D MoC applications. The number of track-segments is very strongly correlated with the run-time of a MoC calculation [1]. The 2-D macroband ray-tracing method had been shown to allow for coarser ray-spacing compared to traditional methods, but it has never been extended for use in 3-D MoC calculations. This work seeks to fill that gap, and perform an investigation into the extension from 2-D to 3-D macroband-like ray-tracing. The ray-tracing technique is referred to as the *macroray*, because the 3-D "macro"paths are no longer bands, but instead parallel-pipes.

This work implemented a 3-D ray-tracing and transport solver library in MPACT based on the macroray method. The initial step in this implementation was to implement a 2-D version [2]. This also serves to confirm the results of previous studies on 2-D macroband [1, 2].

6.1 MacroRay Technique

The macroray ray-tracing technique is a 3-D extension of the macroband method; in order to describe this new ray-tracing technique it is first necessary to describe the macroband method. Section 3.4.5 gave a brief summary of the history of macroband, as well as some benefits of the method. This section will describes the macroband method in more detail, and shows how the method is extended to 3-D.

[1] if the MoC calculation performs an entirely separate calculation (i.e does not use the chord-classification method) for each track-segment, this is expected to be directly proportional (ignoring overhead)

[2] which did not support CMFD

6.1.1 Macroband Technique

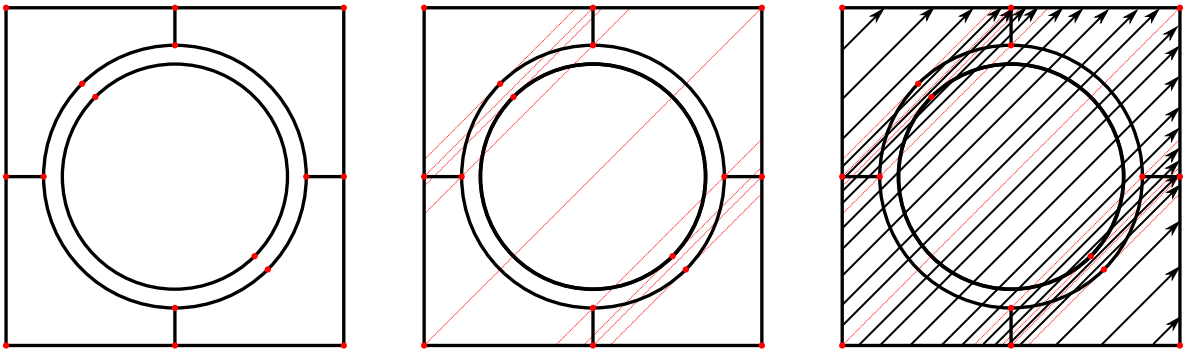
The macroband was first used in the Collision Probability (CP) code HELIOS, and was considered to be essential for the stability of the method [3]. Unlike the traditional uniform/equidistant ray-tracing methods, the macroband method guarantees that each ray within a macroband will pass through the same regions, and not have any regions within its bounds that are not crossed by the rays. This means that the placement of rays, and their widths, are determined by the internal geometry of the pin or subdomain that is being traced. Because there are no material discontinuities within a macroband, the track-segment average values are expected to be smooth in the transverse direction. This allows for the use of more advanced transverse integration, such as Gauss-Legendre quadrature integration [1].

The macroband method ray-tracing is described in Algorithm 9. A downside of this method is that each ray has a unique width which requires additional storage. However, all rays within a macroband pass through the same segments, so the region index for each segment only needs to be stored once, reducing memory by a larger amount. The process is shown visually in Fig. 6.1.

Algorithm 9 Macroband ray-tracing procedure for a pin-cell.

1: Begin with a pin-cell geometry/mesh, and a direction of flight $\widehat{\Omega}_m$.
2: Create a list of points, P, containing:
    1. corners,
    2. mesh intersections,
    3. and tangent points.
3: Sort the points by signed distance in normal direction, $\widehat{n}_m$, from center of pin
4: The macrobands are bounded in the transverse direction by these distances
5: **for all** Macrobands **do**
6: Determine quadrature points and widths in transverse direction
7: Place rays at these points with these widths
8: Trace each ray
9: **end for**

As it is difficult to visualize 3-D ray-tracing data in a 2-D textual media, a quarter integral fuel burnable absorber (IFBA) fuel pin will be used as an example to demonstrate the differences between ray-tracing methods. Figures 6.2 and 6.3 compare the traditional MRT ray-tracing technique with the macroband method with Gauss-Legendre spacing. The rays for a single direction make it clear that the MRT in no way accounts for the internal geometry. In the macroband method, there are clearly regions that have more rays passing through them; in this case, it is the boron layer coating the fuel. When all directions are visualized, again the MRT would use this same pattern of rays whether this was an IFBA rods, or a rectangular reflector (of the same size). In the macroband



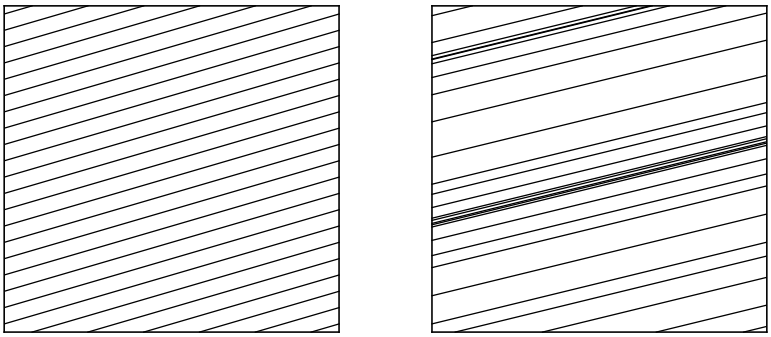
(a) Determine intersection and tangent points

(b) Determine macroband boundaries (through identified points)

(c) Perform ray-tracing within macroband boundaries

Figure 6.1: Ray-tracing process for macroband.

methods, some of the geometry becomes very apparent, the outline of the pin can clearly be seen due to the methods forcing rays through the thin coating.



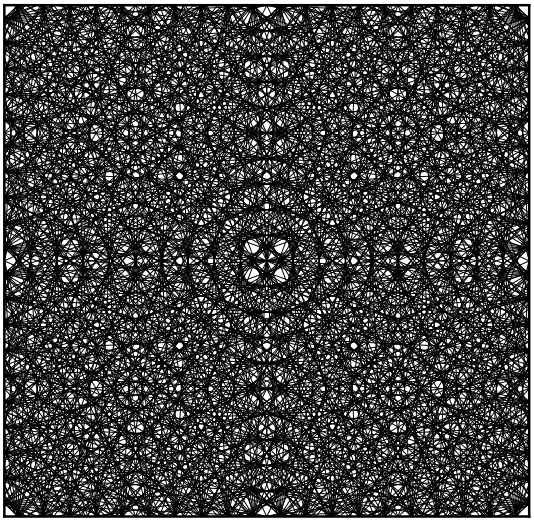
(a) MRT

(b) Macroband (Gauss-Legendre)

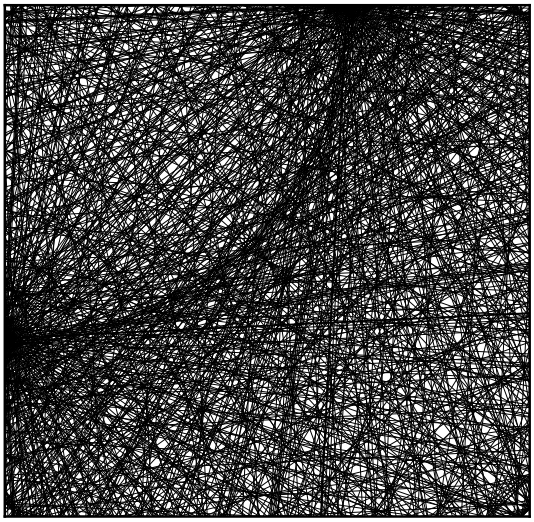
Figure 6.2: 2-D single angle ray-tracing comparison for quarter IFBA pin.

There have been different approaches to 3-D ray-tracing for MoC; the most common approach is to first generate 2-D tracking data, forming characteristic planes. Then 2-D tracking data is generated for each of the characteristic planes. This process was described in detail in Section 3.4.6, and will be referred to as the 2D/2D ray-tracing approach. This was the approach taken for this thesis work, but other approaches may have significant advantages; these will be discussed in Section 6.1.3.

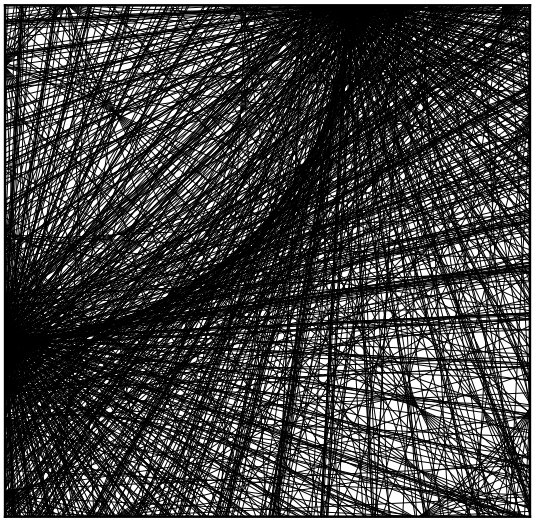
Although previous works have investigated the 2-D macroband ray-tracing method [1, 2, 4], there have not been any studies in the extension of macroband to 3-D MoC calculations. Here, the macroray ray-tracing technique uses the macroband method for both the 2-D radial track generation, and 2-D track generation within each characteristic plane. The setup procedure of the macrorays



(a) MRT



(b) Macroband (Gauss-Legendre)



(c) Macroband-fixed (Gauss-Legendre)

Figure 6.3: 2-D ray-tracing comparison for quarter IFBA pin.

guarantee that their rays pass through the same regions, just as the macroband's did. However, it is no longer guaranteed that any region contained within the volume of a macroray will be passed through. This leads to some issues that are discussed in Section 6.1.3.

Similarly to the macroband method, the macroray method is fundamentally incompatible with the DNPL ray-tracing technique. The MRMB technique can be used to store tracking data for unique geometric subsystems (typically a pin cell), greatly reducing Although tracking data can be generated for unique geometry subsystems (MRMB [1]), some approximation of the angular flux is necessary on interfaces. Generally, the macroray and macroband methods, which provide more accurate integration, exchange regional numerical dispersion for interface numerical dispersion [5]. It is the point of view of the author that this is generally a favorable trade-off, as the engineering quantities of interest are primarily determined through regional integration.

6.1.2 Chord-Classification

The chord-classification ray-tracing method [6] was described in detail in Section 3.4.6.2. One of the key criticisms of this method, by Gunow [7], was that full 3-D tracking data needed to be generated prior to the classification of rays. However, with the addition of the macroray ray-tracing method, classification can be done automatically.

During the axial ray-tracing along a characteristic plane, the computational mesh is used to determine the axial macrobands. Each ray within the macroray is guaranteed to pass through the same regions and surfaces; this means that each ray within an axial macroband is guaranteed to be of the same chord classification. This is demonstrated visually in Fig. 6.4. The chord-classification drastically cuts down on the amount of memory used to store 3-D tracking data, and also may reduce the amount of computational work. The work is reduced because the exponential functions (see Section 5.1) are determined by the material and segment length. For vertical-to-vertical or horizontal-to-horizontal macroray segments, the segment lengths of all rays are the same, thus only a single exponential calculation is necessary.

The chord-classification approach was taken in this work, rather than on-the-fly ray-tracing [7]. This was done due to the ease of implementation (because of automatic classification), and because of the criticisms of on-the-fly ray-tracing described in Section 3.4.6.3.

6.1.3 Interface Angular Flux Approximation

In Section 3.4.5.1 different approaches to approximating the angular flux on the boundaries was outlined for a 2-D macroband-based MoC calculation. A "fraction contribution" sub-boundary averaging technique was described and chosen as the approach for this work. In this approach, each surface of the subsystems (pin cells) is divided into sub-boundaries based on the direction of flight.

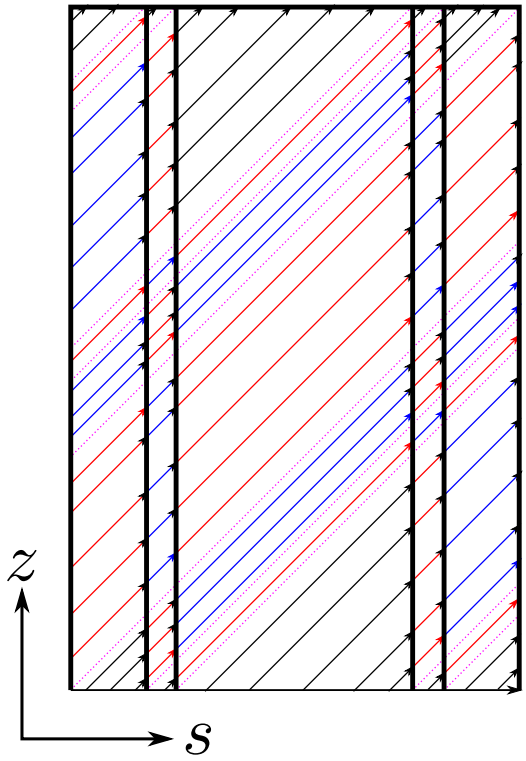


Figure 6.4: 3-D example of chord-classification with Macroray ray-tracing. Colored (red and blue) characteristic tracks represent groups of "V-chords", rays between two vertical surfaces.

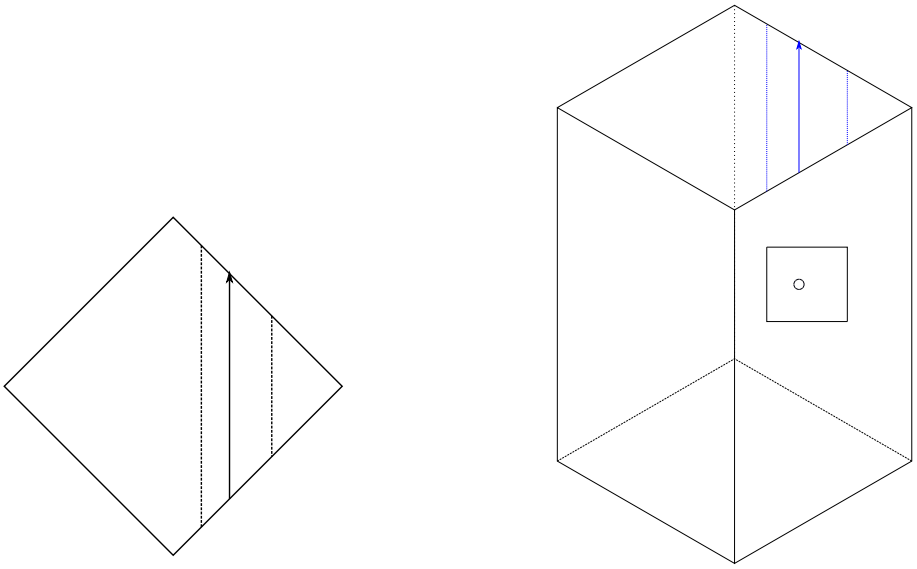
Each ray's width is considered, and partial intersections with each surface are computed and used to determine the ray's fractional contribution to each sub-boundary. In the reverse direction (going from sub-boundary to ray), each sub-boundaries fractional contribution to each ray is determined. It was shown that this method conserved the total area-integrated angular flux on each surface.

However, in 3-D this approach becomes considerably more complicated, due to the nature of 3-D geometries and the 2D/2D ray-tracing approach. This cause of the issues stems from the fact that each ray, as viewed in the direction of flight, is rectangular. Each ray is rectangular because in the radial ray-tracing step, the ray is considered within the bounds of the width in the radially transverse direction, and in the axial ray-tracing on the characteristic plane the ray is again constrained in a height in the axially transverse direction. The rectangular ray projection is demonstrated for a single ray in Fig. 6.5.

For 3-D extruded cartesian pin cells, it is impossible (in arbitrary directions) to "span"[3] the surfaces of our system. Because each ray is rectangular, parts of the pin cell's surfaces will not be "hit" by the projection of any ray, and parts of some rays will live entirely outside the problem

[3]in this context, "span" will be used to describe the idea that each ray's entire area projects onto some surface, and the entire surface is filled with ray projections.

domain. This issue is visualized in Fig. 6.6; it should also be noted that the area outside the domain, and the surface area without any covering rays are not equal in area unless the ray is in the center of the radial width.



(a) 2-D ray viewed from above

(b) 3-D ray viewed in direction of flight

Figure 6.5: Example of the projected rectangular area formed in the 2D/2D ray-tracing approach.

The rays not "spanning" our system causes issues with the conservation of surface flux and leakages as described in Section 3.4.5.1. Previously, the average angular flux in a sub-boundary was determined by

$$\psi_s^i = \sum_j \frac{A(S_i \cap R_j)}{A(S_i)} \psi_r^j. \tag{6.1}$$

It was shown that by summing over all sub-boundaries, the leakage through the surface was conserved. However, because the rays are no longer guaranteed to span the system's surfaces, the summation of intersected ray areas is no longer equal to the sub-boundary areas:

$$\sum_j A(S_i \cap R_j) \leq A(S_i). \tag{6.2}$$

Generally, these areas are very close, and in the limit of infinite rays they are equivalent; but, the motivation of this work has been to reduce the number of rays, so this issue needs to be addressed. Section 6.1.3.1 describes how this issue was addressed in this work, and Section 6.1.3.2 describes possible alternative approaches.

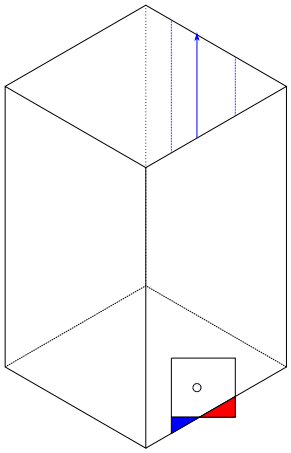


Figure 6.6: An example of a ray that causes issues with the fractional contribution interface flux approximation when using 2D/2D ray-tracing. The highlighted red area is the area of the ray outside the domain (which projects to no surface), and the blue highlighted area is an area on the surface that will not be hit by any ray.

6.1.3.1 MacroRay Area Correction

Two ideas were investigated in order to address the issue of the rays not spanning the surface areas. Here, it should be emphasized that the first approach is *incorrect*. It is described in this section because it seems like a reasonable choice for this correction; in fact, it was not realized for some time in this work that the first approach was incorrect.

The sub-boundary averaging method of this work has been referred to as a "fractional contribution" approach. This comes from Eq. (6.1) where $A(S_i \cap R_j)/A(S_i)$ is the fractional contribution of flux from ray j into sub-boundary i. In 2-D, these fractions will sum to one, because our rays span the surfaces of our system:

$$\sum_j \frac{A(S_i \cap R_j)}{A(S_i)} = 1. \tag{6.3}$$

The first approach taken attempted to preserve this property of fractional contributions summing to unity. This is done by ignoring ray areas outside the domain and surface areas that are not covered by ray projections; the sub-boundary and ray fluxes are determined by

$$\psi_s^i = \sum_j \frac{A(S_i \cap R_j)}{\sum_k A(S_i \cap R_k)} \psi_r^j = \sum_j \frac{A(S_i \cap R_j)}{A_p(S_i)} \psi_r^j, \tag{6.4a}$$

$$\psi_r^j = \sum_i \frac{A(S_i \cap R_j)}{\sum_k A(S_k \cap R_j)} \psi_s^i = \sum_i \frac{A(S_i \cap R_j)}{A_p(R_j)} \psi_s^i, \tag{6.4b}$$

respectively. For convenience, let us define the "total projected area" as the sum of intersections for

that object, for example the total projected area of a ray would be defined by

$$A_p(R_j) \equiv \sum_i A(S_i \cap R_j), \tag{6.5}$$

and similarly for $A_p(S_i)$. Through these definitions the new fractional contributions sum to unity:

$$\sum_j \frac{A(S_i \cap R_j)}{A_p(S_i)} = 1, \tag{6.6a}$$

and

$$\sum_i \frac{A(S_i \cap R_j)}{A_p(R_j)} = 1. \tag{6.6b}$$

While this may seem reasonable [4], this is not actually the property we need to preserve for compatibility with CMFD acceleration. The leakage needs to be preserved, but in this first approach this is not the case.

$$L^s = \widehat{\boldsymbol{\Omega}} \cdot \hat{\boldsymbol{n}} \sum_i A(S_i)\psi_s^i = \widehat{\boldsymbol{\Omega}} \cdot \hat{\boldsymbol{n}} \sum_i A(S_i) \sum_j \frac{A(S_i \cap R_j)}{A_p(S_i)} \psi_r^j \neq \psi_t^r, \tag{6.7a}$$

$$L^r = \widehat{\boldsymbol{\Omega}} \cdot \hat{\boldsymbol{n}} \sum_j A(R_j)\psi_r^j = \widehat{\boldsymbol{\Omega}} \cdot \hat{\boldsymbol{n}} \sum_j A(R_j) \sum_i \frac{A(S_i \cap R_j)}{A_p(R_j)} \psi_s^i. \neq \psi_t^s, \tag{6.7b}$$

Note that the true areas $A(S_i)$ and $A(R_j)$ must be used to sum the fluxes in order to integrate over the entire surface (or ray areas).

This approach actually works for many problems, and while small differences between the solutions of transport and accelerated transport calculations, they are usually small. However, for larger problems, the CMFD acceleration *may* become unstable and cause the iteration scheme to diverge; though this generally only occurs if the convergence criteria is relatively low ($\leq 10^{-6}$). Nevertheless, compatibility with CMFD acceleration, or other acceleration methods, is necessary if this method is ever to be used. This leads to the approach used for the remained of this work.

As this section has emphasized, the property that is important to conserve is the leakage. The leakage can be found by integrating over sub-boundaries, or over rays; it is necessary for these to be equal:

$$L = \widehat{\boldsymbol{\Omega}} \cdot \hat{\boldsymbol{n}} \sum_i A(S_i)\psi_s^i = \sum_j A(R_j)\psi_r^j. \tag{6.8}$$

The sub-boundary flux, ψ_s^i, should be dependent on the ray fluxes, ψ_r^j, and should also follow a similar form as Eq. (6.1).

[4] at least at first to the author

Let us examine the sub-boundary flux in order to show how the leakage may be preserved. Insert a corrective multiplicative term into the summation of Eq. (6.1),

$$\psi_s^i = \sum_j k_s^j \frac{A(S_i \cap R_j)}{A(S_i)} \psi_r^j. \tag{6.9}$$

Substituting this form into Eq. (6.8) yields,

$$L = \widehat{\boldsymbol{\Omega}} \cdot \hat{\boldsymbol{n}} \sum_i A(S_i) \sum_j k_s^j \frac{A(S_i \cap R_j)}{A(S_i)} \psi_r^j = \widehat{\boldsymbol{\Omega}} \cdot \hat{\boldsymbol{n}} \sum_j A(R_j) \psi_r^j. \tag{6.10}$$

So, k_s^j should be chosen such that

$$\sum_i k_s^j A(S_i \cap R_j) = A(R_j), \tag{6.11}$$

solved by

$$k_s^j = \frac{A(R_j)}{\sum\limits_i A(S_i \cap R_j)} = \frac{A(R_j)}{A_p(R_j)}. \tag{6.12}$$

The sub-boundary flux can then be determined by

$$\psi_s^i = \sum_j \frac{A(R_j)}{A_p(R_j)} \frac{A(S_i \cap R_j)}{A(S_i)} \psi_r^j, \tag{6.13a}$$

and similarly, the ray flux can be determined by

$$\psi_r^j = \sum_i \frac{A(S_i)}{A_p(S_i)} \frac{A(S_i \cap R_j)}{A(R_j)} \psi_s^i. \tag{6.13b}$$

As this was derived from Eq. (6.8), these forms guarantee that the leakage is conserved. It should also be noted that if the surfaces are spanned by the rays the factors, $A(R_j)/A_p(R_j)$ and $A(S_i)/A_p(S_i)$, are one, giving back the original form of the equations.

Another consequence of the rays not spanning the problems surfaces, is that larger sub-boundary sizes become necessary to ensure that each sub-boundary is intersected by at least one ray. The larger sub-boundary sizes will have a negative impact on the accuracy of the simulations, particularly in cases where neighboring pins are significantly different.

6.1.3.2 Alternative Approaches

The approach described in Section 6.1.3.1 provided a correction to Eq. (6.1) such that the surface-integrated angular flux is conserved. This approach seems to be valid when operating in the 2D/2D (rectangular) ray-tracing procedure, but there are alternative options. The reason a correction was necessary is because due to the nature of 3-D geometry and 2D/2D ray-tracing, the rays are no longer guaranteed to span the system's surfaces. It is possible to have an alternative ray-tracing approach (still based on the macroray) that does span the system's surfaces.

The choice of rectangular rays was arbitrary, but this choice may be exchanged for any shape, such as triangular, or even arbitrarily shaped rays. Non-rectangular rays were not investigated as part of this work, and to the best of our knowledge, have not been investigated by any works. As previously mentioned, 3-D MoC codes up to this point have used the uniform ray-spacing assumption in order to comply with constraints of MRT and DNPL. When rays are uniformly spaced and have DNPL, the shape of the ray is largely ignored because the procedures only care about the ray's centroid. It only becomes important when considering the integration volume of the ray, such as in the sub-boundary averaging method with fractional contributions.

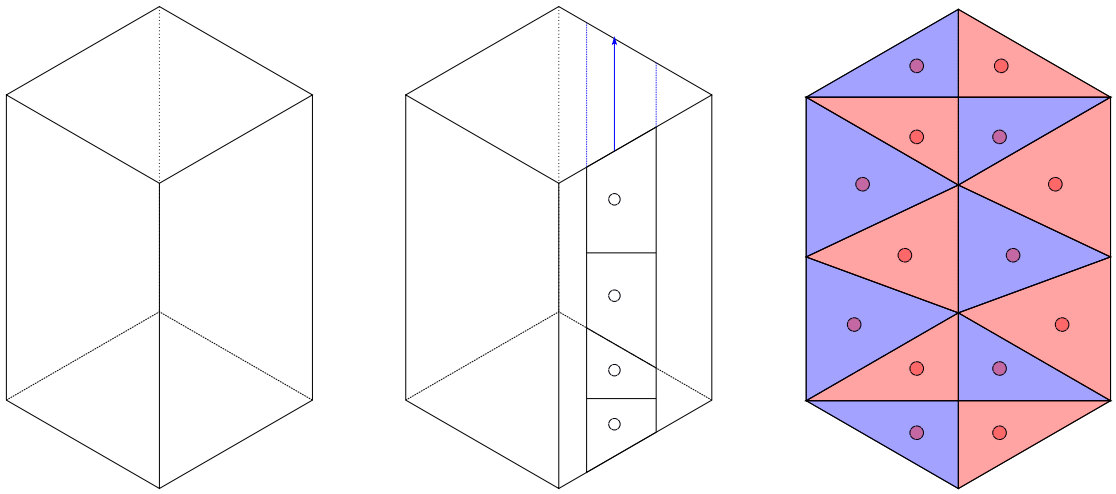
If rays are constructed so that they span the system's surfaces, there is no need to correct Eq. (6.1). It is not clear at this point whether or not this would have benefits for the accuracy of the method. However, it is possible to also preserve some of the desirable features for general geometries of the macroband which were lost in the transition to 3-D (in the 2D/2D ray-tracing framework). If the geometry is not locally extruded (in this work this was an assumption), then it is possible for macrorays to have some of their transverse area in a region that none of its rays pass through. Examples of how non-rectangular rays might be used are shown in Fig. 6.7.

6.2 Results

This section presents results using the macroray ray-tracing method, and comparing it against the modular ray-tracing (MRT) method. All results presented use the LSA approximation developed in Chapter 5.

6.2.1 VERA Problem 1E

VERA progression problem 1E [8] is a single IFBA pin with a very thin coating of boron on the fuel. The boron will shield the pin from incident thermal neutrons, significantly dampening the reactivity. These rods are used in reactor designs, and as the fuel cycle continues the boron will be depleted and the reactivity will increase.



(a) geometry (b) Axially stacked rays (c) General triangular rays

Figure 6.7: Two alternative (non-rectangular) ray-tracing ideas. (a) shows the geometry for clarity. (b) shows an example of rays that are generated to be axially aligned in a characteristic plane, but consider the full volume they pass through. (c) shows an example of triangulated rays. In both (b) and (c) the rays are formed by rhombuses or triangles, but if there were a cylinder in this pin, they could have curved boundaries as well.

Such a thin coating which has a very significant impact on the reactivity can be a difficult problem for transport methods. The MoC is able to handle this case, but generally requires very fine ray-spacing in order to resolve these thin coating regions. This makes VERA problem 1E a perfect test case for the macroband and macroray methods, which will automatically place rays through these thin regions. Ideally, this would allow coarser ray-spacing to be used elsewhere in the pin.

6.2.1.1 VERA Problem 1E: 2-D

Each calculation in this section used a Tabuchi-Yamamoto [9] directional quadrature with 64 azimuthal angles and 4 polar angles over 4π. Additionally, a self-shielding calculation was run to compute macroscopic cross sections for the problem. The reference case uses the same directional quadrature and mesh with a very fine ray-spacing; this was chosen as the reference as this study is intended to isolate effects due to ray-tracing discretizations. Two different approaches were tested to determine the number of rays within a macroband: first, the width of the band is divided by the ray-spacing and the ceiling is taken. The second approach is to place a fixed number of rays within each band.

Figures 6.8 and 6.9 show the eigenvalue results for a variety of effective ray-spacing, and the eigenvalue errors for the the number of ray-segments. Effective ray-spacing is computed by averaging the widths of all rays in the problem. In this case, the eigenvalue error is taken as the

difference from the converged MRT result. These figures show a clear advantages for the macroband ray-tracing method; even for the coarsest macroband spacing, the resulting eigenvalue was within 150 pcm of the converged result. The method using a fixed number of rays in each macroband seems to have an advantage when using coarse ray-spacings, and converges slightly faster than the width-determined number of rays per band.

The convergence of the eigenvalue as more rays (segments) are added is very close to monotonic for the macroband methods. This is certainly not the case for the MRT method, which sees very large swings in the eigenvalue from small changes in the ray-spacing, as shown by the large spikes in Fig. 6.10. The large spikes on the right of this plot show that for small changes in the ray-spacing, a large (> 2000 pcm) reactivity swing can occur when using MRT, whereas both macroband approaches have low differences between cases with similar ray-spacing. The lack of these large changes, allows the solver to act as more of a "black box", where the user does not need knowledge of the method to use it. Additionally, because of the near monotonic convergence behavior, it may be possible to use Richardson extrapolation to further reduce costs.

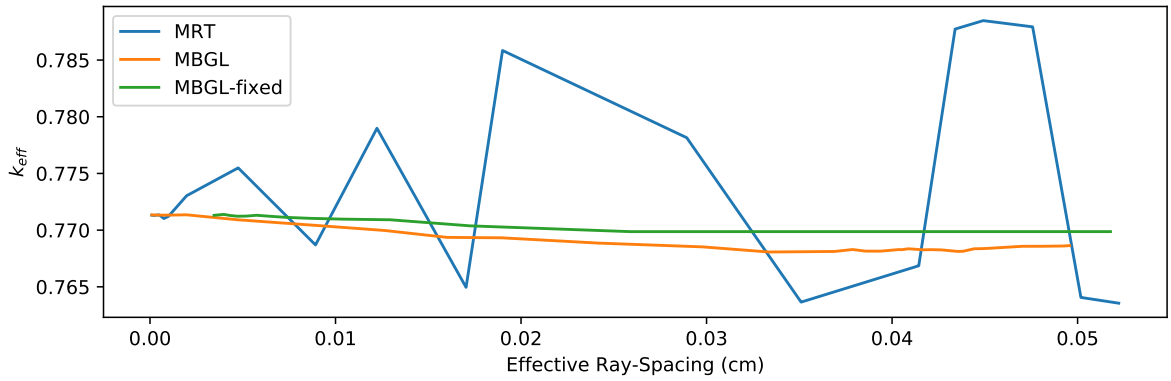


Figure 6.8: VERA Problem 1E: Eigenvalue results for different 2-D ray-tracing methods.

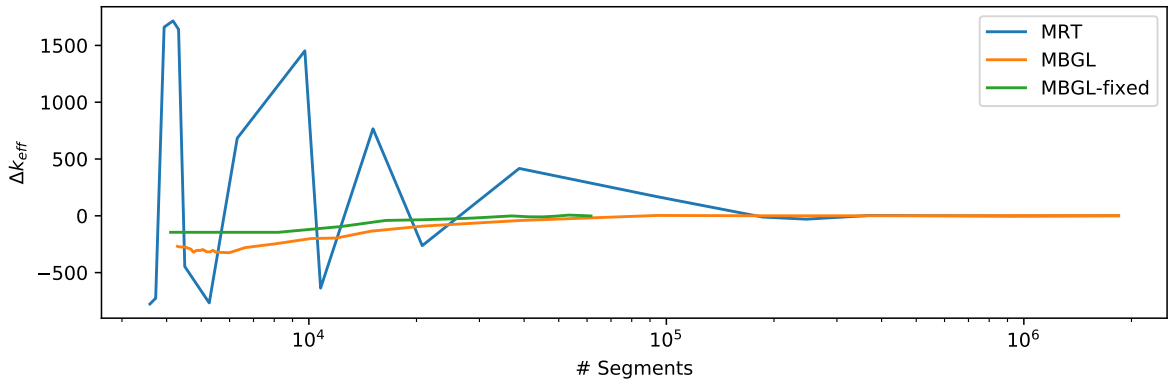


Figure 6.9: VERA Problem 1E: Eigenvalue errors for different 2-D ray-tracing methods.

6.2.1.2 VERA Problem 1E: 3-D

MPACT currently lacks a multigroup sweeper for 3-D self-shielding calculations, so this calculation was run without self-shielding. This will lead to a significantly different converged eigenvalue than for the 2-D case. However, even without self-shielding, this case can be used to show the benefits of the macroray ray-tracing method in 3-D MoC calculations. Due to the 3-D rays, it is no longer useful to compare the results as a function of ray-spacing, as there is now the effective axial and effective radial ray-spacing. As it is a far better metric for the amount of work, all the 3-D results will simply use the number of track-segments for accuracy comparisons. Each calculation was run using odCMFD acceleration [10], and a directional quadrature with 64 azimuthal angles and 6 polar angles over 4π.

Figures 6.11 and 6.12 show the k-eigenvalue as a function of the number of track segments, and the sequential differences in eigenvalue for each refinement of rays. These results compare the traditional MRT method against the macroray method with a fixed number of rays in each radial and axial band (different amounts in radial and axial). A Gauss-Legendre quadrature was used to determine ray placement and width within each band. As this problem is axially homogeneous, a single ray was used in each axial band, and an axial ray-spacing of 0.75 cm was used for the MRT method.

These results suggest that the macroray method is able to converge toward the final result at a faster rate than the traditional MRT method; the eigenvalue change between successive calculations (refining the number of rays) drops below 10 pcm with around 5.6 million segments, whereas the MRT method was unable to converge within 10 pcm even using 14 million. Additionally, it is clear that the near-monotonic convergence behavior which the macroband had in 2-D holds in 3-D.

The MRT calculations were not continued past 14 million segments, as using finer ray-spacing resulted in significantly increased run-times, much of which was spent in the actual ray-tracing procedures. While the macroray does significantly simplify the ray-tracing procedure, the author believes that these long ray-tracing run-times in MRT are likely due to unoptimized code, rather than an inherent flaw in the method. For 5.6 million segments, the macroray ray-tracing procedures took under 10 seconds, while for 4.8 million segments in the MRT took 190 seconds.

6.2.2 Shielded Fuel Box

The shielded fuel-box problem was created for this work to more clearly demonstrate the benefits of the macroray method for problems that pose challenges to the traditional ray-tracing methods. This problem consists of a $0.4\times0.4\times0.4$ cm^3 fuel block surrounded on all sides by 0.1 cm of a shielding material. This entire shielded cube is surrounded by $0.5\times0.5\times0.5$ cm^3 cubes of moderator (26 in total). The calculation uses the C5G7 benchmark cross sections; the fuel is the 8.7% MOX, the

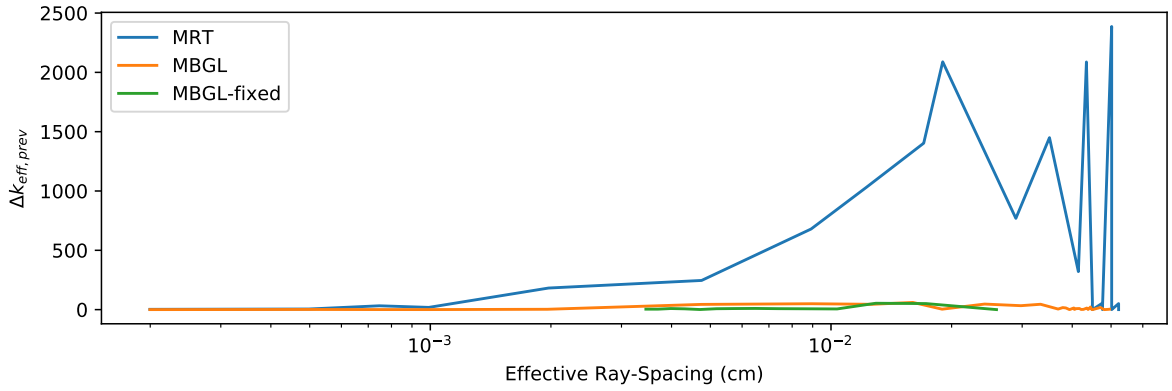


Figure 6.10: VERA Problem 1E: Convergence of eigenvalue for different 2-D ray-tracing methods.

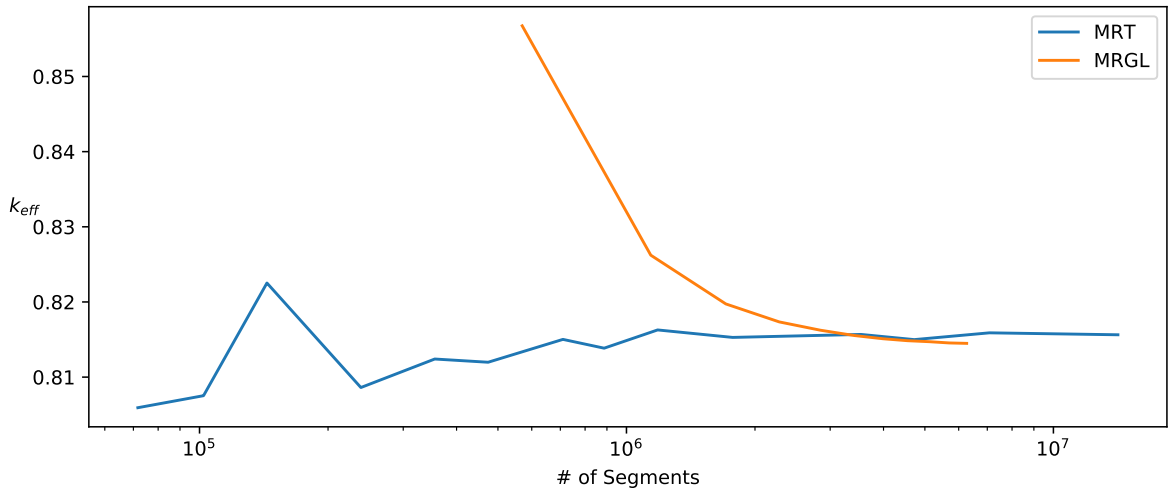


Figure 6.11: VERA Problem 1E: Eigenvalue results for different ray-tracing methods.

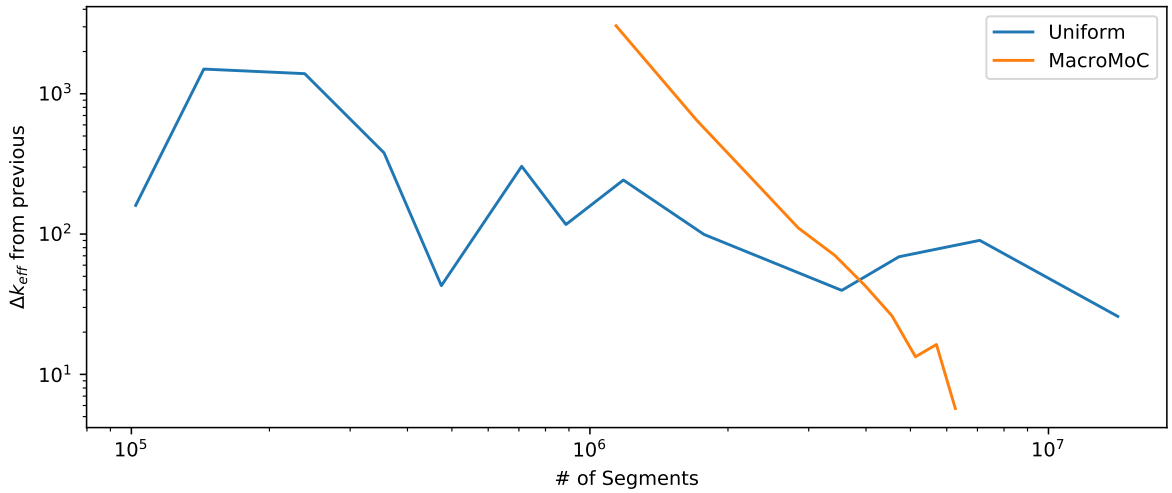


Figure 6.12: VERA Problem 1E: Convergence of eigenvalue for different ray-tracing methods.

shielding material uses the control rod cross sections. A 2-D cross-sectional view of the problem is shown in Fig. 6.13, and reflective boundary conditions are used on each face. This problem is meant to be similar to VERA problem 1E, though the surrounding material was made thicker so that resolving the region did not require an extreme number of rays.

The OpenMC Monte Carlo code [11] was used to generate a reference eigenvalue to compare the results against. The reference solution was generated with 500 batches of 50000 particles, with 50 inactive batches. This yielded a reference eigenvalue with 4 pcm uncertainty. Each calculation used a Chebyshev-Chebyshev directional quadrature with 24 azimuthal and 12 polar angles over 4π; this quadrature was chosen, rather than a Chebyshev-Gauss, as the problem has the same axial and radial descriptions. Similarly, the radial and axial ray-spacing used by the MRT are the same for each calculation, and the number of rays per radial or axial band are the same.

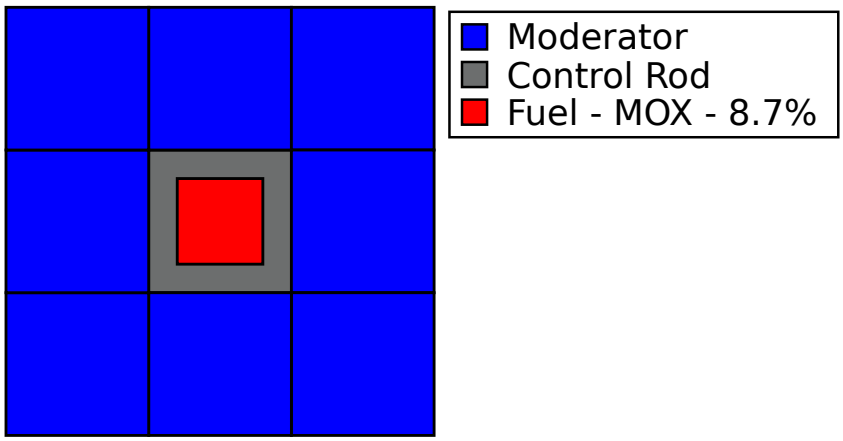


Figure 6.13: Cross-sectional diagram of the shielded-box problem from x-y, x-z, or y-z directions.

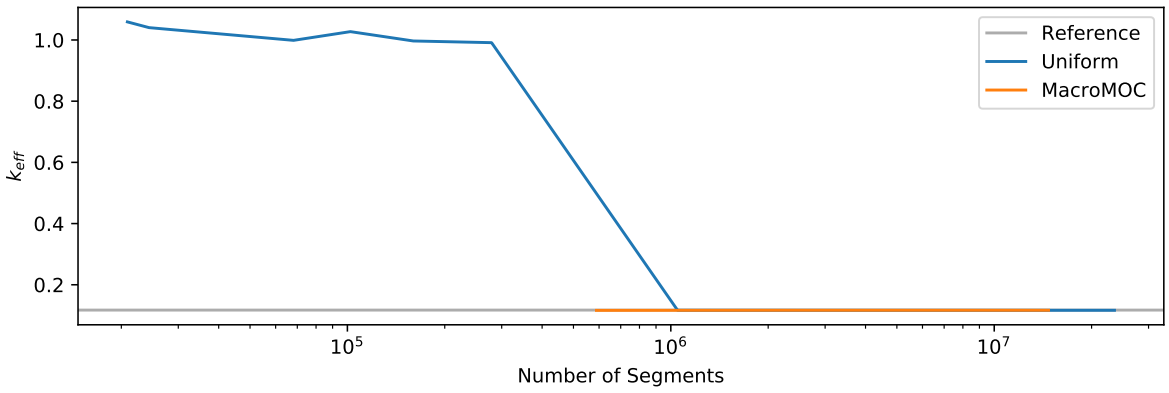


Figure 6.14: Shielded-Box Problem: Eigenvalue results for different 3-D ray-tracing methods.

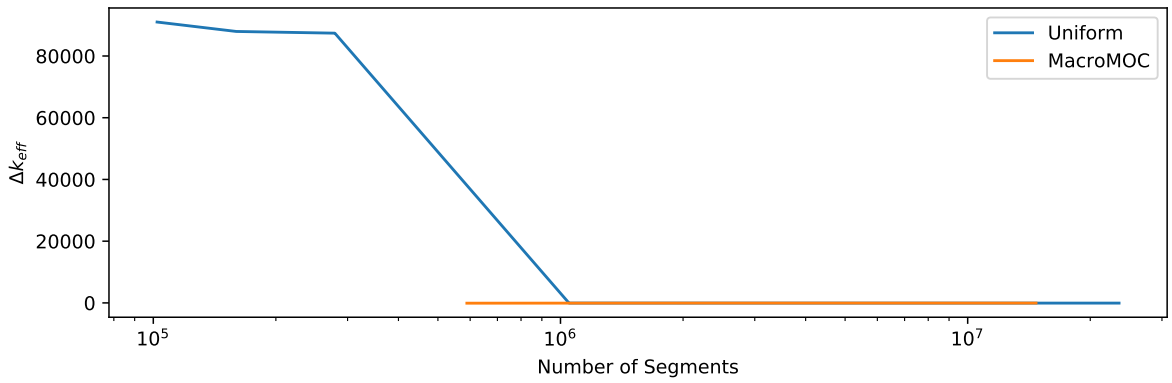


Figure 6.15: Shielded-Box Problem: Eigenvalue errors for different 3-D ray-tracing methods.

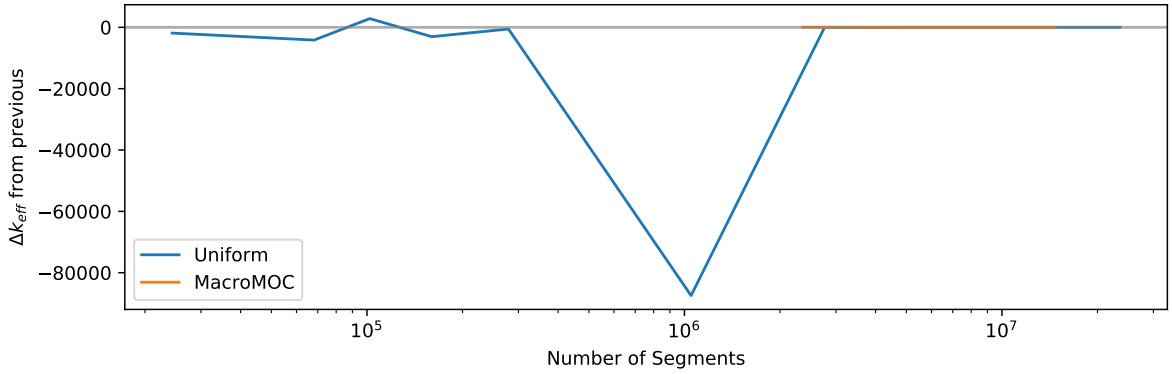


Figure 6.16: Shielded-Box Problem: Convergence of eigenvalue for different 3-D ray-tracing methods.

The results of the ray-tracing methods are summarized in Figs. 6.14 and 6.15. The traditional MRT method generates very large eigenvalue errors when a coarse ray-spacing is used; in fact, for the first several ray-spacing values used, the eigenvalue estimate was near critical even though the reference is very sub-critical. This shows a major flaw in the MRT methods, they do not consider the internal geometry and allow for such coarse spacings that the results can be entirely useless. In contrast, the macroray method enforces a minimum number of rays, which seems to give somewhat reasonable results.

Figure 6.16 shows the change in eigenvalue for each sequential calculation (a refinement in the number of rays/segments). This figure also shows that for very coarse ray-spacings (resulting in near critical eigenvalues) do not have large differences in the resulting eigenvalues. An inexperienced user may believe that the solution is near convergence, which is certainly not the case. Once the ray-spacing is smaller than the thickness of the shielding, the results become reasonable, although still further off from the reference than the macroray method for the same number of segments. However, a user without knowledge of the MoC would likely not be aware that this is a requirement of the user input. Although this was demonstrated with very coarse ray-spacings, a similar effect would likely be present if the shielding were thinner than the default ray-spacing of MPACT. This shows a clear advantage for the macroray method, in that it allows the user to more safely use the method without detailed knowledge of the method.

6.2.3 C5G7

The C5G7 [12, 13] benchmarks are a set of benchmark problems commonly used to help the verification of 2-D and 3-D transport methods. The benchmarks provide a reference Monte Carlo solution which can be used to evaluate the accuracy of methods used to solve the problems. The original benchmark specified a 3-D problem, and a second benchmark reduced the axial size of this problem and added configurations with inserted control rods. In this work, each of the three extended 3-D benchmark problems are solved using both the traditional ray-tracing method (MRT), and the new macroray ray-tracing method. Fig. 6.17 shows the configuration of the C5G7 core benchmark cases; the different problems differ only in their axial description. The radial layout of each fuel assembly is shown in Fig. 6.18.

The LSA was used for all of the calculations in this work, and the meshing parameters used were found to be sufficient in previous studies on the LSA [14]. In the axial direction, the problem was meshed into 2.38 cm slices. The rods (fuel, guide-tubes, and control rods), used a single radius of 0.540 cm, with 4 azimuthal divisions, and 8 azimuthal divisions in the surrounding moderator. The reflector pins used a radial mesh of 0.42×0.42 cm^2. A Chebyshev-Gauss directional quadrature with 64 azimuthal angles and 8 polar angles over 4π was used, as it was shown to yield good

agreement in previous works with 3-D MoC [15]. A pin-wise modular ray-tracing was used, with each ray-tracing module being $1.26 \times 1.26 \times 2.38$ cm^3. Convergence criteria of 1e-6 and 1e-5 were used for the eigenvalue and fission norm respectively. 1080 spatial domains, determined by the RSB graph-partitioning method, were used for each calculation.

The macroray method calculations used 3 rays in each radial band and 5 rays in each axial band. The MRT method calculations used 0.05 cm ray-spacing in the radial direction, and 0.15 cm in the axial direction. Massachusetts Institute of Technology (MIT) has reported that for these problems they were able to use 1.5 cm ray-spacing [14], but in this study this was not found to be sufficient. This is most likely due to differences in how rays are set up within the MRT procedures in MPACT and OpenMOC.

Unfortunately, due to an inefficient implementation determining surface indices on parallel boundaries for 3-D MoC in MPACT, CMFD could not be utilized for the MRT calculations. However, the time that the accelerated method would take can be estimated roughly by the ratio of iterations from the MRT to the macrorray calculations (that used CMFD accleration).

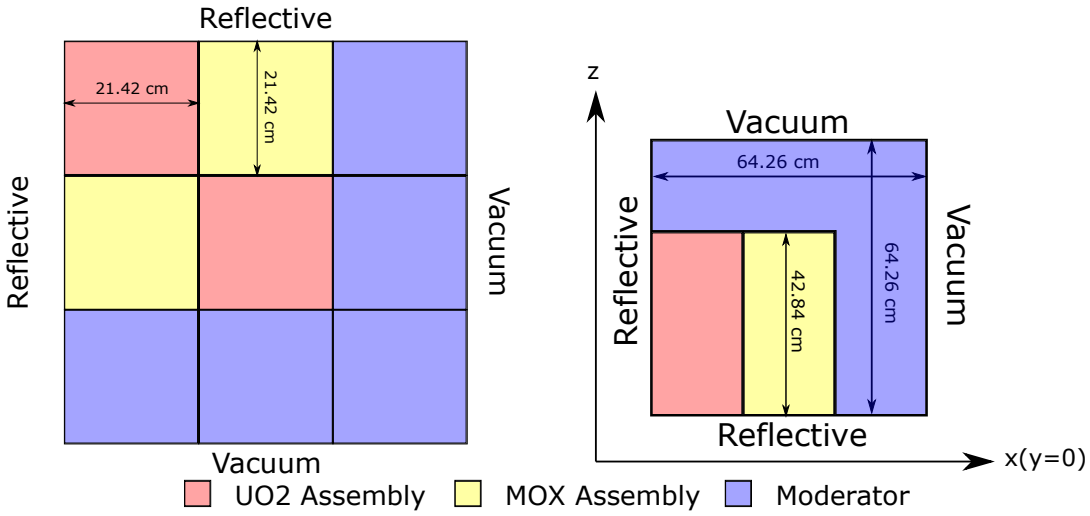


Figure 6.17: C5G7 extended benchmark core layouts.

6.2.3.1 C5G7 Benchmark: Unrodded

The unrodded case has no control rods inserted into the fuel, but control rods are present in the axial reflector region above the fuel region. This is shown in Fig. 6.19.

Eigenvalue and errors are listed in Table 6.1, and pin-power results are summarized in Tables 6.2 and 6.3. The eigenvalues for either ray-tracing method are within 100 pcm of the reference value. Although there is an 8 pcm difference between the two methods, this is not a significant difference.

Overall, the macroray method results in better pin-power comparisons in each metric except for slightly higher error in the maximum pin-power. Additionally, the number of pins within the

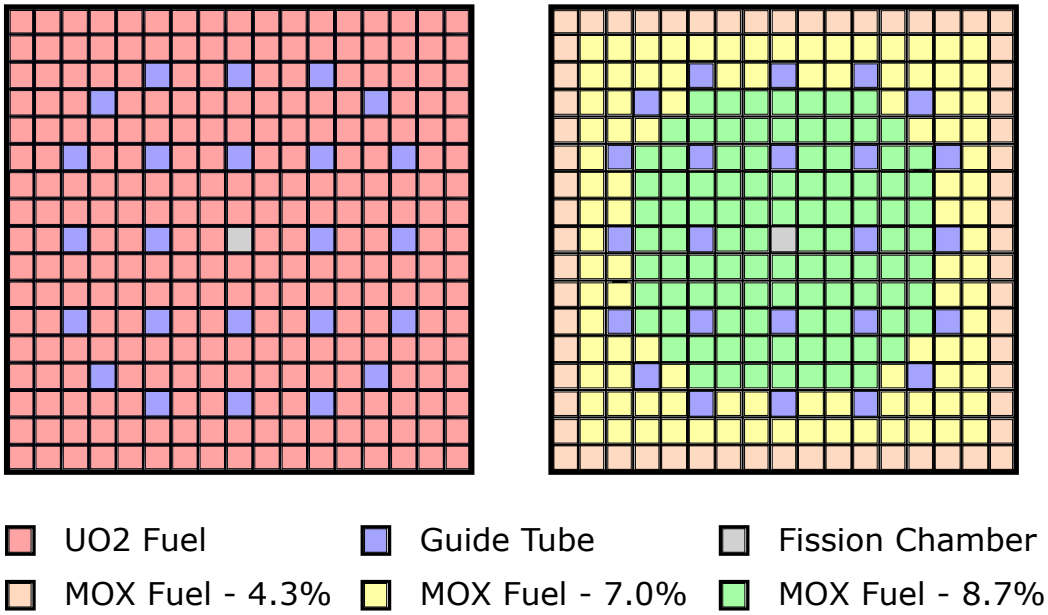


Figure 6.18: C5G7 Assembly layouts.

uncertainty levels of the reference is much higher for the macroray method, indicating better overall predictions in pin-power. However, the macroray method seems to have a bias in the top 1/3 of the fuel, where the power level is lower. The pin-power comparisons within this axial level are significantly worse than in the lower 2/3.

Table 6.1: C5G7 Benchmark: Unrodded eigenvalue comparisons.

Case	Eigenvalue	Error (pcm)
Reference	1.143080	$\pm$ 7
MRT	1.142453	-63
MacroRay	1.142366	-71

Table 6.4 lists the total run-time of each calculation. The MRT calculation took 911 iterations to converge (unaccelerated), while the macroray method took 30 to converge. This gives an estimate for the accelerated MRT method at 240 core-hours. Unfortunately, this is quite significantly better in terms of run-time than the macroray method. A further discussion of this will be presented in Section 6.3.

6.2.3.2 C5G7 Benchmark: Rodded A

The C5G7 rodded A case has control rods partially inserted 14.28 cm, one third of the fuel height, into the central UO_2 assembly. The remaining UO_2 assembly and MOX assemblies have no inserted control rods. The core layout is depicted in Fig. 6.20.

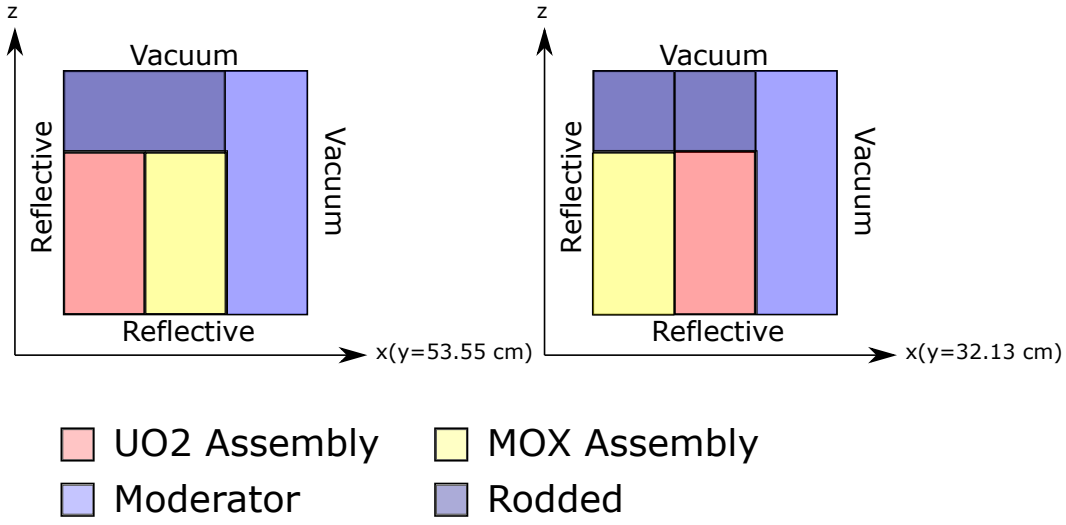


Figure 6.19: C5G7 Benchmark: Unrodded core configuration.

Table 6.2: C5G7 Benchmark: Unrodded pin-power comparisons for the MRT method.

	Z Slice # 1		Z Slice # 2		Z Slice # 3		Overall	
	Reference	MRT	Reference	MRT	Reference	MRT	Reference	MRT
Maximum Pin Power	1.108	1.107	0.882	0.881	0.491	0.488	2.481	2.476
Percent Error	0.090	-0.093	0.100	-0.184	0.130	-0.545	0.060	-0.215
Maximum Error	0.290	2.483	0.320	2.538	0.430	2.618	0.192	2.454
AVG Error	0.164	0.354	0.183	0.358	0.245	0.363	0.109	0.329
RMS Error	0.171	0.522	0.190	0.518	0.255	0.505	0.114	0.486
MRE Error	0.062	0.114	0.055	0.094	0.042	0.058	0.093	0.241
# Pins within 68%	371	177	371	197	371	234	371	142
# Pins within 95%	518	323	518	335	518	392	518	248
# Pins within 99%	540	388	540	414	540	457	540	310
# Pins within 99.9%	544	438	544	456	544	499	544	368
UO2-1 Power	219.04	218.63	174.24	173.91	97.93	97.73	491.21	490.27
MOX Power	94.53	94.61	75.25	75.27	42.92	42.97	212.70	212.85
UO2-2 Power	62.12	62.43	49.45	49.68	27.82	27.91	139.39	140.02
UO2-1 Power % Error	0.082	-0.185	0.073	-0.188	0.055	-0.212	0.123	-0.192
MOX Power % Error	0.061	0.082	0.054	0.032	0.041	0.116	0.092	0.071
UO2-2 Power % Error	0.043	0.508	0.038	0.466	0.029	0.334	0.065	0.458

Table 6.3: C5G7 Benchmark: Unrodded pin-power comparisons for the macroray method.

	Z Slice # 1		Z Slice # 2		Z Slice # 3		Overall	
	Reference	MacroRay	Reference	MacroRay	Reference	MacroRay	Reference	MacroRay
Maximum Pin Power	1.108	1.105	0.882	0.880	0.491	0.491	2.481	2.475
Percent Error	0.090	-0.331	0.100	-0.309	0.130	0.108	0.060	-0.236
Maximum Error	0.280	0.862	0.250	1.131	0.330	2.014	0.192	0.910
AVG Error	0.164	0.257	0.183	0.235	0.245	0.830	0.109	0.190
RMS Error	0.171	0.300	0.190	0.294	0.255	0.902	0.114	0.248
MRE Error	0.062	0.130	0.055	0.083	0.042	0.149	0.093	0.161
# Pins within 68%	371	188	371	229	371	9	371	180
# Pins within 95%	518	328	518	396	518	74	518	342
# Pins within 99%	540	399	540	462	540	132	540	411
# Pins within 99.9%	544	441	544	508	544	232	544	480
UO2-1 Power	219.04	218.21	174.24	173.79	97.93	98.48	491.21	490.47
MOX Power	94.53	94.40	75.25	75.23	42.92	43.33	212.70	212.97
UO2-2 Power	62.12	62.09	49.45	49.48	27.82	28.03	139.39	139.59
UO2-1 Power % Error	0.082	-0.380	0.073	-0.259	0.055	0.554	0.123	-0.151
MOX Power % Error	0.061	-0.134	0.054	-0.021	0.041	0.957	0.092	0.126
UO2-2 Power % Error	0.043	-0.047	0.038	0.053	0.029	0.744	0.065	0.147

Table 6.4: C5G7 Benchmark: Unrodded Performance.

Case	Acclerated?	Time (core-hours)
LSMoC - MRT	F	7293
LSMoC - MacroRay	T	618

Eigenvalue and errors are listed in Table 6.5, and pin-power results are summarized in Tables 6.6 and 6.7. Similarly to the unrodded case, the eigenvalues for each ray-tracing method are within 100 pcm; however, the error is 34 pcm higher for the macroray method. Again, in overall pin-power comparisons, the macroray method outperforms the MRT method in all except the error in the maximum pin-power. Additionally, the number of pins predicted within the uncertainty levels of the reference are significantly higher for the macroray method. But, just like in the unrodded case, the macroray method performs significantly worse in the low-power upper third of the fuel.

Table 6.5: C5G7 Benchmark: Rodded A eigenvalue comparisons.

Case	Eigenvalue	Error (pcm)
Reference	1.128060	$\pm$ 7
MRT	1.127585	-47
MacroRay	1.127248	-81

Table 6.8 lists the total run-time of each calculation. The MRT calculation took 936 iterations to converge (unaccelerated), while the macroray method took 31 to converge. This gives an estimate for the accelerated MRT method at 246 core-hours. Again, this is much better than how the macroray method performed.

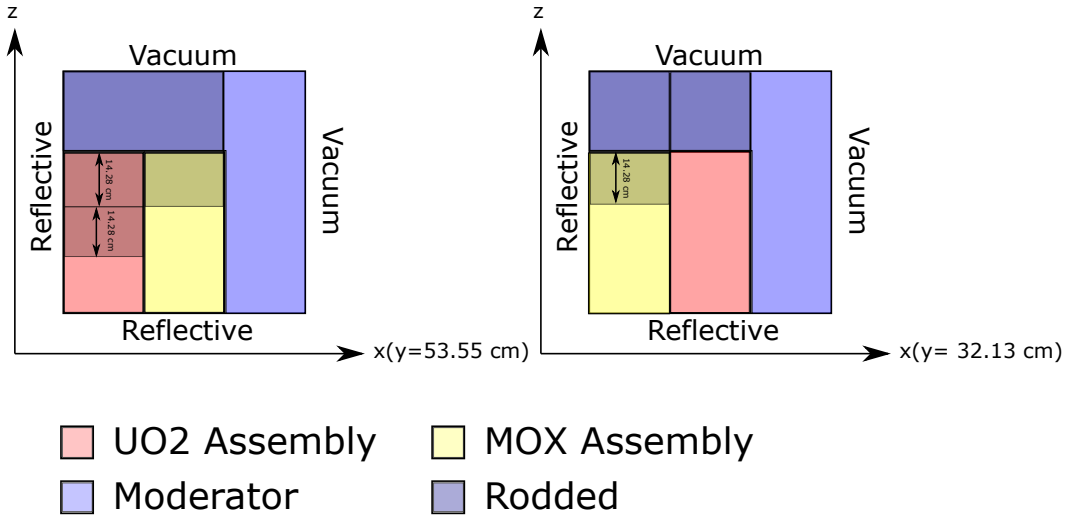


Figure 6.20: C5G7 Benchmark: Rodded A core configuration.

Table 6.6: C5G7 Benchmark: Rodded A pin-power comparisons for the MRT method.

	Z Slice # 1		Z Slice # 2		Z Slice # 3		Overall	
	Reference	MRT	Reference	MRT	Reference	MRT	Reference	MRT
Maximum Pin Power	1.197	1.198	0.832	0.830	0.304	0.300	2.253	2.249
Percent Error	0.080	0.085	0.100	-0.151	0.200	-1.237	0.059	-0.211
Maximum Error	0.270	1.948	0.310	2.168	0.180	2.014	0.183	1.934
AVG Error	0.157	0.429	0.180	0.334	0.260	0.837	0.108	0.294
RMS Error	0.163	0.566	0.186	0.471	0.266	0.962	0.111	0.430
MRE Error	0.066	0.156	0.056	0.093	0.037	0.146	0.094	0.220
# Pins within 68%	371	106	371	197	371	93	371	151
# Pins within 95%	518	237	518	350	518	175	518	270
# Pins within 99%	540	312	540	415	540	223	540	338
# Pins within 99.9%	544	386	544	471	544	281	544	391
UO2-1 Power	237.41	237.80	167.51	167.23	56.26	55.47	461.18	460.49
MOX Power	104.48	104.85	78.01	77.99	39.23	38.95	221.71	221.78
UO2-2 Power	69.80	70.22	53.39	53.58	28.21	28.14	151.39	151.94
UO2-1 Power % Error	0.087	0.164	0.071	-0.170	0.040	-1.404	0.119	-0.149
MOX Power % Error	0.065	0.351	0.056	-0.022	0.040	-0.716	0.094	0.031
UO2-2 Power % Error	0.047	0.608	0.040	0.356	0.029	-0.232	0.068	0.363

Table 6.7: C5G7 Benchmark: Rodded A pin-power comparisons for the macrorray method.

	Z Slice # 1		Z Slice # 2		Z Slice # 3		Overall	
	Reference	MacroRay	Reference	MacroRay	Reference	MacroRay	Reference	MacroRay
Maximum Pin Power	1.197	1.190	0.832	0.828	0.304	0.307	2.253	2.245
Percent Error	0.080	-0.542	0.100	-0.387	0.200	1.088	0.059	-0.387
Maximum Error	0.210	0.976	0.240	1.120	0.350	1.881	0.143	0.985
AVG Error	0.157	0.240	0.180	0.256	0.260	0.937	0.108	0.222
RMS Error	0.163	0.288	0.186	0.320	0.266	0.986	0.111	0.275
MRE Error	0.066	0.136	0.056	0.091	0.037	0.140	0.094	0.198
# Pins within 68%	371	212	371	229	371	8	371	145
# Pins within 95%	518	341	518	369	518	38	518	272
# Pins within 99%	540	390	540	429	540	100	540	360
# Pins within 99.9%	544	433	544	490	544	194	544	439
UO2-1 Power	237.41	236.50	167.51	167.00	56.26	56.74	461.18	460.24
MOX Power	104.48	104.40	78.01	78.04	39.23	39.61	221.71	222.04
UO2-2 Power	69.80	69.76	53.39	53.46	28.21	28.45	151.39	151.67
UO2-1 Power % Error	0.087	-0.382	0.071	-0.305	0.040	0.855	0.119	-0.203
MOX Power % Error	0.065	-0.077	0.056	0.040	0.040	0.972	0.094	0.150
UO2-2 Power % Error	0.047	-0.050	0.040	0.128	0.029	0.849	0.068	0.180

Table 6.8: C5G7 Benchmark: Rodded A Performance.

Case	Acclerated?	Time (core-hours)
LSMoC - MRT	F	7433
LSMoC - MacroRay	T	587

6.2.3.3 C5G7 Benchmark: Rodded B

The C5G7 rodded B case has control rods partially inserted into three of the four fuel assemblies. Each of the MOX assemblies has control rods inserted 14.28 cm. The center UO_2 assembly has control rods inserted 28.56 cm, two thirds of the fuel height; the outer UO_2 assembly has no inserted control rods. This layout is depicted in Fig. 6.21.

Eigenvalue and errors are listed in Table 6.9, and pin-power results are summarized in Tables 6.10 and 6.11. In this case, the macroray method yields eigenvalue error just over 100 pcm, while the MRT method is well under 100. The difference in eigenvalue error is likely primarily due to the fact that the MRT benefits from cancellation of errors, while the macrorray generally does not.

Interestingly, the overall pin-power comparisons are much closer in this case compared to the unrodded and rodded A configurations. The average, RMS and mean relative errors are all similar compared to the MRT method; the error in the maximum power is higher for the macroray method, just as in the previous cases. The macroray method also does not outperform the MRT method in the number of pins within each uncertainty level of the reference. Finally, the macroray method again shows the most significant errors in the upper 1/3 of the fuel, as in previous cases.

Table 6.12 lists the total run-time of each calculation. The MRT calculation took 933 iterations to converge (unaccelerated), while the macroray method took 30 to converge. This gives an estimate for

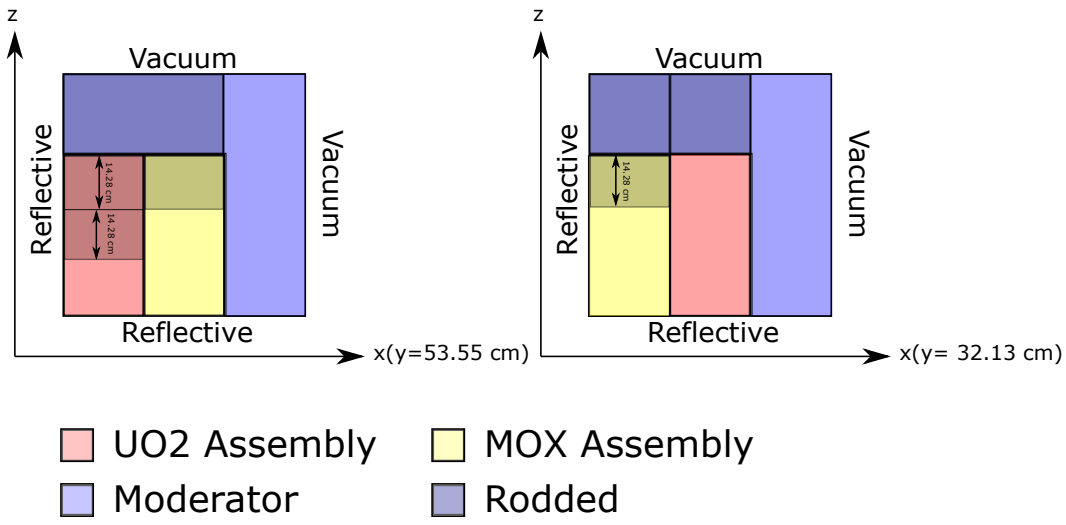


Figure 6.21: C5G7 Benchmark: Rodded B core configuration.

Table 6.9: C5G7 Benchmark: Rodded B eigenvalue comparisons.

Case	Eigenvalue	Error (pcm)
Reference	1.077770	$\pm$ 7
MRT	1.077268	-50
MacroRay	1.076628	-114

Table 6.10: C5G7 Benchmark: Unrodded pin-power comparisons for the MRT method.

	Z Slice # 1		Z Slice # 2		Z Slice # 3		Overall	
	Reference	MRT	Reference	MRT	Reference	MRT	Reference	MRT
Maximum Pin Power	1.200	1.208	0.554	0.549	0.217	0.211	1.835	1.834
Percent Error	0.090	0.691	0.150	-0.909	0.240	-2.738	0.083	-0.065
Maximum Error	0.240	2.449	0.270	1.641	0.240	2.738	0.163	1.850
AVG Error	0.146	0.639	0.181	0.546	0.285	1.593	0.105	0.251
RMS Error	0.150	0.730	0.184	0.619	0.290	1.712	0.108	0.366
MRE Error	0.073	0.348	0.055	0.183	0.034	0.208	0.098	0.203
# Pins within 68%	371	28	371	78	371	25	371	167
# Pins within 95%	518	83	518	161	518	61	518	308
# Pins within 99%	540	124	540	225	540	80	540	375
# Pins within 99.9%	544	183	544	290	544	110	544	427
UO2-1 Power	247.75	249.43	106.56	105.72	41.12	40.23	395.43	395.38
MOX Power	125.78	126.48	81.41	81.09	29.42	28.94	236.62	236.51
UO2-2 Power	91.64	92.22	65.02	65.02	30.68	30.36	187.34	187.60
UO2-1 Power % Error	0.091	0.681	0.056	-0.791	0.035	-2.180	0.112	-0.013
MOX Power % Error	0.073	0.556	0.058	-0.394	0.034	-1.635	0.100	-0.043
UO2-2 Power % Error	0.055	0.636	0.046	-0.010	0.032	-1.046	0.078	0.137

Table 6.11: C5G7 Benchmark: Rodded B pin-power comparisons for the macroray method.

	Z Slice # 1		Z Slice # 2		Z Slice # 3		Overall	
	Reference	MacroRay	Reference	MacroRay	Reference	MacroRay	Reference	MacroRay
Maximum Pin Power	1.200	1.190	0.554	0.553	0.217	0.219	1.835	1.831
Percent Error	0.090	-0.801	0.150	-0.128	0.240	0.832	0.083	-0.215
Maximum Error	0.100	0.925	0.240	1.207	0.370	2.771	0.157	1.100
AVG Error	0.146	0.329	0.181	0.216	0.285	1.322	0.105	0.222
RMS Error	0.150	0.391	0.184	0.291	0.290	1.367	0.108	0.290
MRE Error	0.073	0.225	0.055	0.060	0.034	0.160	0.098	0.203
# Pins within 68%	371	133	371	286	371	1	371	172
# Pins within 95%	518	268	518	444	518	3	518	288
# Pins within 99%	540	332	540	501	540	14	540	342
# Pins within 99.9%	544	385	544	530	544	65	544	405
UO2-1 Power	247.75	246.26	106.56	106.63	41.12	41.63	395.43	394.51
MOX Power	125.78	125.56	81.41	81.48	29.42	29.84	236.62	236.89
UO2-2 Power	91.64	91.55	65.02	65.15	30.68	31.02	187.34	187.71
UO2-1 Power % Error	0.091	-0.600	0.056	0.063	0.035	1.227	0.112	-0.231
MOX Power % Error	0.073	-0.179	0.058	0.095	0.034	1.423	0.100	0.115
UO2-2 Power % Error	0.055	-0.104	0.046	0.192	0.032	1.120	0.078	0.199

the accelerated MRT method at 241 core-hours. Just as the previous two cases, this is significantly lower than the time the macroray method took.

Table 6.12: C5G7 Benchmark: Rodded B Performance.

Case	Acclerated?	Time (core-hours)
LSMoC - MRT	F	7493
LSMoC - MacroRay	T	579

6.3 Discussion on the Performance of MacroRay

While the macroray method was able to reduce the number of segments in some cases, macroray calculations generally took longer for the same number of segments compared to the MRT method. To determine where the performance was lacking, a miniaturized C5G7 unrodded case was profiled, without using CMFD acceleration, and with 18 spatial domains, using both the MRT and macroray methods. For each calculation, nearly all the time was spent in the transport routines (>95%). The transmission calculations and moment accumulation are typically the largest run-time contributors in MRT calculations, and the profiler revealed that 61.8% of the time was spent in these loops. An additional 33% was spent on waiting for messages from other domains.

However, in the macroray calculation, only 22% of the total time was spent on these loops and 15.1 % was spent on the conversion between ray fluxes and surface fluxes. The largest contributor to run-time was waiting for MPI messages, at 57.4%. In the current implementation, a MPI waitall

command is used after sweeping each direction. The fractional run-time can be reduced to 26% if the waitall is moved to after sweeping all directions. This is the same approach used by the MRT sweepers in MPACT. However, the method of generating unique tags must be improved for the macrorray sweepers, because the current method results in a tag overflow on larger cases such as the C5G7 benchmarks.

While the conversion between ray and surface fluxes will never be zero, it is unlikely that the naive implementation of these routines in this work are nearly optimal. A better implementation of these routines may significantly cut down on run-time costs.

6.4 Conclusions

The macroray method was motivated by previous studies on the macroband method that claimed significantly reduced number of rays with maintained accuracy. It was expected that the method, if extended to 3-D, would allow for significantly fewer track-segments, and thus lead to less computational work. For some problems, this is certainly the case; for VERA progression problem 1E, a single IFBA rod, this method was able to significantly reduce the number of segments to reach a converged result. Additionally, for the shielded-box problem, this method is able to act as more of a "black box" solver, allowing for users without deep knowledge of the method to use it.

The method was then tested on the extended 3-D C5G7 benchmarks, and proved it was able to handle larger calculations. Compared to the MRT method on the same mesh, the macroray method typically better-predicted the pin-powers, even when eigenvalue errors were larger. Additionally, the macroray method consistently showed the highest errors in the upper 1/3 of the fuel in the C5G7 benchmark cases, where the pin-powers are lower.

In the C5G7 benchmark cases, which do not have small absorber regions, the method was unable to significantly reduce the number of segments for similar accuracy. It is expected that the macroray method may show more beneficial results in cases where there are small strong absorber regions present. Additionally, the macroray method may show significant benefits for more arbitrary geometries, such as the shielded-box.

This work is the first implementation of the macroray method for 3-D MoC calculations. Due to the significantly different ray-tracing data structure, and sweeping order, separate routines needed to be written for this work. It is not surprising that the initial performance of these routines is worse than that of the MRT-based routines, for similar numbers of segments, given that the MRT sweeping routines have been optimized over the past several years of the CASL project. The original performance of the 3-D MoC using MRT in MPACT was significantly worse [15]. While some of the improvements made to these methods were utilized in this work, it is very likely that optimization work on this method would significantly reduce run-times.

Bibliography

[1] Akio Yamamoto et al. “Non-Equidistant Ray Tracing for the Method of Characteristics”. In: *M&C 2005* (2005), pp. 1–10.

[2] François Févotte, Simone Santandrea, and Richard Sanchez. “Advanced Transverse Integration for the Method of Characteristics”. In: *M&C 2007*. 2007, pp. 1–12. ISBN: 0894480596.

[3] Eduardo A. Villarino et al. “HELIOS: Angularly Dependent Collision Probabilities”. In: *Nuclear Science and Engineering* 112.1 (1992), pp. 16–31. ISSN: 0029-5639. DOI: 10.13182/NSE112-16.

[4] Akio Yamamoto. “Reduction in spatial discretization error in the method of characteristics by using the mobile-chord ray tracing method”. In: *Annals of Nuclear Energy* 35.5 (2008), pp. 783–789. ISSN: 03064549. DOI: 10.1016/j.anucene.2007.09.018.

[5] Richard Sanchez. “Prospects in deterministic three-dimensional whole-core transport calculations”. In: *Nuclear Engineering and Technology* 44.5 (2012), pp. 113–150. ISSN: 17385733. DOI: 10.5516/NET.01.2012.501.

[6] Daniele Sciannandrone, Simone Santandrea, and Richard Sanchez. “Optimized tracking strategies for step MOC calculations in extruded 3D axial geometries”. In: *Annals of Nuclear Energy* 87 (2016), pp. 49–60. ISSN: 18732100. DOI: 10.1016/j.anucene.2015.05.014.

[7] Geoffrey Alexander Gunow. “Full Core 3D Neutron Transport Simulation Using the Method of Characteristics with Linear Sources”. Doctoral. Massachusetts Institute of Technology, 2018.

[8] A. T. Godfrey. *VERA Core Physics Benchmark Progression Problem Specifications*. Tech. rep. 4. 2014, pp. 1–173. URL: https://www.casl.gov/sites/default/files/docs/CASL-U-2012-0131-004.pdf.

[9] Akio Yamamoto et al. “Derivation of optimum polar angle quadrature set for the method of characteristics based on approximation error for the bickley function”. In: *Journal of Nuclear Science and Technology* 44.2 (2007), pp. 129–136. ISSN: 00223131. DOI: 10.1080/18811248.2007.9711266.

[10] Ang Zhu 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. ISSN: 18732100. DOI: 10.1016/j.anucene.2016.05.004.

[11] Paul K. Romano et al. “OpenMC: A state-of-the-art Monte Carlo code for research and development”. In: *Annals of Nuclear Energy* (2015). ISSN: 18732100. DOI: 10.1016/j.anucene.2014.07.048.

[12] K.S. Smith and J.D. Rhodes III. “Full-Core, 2-D, LWR Core Calculations with CASMO-4E”. In: *Physor* 1 (2002), pp. 1–13.

[13] M. A. Smith, E. E. Lewis, and Byung Chan Na. “Benchmark on Deterministic 2-D MOX Fuel Assembly Transport Calculations without Spatial Homogenization”. In: *Progress in Nuclear Energy* 48.5 (2006), pp. 383–393. ISSN: 01491970. DOI: 10.1016/j.pnucene.2006.01.002.

[14] Geoffrey Gunow et al. “Reducing 3D MOC Storage Requirements with Axial On- the-fly Ray Tracing”. In: *Physics of Reactors 2016, PHYSOR 2016: Unifying Theory and Experiments in the 21st Century*. Sun Valley: American Nuclear Society, 2016. URL: http://hdl.handle.net/1721.1/109864{\%}0ACreative.

[15] Brendan Matthew Kochunas. “A Hybrid Parallel Algorithm for the 3-D Method of Characteristics Solution of the Boltzmann Transport Equation on High Performance Compute Clusters”. Doctoral. University of Michigan, 2013, pp. 1–194.

CHAPTER 7

Summary, Conclusions, and Future Work

This chapter gives a brief summary of the motivations for this work, and what was accomplished. The key findings for each study are listed. Also, possible paths for future research in these topics are given.

7.1 Summary of Work

7.1.1 Summary: Spatial Decomposition

Prior to this work, MPACT [1] used an automated spatial decomposition method based on the fuel assemblies. This method was limited, and only allowed for specific numbers of spatial domains to be used. Additionally, such methods have been shown, in other codes, to result in poor load balance, particularly when a large coarsely-meshed radial reflector is present [2]. Stimpson [3] improved MPACT's parallel capabilities, allowing for a problem to be decomposed into groups of ray-tracing modules. However, that work did not introduce a method for generating better spatial decompositions, and relied on the user to enter a decomposition map. This was tedious for the user, and ofter resulted in non-optimal decompositions.

This dissertation's work on spatial decomposition sought to develop decomposition methods to

1. improve load-balance,
2. improve user experience (no additional work),
3. and, allow for less restrictions on the number of domains.

These improvements were made by utilizing graph partitioning methods, which have been highly studied in computer science [4]. Software libraries exist for performing graph partitioning (METIS [5], Zoltan [6], etc.). However, due to MPACT's strict requirements on the decomposition (see Section 4.2), these libraries could not easily be used in MPACT. Several graph partitioning methods

were implemented in MPACT. For unweighted graphs, and not conforming to MPACT's restrictions, these method were shown to be comparable to METIS [7].

The methods were then applied to MPACT and made to conform with the restrictions MPACT places on its spatial decomposition [8]. This work introduced a new graph partitioning method based on the recursive expansion methods, and an improvement to the Kernighan-Lin [9] algorithm for the application to MoC. These new methods, as well as the recursive spectral bisection (RSB) and recursive intertial bisection (RIB), were used to decompose a 2-D transport problem, and resource comparisons of the simulations were compared. In 3-D, several different decomposition approaches were investigated, and a 3-D core was decomposed. For the 3-D calculations, only metrics of the decomposition were compared, rather than actual simulation results.

7.1.2 Summary: Linear Source

In the past, Method of Characteristics (MoC) codes have used the flat-source approximation (FSA), but more recently codes have been using the linear-source approximation (LSA) [2, 10–16]. The general goal of these LSAs is to coarsen the computational mesh, leading to overall improved run-times. However, many of these previous LSAs have been *ad-hoc*. Several of these methods [12, 15, 16] have been based on particle conservation. The LSA developed by Ferrer and Rhodes [16] was implemented into the MPACT code [17, 18].

Once implemented in MPACT, the LSA was tested on several problems, and two flaws were observed. The first was an implementation detail; typically, exponential functions in the MoC are approximated to reduce run-time [19]. However, when the exponential functions in the LSA were approximated in problems with near-void regions, such as the fuel-clad gap, iterative instability was observed. Different methods for addressing this instability were studied in this work [17].

The second flaw was that for problems with multiple physics (thermal-hydraulic (T/H) feedback, or isotropic depletion) or in the 2D/1D approximation [20], the method was far less efficient. This was due to the changing macroscopic cross sections, which led to coefficients needing to be recalculated at every state-point, or even every iteration. This work sought to eliminate the need to recompute these coefficients by eliminating their dependence on the cross section [18]. The elimination of the group-dependent coefficients in the LSA was achieved by re-formulating the method Chapter 5.

7.1.3 Summary: MacroRay

Three-dimensional MoC calculations are generally considered to be too expensive for use in realistic calculations; the methods rarely see use outside academia or national laboratories. The primary factor in determining MoC run-time is the number of track segments. By using a coarser ray-spacing,

far fewer track-segments are generated, which can significantly improve run-time (Section 5.2). However, the presence of small strong absorber regions anywhere in a problem can prevent the use of coarser rays. This motivates the use of a ray-tracing method that is able to use different effective ray-spacings in regions with different requirements.

The macroband [11, 21–23] is a ray-tracing method which bases the placement of rays on the computational mesh of the problem. This method forces the use of finer ray-spacing where small regions are present, and allows for coarser ray-spacing where there are not small regions. With the use of non-uniform ray-spacing [22], these methods can reduce the number of rays necessary for accurate integration. However, the method is not compatible with direct neutron path linking (DNPL), meaning that an approximation of the angular flux at interfaces is necessary, unlike the traditional modular ray-tracing (MRT) ray-tracing methods.

While the macroband has been previously studied for 2-D MoC, there has been no extension or investigation of the method for 3-D MoC calculations. This dissertation seeks to fill that gap, and provides one possible extension of the macroband to 3-D problems. This extension, the macroray ray-tracing method, was implemented in a library dependent on MPACT, and tested on several 3-D transport problems. The method was compared against the traditional MRT ray-tracing method, with a particular focus on accuracy versus the number of track-segments.

7.2 Conclusions

7.2.1 Conclusions: Spatial Decomposition

The use of graph partitioning for spatial decomposition in MPACT has been very successful. The methods have greatly improved user experience, as they allow for an arbitrary number of radial domains in 2D/1D calculations [1]. Unlike previous capabilities in MPACT, these methods also require no additional effort from the user. User feedback has been very positive on these methods, and the default partitioning in MPACT is now the REB graph partitioning method.

For 2-D (or 2D/1D) calculations, the three graph partitioning methods tested did not yield significantly different results. However, for general 3-D decompositions, the RSB and RIB methods generally performed better than the REB method for low to medium numbers of domains. The REB method showed significant advantage for highly decomposed cores. These methods greatly improved the parallel efficiency of the MoC calculation (per iteration).

However, as more spatial domains are used, the convergence rate decreases and more iterations are necessary. This problem is present in previous methods, but seems to be worsened by using the graph partitioning methods; this is likely due to re-entrant rays becoming possible in this methods.

[1] or an arbitrary number of domains in 3-D MoC calculations

Additionally, the parallel scaling of the CMFD solvers in MPACT is worse than that of the MoC solvers. In fact, for highly decomposed cores, the CMFD acceleration takes up more run-time than the MoC.

7.2.2 Conclusions: Linear Source

The modified exponential function used to address the instability in LSMoC calculations with near void regions was successful. Problems that previously demonstrated instabilities were no longer unstable, even when using an interpolation table with the same accuracy as that used with FS calculations. Additionally, the study performed on more accurate tables showed that using an interpolation table for a single function and computing other functions was more efficient than tabulating the three functions. This was able to improve exponential calculation time by 9% and total MoC run-time by 3%, compared to the previous method of tabulating all three functions.

The improved formulation of the LSA was able to entirely eliminate the group-dependent coefficients by reducing them to only spatial dependent terms. In fact, these terms were reduced to a form of the same matrix used to transform from spatial moments to the spatial expansion coefficients. It is the author's opinion that the form found in this work is more elegant than those previously presented, and does not introduce any additional operations (assuming a modern computer architecture, with fused multiply add (FMA) instructions). The elimination of the group-dependence in these coefficients was able to reduce run-times in multiphysics calculations by up to 15%, as the recomputation of these coefficients was no longer necessary.

Mesh sensitivity studies were performed on several VERA progression problems [24]. Coarse mesh parameters were found, which maintained accuracy, while reducing the number of source regions in realistic calculations by up to 85%. On VERA problem 9, a 3-D cycle depletion of a quarter core with T/H feedback showed 10% improvement in total run-time, and 30% reduction in memory. Previous studies have shown that the LSA was able to improve run-times in neutronics only calculations [2, 16]; this work has shown that the use of the improved formulation allows for advantages over the FSA even in cases with multiple physics.

7.2.3 Conclusions: MacroRay

The macroray ray-tracing method is a novel ray-tracing method studied as part of this dissertation work. The aim of this method is to allow for maintained accuracy while reducing the number of track-segments generated during ray-tracing, and thus reducing the amount of work during computation. The underlying idea is to use advanced transverse integration methods to improve the integration accuracy within each region. A similar idea, the macroband, had been studied in 2-D, but no extension to 3-D has been made.

The method was shown to reduce the number of track-segments necessary for calculations in some cases with thin strong absorbers. Though, in benchmarks without these features, the method did not show significant advantages over traditional ray-tracing methods. Unlike traditional ray-tracing methods, this method exhibits near uniform convergence with number of segments. This means as the number of rays is increased, the user can expect that the result will become closer to the converged result. Additionally, because of the dependence on internal geometry for ray-tracing, rays are forced through thin regions; this means the method can be used as more of a "black box" solver, without requiring knowledge of the method by the user.

7.3 Future Work

7.3.1 Future Work: Spatial Decomposition

The spatial decomposition in MPACT has several avenues for future improvement. This most obvious is to improve the parallel efficiency of the CMFD solvers; CMFD begins to dominate run-time as large numbers of spatial domains are used. The use of a multigrid CMFD solver [25], may reduce the total run-time fraction of CMFD relative to what was found in this work, but the multigrid solver does not address the poor parallel scaling. The next is that, the methods of this work created and partitioned a graph that only considered the computational expenses of MoC. As CMFD becomes a significant component of run-time in highly decomposed cases, it is warranted to investigate constructing a graph that considers both MoC and CMFD costs. Finally, it may be possible to modify the graph partitioning methods to avoid "jagged" domains, which negatively impact convergence, without negatively affecting load-balance.

7.3.2 Future Work: Linear Source

The LSA has been shown to allow for significantly coarser meshes while maintaining accuracy. While unlikely to improve run-times for 2-D calculations, some groups have begun to investigate quadratic axial sources for 3-D MoC calculations. The use of these methods would allow the axial mesh to be significantly coarsened. Due to the success of the LSA, a quadratic axial source may be able to improve run-times for axially large calculations, such as reactors. It may also be possible to extend Ferrer and Azmy's [26] work for an arbitrary source-order MoC for general geometries.

Finally, the use of GPGPUs has been studied for FSMoC calculations, but not for the LSMoC. Many recent computing clusters require codes to utilize GPGPUs; if this method is to see continued use, the efficient implementation on GPGPUs is a necessity. Even if clusters are ignored, GPGPUs allow for significant speed-up of MoC calculations, and it is the author's opinion that all modern

MoC codes should utilize them.

7.3.3 Future Work: MacroRay

While the macroray was shown to reduce the number of segments for some cases, it is generally still slower than the solvers using the traditional ray-tracing methods. This is not entirely surprising, as the traditional solvers have been heavily optimized over the past decade of the CASL project, whereas the macroray solver was a first-pass implementation for this research. Nonetheless, if the method is to be used, the efficiency of the implementation must be improved; in particular, the MPI communication, and the cost of the interface approximations must be made more efficient. One possibility to improve these is to use a different approximation of the flux on the interfaces. While this work has been initial research into the use of the macroray method, if the method is to be considered for use in production, it will need to be implemented efficiently on GPGPUs.

Additionally, one of the primary difficulties in this work was due to the choice of rectangular rays in the 2D-2D ray-tracing approach. The use of 2D-2D ray-tracing is common for MRT method but may not be the best choice for macroray methods. Indeed, there is a good argument against its use due to the fact that rays are guaranteed not to "span" the problem's surfaces. A reasonable next step for this work would be to investigate the use of non-rectangular rays, which eliminate the need for some of the fix-ups used in this work.

Bibliography

[1] MPACT Team. *MPACT Theory Manual v2.2.0*. Tech. rep. Consortium for Advanced Simulation of Light Water Reactors, 2016.

[2] Geoffrey Alexander Gunow. "Full Core 3D Neutron Transport Simulation Using the Method of Characteristics with Linear Sources". Doctoral. Massachusetts Institute of Technology, 2018.

[3] Shane Stimpson and Benjamin Collins. "Flexible Spatial Partitions in MPACT Through Module-Based Data Passing". In: *Transactions of the American Nuclear Society*. Vol. 116. San Francisco, 2017, pp. 654–657.

[4] Ulrich Elsner. *Graph partitioning A Survey*. Tech. rep. Technische Universitat Chemnitz, 1997, p. 52. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.54.2060.

[5] George Karypis and Vipin Kumar. "A fast and high quality multilevel scheme for partitioning irregular graphs". In: *SIAM Journal on Scientific Computing* 20.1 (1998), pp. 359–392.

[6] E. G. Boman et al. "The Zoltan and Isorropia Parallel Toolkits for Combinatorial Scientific Computing: Partitioning, Ordering, and Coloring". In: *Scientific Programming* 20.2 (2012), pp. 129–150.

[7] Andrew Fitzgerald et al. "Automated Decomposition of a Structured Grid". In: *Transactions of the American Nuclear Society*. Vol. 117. Washington D.C., 2017, pp. 731–734.

[8] Andrew P Fitzgerald et al. "Spatial decomposition of structured grids for nuclear reactor simulations". In: *Annals of Nuclear Energy* 132 (2019), pp. 686–701. ISSN: 0306-4549. DOI: 10.1016/j.anucene.2019.06.054.

[9] B.W. Kernighan and S. Lin. "An efficient heuristic for partitioning graphs". In: *Bell Systems Technical Journal* 49 (1970), pp. 291–308.

[10] M. J. Halsall. "Neutron Transport in WIMS by the Characteristics Methods". In: *Transactions of the American Nuclear Society1*. 1993, pp. 454–455.

[11] Petko Petkov, Toshikazu Takeda, and Takamasa Mori. “Comparison of the Flat and Linear Source Variants of the Method of Characteristics”. In: *Annals of Nuclear Energy* 26.10 (1999), pp. 935–942. ISSN: 03064549. DOI: 10.1016/S0306-4549(98)00109-1.

[12] S Santandrea and R Sanchez. “POSITIVE LINEAR AND NONLINEAR SURFACE CHARACTERISTIC SCHEMES FOR THE NEUTRON TRANSPORT EQUATION IN UNSTRUCTURED GEOMETRIES”. In: *Physor*. 2002.

[13] Chuntao Tang and Shaohong Zhang. “Development and verification of an MOC code employing assembly modular ray tracing and efficient acceleration techniques”. In: *Annals of Nuclear Energy* 36.8 (2009), pp. 1013–1020. ISSN: 03064549. DOI: 10.1016/j.anucene.2009.06.007.

[14] C Rabiti et al. “Quasi Linear Representation of the Isotropic Scattering Source for the Method of Characteristics”. In: *International Conference on Mathematics, Computational Methods & Reactor Physics* January (2009), pp. 3–7.

[15] Alain Hébert. “High-Order Linear Discontinuous and Diamond Differencing Schemes Along Cyclic Characteristics”. In: *Nuclear Science and Engineering* 184.4 (2016), pp. 591–603. ISSN: 0029-5639. DOI: 10.13182/nse16-82.

[16] Rodolfo M. Ferrer and Joel D. Rhodes. “A Linear Source Approximation Scheme for the Method of Characteristics”. In: *Nuclear Science and Engineering* 182.2 (2016), pp. 151–165. ISSN: 0029-5639. DOI: 10.13182/NSE15-6.

[17] Andrew Fitzgerald and Brendan Kochunas. “Fast Exponential Function Approximations for the Method of Characteristics with Linear Source”. In: *Transactions of the American Nuclear Society*. Orlando, USA, 2018, pp. 645–648.

[18] Andrew P Fitzgerald, Brendan Kochunas, and Thomas Downar. “Improved Formulation of the Method of Characteristics with Linear Source for 2D/1D and Multiphysics Calculations”. In: *M&C*. Portland, OR, 2019, pp. 1093–1103.

[19] Akio Yamamoto, Yasunori Kitamura, and Yoshihiro Yamane. “Computational efficiencies of approximated exponential functions for transport calculations of the characteristics method”. In: *Annals of Nuclear Energy* 31.9 (2004), pp. 1027–1037. DOI: 10.1016/j.anucene.2004.01.003.

[20] Benjamin Collins 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. ISSN: 10902716. DOI: 10.1016/j.jcp.2016.08.022.

[21] Eduardo A. Villarino et al. “HELIOS: Angularly Dependent Collision Probabilities”. In: *Nuclear Science and Engineering* 112.1 (1992), pp. 16–31. ISSN: 0029-5639. DOI: 10.13182/NSE112-16.

[22] Akio Yamamoto et al. “Non-Equidistant Ray Tracing for the Method of Characteristics”. In: *M&C 2005* (2005), pp. 1–10.

[23] François Févotte, Simone Santandrea, and Richard Sanchez. “Advanced Transverse Integration for the Method of Characteristics”. In: *M&C 2007*. 2007, pp. 1–12. ISBN: 0894480596.

[24] A. T. Godfrey. *VERA Core Physics Benchmark Progression Problem Specifications*. Tech. rep. 4. 2014, pp. 1–173. URL: https://www.casl.gov/sites/default/files/docs/CASL-U-2012-0131-004.pdf.

[25] Ben C. Yee. “A Multilevel in Space and Energy Solver for Multigroup Diffusion and Coarse Mesh Finite Difference Eigenvalue Problems”. Doctoral. University of Michigan, 2018.

[26] Rodolfo M. Ferrer and Yousry Y. Azmy. “A Robust Arbitrarily High-Order Transport Method of the Characteristic Type for Unstructured Grids”. In: *Nuclear Science and Engineering* 172.1 (2012), pp. 33–51. ISSN: 0029-5639. DOI: 10.13182/NSE10-106.