TRANSIENT METHOD OF CHARACTERISTICS VIA THE ADIABATIC, THETA, AND MULTIGRID AMPLITUDE FUNCTION METHODS

by

Samuel Christopher Shaner

B.S., Chemical Engineering (2011)
University of California at Santa Barbara

SUBMITTED TO THE DEPARTMENT OF NUCLEAR SCIENCE AND ENGINEERING
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE IN NUCLEAR SCIENCE AND ENGINEERING

AT THE

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2014

© 2014 Massachusetts Institute of Technology
All rights reserved

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
JUL 2 9 2014
LIBRARIES
ARCHIVES

Signature redacted

Signature of Author .....
Samuel Christopher Shaner
Department of Nuclear Science and Engineering
May 13, 2014

Signature redacted

Certified by .....
Kord S. Smith
KEPCO Professor of the Practice of Nuclear Science and Engineering
Thesis Supervisor

Signature redacted

Certified by .....
Benoit Forget
Associate Professor of Nuclear Science and Engineering
Thesis Supervisor

Signature redacted

Accepted by .....
Mujid S. Kazimi
TEPCO Professor of Nuclear Engineering
Chair, Department Committee on Graduate Students

Transient Method of Characteristics via the Adiabatic, Theta, and Multigrid Amplitude Function Methods

by

Samuel Christopher Shaner

Submitted to the Department of Nuclear Science and Engineering
on May 13, 2014, in partial fulfillment of the requirements for the degree of
Master of Science in Nuclear Science and Engineering

Abstract

In this thesis, we investigated the Adiabatic, Theta, and Multigrid Amplitude Function (MAF) methods for solving the Method of Characteristics (MOC) formulation of the time-dependent neutron transport equation. The transient transport versions of the 2D LRA and C5G7 benchmarks were used to assess the performance and accuracy of these methods. We began by deriving the CMFD-accelerated MOC algorithm in 2D steady state form and examining the effects of various MOC and CMFD parameters on eigenvalue convergence. The C5G7 problem showed similar acceleration performance for 2, 4, and 7 CMFD energy group structures. CMFD meshes at or near the pin-cell level resulted in the greatest speedups of 15-45× in run time and 30-240× in number of MOC iterations for both problems. A relaxation factor on the nonlinear diffusion coefficient was required to maintain stability for both problems with optimum values between 0.4-0.7.

A sensitivity study was conducted on the C5G7 and LRA transient problems to understand the effects of time step and spatial mesh sizes on the solution accuracy and run time performance. The shape function time step size had a large effect on the solution accuracy for the MAF and Theta methods in solving the LRA problem. All methods showed moderate sensitivity to the amplitude function step size, where increasing step size shifted the peak power to earlier times. The coarse mesh size did not have a significant effect on solution accuracy, but meshes on the pin-cell level were clearly preferred to reduce run time. The overall run time performance between the three methods was mixed. The MAF and Theta methods displayed ~84% speedup over the Adiabatic method for the LRA problem, but all methods had similar run times for the C5G7 problem. This inconsistency is likely due to the more drastic flux shape change during the LRA transient and the ability of the MAF and Theta methods to more accurately treat the flux shape temporal derivative. These results demonstrate that the Adiabatic, Theta, and MAF methods are computationally efficient methods for solving the time-dependent neutron transport equation and warrant further investigation. There are clear advantages to each method and the optimal method will likely depend on the transient characteristics of the problem being studied.

Thesis Supervisor: Kord S. Smith
Title: KEPCO Professor of the Practice of Nuclear Science and Engineering

Thesis Supervisor: Benoit Forget
Title: Associate Professor of Nuclear Science and Engineering

Acknowledgments

First I would like to thank my advisors, Professor Kord S. Smith and Professor Benoit Forget, for their support, patience, and guidance in my research and academic endeavors. This project began as a collaborative effort with fellow graduate students William Boyd and Lulu Li and to them I extend my sincerest graditude. I would like to acknowledge fellow colleagues and mentors Bryan Herman and Paul Romano for helpful insights. Fruitful discussions with other friends and fellow students are also gratefully acknowledged.

I would also like to thank my parents, Martin and Lisa, and brother, Matt, for their support, love, and encouragement. They have always been supportive of my academic development and helped me remember what is truly important in life. Thank you to two of my good friends, Jacob DeWitte and Andrew Mazzone, for their friendship, advice, and providing me with distractions outside of my work.

This work is supported by the DOE Office of Nuclear Energy's Nuclear Energy University Programs Fellowship. Additional support provided by the DOE's Center for Exascale Simulation of Advanced Reactors (CESAR) is gratefully acknowledged.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Radiation emitted from nuclear reactions in the sun provide the energy for photosynthetic organisms to store chemical energy in organic molecules that eventually become oil, natural gas, coal, biomass, and food. Nuclear reactions within our planet produce geothermal energy that can be captured by drilling down into the Earth's crust. Solar radiation heats our atmosphere producing pressure gradients that drive the wind. The heating of our atmosphere also fuels the hydrologic cycle that replenishes our dams with stored potential energy. All of the energy that our society uses began as nuclear potential energy that was converted into kinetic, potential, and electromagnetic energy through nuclear reactions in our sun and our planet. Our successful endeavor to harness this fundamental energy source in the last 7 decades has led to significant technilogical progress and hope that nuclear energy can be an option to combat climate change. One of man's greatest engineering achievements in the 20th century was the harnessing of nuclear energy to produce electricity.

The study and practice of harnessing nuclear reactions for useful purposes is deemed nuclear engineering, but in practice there is significant overlap with nearly all scientific and engineering disciplines due to the complex material, biological, chemical, and mechanical challenges that arise from interactions with ionizing radiation and harnessing large amounts of energy. Nuclear reactor physics, a subset of nuclear engineering that deals with understanding how to control the nuclear fission chain reaction, is one area where nuclear engineers receive much more training than any

other discipline. The neutron transport equation is the fundamental particle balance equation that nuclear engineers use to understand the neutron behavior inside a medium. The equation describes the motion of non-interacting particles (e.g neutrons or photons) under steady-state or transient conditions. Solving the neutron transport equation for systems of industrial interest requires immense computing power that has led many researchers to reduce the transport equation to the more manageable diffusion equation. The diffusion equation applies Fick's Law, a simple heuristic that postulates that particles will diffuse from areas of high concentration to areas of low concentration with a magnitude proportional to the spatial flux. The diffusion equation is particularly well suited for solving homogenized full-core reactor physics problems, but breaks down on heterogeneous problems where neutron flux and material property gradients can be steep or discontinuous.

In the last 40 years, the exponential growth in computing power has propelled a steady increase in the the capability and accuracy of reactor physics codes. While computers have been used to solve the neutron transport equation since the beginning of the Manhattan Project (albeit not always electronic ones), it is only within the last 10-20 years that significant efforts have been focused on solving the transport equation for practical problems. Methods for solving the transport equation fall into one of two camps: stochastic and deterministic. Stochastic neutron transport is dominated by the Monte Carlo method, which directly simulates a population of neutrons and uses random numbers to model the probabilistic nature of physical events. This approach excels in treating complex geometries and in accuracy over other methods, but often requires hours or days to compute one state point in a real power reactor. Deterministic methods such as the method of characteristics (MOC), the discrete ordinates method (S_n), the method of spherical harmonics (P_n), and collision probability method (CPM) use numerical analysis techniques to approximate the solution of the transport equation. They rely on discretization of the angular, spatial, and energetic dependencies of the neutron flux and material properties in a given domain. While deterministic methods are less accurate than Monte Carlo methods, they often have a much lower computational cost, memory requirement,

and are well suited for parallel implementation on many computer architectures. The method of characteristics in particular has received significant interest in the past few decades owing to its scalability and ability to model complex geometries without undue homogenization. For these reasons, we have chosen to use MOC as our method for transient analysis.

Complementing the rise in interest of using transport methods to analyze steady state reactor physics problems, a shift from diffusion to transport methods for transient simulations has also occurred. Many methods for solving the time-dependent neutron transport and neutron diffusion equations rely on the separation of the spatial (shape) and temporal (amplitude) components of the neutron flux during a transient simulation. Solving for the fine-grained neutron flux at a particular moment in time typically comprises the majority of the run time in a transient simulation. For this reason, we seek to develop methods that minimize the number of fine-grained spatial solves. Solving for the temporal component of the neutron flux often involves a spatial homogenization and/or energy condensation allowing for the neutron flux to be approximated at times between spatial solves. This concept of separating the spatial and temporal portions of a transient simulation can be employed because the spatial component of the neutron flux is often slowly varying whereas the amplitude component varies rapidly.

In 2012, a new spatial kinetics approach called the Multigrid Amplitude Function (MAF) method was proposed to enable long time step transient transport analysis. The MAF method is a generalization of other spatial kinetics methods including the Improved Quasi-static method, the Theta method, and the Frequency Transform method and it can be built on top of nodal diffusion, Monte Carlo, and deterministic transport methods. The MAF method is also flexible in the treating the angular flux derivative directly or applying the isotropic flux derivative approximation.

In this thesis, we begin by deriving the steady state MOC algorithm and discussing its implementation in OpenMOC, an open-source 2D neutron transport code developed at MIT, in Chapter 2. In conjunction with this, we will discuss the coarse mesh finite difference (CMFD) acceleration procedure and how it can be used to accelerate

MOC. In Chapter 3, we will present results from our analysis of two steady state neutron transport problems. In Chapter 4 we will derive the time dependent MOC equations for the Adiabatic, Theta, and MAF methods and discuss their implementation in OpenMOC. In Chapter 5, two transient test problems will be presented. The first problem is the transport version of the LRA benchmark, a 2 group assembly-homogenized BWR diffusion transient problem with thermal feedback. Second, a transient version of the C5G7 benchmark, a steady-state, 7 group deterministic neutron transport benchmark with heterogeneous fuel and moderator materials, is presented. Sensitivity studies will be performed on these problems with respect to shape function time step, amplitude function time step, and coarse mesh size. In Chapter 6, summary, conclusions and recommendations on future work will be discussed.

Chapter 2

Steady State Method of Characteristics

The method of characteristics (MOC) is a widely used technique for solving partial differential equations, including the Boltzmann form of the neutron transport equation. Originally proposed in 1972 for solving reactor physics problems [8], it has seen widespread adoption in the reactor physics community with codes including CASMO, MCCG3D, APOLLO2, DRAGON, MPACT, DeCART, nTRACER, and OpenMOC. While MOC naturally extends to 3D, most implementations have been in 2D due to the computational cost and large memory requirements for modeling the full angular flux dependence in 3D geometries. Additionally, most of the physics effects in lattice calculations can be modeled through 2D MOC due to the relatively uniform material properties in the axial direction of LWR assemblies.

In this study, we limit ourselves to 2D MOC to allow us to perform many trials and understand the effects of simulation parameters on performance and scalability of the methods. We begin with an overview and derivation of the MOC method in section 2.1. Next, we discuss coarse mesh finite difference (CMFD) diffusion and how it can be used to accelerate the convergence of MOC. While most of the work in implementing the MOC solver in OpenMOC was performed by William Boyd ([21], [22]), we present the derivation and implementation of MOC for completeness and as an introduction before deriving time dependent MOC in chapter 4. The time

dependent aspects of OpenMOC have been implemented wholly by the author of this thesis.

2.1 Derivation of the steady state MOC algorithm in 2D

MOC is implemented by discretizing the transport equation over polar and azimuthal angle and integrating the resultant equation over constant direction "tracks". The derivation of the steady state MOC integration method begins with the steady state multi-group form of the neutron transport equation:

$$\Omega \cdot \nabla \Psi_g(r, \Omega) + \Sigma_g^{tr}(r) \Psi_g(r, \Omega) = Q_g(r) \tag{2.1}$$

where g is the energy group index, r is the spatial position vector, Ω is the angular direction vector, $\Psi_g(r, \Omega)$ is the angular flux, $\Sigma_g^{tr}(r)$ is the transport cross-section, and $Q_g(r)$ is the source term. To obtain the characteristic form of Equation 2.1, we integrate the streaming operator, $\Omega \cdot \nabla \Psi_g(r, \Omega)$, over a constant angle track in the direction of the angular flux, Ω. The angular flux is integrated starting at position r and traveling a distance s along a track to a new position $r + s\Omega$:

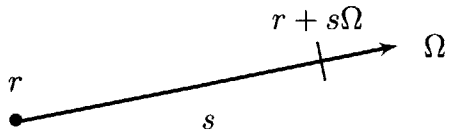

Figure 2-1. Angular flux traveling along a constant angle track with position r, distance traveled s, and direction Ω.

We first us the chain rule to separate out the components of the spatial derivative:

$$\frac{d}{ds} = \frac{dx}{ds}\frac{\partial}{\partial x} + \frac{dy}{ds}\frac{\partial}{\partial y} + \frac{dz}{ds}\frac{\partial}{\partial z} \tag{2.2}$$

Where dr can be expressed as:

$$dr = ds\mathbf{\Omega} = dx\mathbf{i} + dy\mathbf{j} + dz\mathbf{k} \tag{2.3}$$

The dot product of the change in angular flux direction as we move along the track and the unit vectors in each direction is then:

$$ds\Omega \cdot \mathbf{i} = dx \tag{2.4a}$$

$$ds\Omega \cdot \mathbf{j} = dy \tag{2.4b}$$

$$ds\Omega \cdot \mathbf{k} = dz \tag{2.4c}$$

Substituting these into Equation 2.2, we get:

$$\frac{d}{ds} = (\Omega \cdot \mathbf{i})\frac{\partial}{\partial x} + (\Omega \cdot \mathbf{j})\frac{\partial}{\partial y} + (\Omega \cdot \mathbf{k})\frac{\partial}{\partial z} = \Omega \cdot \nabla \tag{2.5}$$

Substituting Equation 2.5 into Equation 2.1 we arrive at the characteristic form of the transport equation:

$$\frac{d\Psi_g(s)}{ds} + \Sigma_g^{tr}(s)\Psi_g(s) = Q_g(s) \tag{2.6}$$

As mentioned before, we are solving the characteristic equation in 1D along a constant angle track. This gives us the angular flux in a particular direction and position in the medium. To obtain the scalar flux we lay down tracks in many different directions and integrate the angular flux in the complete 4π angular space. We represent the different tracks across the geometry by giving them a unique index k. The geometry itself is discretized into flat source regions, or *FSRs*, which homogenize material properties as well as the scalar flux and neutron source. The FSRs further subdivide each track into connected segments with FSR indices denoted r. Figure 2-2 gives an illustration of the track lay down (left), FSR discretization, and track subdivision into segments (right) for a pin cell geometry.

Using the indices for track index, FSR, and energy group, the characteristic equation becomes:

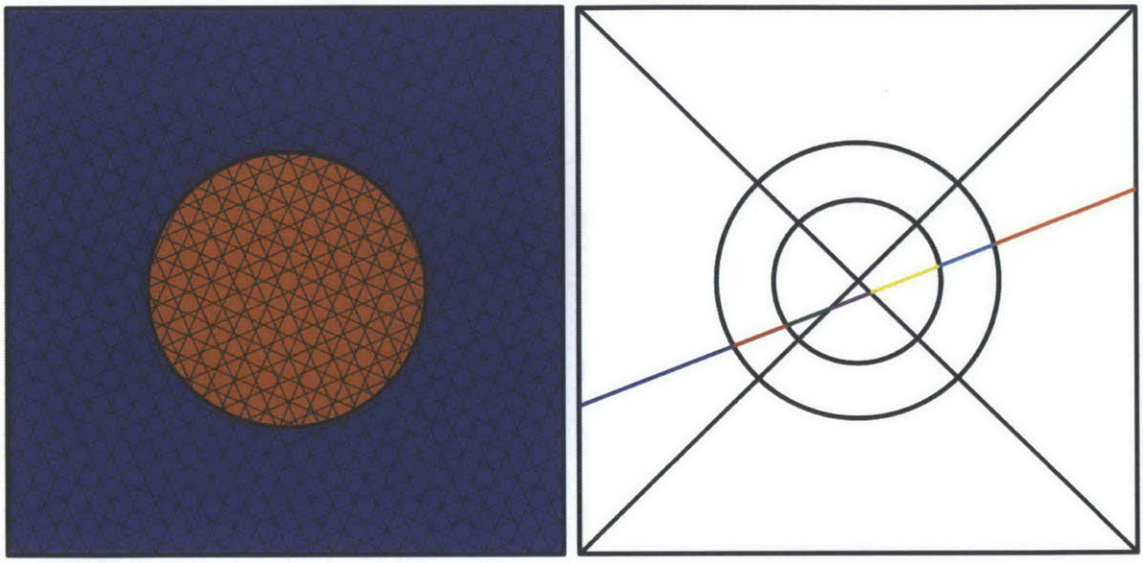

Figure 2-2. Track layout for pin cell geometry (left) and segments for a particular track where the fuel has been divided into two rings and the cell has been divided into four sectors. Each segment is denoted by a unique color corresponding to the FSR it lies in.

$$\frac{d\Psi_{k,r,g}}{ds} + \Sigma^{tr}_{r,g}\Psi_{k,r,g} = Q_{r,g} \tag{2.7}$$

The source term $Q_{r,g}$ is defined in terms of both isotropic fission and scattering from the area-averaged scalar flux, $\Phi_{r,g}$, in each FSR:

$$Q_{r,g} = \frac{1}{4\pi}\left(\frac{\chi_{r,g}}{k_{eff}}\sum_{g\prime=1}^{G}\nu\Sigma^{F}_{r,g\prime}\Phi_{r,g\prime} + \sum_{g\prime=1}^{G}\Sigma^{S}_{r,g\prime\to g}\Phi_{g\prime}\right) \tag{2.8}$$

where $\Sigma^{S}_{r,g'\to g}$ is the scattering cross-section for group g' to group g, $\Sigma^{F}_{r,g\prime}$ is the fission cross-section for group g', ν is the average number of neutrons produced from fission, $\chi_{r,g}$ is the fraction of neutrons produced in group g from fission and k_{eff} is the multiplication factor. Since we are solving the steady state problem, we do not differentiate between prompt and delayed neutrons in the fission source spectrum. In Equation 2.8 we have used the "step characteristic" approximation to model the source as spatially invariant within the FSR. Higher order spatial approximation schemes such as linear source approximation [19] have been shown to decrease both

memory requirements and run times and be compatible with CMFD acceleration. For the purposes of this study, the isotropic source approximation is expected to be sufficient. Equation 2.7 can be integrated for a segment across an FSR from its entry point at s' to exit point at s'' using an integrating factor:

$$\Psi_{k,g}(s'') = \Psi_{k,g}(s')e^{-\tau_{k,r,g}} + \frac{Q_{r,g}}{\Sigma^{tr}_{r,g}}(1 - e^{-\tau_{k,r,g}}) \tag{2.9}$$

with the optical path length defined as $\tau_{k,r,g} = \Sigma^{tr}_{r,g}(s'' - s')$. With minor algebraic rearrangement, the change in the angular flux along the characteristic k is given by the following:

$$\Delta\Psi_{k,g} = \Psi_{k,g}(s') - \Psi_{k,g}(s'') = \left(\Psi_{k,g}(s') - \frac{Q_{r,g}}{\Sigma^{tr}_{r,g}}\right)\left(1 - e^{-\tau_{k,r,g}}\right) \tag{2.10}$$

By defining $l_{k,r} = s'' - s'$, the average angular flux contribution to FSR r from track k is the following integral:

$$\overline{\Psi}_{k,r,g} = \frac{1}{l_{k,r}} \int_{s'}^{s''} \Psi_{k,r,g}(s)\, \mathrm{d}s \tag{2.11}$$

Upon evaluating the integral, the average flux can be reduced to the following algebraic expression:

$$\overline{\Psi}_{k,r,g} = \frac{1}{l_{k,r}} \left[\frac{\Psi_{k,g}(s')}{\Sigma^{tr}_{r,g}}(1 - e^{-\tau_{k,r,g}}) + \frac{l_{k,r}Q_{i,g}}{\Sigma^{tr}_{r,g}}\left(1 - \frac{(1 - e^{-\tau_{k,r,g}})}{\tau_{k,r,g}}\right)\right] \tag{2.12}$$

Equation 2.12 represents the average angular flux along one segment. The area-averaged scalar flux in FSR r with area A_r can be found by performing an area-weighted sum of the segment-wise average angular flux in each region and integrating the average angular flux over azimuthal and polar angles, m and p, using quadrature rules for each track segment in A_r:

$$\Phi_{r,g} = \frac{4\pi}{A_r} \sum_{k \in A_r} \sum_{p=1}^{P} \omega_{m(k)}\omega_p\omega_k l_{k,r} \overline{\Psi}_{k,r,g,p} \tag{2.13}$$

where $m(k)$ represents the azimuthal angle for track k, $\omega_{m(k)}$ and ω_p are the azimuthal and polar quadrature weights, and ω_k represents the *track width* for track k. In OpenMOC, the Yamamoto-Tabuchi quadrature set is used for the polar weights [3]. The azimuthal weight is defined as the fractional angular space owned by a particular angle:

$$\omega_{m(k)} = \frac{\Delta m(k)}{2\pi} \tag{2.14}$$

where $\Delta m(k)$ is the angular space represented by tracks with azimuthal angle $m(k)$. The final form of the scalar flux can be found by substituting the expression for the average angular flux from Equation 2.12 into Equation 2.13, and rearranging in terms of the change in angular flux along the characteristic from Equation 2.10:

$$\Phi_{r,g} = \frac{4\pi}{\Sigma_{r,g}^{tr}} \left[Q_{r,g} + \frac{1}{A_r} \sum_{k \in A_r} \sum_{p=1}^{P} \omega_{m(k)} \omega_p \omega_k \sin\theta_p \Delta\Psi_{k,r,g,p} \right] \tag{2.15}$$

where the polar angle, θ_p, is measured from the z-axis.

Once a track reaches a geometry boundary, a boundary condition must be applied. There are two natural boundary conditions, reflective and vacuum, that can easily be applied to the MOC method. For the reflective boundary condition, the angular flux for a particular track is given to a complementary track that is a reflection on the geometry boundary. This requires that tracks be laid down in a cyclic fashion such that every track has a complementary outgoing and incoming track as shown in Figure 2-3.

Reflective boundary conditions are often used to simplify a problem when the reactor is symmetric about either of the axes. Vacuum boundary conditions are used on the outer boundaries of a reactor. When a track intersects a vacuum boundary, the outgoing neutron leakage is tallied using Equation 2.16.

$$L \mathrel{+}= 2\pi \sum_{g=1}^{G} \sum_{p=1}^{P} \omega_{m(k)} \omega_p \omega_k \sin\theta_p \Psi_{k,g,p}^{\text{outgoing}} \quad \forall \quad \text{vacuum boundary crossings} \tag{2.16}$$

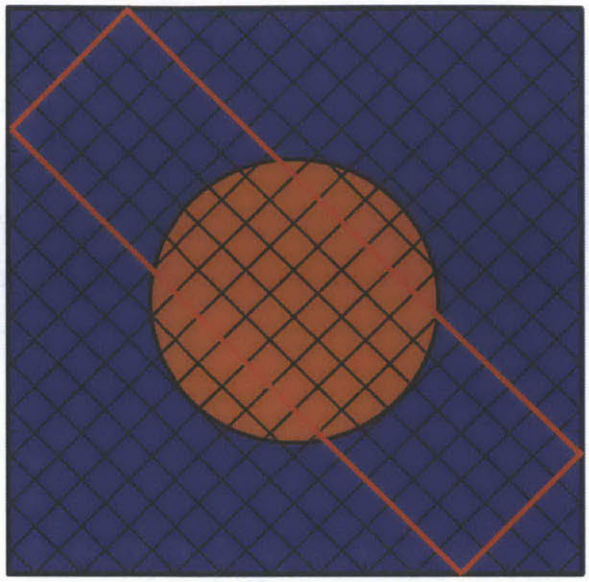

Figure 2-3. Track layout for pin cell geometry with one of the track cycles shown in red.

where L is the total neutron current leaking out of the geometry. Using the tallied leakage and FSR scalar flux, the neutron multiplication factor, k_{eff} can be computed using Equation 2.17.

$$k_{eff} = \frac{\sum_{r=1}^{R}\sum_{g=1}^{G} \nu\Sigma_{r,g}^{F}\Phi_{r,g}A_r}{L + \sum_{r=1}^{R}\sum_{g=1}^{G} \Sigma_{r,g}^{A}\Phi_{g}A_r} \tag{2.17}$$

where R is the number of FSRs in the geometry. In some problems such as the transport version of the LRA benchmark, a critical buckling term, B_z^2, is used to treat leakage in the axial direction. In this case, the expression for k_{eff} becomes:

$$k_{eff} = \frac{\sum_{r=1}^{R}\sum_{g=1}^{G} \nu\Sigma_{r,g}^{F}\Phi_{r,g}A_r}{L + \sum_{r=1}^{R}\sum_{g=1}^{G} \left(\Sigma_{r,g}^{A} + B_z^2 D_{r,g}\right)\Phi_{g}A_r} \tag{2.18}$$

where $D_{r,g}$ is the diffusion coefficient in FSR r and group g.

The goal of the MOC method is to determine the neutron flux within each FSR, which can then be used to compute reaction rates. The MOC method is implemented by performing an iterative two-step procedure. First, the source for each energy group in each FSR, $Q_{r,g}$, is initialized. Then the angular flux is propagated along each track and the contribution for each track segment and each energy group is tallied in the corresponding FSR scalar flux tally using Equation 2.15. Additionally, the leakage for each track that intersects a vacuum boundary is tallied in the total leakage tally, L. This procedure is called a "fixed source iteration" and is shown in Algorithm 1. A single fixed source iteration, denoted by the index n, involves five nested loops over azimuthal angles, tracks, segments in different FSRs, energy groups and polar angles. The sets of all azimuthal angles, tracks, track segments, FSRs, energy groups and polar angles are denoted by M, K, S, R, G and P, respectively. For notational simplicity, the subset of tracks for azimuthal angle m is denoted by $K(m)$, the subset of segments for track k is given by $S(k)$, and the FSR for segment s is represented as $R(s)$.

Algorithm 1 Fixed source iteration for OpenMOC

$\Phi_{r,g} \leftarrow 0 \quad \forall\, r, g \in \{R, G\}$ # Initialize FSR scalar fluxes to zero
while $Q_{r,g}\ \forall\, r$ not converged **do**
  for all $m \in M$ **do** # Loop over azimuthal angles
    for all $k \in K(m)$ **do** # Loop over tracks
      for all $s \in S(k)$ **do** # Loop over segments
        for all $g \in G$ **do** # Loop over energy groups
          for all $p \in P$ **do** # Loop over polar angles
            $r \leftarrow R(s)$ # Get FSR for this segment
            $\Delta\Psi_{k,r,g,p} \leftarrow \left(\Psi_{k,g,p} - \frac{Q_{r,g}}{\Sigma^{tr}_{r,g}}\right)\left(1 - e^{-\tau_{k,r,g,p}}\right)$
            $\Phi_{r,g} \leftarrow \Phi_{r,g} + \frac{4\pi}{A_r}\omega_{m(k)}\omega_p\omega_k \sin\theta_p l_{k,r}\Delta\Psi_{k,r,g,p}$
            $\Psi_{k,g,p} \leftarrow \Psi_{k,g,p} - \Delta\Psi_{k,g,p}$
          end for
        end for
      end for
    end for
    if B.C. are reflective **then** # Set incoming flux for outgoing track
      $\Psi_{k',g,p}(0) \leftarrow \Psi_{k,g,p}$ # Reflective B.C.'s
    else
      $\Psi_{k',g,p}(0) \leftarrow 0$ # Vacuum B.C.'s
      $L \leftarrow L + 2\pi\omega_{m(k)}\omega_p\omega_k \sin\theta_p \Psi_{k,g,p}$
    end if
  end for
  Update k_{eff} and $Q_{r,g}\ \forall\, r$ # Equation 2.17 and Algorithm 2
end while

Algorithm 2 FSR source update for OpenMOC

```
for all r ∈ R do                                  # Loop over FSRs
  for all g ∈ G do                                # Loop over energy groups
    Q_{r,g}^{(n+1)} ← (χ_{r,g} / 4πk_eff) νΣ_{r,g}^F Φ_{r,g}^{(n)}   # Initialize new total source with fission
    for all g' ∈ G do                             # Loop over energy groups
      Q_{r,g}^{(n+1)} ← Q_{r,g}^{(n+1)} + (1/4π) Σ_{r,g'→g}^S Φ_{r,g'}^{(n)}
    end for
  end for
end for
```

The spatial shape and energy distribution of the flux across FSRs is iteratively computed by fixed source iterations and source updates until some measure of convergence is achieved. In the version of OpenMOC used in this thesis, convergence is based on the relative root-mean-square-difference (RMSD) of the group-wise fission source in FSRs between successive fixed source iterations as represented in Equation 2.19.

$$\text{RMSD} = \sqrt{\frac{\sum_{r=1}^{R}\sum_{g=1}^{G}\left(\frac{\nu\Sigma_g^F(r)\phi_g^{(n+1)}(r) - \nu\Sigma_g^F(r)\phi_g^{(n)}(r)}{\nu\Sigma_g^F(r)\phi_g^{(n+1)}(r)}\right)^2}{R * G}} \tag{2.19}$$

2.2 Coarse Mesh Finite Difference Acceleration

While MOC offers many benefits including treatment of complex geometries and amenability to parallelization, it suffers from slow convergence which necessitates the use of acceleration methods. Numerous acceleration schemes have been proposed for MOC such as CMFD [13], coarse mesh rebalance (CMR) [2, 18], hybrid CMFD/macro-group MOC acceleration [10], and low order transport operator acceleration [17] with CMFD being the most widely adopted due to its simplicity and acceleration performance. OpenMOC uses the CMFD nonlinear diffusion acceleration (NDA) scheme to reduce the number of iterations required for convergence. Acceleration schemes, such as NDA, are necessary when solving full-core problems which require thousands of power iterations in LWR problems that tend to have high dominance ratios. CMFD was first proposed by Smith [13] and has been widely used in accelerating neutron diffusion and transport problems for many years [9, 24]. In

particular, it has been shown that CMFD acceleration gives >100× speedups on large LWR problems [14].

CMFD acceleration functions by using the solution of a coarse mesh diffusion problem to accelerate the convergence of a fine mesh transport problem. It is implemented by overlaying a 2D rectilinear mesh over an FSR mesh. Figure 2-4 gives an illustration of the FSR mesh layout and coarse mesh layout used for solving a 17 × 17 PWR assembly problem.

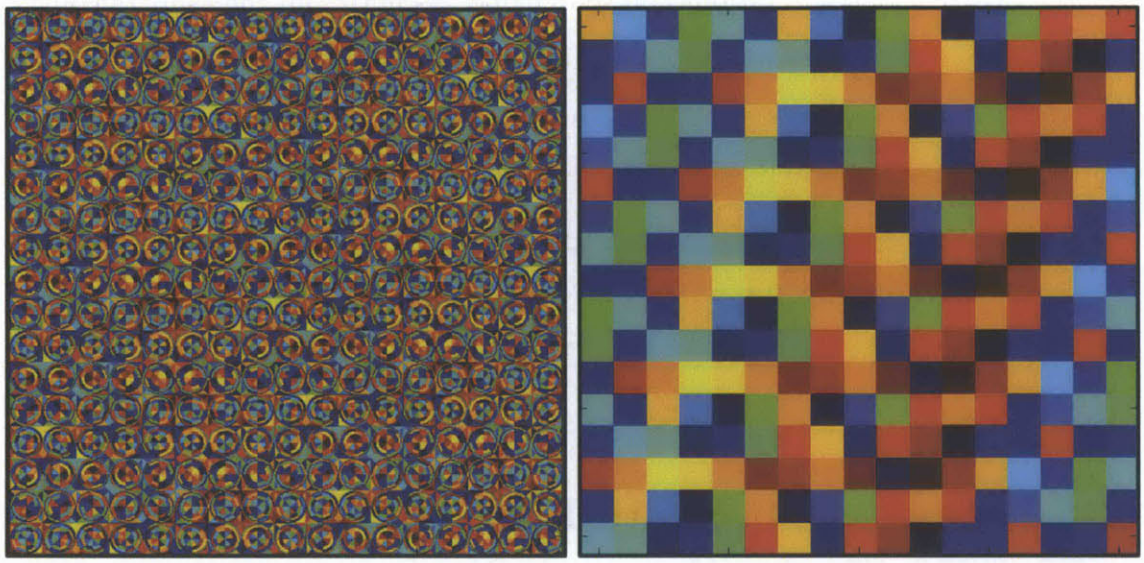

Figure 2-4. Flat source region (left) and CMFD (right) mesh layout for a 17 × 17 PWR assembly where each colored cell denotes a different region.

To derive the CMFD equations, we begin with the 2D, steady state multi-group neutron diffusion equation:

$$-\nabla \cdot D_{\mathbf{g}}(x,y)\nabla\phi_{\mathbf{g}}(x,y) + \Sigma^{A}_{\mathbf{g}}(x,y)\phi_{\mathbf{g}}(x,y) + \sum_{\substack{\mathbf{g}\prime=1 \\ \mathbf{g}\prime\neq\mathbf{g}}}^{\mathbf{G}} \Sigma^{S}_{\mathbf{g}\rightarrow\mathbf{g}\prime}(x,y)\phi_{\mathbf{g}}(x,y) \tag{2.20}$$

$$= \frac{\chi_{\mathbf{g}}(x,y)}{k_{eff}} \sum_{\mathbf{g}\prime=1}^{\mathbf{G}} \nu\Sigma^{F}_{\mathbf{g}\prime}(x,y)\phi_{\mathbf{g}\prime}(x,y) + \sum_{\substack{\mathbf{g}\prime=1 \\ \mathbf{g}\prime\neq\mathbf{g}}}^{\mathbf{G}} \Sigma^{S}_{\mathbf{g}\prime\rightarrow\mathbf{g}}(x,y)\phi_{\mathbf{g}\prime}(x,y)$$

where terms in Equation 2.20 are defined in Table 2.1

Table 2.1. Descriptions of the diffusion equation terms

Variable	Description	Variable	Description
D	Diffusion coefficient	ϕ	Scalar flux on coarse mesh
Σ^A	Coarse mesh absorption XS	χ	Fission spectrum
Σ^S	Coarse mesh scattering XS	k_{eff}	Neutron multiplication factor
Σ^F	Coarse mesh fission XS	ν	Neutrons per fission
$\mathbf{g}, \mathbf{g}'$	Energy group index	x, y	Position variable
$\mathbf{G}$	Coarse mesh energy groups		

2.2.1 Generation of cross sections for the coarse mesh

The cross sections for the CMFD diffusion equation are generated by energy-condensation and area-averaging of the cross sections from the fine mesh as shown in Equation 2.21. The energy group structure of the CMFD diffusion equation does not have to be the same as the energy group structure used in MOC. For example, Figure 2-5 shows the various ways to formulate the CMFD group structure to accelerate a three group MOC problem.

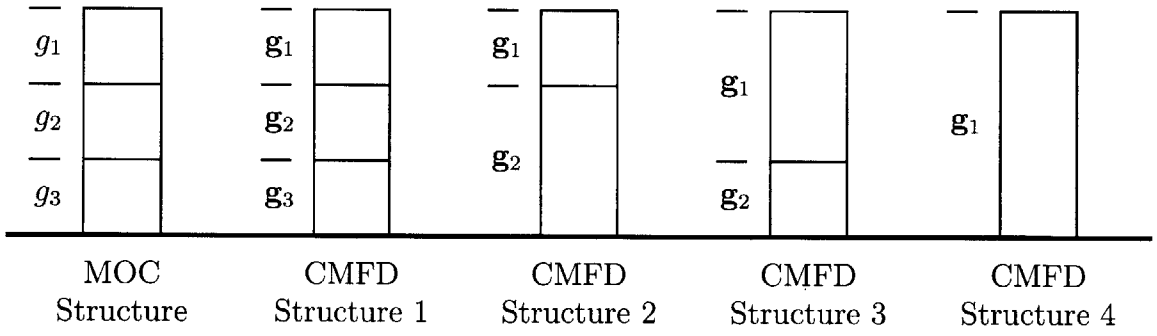

Figure 2-5. Illustration of possible CMFD energy group structures for a three group MOC calculation.

where energy groups in MOC are denoted with g and energy groups in CMFD are denoted with $\mathbf{g}$. The generalized equations for computing cross sections on the coarse mesh are then described by Equation 2.21.

$$\Sigma_{\mathbf{g}}^{A,i,j} = \frac{\sum_{g\in\mathbf{g}}\sum_{r\in(i,j)} \Sigma_{r,g}^{A}\Phi_{r,g}A_r}{\sum_{g\in\mathbf{g}}\sum_{r\in(i,j)} \Phi_{r,g}A_r} \tag{2.21a}$$

$$\Sigma_{\mathbf{g}}^{F,i,j} = \frac{\sum_{g\in\mathbf{g}}\sum_{r\in(i,j)} \Sigma_{r,g}^{F}\Phi_{r,g}A_r}{\sum_{g\in\mathbf{g}}\sum_{r\in(i,j)} \Phi_{r,g}A_r} \tag{2.21b}$$

$$\nu\Sigma_{\mathbf{g}}^{F,i,j} = \frac{\sum_{g\in\mathbf{g}}\sum_{r\in(i,j)} \nu\Sigma_{r,g}^{F}\Phi_{r,g}A_r}{\sum_{g\in\mathbf{g}}\sum_{r\in(i,j)} \Phi_{r,g}A_r} \tag{2.21c}$$

$$\Sigma_{\mathbf{g}\to\mathbf{g}\prime}^{S,i,j} = \frac{\sum_{g\in\mathbf{g}}\sum_{g\prime\in\mathbf{g}\prime}\sum_{r\in(i,j)} \Sigma_{r,g\to g\prime}^{S}\Phi_{r,g}A_r}{\sum_{g\in\mathbf{g}}\sum_{r\in(i,j)} \Phi_{r,g}A_r} \tag{2.21d}$$

$$D_{\mathbf{g}}^{i,j} = \frac{\sum_{g\in\mathbf{g}}\sum_{r\in(i,j)} \frac{1}{3\Sigma_{r,g}^{tr}}\Phi_{r,g}A_r}{\sum_{g\in\mathbf{g}}\sum_{r\in(i,j)} \Phi_{r,g}A_r} \tag{2.21e}$$

$$\chi_{\mathbf{g}}^{i,j} = \frac{\sum_{g\in\mathbf{g}}\sum_{r\in(i,j)}\sum_{g\prime=1}^{G} \chi_{r,g}\nu\Sigma_{r,g\prime}^{F}\Phi_{r,g\prime}A_r}{\sum_{r\in(i,j)}\sum_{g\prime=1}^{G}\sum_{g\prime\prime=1}^{G} \chi_{r,g\prime\prime}\nu\Sigma_{r,g\prime}^{F}\Phi_{r,g\prime}A_r} \tag{2.21f}$$

$$\phi_{\mathbf{g}}^{i,j} = \frac{\sum_{g\in\mathbf{g}}\sum_{r\in(i,j)} \Phi_{r,g}A_r}{\sum_{r\in(i,j)} A_r} \tag{2.21g}$$

where i and j denote the mesh cell indices in the x and y directions, respectively.

2.2.2 Finite difference approximation of flux between neighboring cells

The diffusion operator in Equation 2.20 can be expanded to yield:

$$-\frac{\partial}{\partial x}D_{\mathbf{g}}(x,y)\frac{\partial}{\partial x}\phi_{\mathbf{g}}(x,y)-\frac{\partial}{\partial y}D_{\mathbf{g}}(x,y)\frac{\partial}{\partial y}\phi_{\mathbf{g}}(x,y)+\Sigma^{R}_{\mathbf{g}}(x,y)\phi_{\mathbf{g}}(x,y) \tag{2.22}$$
$$=\frac{\chi_{\mathbf{g}}(x,y)}{k_{eff}}\sum_{\mathbf{g}\prime=1}^{\mathbf{G}}\nu\Sigma^{F}_{\mathbf{g}\prime}(x,y)\phi_{\mathbf{g}\prime}(x,y)+\sum_{\substack{\mathbf{g}\prime=1\\ \mathbf{g}\prime\neq\mathbf{g}}}^{\mathbf{G}}\Sigma^{S}_{\mathbf{g}\prime\rightarrow\mathbf{g}}(x,y)\phi_{\mathbf{g}\prime}(x,y)$$

where the removal cross section, $\Sigma^{R}_{\mathbf{g}}$ is defined as:

$$\Sigma^{R}_{\mathbf{g}}\equiv\Sigma^{A}_{\mathbf{g}}+\sum_{\substack{\mathbf{g}\prime=1\\ \mathbf{g}\prime\neq\mathbf{g}}}^{\mathbf{G}}\Sigma^{S}_{\mathbf{g}\rightarrow\mathbf{g}\prime} \tag{2.23}$$

In cases where axial buckling is specified the removal cross section takes the following form:

$$\Sigma^{R}_{\mathbf{g}}\equiv D_{\mathbf{g}}B_{z}^{2}+\Sigma^{A}_{\mathbf{g}}+\sum_{\substack{\mathbf{g}\prime=1\\ \mathbf{g}\prime\neq\mathbf{g}}}^{\mathbf{G}}\Sigma^{S}_{\mathbf{g}\rightarrow\mathbf{g}\prime} \tag{2.24}$$

We can integrate Equation 2.22 over a mesh cell to get the neutron balance in that cell. Performing the integration on cell (i,j) and simplifying we get:

$$-\int_{i-\frac{1}{2}}^{i+\frac{1}{2}}\int_{j-\frac{1}{2}}^{j+\frac{1}{2}}\frac{\partial}{\partial x}D^{i,j}_{\mathbf{g}}\frac{\partial}{\partial x}\phi^{i,j}_{\mathbf{g}}dxdy-\int_{i-\frac{1}{2}}^{i+\frac{1}{2}}\int_{j-\frac{1}{2}}^{j+\frac{1}{2}}\frac{\partial}{\partial y}D^{i,j}_{\mathbf{g}}\frac{\partial}{\partial y}\phi^{i,j}_{\mathbf{g}}dxdy \tag{2.25}$$
$$+\Delta x^{i,j}\Delta y^{i,j}\Sigma^{R,i,j}_{\mathbf{g}}\phi^{i,j}_{\mathbf{g}}=\Delta x^{i,j}\Delta y^{i,j}\frac{\chi^{i,j}_{\mathbf{g}}}{k_{eff}}\sum_{\mathbf{g}\prime=1}^{\mathbf{G}}\nu\Sigma^{F,i,j}_{\mathbf{g}\prime}\phi^{i,j}_{\mathbf{g}\prime}+\Delta x^{i,j}\Delta y^{i,j}\sum_{\substack{\mathbf{g}\prime=1\\ \mathbf{g}\prime\neq\mathbf{g}}}^{\mathbf{G}}\Sigma^{S,i,j}_{\mathbf{g}\prime\rightarrow\mathbf{g}}\phi^{i,j}_{\mathbf{g}\prime}$$

where $\Delta x^{i,j}$ and $\Delta y^{i,j}$ denote the width and height of cell (i,j), respectively. Using the Divergence Theorem, we can reduce the volume integrals of the streaming terms to surface integrals over the bounding surfaces of a cell. The surface integrals will then represent currents across the surfaces of a cell. Using streaming in the x-direction as an example, we can rewrite the volume integral of the streaming term in the x-direction as:

$$-\int_{i-\frac{1}{2}}^{i+\frac{1}{2}}\int_{j-\frac{1}{2}}^{j+\frac{1}{2}}\frac{\partial}{\partial x}D_{\mathbf{g}}^{i,j}\frac{\partial}{\partial x}\phi_{\mathbf{g}}^{i,j}dxdy = -\Delta y^{i,j}(J_{\mathbf{g}}^{i+\frac{1}{2},j} - J_{\mathbf{g}}^{i-\frac{1}{2},j}) \tag{2.26}$$

where $J_{\mathbf{g}}^{i+\frac{1}{2},j}$ and $J_{\mathbf{g}}^{i-\frac{1}{2},j}$ represent the surface-averaged net current across the right surface and across the left surface, respectively. An analogous equation can be written for streaming in the y-direction.

As an example, we will now solve for the currents on the right surface of a cell (i,j) with neighboring cell $(i+1,j)$ as illustrated in Figure 2-6.

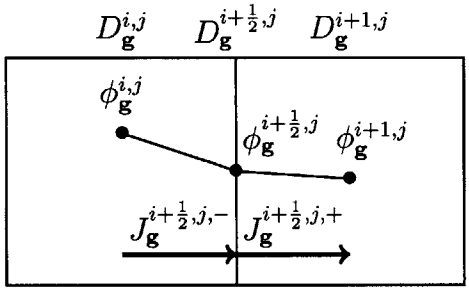

Figure 2-6. Illustration of terms required to solve for streaming of neutrons between cells (i,j) and $(i+1,j)$ in CMFD diffusion.

To solve for the current at cell boundaries, we can perform finite difference approximations of the flux at a surface using the flux in the adjacent cells:

$$\begin{aligned} J_{\mathbf{g}}^{i+\frac{1}{2},j,+} &= -D_{\mathbf{g}}^{i+1,j}\frac{d\phi_{\mathbf{g}}^{i+\frac{1}{2},j}}{dx}\bigg|_{x^{i+\frac{1}{2},j,+}} = -D_{\mathbf{g}}^{i+1,j}\frac{\phi_{\mathbf{g}}^{i+1,j} - \phi_{\mathbf{g}}^{i+\frac{1}{2},j}}{\frac{\Delta x^{i+1,j}}{2}} \\ J_{\mathbf{g}}^{i+\frac{1}{2},j,-} &= -D_{\mathbf{g}}^{i,j}\frac{d\phi_{\mathbf{g}}^{i+\frac{1}{2},j}}{dx}\bigg|_{x^{i+\frac{1}{2},j,-}} = -D_{\mathbf{g}}^{i,j}\frac{\phi_{\mathbf{g}}^{i+\frac{1}{2},j} - \phi_{\mathbf{g}}^{i,j}}{\frac{\Delta x^{i,j}}{2}} \end{aligned} \tag{2.27}$$

where $J_{\mathbf{g}}^{i+\frac{1}{2},j,+}$ and $J_{\mathbf{g}}^{i+\frac{1}{2},j,-}$ are the surface-averaged net currents as approximated using a backward difference approximation from the cell at (i,j) and a forward difference approximation from the cell at $(i+1,j)$, respectively. Equating these representations of the current we can solve for the flux at the surface:

$$\phi_{\mathbf{g}}^{i+\frac{1}{2},j} = \frac{D_{\mathbf{g}}^{i,j}\phi_{\mathbf{g}}^{i,j}\Delta x^{i+1,j} + D_{\mathbf{g}}^{i+1,j}\phi_{\mathbf{g}}^{i+1,j}\Delta x^{i,j}}{D_{\mathbf{g}}^{i,j}\Delta x^{i+1,j} + D_{\mathbf{g}}^{i+1,j}\Delta x^{i,j}} \tag{2.28}$$

The net current across the surface at $(i+\frac{1}{2},j)$ is then:

$$J_{\mathbf{g}}^{i+\frac{1}{2},j} = -\hat{D}_{\mathbf{g}}^{i+\frac{1}{2},j}(\phi_{\mathbf{g}}^{i+1,j} - \phi_{\mathbf{g}}^{i,j}) \tag{2.29}$$

where:

$$\hat{D}_{\mathbf{g}}^{i+\frac{1}{2},j} = \frac{2D_{\mathbf{g}}^{i,j}D_{\mathbf{g}}^{i+1,j}}{D_{\mathbf{g}}^{i,j}\Delta x^{i+1,j} + D_{\mathbf{g}}^{i+1,j}\Delta x^{i,j}} \tag{2.30}$$

The neutron balance equation in a cell then becomes:

$$\begin{aligned} -\Delta y^{i,j}(J_{\mathbf{g}}^{i+\frac{1}{2},j} - J_{\mathbf{g}}^{i-\frac{1}{2},j}) - \Delta x^{i,j}(J_{\mathbf{g}}^{i,j+\frac{1}{2}} - J_{\mathbf{g}}^{i,j-\frac{1}{2}}) + \Delta x^{i,j}\Delta y^{i,j}\Sigma_{\mathbf{g}}^{R,i,j}\phi_{\mathbf{g}}^{i,j} \\ = \Delta x^{i,j}\Delta y^{i,j}\frac{\chi_{\mathbf{g}}^{i,j}}{k_{eff}}\sum_{\mathbf{g}'=1}^{\mathbf{G}}\nu\Sigma_{\mathbf{g}'}^{F,i,j}\phi_{\mathbf{g}'}^{i,j} + \Delta x^{i,j}\Delta y^{i,j}\sum_{\substack{\mathbf{g}'=1 \\ \mathbf{g}'\neq\mathbf{g}}}^{\mathbf{G}}\Sigma_{\mathbf{g}'\rightarrow\mathbf{g}}^{S,i,j}\phi_{\mathbf{g}'}^{i,j} \end{aligned} \tag{2.31}$$

Note that Equation 2.29 is the algebraic net current based on the finite difference approximation being applied across the surface of two neighboring cells and not the actual net current in the MOC problem. The actual current from the MOC problem is computed by accumulating the current contribution from every segment that crosses a surface as will be shown in the subsection 2.2.3.

2.2.3 Introduction of nonlinear diffusion correction factors

In order to conserve neutron balance between the CMFD and MOC problems, the net currents across the coarse mesh cell surfaces must be equal. The surface diffusion coefficient expression in Equation 2.30 results in a neutron current close to the actual current produced via the MOC solve, but there is no guarantee the currents will be equal. To compute the currents from MOC, the net currents are tallied during a fixed

source iteration by summing the current contributions from each track that intersects a surface. For example, Equation 2.32 represents the net current tally expression for surface $(i+\frac{1}{2}, j)$.

$$\tilde{J}_{\mathbf{g}}^{i+\frac{1}{2},j} = \sum_{k \cap (i+\frac{1}{2},j)} \sum_{g \in \mathbf{g}} 2\pi \omega_{m(k)} \tilde{\omega}_k \omega_p \sin\theta_p \Psi_{k,g,p} \cdot \hat{n} \tag{2.32}$$

where $\hat{n}$ is the unit surface normal and $\tilde{\omega}_k$ is the length of surface crossed by the track, as illustrated in Figure 2-7. The first sum in Equation 2.32 is over all tracks that cross the surface between mesh cells (i, j) and $(i+1, j)$.

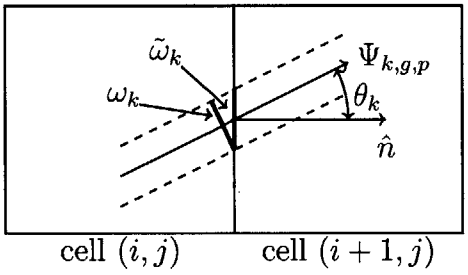

Figure 2-7. Angular flux from track k with energy group g and polar angle p crossing surface $(i+\frac{1}{2}, j)$.

The length of surface crossed by the track is defined as:

$$\tilde{\omega}_k = \frac{\omega_k}{\cos\theta_k} \tag{2.33}$$

Inserting Equation 2.33 into Equation 2.32 gives us:

$$\tilde{J}_{\mathbf{g}}^{i+\frac{1}{2},j} = \sum_{k \cap (i+\frac{1}{2},j)} \sum_{g \in \mathbf{g}} 2\pi \omega_{m(k)} \frac{\omega_k}{\cos\theta_k} \omega_p \sin\theta_p \Psi_{k,g,p} \cdot \hat{n} \tag{2.34}$$

When we apply the scalar product between the track azimuthal angle with the unit surface normal of the track, we get $\cos\theta_k$, which reduces Equation 2.34 to:

$$\tilde{J}_{\mathbf{g}}^{i+\frac{1}{2},j} = \sum_{k \cap (i+\frac{1}{2},j)} \sum_{g \in \mathbf{g}} 2\pi \omega_{m(k)} \omega_k \omega_p \sin\theta_p \Psi_{k,g,p} \tag{2.35}$$

In order for the tallied net currents to equal the net current expression, a nonlinear

diffusion coefficient term is added to Equation 2.29:

$$\frac{\tilde{J}_{\mathbf{g}}^{i+\frac{1}{2},j}}{\Delta y^{i,j}} = -\hat{D}_{\mathbf{g}}^{i+\frac{1}{2},j}(\phi_{\mathbf{g}}^{i+1,j} - \phi_{\mathbf{g}}^{i,j}) - \tilde{D}_{\mathbf{g}}^{i+\frac{1}{2},j}(\phi_{\mathbf{g}}^{i+1,j} + \phi_{\mathbf{g}}^{i,j}) \tag{2.36}$$

where $\tilde{D}$ is the nonlinear diffusion coefficient correction factor. Note that the current expression on the right hand side of Equation 2.36 computes the *surface-averaged* net current whereas the current tallied from MOC has not been averaged over the surface; therefore, in Equation 2.36 the net current tallied from MOC has been divided by the length of the surface that is being crossed. $\tilde{D}$ is computed to make Equation 2.36 valid for the tallied net surface current in MOC for the most recent fixed source iteration:

$$\tilde{D}_{\mathbf{g}}^{i+\frac{1}{2},j} = \frac{-\hat{D}_{\mathbf{g}}^{i+\frac{1}{2},j}(\phi_{\mathbf{g}}^{i+1,j} - \phi_{\mathbf{g}}^{i,j}) - \frac{\tilde{J}_{\mathbf{g}}^{i+\frac{1}{2},j}}{\Delta y^{i,j}}}{(\phi_{\mathbf{g}}^{i+1,j} + \phi_{\mathbf{g}}^{i,j})} \tag{2.37}$$

2.2.4 Treatment of optically thick regions

As shown in Figure 2-4 the CMFD mesh is often applied at the pin cell level with cells on the order of 1-2 cm. By conserving reaction and leakage rates within cells, CMFD guarantees preservation of area-averaged scalar fluxes and net surface currents from the MOC fixed source iteration if the CMFD equations can be converged. However, when the fine mesh cell size becomes significantly larger than the neutron mean free path in that cell, the step characteristics no longer preserve the linear infinite medium solution to the transport equation [6]. While the nonlinear diffusion correction term in CMFD is guaranteed to preserve reaction rates and surface net currents for any choice of diffusion coefficient, convergence (and convergence rate) of the nonlinear iteration acceleration of CMFD is affected by the choice of diffusion coefficient. All flat source methods, when applied for thick optical meshes, artificially distribute neutrons in space. This is the reason that Larsen's effective diffusion coefficient [6] is useful in assuring that the CMFD acceleration equations have a diffusion coefficient (on the flux gradient term) that is consistent, not with the physical transport problem, but with the transport problem that is being accelerated by the CMFD equations. Larsen's

effective diffusion coefficient is precisely this term in the one-dimensional limit. The effective diffusion coefficient in the x-direction for cell (i,j) can be expressed as:

$$D_{\mathbf{g}}^{i,j,eff,x} = D_{\mathbf{g}}^{i,j}\left(1+\frac{\Delta x^{i,j}\rho_{\mathbf{g}}^{i,j,x}}{2D_{\mathbf{g}}^{i,j}}\right) \tag{2.38a}$$

$$\rho_{\mathbf{g}}^{i,j,x} = \frac{\sum_{p=1}^{P}\cos(\theta_p)\omega_p\alpha_{\mathbf{g},p}^{i,j,x}}{\sum_{p=1}^{P}\omega_p} \tag{2.38b}$$

$$\alpha_{\mathbf{g},p}^{i,j,x} = \left(\frac{1+exp[-\gamma_{\mathbf{g},p}^{i,j,x}]}{1-exp[-\gamma_{\mathbf{g},p}^{i,j,x}]}\right)-\frac{2}{\gamma_{\mathbf{g},p}^{i,j,x}} \tag{2.38c}$$

$$\gamma_{\mathbf{g},p}^{i,j,x} = \frac{\Delta x^{i,j}}{3D_{\mathbf{g}}^{i,j}\cos(\theta_p)} \tag{2.38d}$$

Note that the effective diffusion coefficient depends on the width of the cell and is therefore directional in a 2D mesh. Equation 2.38 can also be used to compute the effective diffusion coefficient in the y-direction, which will differ from the effective diffusion coefficient in the x-direction if the cell is not square. As the size of the cell approaches zero and the optical thickness of the cell approaches the optically thin limit, the effective diffusion coefficient will approach the material diffusion coefficient. For simplicity, we continue to use the surface diffusion coefficient terms in the rest of this thesis without the "eff" superscript.

2.2.5 Treatment of coarse mesh cell corner crossings

In the CMFD formalism introduced in the previous sections, we only treat transport to adjacent cells. However, MOC produces tracks that directly and indirectly intersect mesh cell corners. A direct crossing is defined as a crossing where the centerline of a track directly intersects a mesh cell corner. An indirect crossing is defined as a crossing where the track *sweeps* through a corner but the track centerline does not directly cross through it. Illustrations of these two crossing types are shown in Figure 2-8.

There are three main approximations to treat corner crossings:

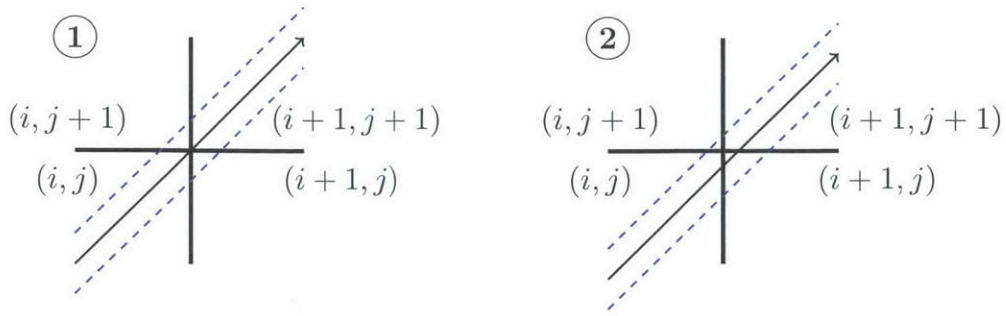

Figure 2-8. Illustration of direct (**1**) and indirect (**2**) track corner crossings.

1. Consistently tallying the current from direct corner crossings to only one of the surfaces. This approximation effectively moves the tracks that directly cross through a corner far enough to one side such that the entire segment crosses a single surface. In order to maintain neutron balance, tracks must be consistently moved to the same side for tracking forward and backwards along a track. Indirect crossings are ignored.

2. Split the current from direct corner crossings to each of the neighboring surfaces. This approximation effectively splits the track into two half-weighted tracks and moves them to either side of the corner such that each new half-weighted track only sweeps across one surface. Indirect crossings are ignored. This is essentially a special case of approximation 3 where only direct crossings are treated.

3. Split the current from direct and indirect corner crossings to each of the neighboring surfaces and weight the current contribution to each surface based on the length of that surface swept through by the track.

Note that tracks are not physically moved in any of these cases; rather, we make the assumption that they are moved when we tally the surface currents. In OpenMOC we have implemented approximation 2 where only direct corner crossings are treated; indirect crossings are tallied only on the surface that is directly crossed by a track. While applying approximation 3 would be more accurate, this would incur additional storage requirements as each segment that crosses a surface needs to know which surface(s) it crosses *and* a weight for splitting the current contribution to each surface

crossed. Illustrations of the approximations applied to direct and indirect surface crossings are illustrated in Figure 2-9.

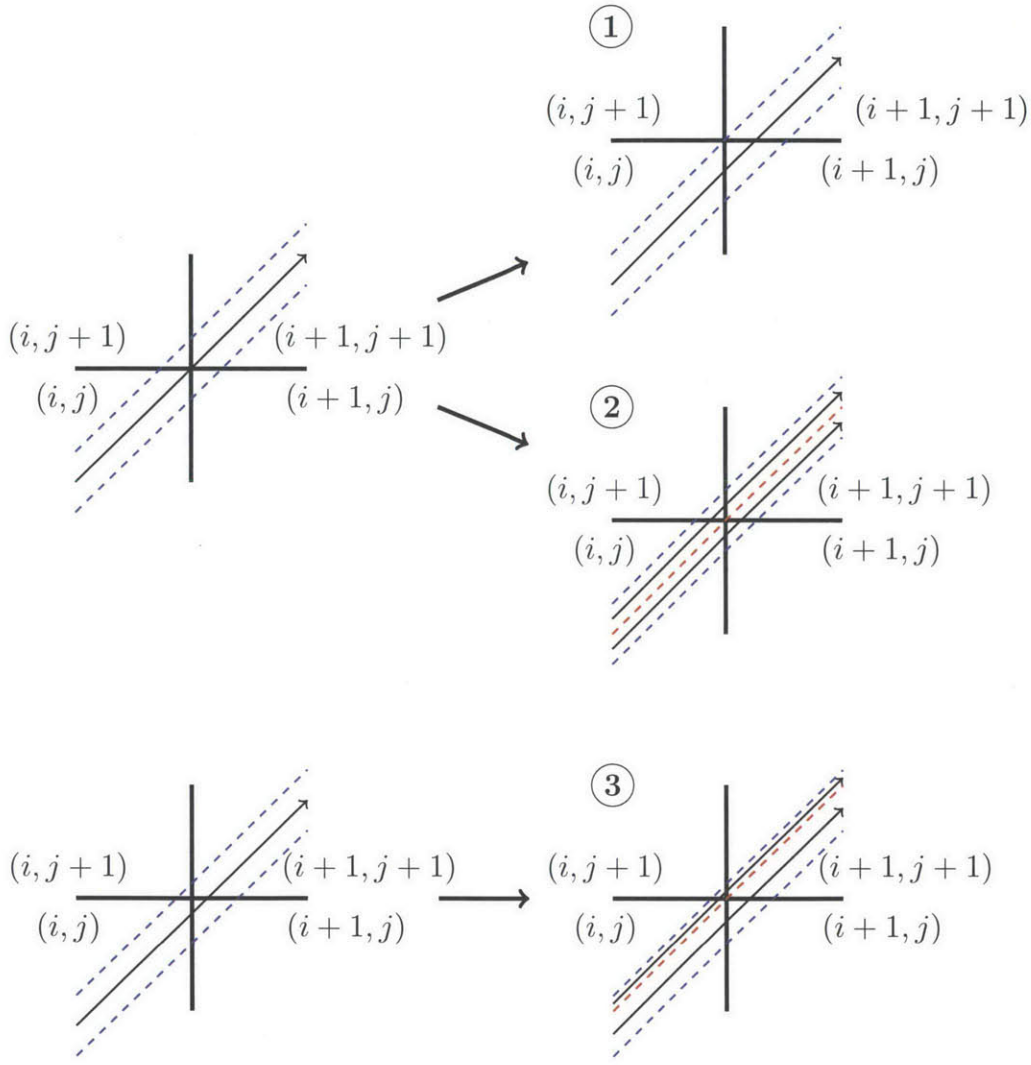

Figure 2-9. Illustration of approximations applied to direct (above) and indirect (below) corner crossings. The approximation applied is denoted by the number in the circle. The blue dashed lines bound the track sweeping area and the red dashed lines separate partially weighted tracks.

The tallies for a track that crosses from cell (i, j) to $(i+1, j+1)$ with approximation 1 include a tally on the surface between cell (i, j) and cell $(i+1, j)$ *and* on the surface between cells $(i+1, j)$ and cell $(i+1, j+1)$. The current must be tallied on the second surface in order to preserve neutron balance and ensure that neutrons traveling on the track get transferred from cell (i, j) to cell $(i+1, j+1)$. It is also important that

the track be assigned to the same surface for both forward and reverse tracking.

The tallies with approximation 2 are simply half-weighted tallies for tracks that pass on either side of the corner:

$$\tilde{J}_{\mathbf{g}}^{i+\frac{1}{2},j} \mathrel{+}= \frac{1}{2} \sum_{k \cap (i+\frac{1}{2},j+\frac{1}{2})} \sum_{g \in \mathbf{g}} \epsilon_{k,g,p} \tag{2.39a}$$

$$\tilde{J}_{\mathbf{g}}^{i,j+\frac{1}{2}} \mathrel{+}= \frac{1}{2} \sum_{k \cap (i+\frac{1}{2},j+\frac{1}{2})} \sum_{g \in \mathbf{g}} \epsilon_{k,g,p} \tag{2.39b}$$

$$\tilde{J}_{\mathbf{g}}^{i+1,j+\frac{1}{2}} \mathrel{+}= \frac{1}{2} \sum_{k \cap (i+\frac{1}{2},j+\frac{1}{2})} \sum_{g \in \mathbf{g}} \epsilon_{k,g,p} \tag{2.39c}$$

$$\tilde{J}_{\mathbf{g}}^{i+\frac{1}{2},j+1} \mathrel{+}= \frac{1}{2} \sum_{k \cap (i+\frac{1}{2},j+\frac{1}{2})} \sum_{g \in \mathbf{g}} \epsilon_{k,g,p} \tag{2.39d}$$

where:

$$\epsilon_{k,g,p} = 2\pi \omega_{m(k)} \omega_k \omega_p \sin\theta_p \Psi_{k,g,p} \tag{2.40}$$

The first summation in Equation 2.39 is over the tracks that directly cross through corner $(i+\frac{1}{2}, j+\frac{1}{2})$. Like the tallies in approximation 1, the tallies used in approximation 2 include tallies for the surface of the adjacent cell to the diagonal cell. The tallies with approximation 3 are slightly more complicated as they include the position at which the track crosses the surface and the position of the corner. Let's assume that a track crosses the surface between cells (i,j) and cell $(i+1,j)$ at point (x_k, y_k) and the corner of interest is at point $(x_{i+\frac{1}{2}}, y_{j+\frac{1}{2}})$ as shown in Figure 2-10.

The tallies for tracks that directly or indirectly intersect the corner illustrated in Figure 2-10 and described in Equation 2.41.

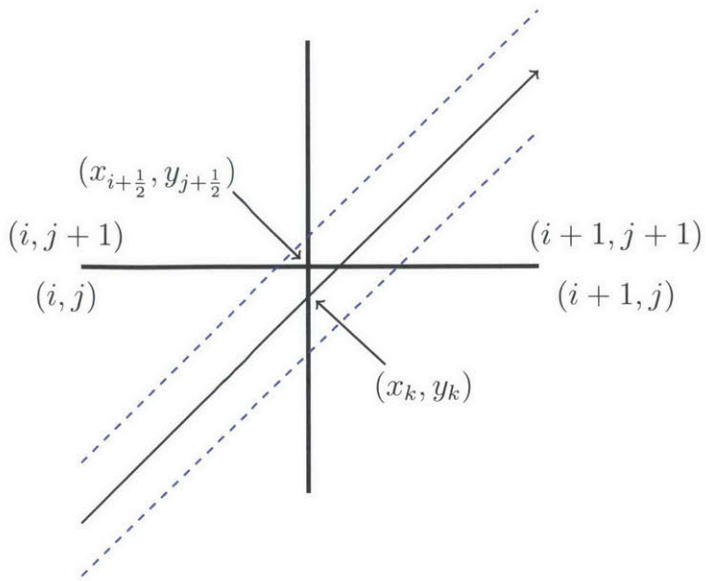

Figure 2-10. Illustration of an indirect corner crossing with labeled surface intersections and corner point.

$$\tilde{J}_{\mathbf{g}}^{i+\frac{1}{2},j} \mathrel{+}= \sum_{k\cap(i+\frac{1}{2},j+\frac{1}{2})} \sum_{g\in\mathbf{g}} \left(0.5 - \frac{\sqrt{(x_k - x_{i+\frac{1}{2}})^2}}{\tilde{\omega_k}} + \frac{\sqrt{(y_k - y_{j+\frac{1}{2}})^2}}{\tilde{\omega_k}}\right) \epsilon_{k,g,p} \quad (2.41a)$$

$$\tilde{J}_{\mathbf{g}}^{i,j+\frac{1}{2}} \mathrel{+}= \sum_{k\cap(i+\frac{1}{2},j+\frac{1}{2})} \sum_{g\in\mathbf{g}} \left(0.5 + \frac{\sqrt{(x_k - x_{i+\frac{1}{2}})^2}}{\tilde{\omega_k}} - \frac{\sqrt{(y_k - y_{j+\frac{1}{2}})^2}}{\tilde{\omega_k}}\right) \epsilon_{k,g,p} \quad (2.41b)$$

$$\tilde{J}_{\mathbf{g}}^{i+1,j+\frac{1}{2}} \mathrel{+}= \sum_{k\cap(i+\frac{1}{2},j+\frac{1}{2})} \sum_{g\in\mathbf{g}} \left(0.5 - \frac{\sqrt{(x_k - x_{i+\frac{1}{2}})^2}}{\tilde{\omega_k}} + \frac{\sqrt{(y_k - y_{j+\frac{1}{2}})^2}}{\tilde{\omega_k}}\right) \epsilon_{k,g,p} \quad (2.41c)$$

$$\tilde{J}_{\mathbf{g}}^{i+\frac{1}{2},j+1} \mathrel{+}= \sum_{k\cap(i+\frac{1}{2},j+\frac{1}{2})} \sum_{g\in\mathbf{g}} \left(0.5 + \frac{\sqrt{(x_k - x_{i+\frac{1}{2}})^2}}{\tilde{\omega_k}} - \frac{\sqrt{(y_k - y_{j+\frac{1}{2}})^2}}{\tilde{\omega_k}}\right) \epsilon_{k,g,p} \quad (2.41d)$$

where the summation is over the tracks that directly and indirectly cross through corner $(i+\frac{1}{2}, j+\frac{1}{2})$. All other surface crossings are treated with Equation 2.35.

2.2.6 Matrix form of CMFD method

Going back to Equation 2.31 and inserting the nonlinear diffusion coefficients from Equation 2.36, the finite difference form of the diffusion equation over a mesh cell becomes:

$$\begin{aligned}
&\Delta y^{i,j}(\hat{D}_{\mathbf{g}}^{i-\frac{1}{2},j}[\phi_{\mathbf{g}}^{i,j} - \phi_{\mathbf{g}}^{i-1,j}] + \tilde{D}_{\mathbf{g}}^{i-\frac{1}{2},j}[\phi_{\mathbf{g}}^{i,j} + \phi_{\mathbf{g}}^{i-1,j}]) - \\
&\Delta y^{i,j}(\hat{D}_{\mathbf{g}}^{i+\frac{1}{2},j}[\phi_{\mathbf{g}}^{i+1,j} - \phi_{\mathbf{g}}^{i,j}] + \tilde{D}_{\mathbf{g}}^{i+\frac{1}{2},j}[\phi_{\mathbf{g}}^{i+1,j} - \phi_{\mathbf{g}}^{i,j}]) + \\
&\Delta x^{i,j}(\hat{D}_{\mathbf{g}}^{i,j-\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j} - \phi_{\mathbf{g}}^{i,j-1}] + \tilde{D}_{\mathbf{g}}^{i,j-\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j} + \phi_{\mathbf{g}}^{i,j-1}]) - \\
&\Delta x^{i,j}(\hat{D}_{\mathbf{g}}^{i,j+\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j+1} - \phi_{\mathbf{g}}^{i,j}] + \tilde{D}_{\mathbf{g}}^{i,j+\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j+1} + \phi_{\mathbf{g}}^{i,j}]) + \\
\Delta x^{i,j}\Delta y^{i,j}\Sigma_{\mathbf{g}}^{R,i,j}\phi_{\mathbf{g}}^{i,j} &= \Delta x^{i,j}\Delta y^{i,j}\frac{\chi_{\mathbf{g}}^{i,j}}{k_{eff}}\sum_{\mathbf{g}'=1}^{\mathbf{G}}\nu\Sigma_{\mathbf{g}'}^{F,i,j}\phi_{\mathbf{g}'}^{i,j} + \Delta x^{i,j}\Delta y^{i,j}\sum_{\substack{\mathbf{g}'=1 \\ \mathbf{g}'\neq\mathbf{g}}}^{\mathbf{G}}\Sigma_{\mathbf{g}'\rightarrow\mathbf{g}}^{S,i,j}\phi_{\mathbf{g}'}^{i,j}
\end{aligned} \tag{2.42}$$

We can condense the CMFD diffusion equations down to matrix form to get the following generalized non-hermitian eigenvalue problem:

$$A\phi = \frac{1}{k_{eff}}M\phi \tag{2.43}$$

The matrices can be arranged in either a group-wise or cell-wise ordering. The CMFD implementation in OpenMOC uses a cell-wise order where the A matrix is composed of a block diagonal with blocks of size $\mathbf{G}\times\mathbf{G}$ and four off-diagonals for transport to neighboring cells as shown in Figure 2-11.

In subsection 2.2.7 we discuss how this matrix equation will be solved and used to accelerate the solution of the MOC solve.

2.2.7 CMFD accelerated MOC algorithm

The general flowchart for MOC algorithm and CMFD acceleration are shown in Figure 2-12.

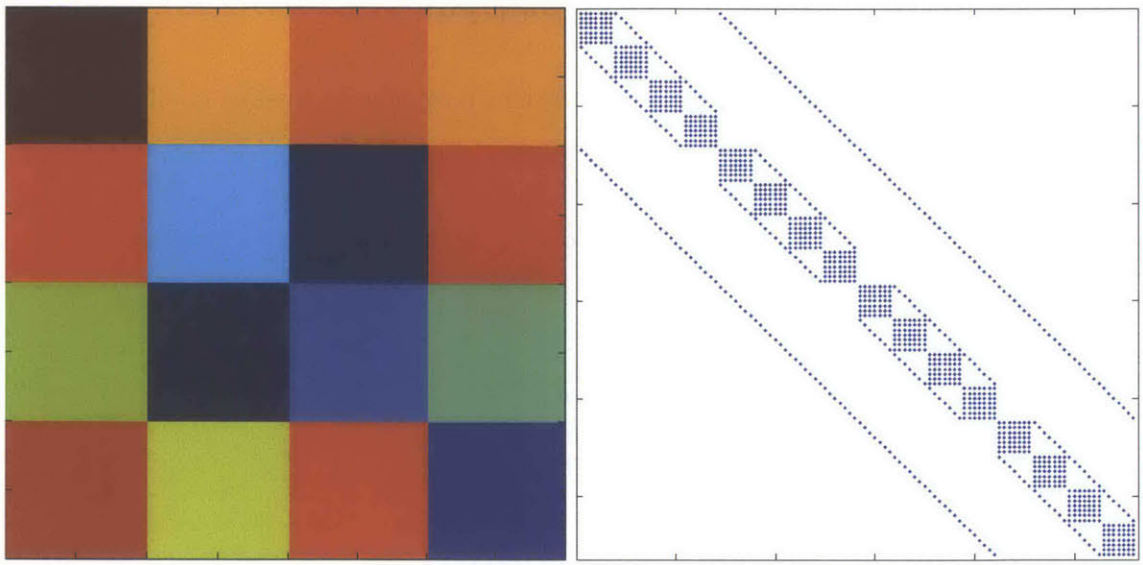

Figure 2-11. CMFD mesh layout (left) and spy of CMFD A matrix (right) for a 4 $\times$ 4 infinite lattice pin-cell problem with 7 energy group cross sections.

CMFD acceleration is implemented in OpenMOC by overlaying a rectilinear coarse mesh on top of the unstructured flat source region mesh as shown in Figure 2-4. During an MOC fixed source iteration, OpenMOC tallies the net currents across the surfaces of each mesh cell. The fixed source iteration algorithm then becomes Algorithm 3.

At the end of the fixed source iteration, OpenMOC proceeds to condense the cross sections, flux, and diffusion coefficients according to Equation 2.21. The diffusion coefficient coupling terms that link neighboring cells, $\hat{D}$ and $\tilde{D}$, are then computed using Equation 2.30 and Equation 2.37. There are two subtle points in computing the nonlinear coupling coefficients $\tilde{D}$. First, the condition $|\tilde{D}| < |\hat{D}|$ must be met in order to guarantee the diagonal dominance in the destruction matrix, A. If this condition is not met, the surface diffusion coefficients will be re-computed such that they are equal in magnitude and satisfy Equation 2.36. Second, under-relaxation of the nonlinear coupling coefficient is used to accelerate and maintain stability of the MOC problem for large, heterogeneous geometries. OpenMOC does so by applying a fixed damping factor on the $\tilde{D}$ terms. The nonlinear diffusion coefficients are initially set to zero and modified according to Equation 2.44.

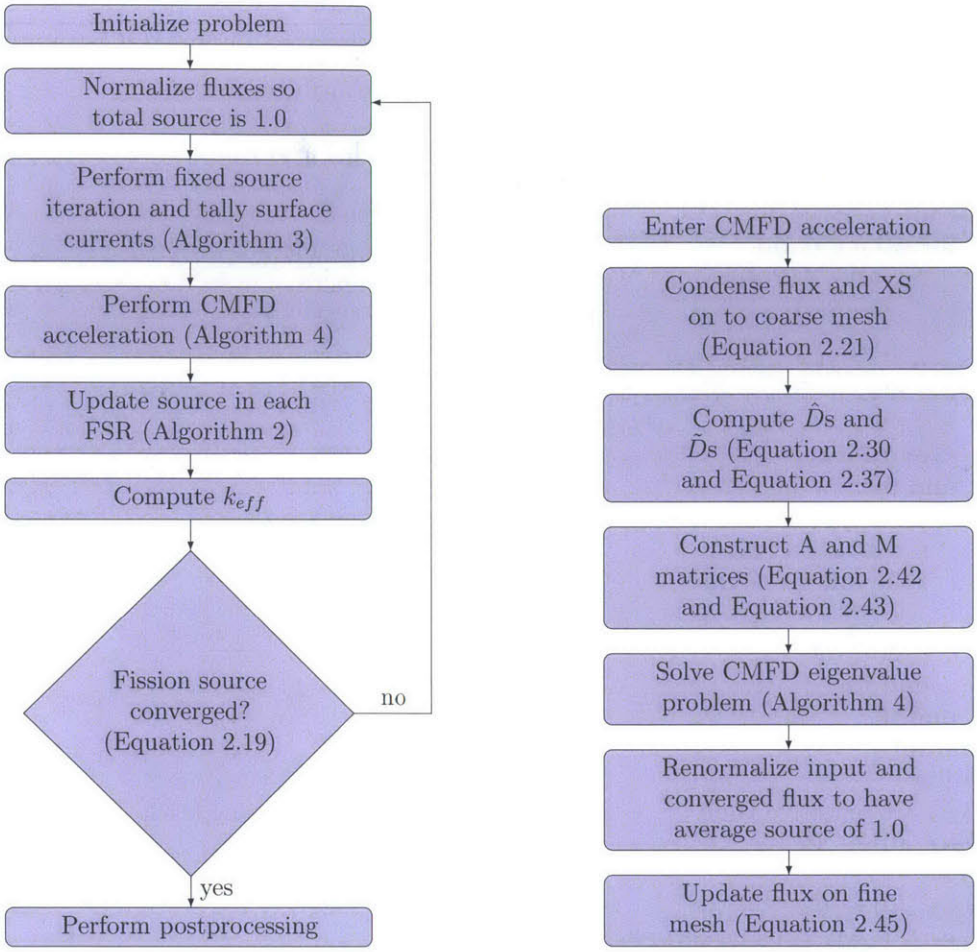

Figure 2-12. The solution procedure for CMFD accelerated MOC.

$$\tilde{D}_{\mathbf{g}}^{i+\frac{1}{2},j,(n)} = (1-\omega_d)\tilde{D}_{\mathbf{g}}^{i+\frac{1}{2},j,(n-1)} + \omega_d \frac{-\hat{D}_{\mathbf{g}}^{i+\frac{1}{2},j,(n)}(\phi_{\mathbf{g}}^{i+1,j,(n)} - \phi_{\mathbf{g}}^{i,j,(n)}) - \frac{\tilde{J}_{\mathbf{g}}^{i+\frac{1}{2},j,(n)}}{\Delta y^{i,j}}}{(\phi_{\mathbf{g}}^{i+1,j,(n)} + \phi_{\mathbf{g}}^{i,j,(n)})} \quad (2.44)$$

where ω_d is the under-relaxation damping factor and (n) is the fixed source iteration. Within each CMFD iteration, OpenMOC uses power iterations to solve the generalized non-Hermitian eigenvalue problem as shown in Algorithm 4. In each power iteration, the linear system is solved using a parallel (red-black) implementation of the Gauss-Seidel (GS) method as shown in Algorithm 5. Upon convergence of the CMFD diffusion problem, OpenMOC performs prolongation by multiplying each FSR's scalar flux by the ratio of the converged coarse mesh scalar flux to the initial

Algorithm 3 Fixed source iteration for CMFD accelerated OpenMOC

$\Phi_{r,g} \leftarrow 0 \quad \forall\, r, g \in \{R, G\}$ # Initialize FSR scalar fluxes to zero
while $Q_{r,g}\ \forall\, r$ not converged **do**
  for all $m \in M$ **do** # Loop over azimuthal angles
    for all $k \in K(m)$ **do** # Loop over tracks
      for all $s \in S(k)$ **do** # Loop over segments
        for all $g \in G$ **do** # Loop over energy groups
          for all $p \in P$ **do** # Loop over polar angles
            $r \leftarrow R(s)$ # Get FSR for this segment
            $\Delta\Psi_{k,r,g,p} \leftarrow \left(\Psi_{k,g,p} - \frac{Q_{r,g}}{\Sigma^{tr}_{r,g}}\right)\left(1 - e^{-\tau_{k,r,g,p}}\right)$
            $\Phi_{r,g} \leftarrow \Phi_{r,g} + \frac{4\pi}{A_r}\omega_{m(k)}\omega_p\omega_k \sin\theta_p l_{k,r}\Delta\Psi_{k,r,g,p}$
            $\Psi_{k,g,p} \leftarrow \Psi_{k,g,p} - \Delta\Psi_{k,g,p}$
          end for
        end for
        if $\Psi_{k,r,g,p} \cap$ surface **then** # Check if segment on a surface
          for all $g \in G$ **do** # Loop over energy groups
            for all $p \in P$ **do** # Loop over polar angles
              $\tilde{j}^{i\pm\frac{1}{2},j\pm\frac{1}{2}}_{\mathbf{g}} \leftarrow 2\pi\omega_{m(k)}\omega_k\omega_p \sin\theta_p \Psi_{k,g,p}$
            end for
          end for
        end if
      end for
    end for
    if B.C. are reflective **then** # Set incoming flux for outgoing track
      $\Psi_{k',g,p}(0) \leftarrow \Psi_{k,g,p}$ # Reflective B.C.'s
    else
      $\Psi_{k',g,p}(0) \leftarrow 0$ # Vacuum B.C.'s
      $L \leftarrow L + 2\pi\omega_{m(k)}\omega_p\omega_k \sin\theta_p \Psi_{k,g,p}$
    end if
  end for
  Solve CMFD diffusion problem # Algorithm 4
  Update MOC scalar flux # Equation 2.45
  Update k_{eff} and $Q_{r,g}\ \forall\, r$ # Algorithm 2
end while

coarse mesh scalar flux in the acceleration step:

$$\Phi_{r,g} = \Phi_{r,g}\frac{\phi^{i,j,new}_{\mathbf{g}}}{\phi^{i,j,old}_{\mathbf{g}}} \qquad \forall \quad r \in (i,j) \tag{2.45}$$

Where $\phi^{i,j,old}_{\mathbf{g}}$ is the coarse mesh scalar flux computed using equation Equation 2.21 and $\phi^{i,j,new}_{\mathbf{g}}$ is the converged CMFD coarse mesh scalar flux for coarse group $\mathbf{g}$ that contains group g.

Instead of splitting the corner currents during the MOC fixed source iteration, OpenMOC tallies the corner currents as independent surfaces and then splits the

Algorithm 4 Power Iteration Outer Loop Source Iteration

$\chi, \Sigma^{S,A,F,tr}, \Phi \rightarrow \chi, \Sigma^{S,A,F}, D, \phi$	# Condense XS using Equation 2.21
Compute $\hat{D}, \tilde{D}$	# Equation 2.30 and Equation 2.37
$S_{old} = M\phi$	# Compute initial source
$S_{old} /= \frac{\sum S_{old}}{N*\mathbf{G}}$	# Normal source to average 1.0
while S_{new} not converged **do**	
$\phi = A^{-1} S_{old}$	# Algorithm 5
$S_{new} = M\phi$	# Compute the new source
$k_{eff} = \frac{\sum S_{new}}{\sum S_{old}}$	# Compute k_{eff}
$S_{old} *= k_{eff}$	# Scale the old source
$R = (S_{new} - S_{old})/S_{new}$	# Compute the fission source residual
$EPS = \sqrt{\sum R^2/(N * \mathbf{G})}$	# Compute the fission source RMSD
$S_{new} /= \frac{\sum S_{new}}{N*\mathbf{G}}$	# Normal source to average 1.0
$S_{old} = S_{new}$	# Pass new source to old source
end while	

corner currents between their neighboring surfaces in step 2 of Algorithm 4. With the corner currents accounted for, the surface diffusion coefficients are then computed just prior to the power method eigenvalue solve. The first step in the power method inner loop iteration is to solve a linear fixed fission source problem. The GS method is used to solve the linear system as described by Algorithm 5. While more computationally efficient methods exist for solving generalized non-Hermitian eigenvalue problems like Krylov-subspace methods, we chose the power method for its simplicity and stability. Additionally, more computationally efficient methods exist for solving the linear system like Generalized Minimum Residual (GMRES) and stabilized bi-conjugate gradient (BiCGStab). As we will show in the timing results in Chapter 5, the power method with GS performs sufficiently well for this application where the time to complete the fixed source iteration with dominate the problem run time.

Algorithm 5 Parallel (red-black) Gauss-Seidel numerical flux inversion

while ϕ not converged **do**
  $\phi^{old} = \phi$ # Copy ϕ to ϕ^{old}
  for all $y \in Y$ **do** # Loop over red cells
    for $x = mod(y, 2) : x{+}{=}2 : x < X$ **do**
      for all $\mathbf{g} \in \mathbf{G}$ **do** # Loop over groups

$$\phi_{\mathbf{g}}^{x,y,(k+1)} = \frac{1}{A_{xx}} \left(S_{old,x} - \sum_{y<x} A_{xy} \phi_{\mathbf{g}}^{x,y,(k+1)} - \sum_{y>x} A_{xy} \phi_{\mathbf{g}}^{x,y,(k)} \right)$$

      end for
    end for
  end for
  for all $y \in Y$ **do** # Loop over black cells
    for $x = 1 - mod(y, 2) : x \mathrel{+}= 2 : x < X$ **do**
      for all $\mathbf{g} \in \mathbf{G}$ **do** # Loop over groups

$$\phi_{\mathbf{g}}^{x,y,(k+1)} = \frac{1}{A_{xx}} \left(S_{old,x} - \sum_{y<x} A_{xy} \phi_{\mathbf{g}}^{x,y,(k+1)} - \sum_{y>x} A_{xy} \phi_{\mathbf{g}}^{x,y,(k)} \right)$$

      end for
    end for
  end for
  $R = (M\phi^{(k+1)} - M\phi^{(k)})/(M\phi^{(k+1)})$ # Compute the fission source residual
  $EPS = \sqrt{\sum R^2/(N * \mathbf{G})}$ # Compute the fission source RMSD
end while

Chapter 3

Steady State Results

In this chapter we analyze the sensitivity of simulation parameters and model the performance of OpenMOC for solving the steady state transport specifications of the LRA and C5G7 benchmarks. The sensitivity analysis was performed to determine the simulation parameters (e.g. track spacing, number of azimuthal angles, and geometry discretization) required to converge the eigenvalue. A performance analysis was performed to understand the acceleration capabilities of CMFD as a function of coarse mesh cell size and energy group structure.

3.1 LRA Transport Benchmark

The transport version of the 2D LRA transient benchmark problem is specified in section A.1. The 2D LRA benchmark is a 2-group, quarter-core BWR transient problem this is specified using diffusion theory cross sections. A transport version of the benchmark was presented by [16] as a test problem for modern transport theory neutronics codes. To our knowledge, no one has presented a spatially converged solution to the initial state of the transport version of this benchmark. Therefore, we use an ultrafine mesh and dense track layout to generate a spatially converged solution that can be used for comparisons.

The geometry of the LRA problem is discretized into homogenized assemblies of size 15 cm $\times$ 15 cm. In order to spatially resolve the flux, the assemblies were

uniformly refined until the eigenvalue converged. The following refinements and corresponding flat source regions sizes were used in the study:

Table 3.1. FSR discretizations used in LRA convergence study

Uniform Refinements	FSRs per assembly	FSR size (cm)
0	1	15 x 15
1	4	7.5 x 7.5
2	16	3.75 x 3.75
3	64	1.875 x 1.875
4	256	0.9375 x 0.9375
5	1024	0.46875 x 0.46875
6	4096	0.2343675 x 0.2343675
7	16384	0.1171875 x 0.1171875

In all simulations the parameters listed in Table 3.2 were used:

Table 3.2. Calculation conditions

Parameter	Value
Number of polar angles	3 (using TY quadrature set)
CMFD damping factor	0.5 (unless otherwise specified)
CPU architecture	2 6-core Intel Xeon processors
Threads	12

For the sensitivity analysis, a three dimensional parameter space was explored to find the conditions necessary to achieve a spatially converged solution. The dimensions include track spacing, number of azimuthal angles, and mesh cell size. The finest mesh cell size and densest track layout was assumed to be the spatially converged solution. The parameters and flux plots for this reference solution are presented in Table 3.3 and Figure 3-1.

3.1.1 Sensitivity analysis to simulation parameters

In solving a problem with the 2D formulation of the method of characteristics, there are three main parameters to manipulate to refine the solution: the track spacing, the number of azimuthal angles, and the number of FSRs. There are obvious memory

Table 3.3. Calculation parameters for converged LRA problem

Parameter	Value
# of azimuthal angles	32
Track spacing	0.05 cm
MOC mesh size	0.1171875 cm x 0.1171875 cm
Converged k_{eff}	0.997213

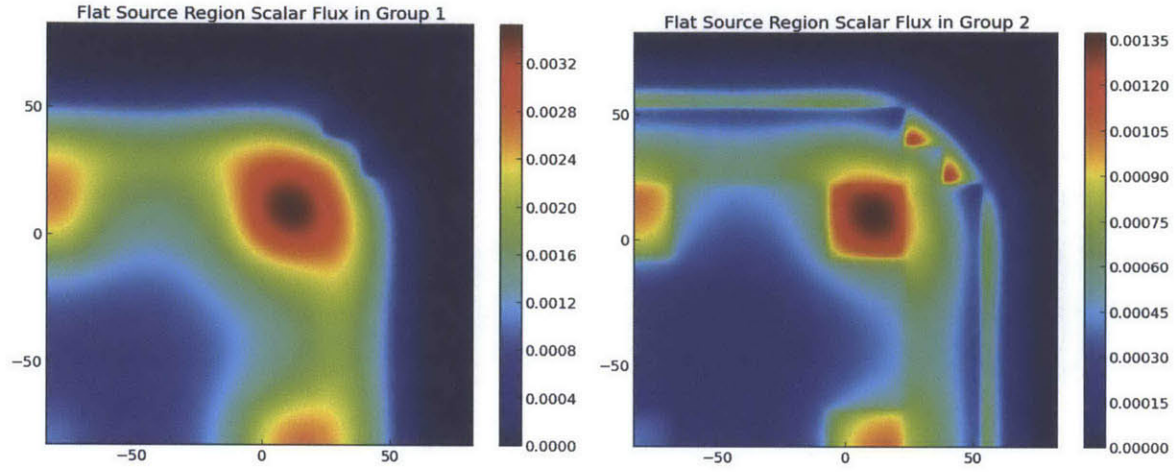

Figure 3-1. Flux plots for fast (left) and thermal (right) groups of the initial state of the transport LRA benchmark.

and runtime advantages to limiting the mesh size and number of tracks that traverse the geometry. However, different problems require different combinations of mesh and track refinement in order to spatially resolve the fission source. For example, regions where there are large flux gradients, such as in fuel pins where spatial self-shielding occurs, can result in fission rates that vary by a factor of 2 or more depending on location. Additionally regions around the core/reflector boundary can require a very fine mesh to resolve the position and magnitude of the flux peak in the reflector.

To understand the influence of track spacing, number of azimuthal angles, and the number of FSRs on the problem eigenvalue, a series of trials were conducted to examine the parameter space. The track spacing and number of azimuthal angles used in trials is presented in Table 3.4 and the spatial discretizations are presented in Table 3.1.

From these trials, we found the eigenvalue to be relatively insensitive to the num-

Table 3.4. Calculation conditions

Parameter	Values
Azimuthal angles	8, 16, 32
Track spacings (cm)	0.1, 0.05, 0.025, 0.01

ber of azimuthal angles and track spacing. For example, Figure 3-2 shows the eigenvalue error, Δk, for trials conducted with an FSR mesh with 5 uniform refinements (0.46875 cm $\times$ 0.46875 cm FSRs). The insensitivity to the number of azimuthal angles and track spacing is expected as the materials in the LRA problem are homogenized over large regions and the energy group structure is condensed to 2 groups. This makes it impossible to resolve the fine spatial and energetic flux features that would result in sharp local flux gradients in the physical geometry. For example, highly absorbing materials (e.g. control rods or burnable poisons) or neighboring fuel assemblies with significantly different U/MOX contents can cause significant local flux gradients that would require a fine track lay down to properly resolve. With this information, we selected a track spacing of **0.05 cm** and **32 azimuthal angles** as sufficient values to converge the eigenvalue.

The eigenvalue was found to be very sensitive to the FSR size. Figure 3-3 shows the eigenvalue error as compared to the spatially converged solution for a series of trials conducted with a track spacing of 0.05 cm and 32 azimuthal angles.

The data in Figure 3-3 demonstrates that in order to achieve an eigenvalue error of $\sim$25 pcm, the FSR mesh needs to be refined to below 0.5 cm $\times$ 0.5 cm. From this analysis, we selected the FSR mesh size of **0.46875 cm x 0.46875 cm** to be used in our transient analysis.

3.1.2 Performance analysis of CMFD acceleration

Coarse mesh finite difference acceleration has been shown to produce speedups of $>$ 100x for large LWR problems [14]. In OpenMOC we have adopted a generalized CMFD scheme that allows users to set the CMFD mesh size independent of FSR mesh size but with the requirement that FSRs can lie within only one CMFD mesh cell. This allows for CMFD acceleration to be performed on a (multi-) pin-cell, (multi-)

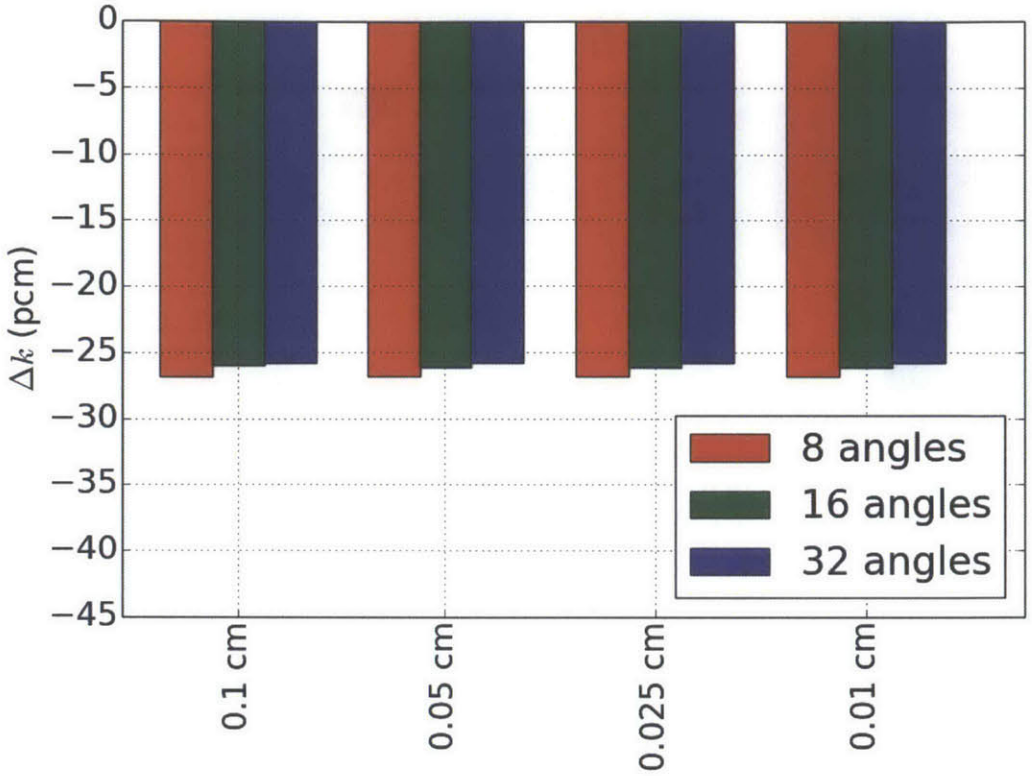

Figure 3-2. Plot of eigenvalue error vs track spacing for various azimuthal angle values for the steady state LRA benchmark. The FSR size was selected to be 0.46875 cm × 0.46875 cm.

assembly, or core level. Furthermore, the energy group structure for CMFD acceleration can be set independent of the group structure used in the MOC problem with the requirement that MOC energy groups must be fully contained within one CMFD energy group. This allows for fine-group transport calculations to be performed and significant acceleration to be achieved without a fine-group CMFD diffusion solve that can require significant compute time when the group structure goes beyond 50-100 groups.

The main parameters that affect the performance of CMFD in accelerating the solution to an MOC transport problem include the CMFD mesh cell size, CMFD energy group structure, and CMFD non-linear diffusion coefficient damping factor. All performance trials were conducted using parameters identified in subsection 3.1.1 as producing a sufficiently converged solution. Figure 3-4 shows the runtime and

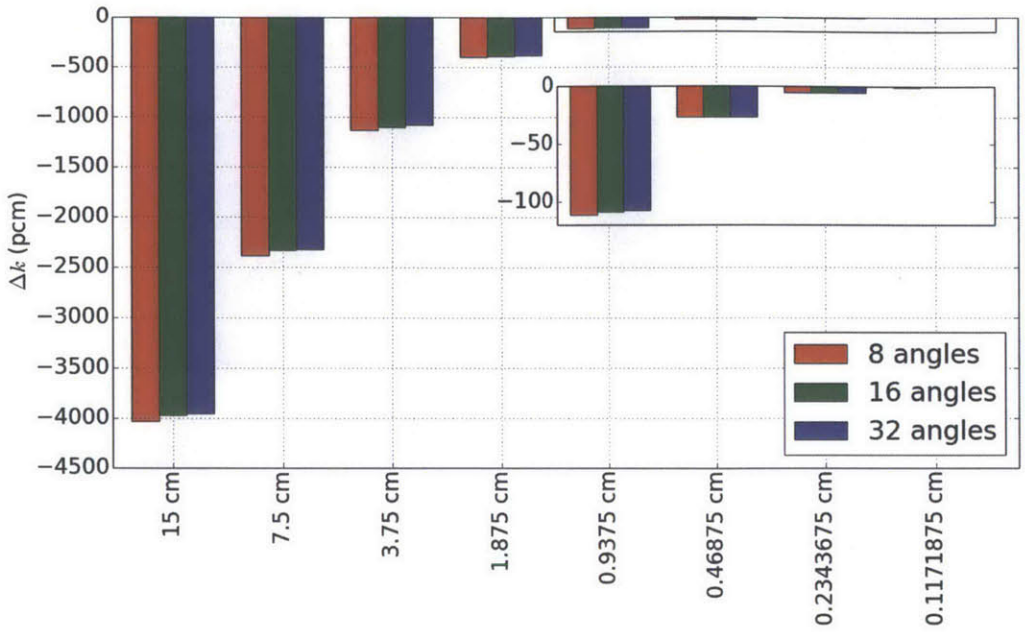

Figure 3-3. Plot of eigenvalue error vs mesh size applied to an assembly to generate the FSR mesh for the steady state LRA benchmark. The inset shows a zoomed-in view of the bounded region.

number of MOC fixed source iterations required to converge the eigenvalue as a function of CMFD mesh size.

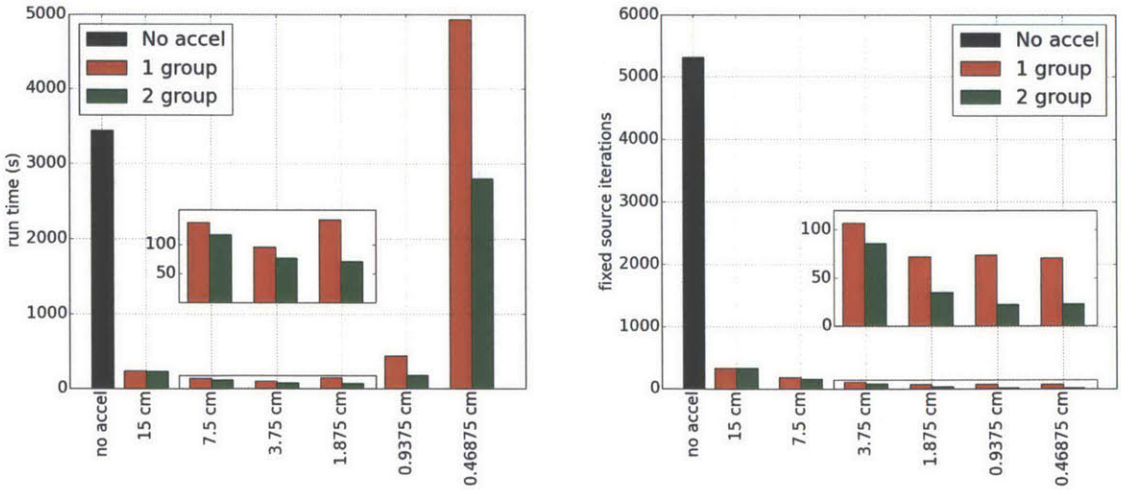

Figure 3-4. Plots of the runtime and # of MOC fixed source iterations vs the number of mesh cell refinements on each assembly in the CMFD mesh. The insets show a zoomed-in view of the bounded regions.

Figure 3-4 shows that there is a tradeoff between minimizing the number of MOC fixed source iterations and the runtime to converge the fission source in our implementation of CMFD. The assembly-wise CMFD meshes (0 refinements) produced poor performance due to the excessive number of MOC fixed source iterations. Similarly, the fine CMFD meshes (4 and 5 refinements) had poor runtime performance due to excessive computing time to solve the CMFD diffusion problem after each MOC fixed source iteration. The CMFD energy group structure had a small effect on the runtime and number of MOC fixed source iterations required to converge the solution. In all cases, the performance of the 2 group structure was better than the 1 group case. This performance advantage can be attributed to the reduction in MOC fixed source iterations required to converge the fission source for the 2 group structure. It should also be noted that the eigenvalue for all these trials was within 10 pcm of each other demonstrating consistency between answers produced with various CMFD mesh levels and unaccelerated MOC. A small difference in eigenvalues can be expected between the unaccelerated and accelerated cases due to slightly premature convergence for the unaccelerated and coarse spatial/energy discretization cases. Using this information, we selected CMFD mesh sizes of **1.875 cm x 1.875 cm** and **3.75 cm x 3.75 cm** to be further studied in transient analysis due to their ability to rapidly accelerate the convergence of the MOC problem. For completeness, we have included the runtime, MOC fixed source iterations, and eigenvalue for each of the cases plotted in Figure 3-4 in Table 3.5.

The effect of a fixed damping factor on the CMFD method was also investigated. In all cases the same 2 group structure was used in MOC and CMFD. Table 3.6 and Figure 3-5 show the number of fixed source iterations required to converge the fission source distribution to 1e-6 for the LRA benchmark problem using various damping factors. The effective diffusion coefficient is applied in solving the CMFD equations for simulations with CMFD mesh sizes of 15 cm, 7.5 cm, and 3.75 cm. As shown in Table 3.6 the CMFD method with a damping factor of less than 0.6 reduces the number of MOC fixed source iterations by a factor of >100 and the runtime by a factor of >45. For this specific case, the optimal damping factor was found to be

Table 3.5. The eigenvalue, runtime, and number of fixed source iterations required to converge the LRA benchmark with different CMFD mesh sizes.

CMFD mesh size (cm)	CMFD groups	Fixed source iterations	k_{eff}	Runtime (s)
no accel	-	5305	0.996950	3437.5
15	2	333	0.996955	234.2
7.5	2	161	0.996957	117.1
3.75	2	86	0.996956	77.3
1.875	2	35	0.996955	71.5
0.9375	2	22	0.996955	177.8
0.46875	2	23	0.996954	2798.5
15	1	330	0.996955	239.5
7.5	1	181	0.996956	137.6
3.75	1	107	0.996956	96.4
1.875	1	72	0.996957	142.7
0.9375	1	74	0.996956	439.6
0.46875	1	71	0.996956	4928.1

$\sim$0.5 and a damping factor of 0.7 or greater would fail to converge the problem.

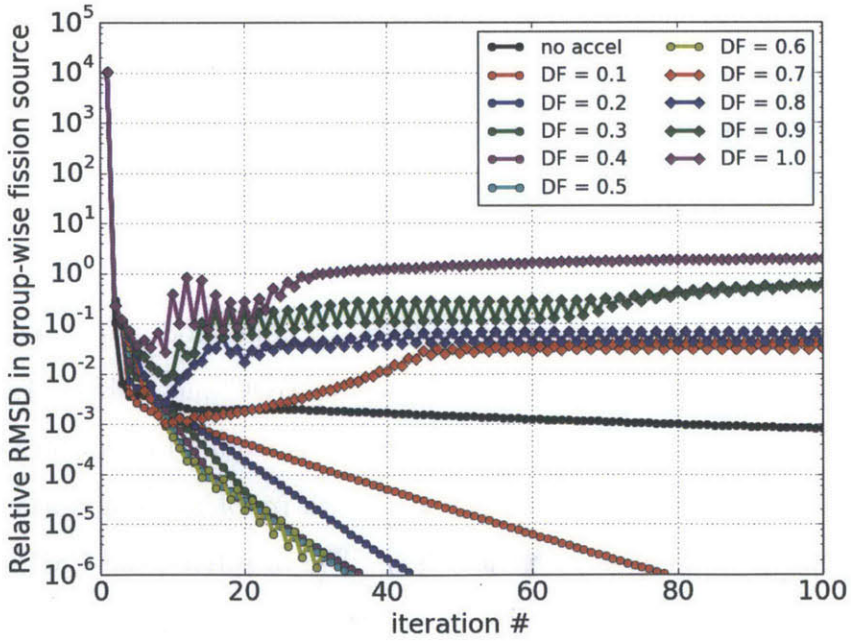

Figure 3-5. Plot of relative RMSD in group-wise fission source (Equation 2.19) vs iteration # for trials with various damping factors.

Table 3.6. The eigenvalue, runtime, and number of fixed source iterations required to converge the LRA benchmark with different CMFD damping factors.

CMFD	Damping factor	fixed source iterations	k_{eff}	Runtime (s)
N	-	5305	0.996950	3437.5
Y	0.1	78	0.996955	146.9
Y	0.2	43	0.996955	95.6
Y	0.3	36	0.996955	77.0
Y	0.4	36	0.996955	72.3
Y	0.5	35	0.996955	71.5
Y	0.6	31	0.996956	68.0
Y	0.7	-	-	-
Y	0.8	-	-	-
Y	0.9	-	-	-
Y	1.0	-	-	-

Based on the runtime performance and stability for the various damping factors used, we selected a damping factor of **0.5** to be used in the transient analysis.

3.2 C5G7 Transport Benchmark

The transient version of the 2D C5G7 transport benchmark problem is specified in section A.2. To summarize, the C5G7 problem was developed as a modern benchmark for deterministic neutron transport methods without spatial homogenization. The problem contains sixteen 17 × 17 pin cell assemblies surrounded by water with vacuum boundary conditions on all sides. The benchmark maintains heterogeneity between the fuel and moderator regions. Typically the problem is presented as a quarter core problem with reflective boundary conditions on inner surfaces (Figure A-2). In the steady state analysis, the quarter core geometry is used.

The C5G7 benchmark specification provides the reference eigenvalue and pin powers to allow code developers to compare the accuracy of their codes. Before performing the sensitivity analysis, we tested the accuracy of OpenMOC by simulating the C5G7 benchmark problem with an ultrafine FSR discretization, 128 azimuthal angles, and 0.05 track spacing. The pin cells in the ultrafine FSR case discretized into three equal

volume fuel rings, three water rings of inner radius 0.54 cm, 0.58 cm, and 0.62 cm, and eight sectors. The reflector region adjacent to the bundles is discretized into two regions of thickness 13.86 cm and 7.56 cm when going from the fueled assemblies to the boundaries. The region closest to the fueled assemblies is discretized into square cells of side length 0.126 cm followed by the outermost region with square cells of side length 1.26 cm. A summary of the results and plots of the fast and thermal flux for this simulation are shown in Table 3.7 and Figure 3-6, respectively.

Table 3.7. Calculation parameters for converged C5G7 problem

Parameter	Value
# of azimuthal angles	128
Track spacing	0.05 cm
Flat source regions	142,964
Converged k_{eff}	1.18672
Reference k_{eff}	1.18655
Δk_{eff} (pcm)	+17 pcm

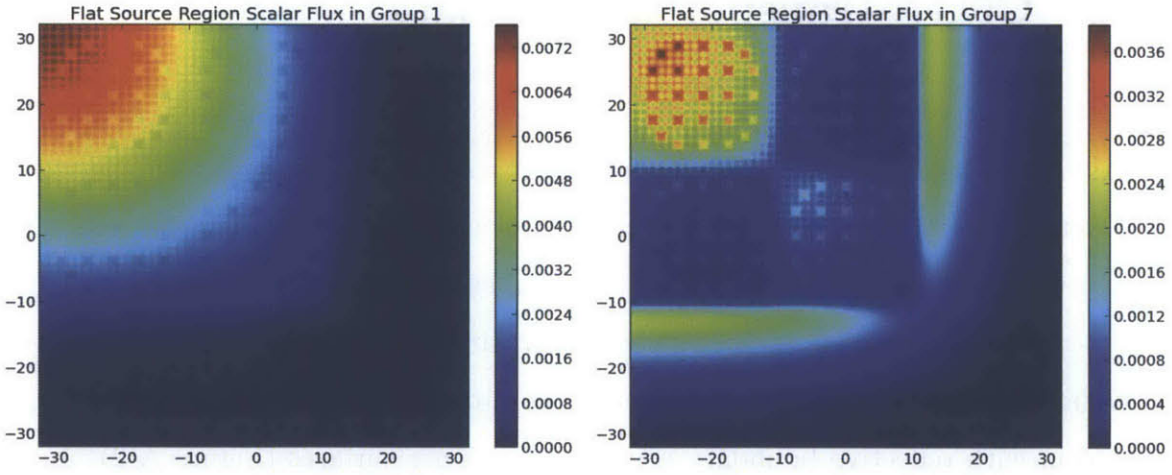

Figure 3-6. Flux plots for fast (left) and thermal (right) groups of the C5G7 benchmark.

While we would ideally run our transient simulations with an ultrafine FSR discretization and track layout, the runtime performance will be significantly affected by our choice of these parameters. Therefore, we conducted a sensitivity analysis of the eigenvalue on the the FSR discretization, number of azimuthal angles, and track

spacing. In order to spatially resolve the flux within the fueled regions, the pin cells were discretized using axial divisions and radial rings in the the fuel and moderator. Four discretization schemes were used and are described in Table 3.8 and Figure 3-7. The reflector region adjacent to the bundles was also discretized as shown in Figure 3-8. The reflector region is broken up into three regions of thickness 6.3 cm, 7.56 cm, and 7.56 cm when going from the fueled assemblies to the boundaries. The region closest to the fueled assemblies is discretized into square cells of side length 0.21 cm, followed by a region of square cells of side length 0.315 cm, followed by the outermost region with square cells of side length 1.26 cm.

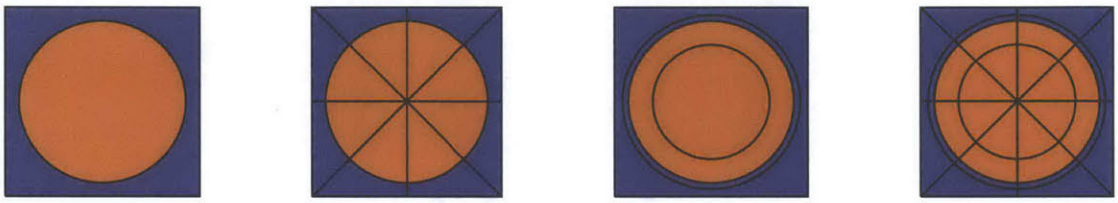

Figure 3-7. Illustration of spatial discretization schemes used to model the pin cells in the C5G7 problem. The moderator and fuel regions are shown in blue and orange, respectively.

Table 3.8. Discretization schemes for pin cell

Pin-Cell Scheme	Fuel Ring Radii (cm)	Water Ring Radii (cm)	Sectors
1	0.54	0.54	0
2	0.54	0.54	8
3	0.382, 0.54	0.54, 0.58	0
4	0.382, 0.54	0.54, 0.58	8

In all simulations the parameters listed in Table 3.9 were used.

For the sensitivity analysis, a three dimensional parameter space was explored to find the conditions necessary to achieve a spatially converged solution. The dimensions include track spacing, number of azimuthal angles, and mesh cell size.

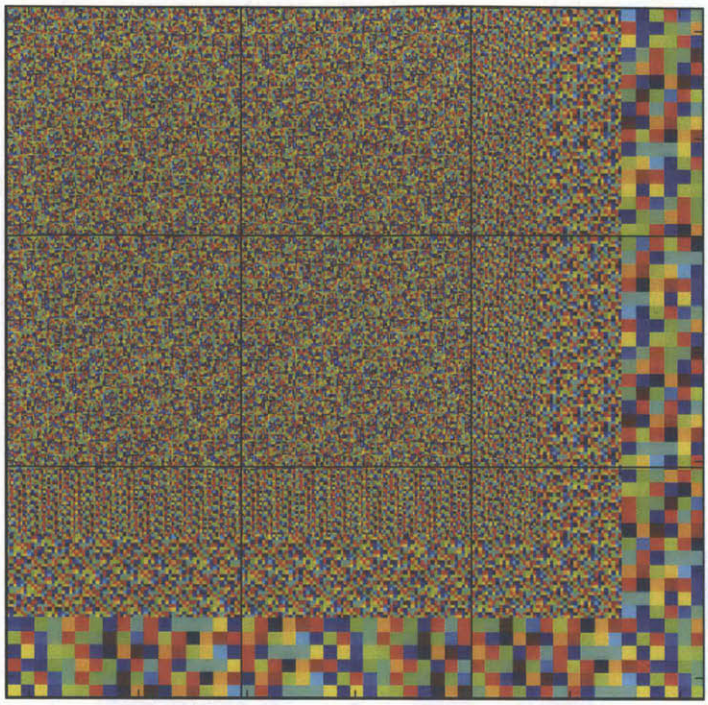

Figure 3-8. Flat source region discretization scheme for the quarter-core C5G7 geometry with pin-cell discretization scheme 4.

Table 3.9. Calculation conditions for C5G7 steady state benchmark

Parameter	Value
Number of polar angles	3 (using TY quadrature set)
MOC convergence criteria	1e-6 on root-mean-square-difference of fission source
CMFD convergence criteria	1e-6 on root-mean-square-difference of fission source
GS convergence criteria	1e-8 on root-mean-square-difference of fission source
CMFD damping factor	0.6 (unless otherwise specified)
CPU architecture	2 6-core Intel Xeon processors
Threads	12

3.2.1 Sensitivity analysis to simulation parameters

To understand the influence of track spacing, number of azimuthal angles, and the number of FSRs on converging the C5G7 eigenvalue, a series of trials were performed to map out the parameter space. The track spacings and number of azimuthal angles used are presented in Table 3.10 and the spatial discretizations are provided in Table 3.8.

Table 3.10. Calculation conditions

Parameter	Values
Azimuthal angles	8, 16, 32, 64, 128
Track spacings (cm)	0.1, 0.05, 0.025, 0.01

From these trials, we found the eigenvalue to be relatively insensitive to the track spacing, but sensitive to the number of azimuthal angles. For example, Figure 3-9 shows the eigenvalue error for trials conducted with pin-cell discretization scheme 4. The sensitivity to number of azimuthal angles is likely due to the sharp flux gradients in the reflector just outside the core and the presence of neighboring U/MOX assemblies. With this information, we selected a track spacing of **0.05 cm** and **64 azimuthal angles** as sufficient values to converge the eigenvalue.

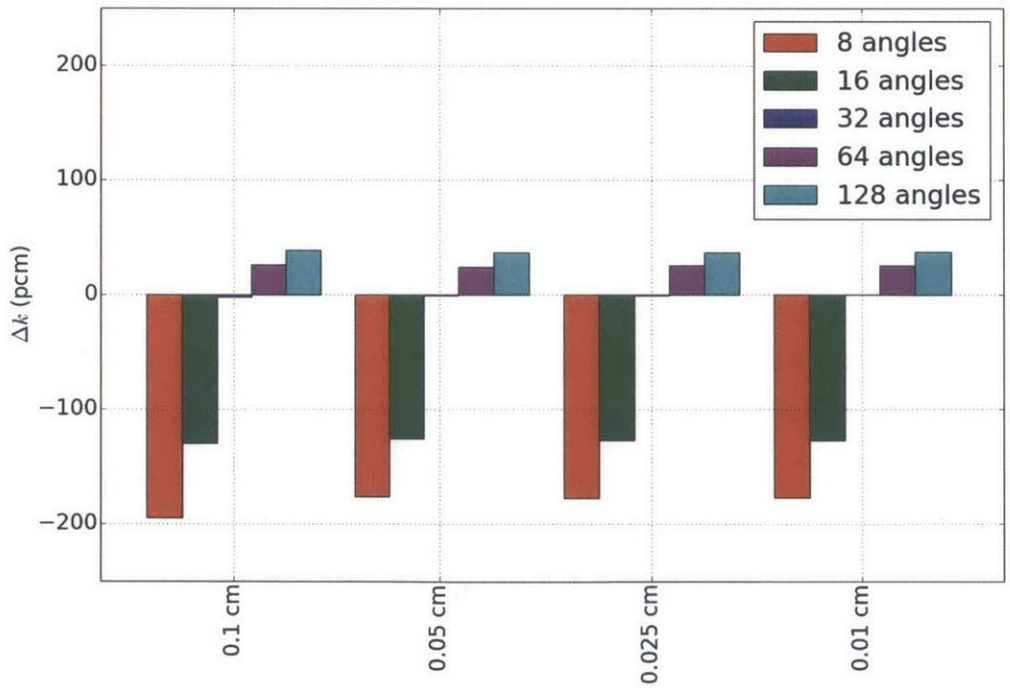

Figure 3-9. Plot of eigenvalue error vs track spacing for various azimuthal angle values for the steady state C5G7 benchmark. Scheme 4 was used for the pin cell spatial discretization.

The eigenvalue was found to be very sensitive to the FSR size. Figure 3-10 shows

the eigenvalue error as compared to the reference solution for a series of trials conducted with a track spacing of 0.05 cm and 64 azimuthal angles.

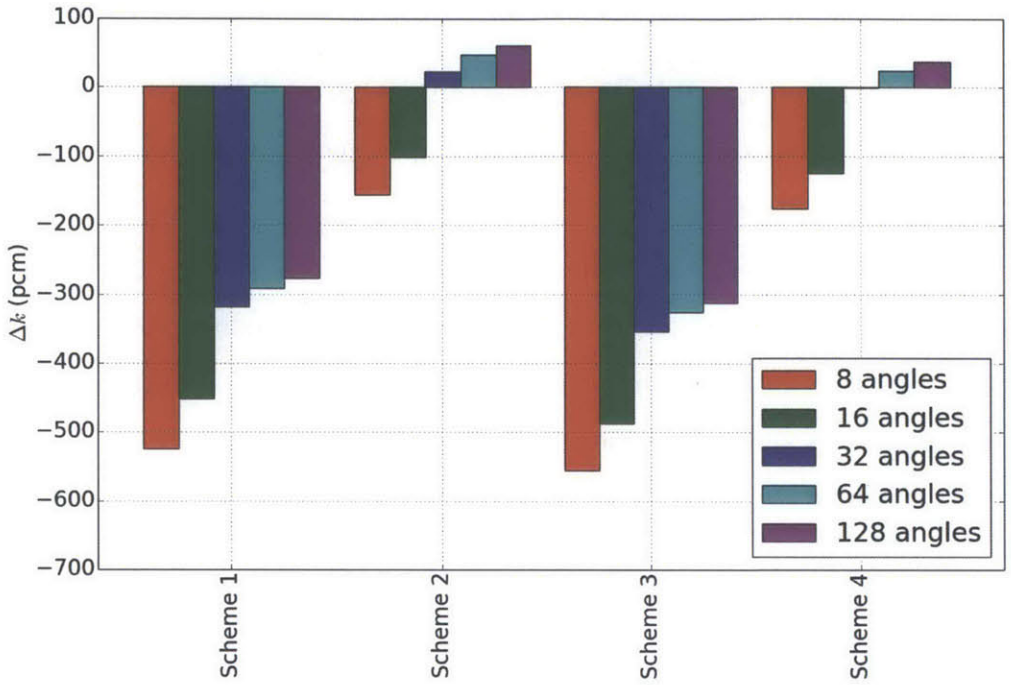

Figure 3-10. Plot of absolute eigenvalue error vs track spacing for the steady state C5G7 benchmark.

The data in Figure 3-10 demonstrates that the eigenvalue is highly sensitive to the spatial discretization. In particular, the presence of sectors (schemes 2 and 4) is a crucial characteristic in reducing the eigenvalue error. In order to achieve an eigenvalue error of ~25 pcm, at least 32 azimuthal angles and a pin cell spatial discretization of 2 water rings, 2 fuel rings, and 8 sectors was required. From this analysis, we selected a track spacing of **0.05 cm**, number of azimuthal angles of **64 angles**, and pin cell discretization of **2 water rings, 2 fuel rings, and 8 sectors** to use in our transient analysis.

3.2.2 Performance analysis of CMFD acceleration

The main parameters that affect the performance of CMFD in accelerating the solution to an MOC transport problem include the CMFD mesh cell size, CMFD energy

group structure, and CMFD non-linear diffusion coefficient damping factor. All performance trials were conducted using parameters identified in subsection 3.2.1 as producing a sufficiently converged solution. The four different CMFD energy group structures studied are shown in Figure 3-11. Figure 3-12 shows the runtime and number of MOC fixed source iterations required to converge the fission source as a function of CMFD mesh size.

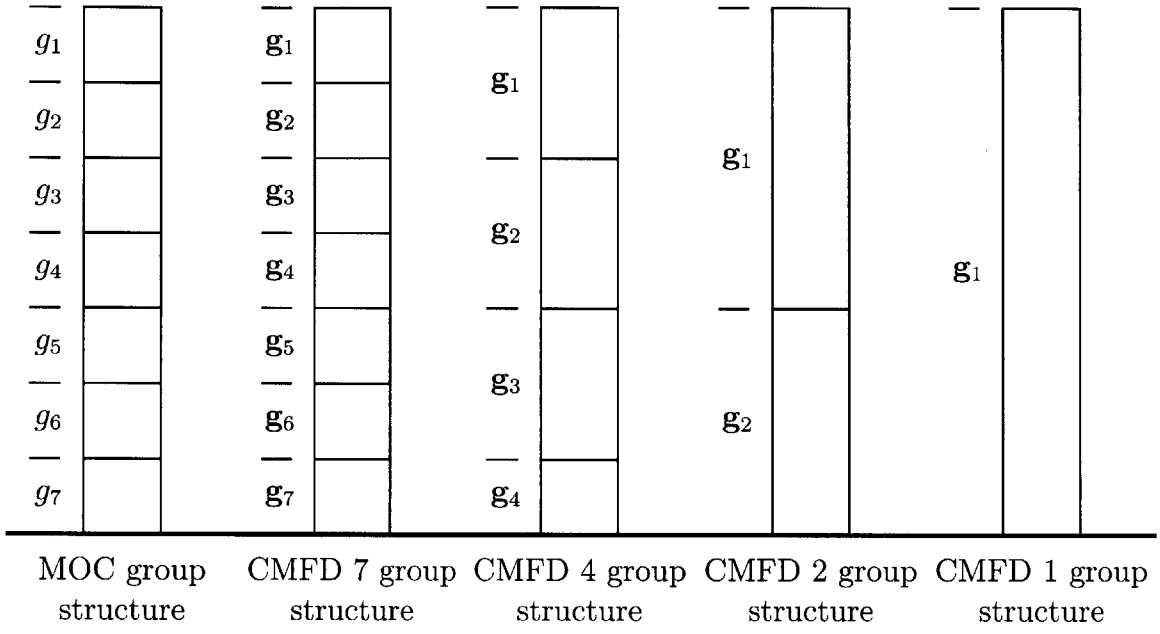

Figure 3-11. Illustration of CMFD energy group structures used in solving the C5G7 benchmark.

As was the case with the LRA benchmark, the coarse CMFD meshes produced poor performance due to an excessive number of MOC fixed source iterations. The CMFD energy group structure had a small effect on the runtime and number of iterations required to converge the solution. For the pin-wise CMFD mesh, the multigroup structures were clearly favored in terms of both runtime and number of MOC fixed source iterations. It should also be noted that the eigenvalue for all these trials was within 10 pcm of each other demonstrating consistency between answers produced with various CMFD mesh levels and unaccelerated MOC. A small difference can be expected due to slightly premature convergence for the unaccelerated and coarse spatial/energy discretization cases. Using this information, we selected a pin-cell CMFD

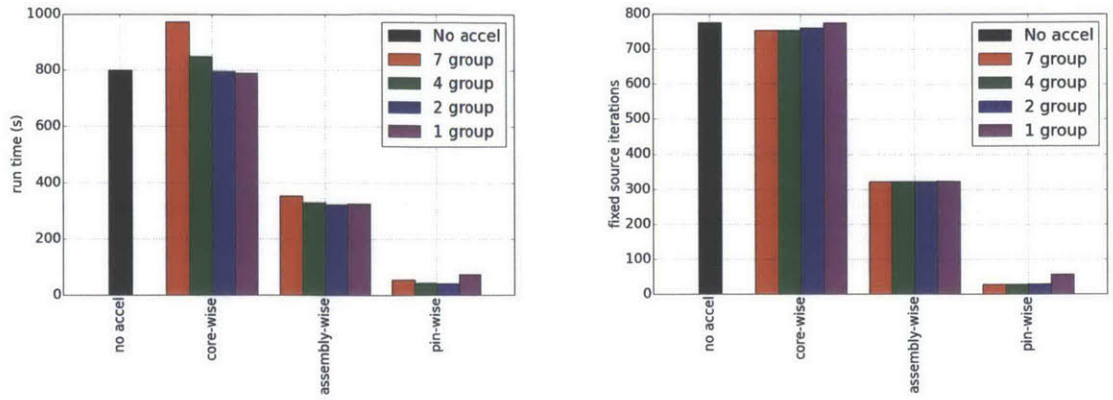

Figure 3-12. Plots of the runtime and number of fixed source iterations vs the CMFD mesh size.

mesh along with the MOC simulation parameters identified in subsection 3.2.1 to be the focus in our transient analysis. For completeness, we have included the runtime, MOC fixed source iterations, and eigenvalue for each of the cases plotted in Figure 3-12 in Table 3.11.

Table 3.11. The eigenvalue, runtime, and number of fixed source iterations required to converge the C5G7 benchmark with different CMFD spatial discretization schemes.

CMFD mesh level	CMFD groups	Fixed source iterations	k_{eff}	Runtime (s)
no accel	-	775	1.186743	798.5
core-wise	1	775	1.186743	788.9
core-wise	2	760	1.186748	794.6
core-wise	4	754	1.186750	847.9
core-wise	7	754	1.186750	971.8
assembly-wise	1	323	1.186788	324.2
assembly-wise	2	321	1.186789	321.5
assembly-wise	4	322	1.186790	328.1
assembly-wise	7	321	1.186790	352.0
pin-wise	1	56	1.186787	73.8
pin-wise	2	29	1.186788	41.7
pin-wise	4	27	1.186789	42.9
pin-wise	7	27	1.186789	55.0

The effect of a fixed damping factor on the CMFD method is further investigated. In all cases the same group structure was used in MOC and CMFD. Table 3.12 and

Figure 3-13 show the number of fixed source iterations required to converge the fission source distribution to 1e-6 for the C5G7 benchmark problem. As shown in Table 3.12 the CMFD method with a damping factor of ~0.6-0.7 reduces the number of MOC fixed source iterations by a factor of ~30 and the runtime by a factor of ~15. For this specific case, the optimal damping factor is about 0.6 and a damping factor of 0.9 or greater fails to converge the problem.

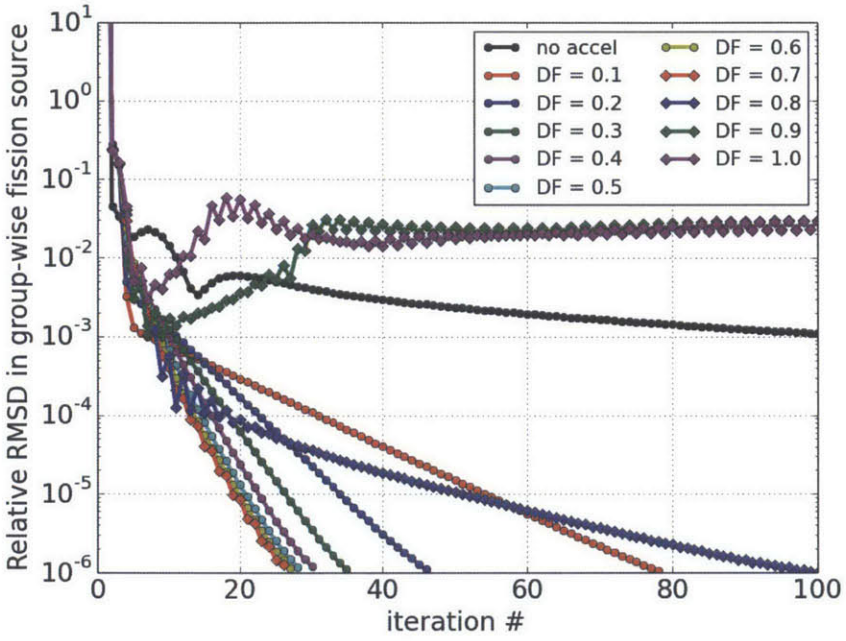

Figure 3-13. Plot of relative RMSD in group-wise fission source (Equation 2.19) vs iteration # for trials with various damping factors.

Based on the runtime performance and stability for the various damping factors used, we selected a damping factor of **0.6** to be used in the transient analysis.

Table 3.12. The eigenvalue, runtime, and number of fixed source iterations required to converge the LRA benchmark with different CMFD damping factors.

CMFD	Damping factor	fixed source iterations	k_{eff}	Runtime (s)
N	-	775	1.186743	798.5
Y	0.1	78	1.186786	144.6
Y	0.2	46	1.186788	90.5
Y	0.3	35	1.186789	71.8
Y	0.4	30	1.186789	61.4
Y	0.5	28	1.186789	58.0
Y	0.6	27	1.186789	55.0
Y	0.7	26	1.186789	51.8
Y	0.8	100	1.186790	163.9
Y	0.9	-	-	-
Y	1.0	-	-	-

Chapter 4

Time-dependent Method of Characteristics

Many methods for solving the time-dependent neutron transport and neutron diffusion equations rely on the separation of the spatial (shape) and temporal (amplitude) components of the neutron flux during a transient simulation [20]. Solving for the fine-grained neutron flux at a particular moment in time typically comprises the majority of the run time in a transient simulation. For this reason we seek to develop methods that minimize the number of fine-grained spatial solves. In the last few decades, several approaches of using the CMFD framework for solving time dependent MOC and nodal diffusion problems have been developed ([23], [11], [7]). Solving for the temporal component of the neutron flux often involves a spatial homogenization and/or energy condensation allowing for the neutron flux to be approximated at times between spatial solves as illustrated in Figure 4-1.

In Figure 4-1 Δt_a and Δt_s represent the time step size for the amplitude function and shape function, respectively. The time values t^a and t^{a-1} represent the time at the current and previous amplitude function time step, respectively. The time values t^s and t^{s-1} represent the time of the current and previous shape function solve. This concept of separating the spatial and temporal solves of a transient simulation can be employed because the spatial component of the neutron flux is often slowly varying whereas the amplitude component varies rapidly. In this thesis we have selected

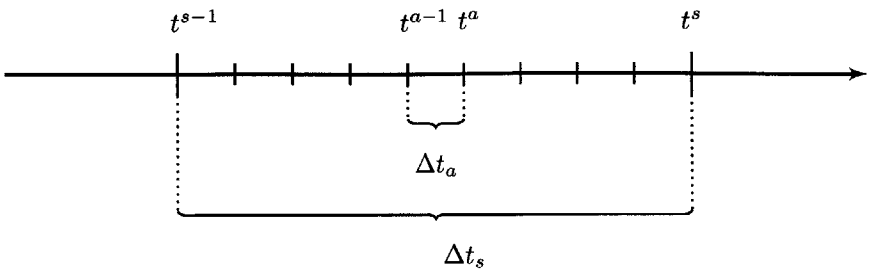

Figure 4-1. The relative size of the shape function and amplitude function time step sizes. The notation for time values during an arbitrary amplitude function solve a within shape function solve s is also shown.

three spatial methods, the Adiabatic, Theta, and MAF methods, to investigate. The derivation of these methods is provided in section 4.1. All methods utilized the fully implicit formulation of the time-dependent coarse mesh finite difference (TCMFD) equations with non-linear coupling coefficients for the temporal solve, which will be derived in section 4.3. In section 4.4 we will discuss our implementation of the transient methods in OpenMOC.

4.1 Derivation of Time-dependent Method of Characteristics Methods

The derivation for the Adiabatic, Theta, and Multigrid Amplitude Function spatial methods begins with the time-dependent neutron transport equation **with assumed isotropic angular flux time derivative** [15] and the delayed neutron precursor density balance equation:

$$\frac{1}{4\pi v_g}\frac{\partial \Phi_g(r,t)}{\partial t} = -\Omega \cdot \nabla \Psi_g(r,\Omega,t) - \Sigma_g^{tr}(r,t)\Psi_g(r,\Omega,t) + Q_g(r,t),$$

$$Q_g(r,t) = \frac{1}{4\pi}\left(\frac{\chi_{r,g}(1-\beta)}{k_{eff}^0}\sum_{g\prime=1}^{G}\nu\Sigma_{g\prime}^F(r,t)\Phi_{g\prime}(r,t) + \sum_{g\prime=1}^{G}\Sigma_{g\prime\rightarrow g}^S(r,t)\Phi_{g\prime}(r,t) + \sum_{m=1}^{M}\chi_{r,g}\lambda_m C_m(r,t)\right) \tag{4.1}$$

$$\frac{\partial C_m(r,t)}{\partial t} = \frac{\beta_m}{k^0_{eff}} \sum_{g'=1}^{G} \nu\Sigma^F_{g'}(r,t)\Phi_{g'}(r,t) - \lambda_m C_m(r,t) \quad \forall \quad m = 1, ..., M \tag{4.2}$$

Terms in the balance equation are defined in Table 4.1. When solving a problem using the time-dependent methods, we typically assume the initial state is critical. To ensure criticality in the neutron balance equation, we include the initial eigenvalue, k^0_{eff}, on the fission source term as shown in Equation 4.1 and Equation 4.2. The neutron spectrum from fission, $\chi_{r,g}$, has been assumed to be the same for both prompt and delayed neutrons from a given material.

Treating the temporal derivative of the angular flux explicitly instead of imposing the isotropic assumption has been studied, but incurs either significant memory requirements to store the segment-averaged angular fluxes at the current and previous time steps or computational overhead to recompute the angular flux at the previous time step on-the-fly [1, 16]. Furthermore, the purpose of our study was to focus more on the sensitivity of time-dependent MOC methods and less on the implementing high-precision temporal flux approximations for the MOC method.

Table 4.1. Descriptions of the transport equation terms

Variable	Description	Variable	Description
υ	Velocity	Ω	Direction of neutron travel
Φ	Scalar flux	Ψ	Angular flux
Σ^{tr}	Transport cross section	Σ^F	Fission cross section
Σ^S	Scattering cross section	ν	Neutrons per fission
Q	Isotropic neutron source	C	Delayed neutron precursor density
χ	Fission spectrum	β	Delayed neutron fraction
t	Time	r	Position vector
g, g'	Energy group	λ	Precursor decay constant
m	Precursor group		

In these methods, the fine mesh scalar flux, Φ, is factorized into a coarse mesh scalar flux, ϕ, and a fine mesh flux shape function, φ. Equation 4.3 provides the definition for the fine mesh shape.

$$\varphi_g(r,t) \equiv \frac{\Phi_g(r,t)}{\phi_g^{i,j}(t)} \tag{4.3}$$

The superscripts, i, j, on the coarse mesh scalar flux, $\phi_g^{i,j}(t)$, represent the coarse mesh cell indices. In our derivation of time-dependent transport methods we have assumed that the fine mesh and coarse mesh share the same group structure with groups denoted by index g. The coarse mesh scalar flux for a particular mesh cell and group is computed by area averaging the fine mesh flux of the FSRs contained within that cell using the procedure shown in Equation 2.21.

4.1.1 The Adiabatic Method

The Adiabatic method is conceptually the simplest of the transient methods and involves ignoring the time derivative of the neutron flux. This reduces the transport equation to its steady-state form (Equation 2.1). Therefore, at each time step, the neutron flux is computed using a standard steady-state MOC eigenvalue problem (Algorithm 3) where the material properties have been perturbed to reflect any material movements and feedback. The delayed neutron precursors are essentially ignored in the Adiabatic method as they are assumed to be at their steady state values.

An important aspect of the Adiabatic method is that the flux shape produced by the steady-state MOC eigenvalue problem is an eigenvector and therefore independent of power level. Before Equation 2.21 can be used to compute an updated coarse mesh flux, the fine mesh flux must be renormalized to the current power level. This is achieved by using the flux renormalization procedure presented in Equation 4.4.

$$\Phi_g(r,t) = \Phi_g^*(r,t) \frac{\bar{P}(t)}{\frac{1}{A_{core}} \sum_{r=1}^{R} \sum_{g=1}^{G} \kappa\Sigma_g^F(r,t)\Phi_g^*(r,t)A(r)} \tag{4.4}$$

where $\bar{P}(t)$ is the average power density in the core at time t prior to the shape function solve, A_{core} is the area of the fueled material in the core, $\Phi_g^*(t)$ is the fine mesh scalar flux that is output by the MOC solver, $\Phi_g(t)$ is the fine mesh scalar flux

after renormalization, and κ is the thermal energy per fission.

4.1.2 The Theta Method

Before describing the Theta method, we first define a new MOC source term, $\tilde{Q}$, that incorporates the flux temporal derivative:

$$\tilde{Q}_g(r, t^s) = Q_g(r, t^s) - \frac{1}{4\pi v_g} \frac{\partial \Phi_g(r, t)}{\partial t} \tag{4.5}$$

The fully implicit Theta method makes a backwards difference approximation on the flux temporal derivative resulting in the following source term:

$$\tilde{Q}_g(r, t^s) = Q_g(r, t^s) - \frac{1}{4\pi v_g \Delta t_{shape}} \left(\Phi_g(r, t^s) - \Phi_g(r, t^{s-1}) \right) \tag{4.6}$$

Where t^s and t^{s-1} represent the time at the current and previous shape function time steps, respectively. Note that Equation 4.6 contains the scalar flux for the current time step, which is being computed. The initial estimate of this term is the scalar flux computed during the previous fixed source iteration (Algorithm 3). The source term for the Theta method can be computed by Algorithm 6.

Algorithm 6 FSR source update for Theta method in OpenMOC

for all $i \in I$ **do** # Loop over mesh cells
  for all $r \in i$ **do** # Loop over FSRs in cell
    for all $g \in G$ **do** # Loop over energy groups
      $Q_{r,g}^{(n+1)}(t^s) \leftarrow (1 - \beta) \frac{\chi_{r,g}}{4\pi k_{eff}^0} \nu \Sigma_{r,g}^F \Phi_{r,g}^{(n)}(t^s)$
      for all $g' \in G$ **do** # Loop over energy groups
        $Q_{r,g}^{(n+1)}(t^s) \leftarrow Q_{r,g}^{(n+1)} + \frac{1}{4\pi} \Sigma_{r,g' \rightarrow g}^S \Phi_{r,g'}^{(n)}(t^s)$
      end for
      for all $m \in M$ **do** # Loop over delayed groups
        $Q_{r,g}^{(n+1)}(t^s) \leftarrow Q_{r,g}^{(n+1)} + \frac{1}{4\pi} \chi_{r,g} \lambda_m C_{m,r}(t^s)$
      end for
      $Q_{r,g}^{(n+1)}(t^s) \leftarrow Q_{r,g}^{(n+1)}(t^s) - \frac{1}{4\pi v_g \Delta t_s} \left(\Phi_{r,g}^{(n)}(t^s) - \Phi_{r,g}^{(n)}(t^{s-1}) \right)$
    end for
  end for
end for

The fixed source iteration algorithm for the Theta method is identical to the fixed source iteration algorithm for the steady-state MOC problem (Algorithm 3) where

the cross sections at the current time step, t^s, are used.

4.1.3 The Multigrid Amplitude Function Method

In the MAF method, the flux derivative is broken up using the product rule and described as:

$$\frac{\partial \Phi_g(r,t)}{\partial t} = \omega_g^{i,j}(t)\Phi_g(r,t) + \phi_g^{i,j}(t)\frac{\partial \varphi_g(r,t)}{\partial t} \tag{4.7}$$

where:

$$\omega_g^{i,j}(t) = \frac{1}{\phi_g^{i,j}(t)}\frac{\partial \phi_g^{i,j}(t)}{\partial t} \tag{4.8}$$

Inserting $\omega_g^{i,j}(t)$ into the MOC source term we get:

$$\tilde{Q}_g(r,t^s) = Q_g(r,t^s) - \omega_g^{i,j}(t^s)\Phi_g(r,t^s) - \phi_g^{i,j}(t^s)\frac{\partial \varphi_g(r,t^s)}{\partial t} \tag{4.9}$$

Applying the backwards difference approximation on the temporal derivative of the shape function and using algebra to convert the shape function terms into scalar fluxes, we get:

$$\tilde{Q}_g(r,t^s) = Q_g(r,t^s) - \frac{\Phi_g(r,t^s)}{4\pi\upsilon_g}\left(\omega_g^{i,j}(t^s) + \frac{1}{\Delta t_s}\right) + \frac{\Phi_g(r,t^{s-1})}{4\pi\upsilon_g\Delta t_s}\frac{\phi_g^{i,j}(t^s)}{\phi_g^{i,j}(t^{s-1})} \tag{4.10}$$

As with the Theta method, the MAF method contains the scalar flux for the current time step. Additionally, the MAF method source contains the coarse mesh flux and derivative at the current time step. There are two main approximations that could be used here. In the first choice, the coarse mesh flux computed from the most recent time-dependent CMFD solve can be used along with its derivative. With this choice, the coarse mesh flux and derivative are unchanged during the MOC solve. The second choice is to extend the first choice by updating the amplitude function and derivative after each MOC fixed source iteration with the coarse mesh flux after each CMFD update. In OpenMOC we have chosen to implement the first choice, but acknowledge that other valid approximations exist and possibly present

different behavior in solving the time-dependent MOC problem. In order to denote this approximation in the source term, we will rewrite Equation 4.11 with the coarse mesh flux and derivative terms at the current time step denoted with a $*$ on the time variable to indicate that they are constant during the MOC solve. As we will see later in subsection 4.1.4, this notation will be important in helping avoid confusion when we perform CMFD acceleration.

$$\tilde{Q}_g(r,t^s) = Q_g(r,t^s) - \frac{\Phi_g(r,t^s)}{4\pi v_g}\left(\omega_g^{i,j}(t^{s,*}) + \frac{1}{\Delta t_s}\right) + \frac{\Phi_g(r,t^{s-1})}{4\pi v_g \Delta t_s}\frac{\phi_g^{i,j}(t^{s,*})}{\phi_g^{i,j}(t^{s-1})} \tag{4.11}$$

The source term for the MAF method can be represented by Algorithm 7.

Algorithm 7 FSR source update for MAF method in OpenMOC

for all $i \in I$ **do** # Loop over mesh cells
  for all $r \in i$ **do** # Loop over FSRs in cell
    for all $g \in G$ **do** # Loop over energy groups
      $Q_{r,g}^{(n+1)}(t^s) \leftarrow (1-\beta)\frac{\chi_{r,g}}{4\pi k_{eff}^0}\nu\Sigma_{r,g}^F\Phi_{r,g}^{(n)}(t^s)$
      for all $g' \in G$ **do** # Loop over energy groups
        $Q_{r,g}^{(n+1)}(t^s) \leftarrow Q_{r,g}^{(n+1)}(t^s) + \frac{1}{4\pi}\Sigma_{r,g'\rightarrow g}^S\Phi_{r,g'}^{(n)}(t^s)$
      end for
      for all $m \in M$ **do** # Loop over delayed groups
        $Q_{r,g}^{(n+1)}(t^s) \leftarrow Q_{r,g}^{(n+1)}(t^s) + \frac{1}{4\pi}\chi_{r,g}\lambda_m C_{m,r}(t^s)$
      end for
      $Q_{r,g}^{(n+1)}(t^s) \leftarrow Q_{r,g}^{(n+1)}(t^s) - \frac{\Phi_g(r,t^s)}{4\pi v_g}\left(\omega_g^{i,j}(t^{s,*}) + \frac{1}{\Delta t_s}\right) + \frac{\Phi_g(r,t^{s-1})}{4\pi v_g \Delta t_s}\frac{\phi_g^{i,j}(t^{s,*})}{\phi_g^{i,j}(t^{s-1})}$
    end for
  end for
end for

It is important to note one additional point in solving the MOC problem with the Theta and MAF methods. At each shape function solve, a fixed source MOC problem is being solved. The transport equation has fission and scattering source components that will change during each iteration of the MOC solve while the components of the source describing the delayed neutron precursor concentrations and shape at the previous time step will be fixed. As will be shown in chapter 5, this important feature results in a reduction in the run time required for a particular MOC solve with the Theta and MAF methods as compared with the Adiabatic method.

In summary, the Adiabatic, Theta, and MAF methods mainly differ in their representation of the source term. This results in small, but significant changes in the

formulation of the algorithms used to solve the transient MOC problem with each method. For comparison, the source terms have been included again in Table 4.2.

Table 4.2. MOC source terms for transient methods

Method	$\tilde{Q}_g(r, t^{s+1})$
Adiabatic	$\frac{1}{4\pi}\left(\frac{\chi_{r,g}}{k_{eff}}\sum_{g\prime=1}^{G}\nu\Sigma_{g\prime}^{F}\Phi_{g\prime}(r,t^s)+\sum_{g\prime=1}^{G}\Sigma_{g\prime\to g}^{S}(r,t^s)\Phi_{g\prime}(r,t^s)\right)$
Theta (FI)	$Q_g(r,t^s)-\frac{1}{4\pi v_g\Delta t_s}\left(\Phi_g(r,t^s)-\Phi_g(r,t^{s-1})\right)$
MAF	$Q_g(r,t^s)-\frac{\Phi_g(r,t^s)}{4\pi v_g}\left(\omega_g^{i,j}(t^{s,*})+\frac{1}{\Delta t_s}\right)+\frac{\Phi_g(r,t^{s-1})}{4\pi v_g\Delta t_s}\frac{\phi_g^{i,j}(t^{s,*})}{\phi_g^{i,j}(t^{s-1})}$

4.1.4 CMFD acceleration for time-dependent MOC

As shown in the previous two subsections, the Theta and MAF methods change the parameters in the MOC algorithm. More specifically, these methods have additional source terms that approximate the temporal derivative of the neutron flux. In addition to the parameters in the MOC algorithm being changed, the parameters in the CMFD acceleration algorithms will be similarly modified. In section 2.2 the CMFD method was derived by starting with the steady state multi-group neutron diffusion equation (Equation 2.20). In order to use CMFD to accelerate a time-dependent MOC method, the time-dependent multi-group neutron diffusion equation (Equation 4.12) must be used. In this section we present the slight differences in performing time-dependent CMFD acceleration on a time-dependent MOC problem.

The time-dependent multi-group neutron diffusion equation is:

$$\frac{1}{v_g}\frac{\partial\phi_g(x,y,t)}{\partial t} = \nabla\cdot D_g(x,y,t)\nabla\phi_g(x,y,t) - \Sigma_g^R(x,y,t)\phi_g(x,y,t) + \mathcal{Q}_g(x,y,t),$$

$$\mathcal{Q}_g(x,y,t) = \frac{\chi_g(x,y,t)}{k_{eff}^0}(1-\beta)\sum_{g'=1}^{G}\nu\Sigma_{g'}^F(x,y,t)\phi_{g'}(x,y,t) + \sum_{\substack{g'=1\\g'\neq g}}^{G}\Sigma_{g'\to g}^S(x,y,t)\phi_{g'}(x,y,t) + \sum_{m=1}^{M}\chi_g(x,y,t)\lambda_m\mathcal{C}_m(x,y,t) \tag{4.12}$$

where the terms are the same as presented in Table 2.1 and Table 4.1. The delayed neutron precursors in Equation 4.12 are condensed from the fine to the coarse mesh in a similar procedure as that performed for the cross sections. Equation 4.13 shows the procedure for condensing the precursor for a coarse mesh cell, (i,j).

$$\mathcal{C}_m^{i,j} = \frac{\sum_{r\in(i,j)} C_{m,r}A_r}{\sum_{r\in(i,j)} A_r} \tag{4.13}$$

As was the case with the MOC algorithms, the CMFD algorithms for the steady state and time dependent problems are very similar. The main differences being the separation of the fission source into prompt and delayed neutrons and the addition of the flux temporal derivative. The time-dependent multi-group neutron diffusion equation can be integrated over the coarse mesh cells and the finite difference approximation can be applied to the streaming term using the same procedure as for the steady state CMFD method (section 2.2). Applying the integration and finite differencing procedure results in Equation 4.14.

$$\begin{aligned}
\frac{1}{v_g}\frac{d\phi_g^{i,j}(t)}{dt}\Delta x^{i,j}\Delta y^{i,j} &= \mathcal{Q}_g^{i,j}(t)\Delta x^{i,j}\Delta y^{i,j} - \Sigma_g^{R,i,j}(t)\phi_g^{i,j}(t)\Delta x^{i,j}\Delta y^{i,j} \\
&- \Delta y^{i,j}(\hat{D}_{\mathbf{g}}^{i-\frac{1}{2},j}[\phi_{\mathbf{g}}^{i,j} - \phi_{\mathbf{g}}^{i-1,j}] + \tilde{D}_{\mathbf{g}}^{i-\frac{1}{2},j}[\phi_{\mathbf{g}}^{i,j} + \phi_{\mathbf{g}}^{i-1,j}]) \\
&+ \Delta y^{i,j}(\hat{D}_{\mathbf{g}}^{i+\frac{1}{2},j}[\phi_{\mathbf{g}}^{i+1,j} - \phi_{\mathbf{g}}^{i,j}] + \tilde{D}_{\mathbf{g}}^{i+\frac{1}{2},j}[\phi_{\mathbf{g}}^{i+1,j} - \phi_{\mathbf{g}}^{i,j}]) \\
&- \Delta x^{i,j}(\hat{D}_{\mathbf{g}}^{i,j-\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j} - \phi_{\mathbf{g}}^{i,j-1}] + \tilde{D}_{\mathbf{g}}^{i,j-\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j} + \phi_{\mathbf{g}}^{i,j-1}]) \\
&+ \Delta x^{i,j}(\hat{D}_{\mathbf{g}}^{i,j+\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j+1} - \phi_{\mathbf{g}}^{i,j}] + \tilde{D}_{\mathbf{g}}^{i,j+\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j+1} + \phi_{\mathbf{g}}^{i,j}]),
\end{aligned} \tag{4.14}$$

$$\mathcal{Q}_g^{i,j}(t) = \frac{\chi_g^{i,j}(t)}{k_{eff}^0}(1-\beta)\sum_{g\prime=1}^{G}\nu\Sigma_{g\prime}^{F,i,j}(t)\phi_{g\prime}^{i,j}(t) + \sum_{\substack{g\prime=1 \\ g\prime\neq g}}^{G}\Sigma_{g\prime\to g}^{S,i,j}(t)\phi_{g\prime}^{i,j}(t) + \sum_{m=1}^{M}\chi_g^{i,j}(t)\lambda_m C_m^{i,j}(t)$$

We now turn our attention to the treatment of the flux temporal derivative term. For the Theta method, we apply the same backwards difference approximation to the flux temporal derivative as was used in subsection 4.1.2:

$$\frac{d\phi_g^{i,j}(t)}{dt} = \frac{\phi_g^{i,j}(t^s) - \phi_g^{i,j}(t^{s-1})}{\Delta t_s} \tag{4.15}$$

Inserting Equation 4.15 into Equation 4.14 results in the final form of the CMFD equations used to accelerate the Theta method MOC problem.

$$\begin{aligned}
\frac{\phi_g^{i,j}(t^{s-1})}{v_g\Delta t_s}\Delta x^{i,j}\Delta y^{i,j} + \sum_{m=1}^{M}\chi_g^{i,j}\lambda_m C_m^{i,j}(t^s)\Delta x^{i,j}\Delta y^{i,j} &= \frac{\phi_g^{i,j}(t^s)}{v_g\Delta t_s}\Delta x^{i,j}\Delta y^{i,j} \\
&+\Sigma_g^{R,i,j}(t^s)\phi_g^{i,j}(t^s)\Delta x^{i,j}\Delta y^{i,j} - \tilde{\mathcal{Q}}_g^{i,j}(t^s)\Delta x^{i,j}\Delta y^{i,j} \\
&+\Delta y^{i,j}(\hat{D}_{\mathbf{g}}^{i-\frac{1}{2},j}[\phi_{\mathbf{g}}^{i,j} - \phi_{\mathbf{g}}^{i-1,j}] + \tilde{D}_{\mathbf{g}}^{i-\frac{1}{2},j}[\phi_{\mathbf{g}}^{i,j} + \phi_{\mathbf{g}}^{i-1,j}]) \\
&-\Delta y^{i,j}(\hat{D}_{\mathbf{g}}^{i+\frac{1}{2},j}[\phi_{\mathbf{g}}^{i+1,j} - \phi_{\mathbf{g}}^{i,j}] + \tilde{D}_{\mathbf{g}}^{i+\frac{1}{2},j}[\phi_{\mathbf{g}}^{i+1,j} - \phi_{\mathbf{g}}^{i,j}]) \\
&+\Delta x^{i,j}(\hat{D}_{\mathbf{g}}^{i,j-\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j} - \phi_{\mathbf{g}}^{i,j-1}] + \tilde{D}_{\mathbf{g}}^{i,j-\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j} + \phi_{\mathbf{g}}^{i,j-1}]) \\
&-\Delta x^{i,j}(\hat{D}_{\mathbf{g}}^{i,j+\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j+1} - \phi_{\mathbf{g}}^{i,j}] + \tilde{D}_{\mathbf{g}}^{i,j+\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j+1} + \phi_{\mathbf{g}}^{i,j}])
\end{aligned} \tag{4.16}$$

Where the source has been modified to only include inscattering and prompt neutron generation:

$$\tilde{\mathcal{Q}}_g^{i,j}(t^s) = \frac{\chi_g^{i,j}}{k_{eff}^0}(1-\beta)\sum_{g\prime=1}^{G}\nu\Sigma_{g\prime}^{F,i,j}\phi_{g\prime}^{i,j}(t^s) + \sum_{\substack{g\prime=1 \\ g\prime\neq g}}^{G}\Sigma_{g\prime\to g}^{S,i,j}\phi_{g\prime}^{i,j}(t^s) \tag{4.17}$$

For the MAF method, we return to Equation 4.18 where the scalar flux was broken up using the product rule:

$$\frac{\partial \Phi_g(r,t)}{\partial t} = \omega_g^{i,j}(t)\Phi_g(r,t) + \phi_g^{i,j}(t)\frac{\partial \varphi_g(r,t)}{\partial t} \tag{4.18}$$

Applying the backwards difference approximation on the temporal derivative of the shape function and using algebra to convert the shape function terms into scalar fluxes, we get:

$$\frac{\partial \Phi_g(r,t^s)}{\partial t} = \frac{\Phi_g(r,t^s)}{4\pi \upsilon_g}\left(\omega_g^{i,j}(t^{s,*}) + \frac{1}{\Delta t_s}\right) - \frac{\Phi_g(r,t^{s-1})}{4\pi \upsilon_g \Delta t_s}\frac{\phi_g^{i,j}(t^{s,*})}{\phi_g^{i,j}(t^{s-1})} \tag{4.19}$$

Condensing the fine mesh scalar flux terms we get an expression for the coarse mesh scalar flux derivative:

$$\frac{\partial \phi_g^{i,j}(t^s)}{\partial t} = \frac{\phi_g^{i,j}(t^s)}{\upsilon_g}\left(\omega_g^{i,j}(t^{s,*}) + \frac{1}{\Delta t_s}\right) - \frac{\phi_g^{i,j}(t^{s,*})}{\upsilon_g \Delta t_s} \tag{4.20}$$

It is important to remember that in subsection 4.1.3 we assumed that the coarse mesh flux in the second term on the right hand side of Equation 4.20 and the coarse mesh flux derivative in Equation 4.20 were taken to be constant during the MOC solve using the values from the most recent time-dependent CMFD solve. This is indicated in each of those terms with a $*$ on the time variable. By specifying these coarse mesh flux and flux derivative terms to be constant during an MOC solve they effectively become fixed sources. This characteristic makes the CMFD problem linear instead of being an eigenvalue problem. Inserting the coarse mesh flux derivative into the time-dependent CMFD equations, we get the final form of the CMFD equations for accelerating the MAF specification of time-dependent MOC:

$$\frac{\phi_g^{i,j}(t^{s,*})}{v_g \Delta t_s}\Delta x^{i,j}\Delta y^{i,j} + \sum_{m=1}^{M} \chi_g^{i,j}\lambda_m \mathcal{C}_m^{i,j}(t^s)\Delta x^{i,j}\Delta y^{i,j} = \frac{\phi_g^{i,j}(t^s)}{v_g}\left(\omega_g^{i,j}(t^{s,*}) + \frac{1}{\Delta t_s}\right)\Delta x^{i,j}\Delta y^{i,j}$$
$$+\Sigma_g^{R,i,j}(t^s)\phi_g^{i,j}(t^s)\Delta x^{i,j}\Delta y^{i,j} - \tilde{\mathcal{Q}}_g^{i,j}(t^s)\Delta x^{i,j}\Delta y^{i,j}$$
$$+\Delta y^{i,j}(\hat{D}_{\mathbf{g}}^{i-\frac{1}{2},j}[\phi_{\mathbf{g}}^{i,j} - \phi_{\mathbf{g}}^{i-1,j}] + \tilde{D}_{\mathbf{g}}^{i-\frac{1}{2},j}[\phi_{\mathbf{g}}^{i,j} + \phi_{\mathbf{g}}^{i-1,j}])$$
$$-\Delta y^{i,j}(\hat{D}_{\mathbf{g}}^{i+\frac{1}{2},j}[\phi_{\mathbf{g}}^{i+1,j} - \phi_{\mathbf{g}}^{i,j}] + \tilde{D}_{\mathbf{g}}^{i+\frac{1}{2},j}[\phi_{\mathbf{g}}^{i+1,j} - \phi_{\mathbf{g}}^{i,j}])$$
$$+\Delta x^{i,j}(\hat{D}_{\mathbf{g}}^{i,j-\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j} - \phi_{\mathbf{g}}^{i,j-1}] + \tilde{D}_{\mathbf{g}}^{i,j-\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j} + \phi_{\mathbf{g}}^{i,j-1}])$$
$$-\Delta x^{i,j}(\hat{D}_{\mathbf{g}}^{i,j+\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j+1} - \phi_{\mathbf{g}}^{i,j}] + \tilde{D}_{\mathbf{g}}^{i,j+\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j+1} + \phi_{\mathbf{g}}^{i,j}]) \tag{4.21}$$

It is important to notice that Equation 4.16 and Equation 4.21 have the same form, but with slight differences on the fixed source term on the left hand side of the equations and the first term multiplies the flux at the current time step on the right hand side of the equations. This allows us to implement one algorithm that can be used to solve the time-dependent CMFD problem for both methods. We can write the time-dependent CMFD equations in matrix form as:

$$A\phi = b \tag{4.22}$$

This problem can then be solved with Algorithm 5. After solving for the flux, the procedure for updating the fine mesh scalar flux is the same as the method presented in subsection 2.2.7 for steady-state MOC.

4.2 Time integration of delayed neutron precursors

An important part in modeling a nuclear reactor where the power level or shape shifts rapidly (1e-4 to 1e-3 seconds) is computing of the delayed neutron precursor concentrations. During a transient, the precursor concentration shape and amplitude will change in a similar manner to the neutron flux. In this thesis, we chose to solve

the delayed neutron precursor equations independently of the shape function solve. In Equation 4.2 we presented the delayed neutron precursor equation. To solve for the delayed neutron precursors, we directly integrate this equation to yield:

$$C_m(r,t^s) = e^{-\lambda_m \Delta t_s} C_m(r,t^{s-1}) + e^{-\lambda_m \Delta t_s} \int_{t^{s-1}}^{t^s} e^{\lambda(t-t^{s-1})} \frac{\beta_m}{k^0_{eff}} \sum_{g=1}^{G} \nu\Sigma^F_g(r,t)\Phi_g(r,t)dt \tag{4.23}$$

We now assume that the fission source varies linearly over a time step. Equation 4.23 integrates out to:

$$\begin{aligned} C_m(r,t^s) = k_{1,m} C_m(r,t^{s-1}) + k_{2,m} \frac{\beta_m}{\lambda_m k^0_{eff}} \sum_{g=1}^{G} \nu\Sigma^F_g(r,t^s)\Phi_g(r,t^s) \\ - k_{3,m} \frac{\beta_m}{\lambda_m k^0_{eff}} \sum_{g=1}^{G} \nu\Sigma^F_g(r,t^{s-1})\Phi_g(r,t^{s-1}) \end{aligned} \tag{4.24}$$

where:

$$k_{1,m} \equiv e^{-\lambda_m \Delta t_s} \tag{4.25a}$$

$$k_{2,m} \equiv 1 - \frac{1 - e^{-\lambda_m \Delta t_s}}{\lambda_m \Delta t_s} \tag{4.25b}$$

$$k_{3,m} \equiv e^{-\lambda_m \Delta t_s} - \frac{1 - e^{-\lambda_m \Delta t_s}}{\lambda_m \Delta t_s} \tag{4.25c}$$

Equation 4.24 allows us to solve for the delayed neutron precursors on the fine mesh at any point in time. As seen in subsection 4.1.4, the delayed neutron precursors on the fine mesh can be condensed to precursor concentrations on the coarse mesh using Equation 4.13.

4.3 Time dependent CMFD equations

So far we have only described how to solve for the shape function during a transient solve. Now we will change our focus to solving for the amplitude function in between shape function solves using a fine time step size. In section 4.1 we derived three methods to compute the shape functions. In this section we will derive a similar form of the time dependent coarse mesh finite difference (TCMFD) equations as that used to accelerate time-dependent MOC. It is important that we make clear that there are differences between the TCMFD method we derived in section 4.1 and the method derived in this section. Notably, the TCMFD method presented in this section will make use of approximations for the nonlinear diffusion coefficients instead of exact values computed from tallied MOC fluxes and will be solved on a finer time step. To derive a procedure for solving the TCMFD equations on our coarse mesh, we begin with the multigroup TCMFD balance equation integrated over a cell:

$$\begin{aligned}
\frac{1}{v_g}\frac{d\phi_g^{i,j}(t)}{dt}\Delta x^{i,j}\Delta y^{i,j} &= \mathcal{Q}_g^{i,j}(t)\Delta x^{i,j}\Delta y^{i,j} - \Sigma_g^{R,i,j}(t)\phi_g^{i,j}\Delta x^{i,j}\Delta y^{i,j} \\
&- \Delta y^{i,j}(\hat{D}_{\mathbf{g}}^{i-\frac{1}{2},j}[\phi_{\mathbf{g}}^{i,j} - \phi_{\mathbf{g}}^{i-1,j}] + \tilde{D}_{\mathbf{g}}^{i-\frac{1}{2},j}[\phi_{\mathbf{g}}^{i,j} + \phi_{\mathbf{g}}^{i-1,j}]) \\
&+ \Delta y^{i,j}(\hat{D}_{\mathbf{g}}^{i+\frac{1}{2},j}[\phi_{\mathbf{g}}^{i+1,j} - \phi_{\mathbf{g}}^{i,j}] + \tilde{D}_{\mathbf{g}}^{i+\frac{1}{2},j}[\phi_{\mathbf{g}}^{i+1,j} - \phi_{\mathbf{g}}^{i,j}]) \\
&- \Delta x^{i,j}(\hat{D}_{\mathbf{g}}^{i,j-\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j} - \phi_{\mathbf{g}}^{i,j-1}] + \tilde{D}_{\mathbf{g}}^{i,j-\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j} + \phi_{\mathbf{g}}^{i,j-1}]) \\
&+ \Delta x^{i,j}(\hat{D}_{\mathbf{g}}^{i,j+\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j+1} - \phi_{\mathbf{g}}^{i,j}] + \tilde{D}_{\mathbf{g}}^{i,j+\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j+1} + \phi_{\mathbf{g}}^{i,j}]),
\end{aligned} \tag{4.26}$$

$$\mathcal{Q}_g^{i,j}(t) = \frac{\chi_g^{i,j}(t)}{k_{eff}^0}(1-\beta)\sum_{g\prime=1}^{G}\nu\Sigma_{g\prime}^{F,i,j}(t)\phi_{g\prime}^{i,j}(t) + \sum_{\substack{g\prime=1 \\ g\prime\neq g}}^{G}\Sigma_{g\prime\to g}^{S,i,j}(t)\phi_{g\prime}^{i,j}(t) + \sum_{m=1}^{M}\chi_g^{i,j}(t)\lambda_m C_m^{i,j}(t)$$

The neutron current expressions are defined by Equation 2.36. We now perform a backwards difference approximation on the cell flux temporal derivative over a fine time step:

$$\frac{d\phi_g^{i,j}(t)}{dt} = \frac{\phi_g^{i,j}(t^a) - \phi_g^{i,j}(t^{a-1})}{\Delta t_a} \tag{4.27}$$

Inserting Equation 4.27 into Equation 4.26 and rearranging we get:

$$
\begin{aligned}
&\frac{\phi(t^{a-1})}{\upsilon_g \Delta t_a}\Delta x^{i,j}\Delta y^{i,j} = \mathcal{Q}_g^{i,j}(t^a)\Delta x^{i,j}\Delta y^{i,j} \\
&-\Sigma_g^{R,i,j}(t^a)\phi_g^{i,j}\Delta x^{i,j}\Delta y^{i,j} + \frac{\phi(t^a)}{\upsilon_g \Delta t_a}\Delta x^{i,j}\Delta y^{i,j} \\
&+\Delta y^{i,j}(\hat{D}_{\mathbf{g}}^{i-\frac{1}{2},j}[\phi_{\mathbf{g}}^{i,j} - \phi_{\mathbf{g}}^{i-1,j}] + \tilde{D}_{\mathbf{g}}^{i-\frac{1}{2},j}[\phi_{\mathbf{g}}^{i,j} + \phi_{\mathbf{g}}^{i-1,j}]) \\
&-\Delta y^{i,j}(\hat{D}_{\mathbf{g}}^{i+\frac{1}{2},j}[\phi_{\mathbf{g}}^{i+1,j} - \phi_{\mathbf{g}}^{i,j}] + \tilde{D}_{\mathbf{g}}^{i+\frac{1}{2},j}[\phi_{\mathbf{g}}^{i+1,j} - \phi_{\mathbf{g}}^{i,j}]) \\
&+\Delta x^{i,j}(\hat{D}_{\mathbf{g}}^{i,j-\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j} - \phi_{\mathbf{g}}^{i,j-1}] + \tilde{D}_{\mathbf{g}}^{i,j-\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j} + \phi_{\mathbf{g}}^{i,j-1}]) \\
&-\Delta x^{i,j}(\hat{D}_{\mathbf{g}}^{i,j+\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j+1} - \phi_{\mathbf{g}}^{i,j}] + \tilde{D}_{\mathbf{g}}^{i,j+\frac{1}{2}}[\phi_{\mathbf{g}}^{i,j+1} + \phi_{\mathbf{g}}^{i,j}])
\end{aligned}
\tag{4.28}
$$

The coarse mesh flux at the forward time step can then be computed using Algorithm 8.

Algorithm 8 TCMFD Equations Solution Procedure

Look up material XS	# Ensure XS are at current time and temperature values
Interpolate $\tilde{D}$	# Equation 4.30
Update precursor concentrations	# Equation 4.24
while $M\phi$ not converged **do**	
Solve TCMFD equations	# Solve Equation 4.28 using Algorithm 5
Update material temperatures	
Update precursor concentrations	# Equation 4.24
Look up material XS	# Ensure XS are at current time and temperature values
end while	

4.3.1 Interpolation of shape function and nonlinear diffusion coefficients

In between the large shape function time steps, the fine mesh flux shape and net currents between coarse mesh cells often varies. To approximate the fine mesh flux shape, it is linearly interpolated using Equation 4.29.

$$
\varphi_g(r,t^a) = (1 - \frac{t^a - t^{s-1}}{\Delta t_s})\varphi_g(r,t^{s-1}) + \frac{t^a - t^{s-1}}{\Delta t_s}\varphi_g(r,t^s) \tag{4.29}
$$

The net current across coarse mesh cell boundaries is treated indirectly through the nonlinear coupling coefficient term. After the time-dependent MOC solve, the non-

linear diffusion coefficients from the last fixed source iteration are saved. In between shape function time steps, the nonlinear diffusion coefficients are linearly interpolated using Equation 4.30.

$$\tilde{D}_g^{i\pm\frac{1}{2},j}(r,t^a) = (1-\frac{t^a-t^{s-1}}{\Delta t_s})\tilde{D}_g^{i\pm\frac{1}{2},j}(r,t^{s-1}) + \frac{t^a-t^{s-1}}{\Delta t_s}\tilde{D}_g^{i\pm\frac{1}{2},j}(r,t^s) \quad (4.30a)$$

$$\tilde{D}_g^{i,j\pm\frac{1}{2}}(r,t^a) = (1-\frac{t^a-t^{s-1}}{\Delta t_s})\tilde{D}_g^{i,j\pm\frac{1}{2}}(r,t^{s-1}) + \frac{t_a-t^{s-1}}{\Delta t_s}\tilde{D}_g^{i,j\pm\frac{1}{2}}(r,t^s) \quad (4.30b)$$

4.3.2 Convergence criteria on shape and amplitude function

When solving for the shape or amplitude function using an implicit method, a convergence relation must be adopted. For the shape function we have chosen to set the convergence based on the fine mesh fission source. The error estimate is computed as the relative root-mean-square-difference in the fission source from the previous iteration compared to the fission source in the current iteration. The expression for the fission source error is shown in Equation 4.31.

$$\text{RMSD} = \sqrt{\frac{\sum_{(i,j)}^{(I,J)}\sum_{r\in(i,j)}\sum_{g=1}^{G}\left(\frac{\nu\Sigma_g^F(r,t^s)\phi_g^{i,j}(t^s)\varphi_g(r,t^s) - \nu\Sigma_g^F(r,t^{s,*})\phi_g^{i,j}(t^{s,*})\varphi_g(r,t^s)}{\nu\Sigma_g^F(r,t^s)\phi_g^{i,j}(t^s)\varphi_g(r,t^s)}\right)^2}{R*G}} \quad (4.31)$$

where terms evaluated at $t^{s,*}$ use the values from the previous fully implicit iteration. Similar to the shape function convergence, the amplitude function convergence is based on the relative root-mean-square-difference of the fission source. The expression for the fission source error is shown in Equation 4.32.

$$\text{RMSD} = \sqrt{\frac{\sum_{(i,j)}^{(I,J)}\sum_{g=1}^{G}\left(\frac{\nu\Sigma_g^{F,i,j}(t^a)\phi_g^{i,j}(t^a) - \nu\Sigma_g^{F,i,j}(t^{a,*})\phi_g^{i,j}(t^{a,*})}{\nu\Sigma_g^{F,i,j}(t^a)\phi_g^{i,j}(t^a)}\right)^2}{I*J*G}} \quad (4.32)$$

4.4 Implementation in OpenMOC

The Adiabatic, FI Theta, and MAF methods have been implemented in the OpenMOC code. The implementation consists of one base implementation with the optional amplitude function prolongation and the requirement for each step to be solved twice to ensure convergence. In all cases the solution procedures can be broken up into two solves, the outer shape function solve and the inner amplitude function solve, and several intermediate steps to propagate delayed neutron precursors, compute variables, look up material cross sections at the current simulation time, and check for convergence. In this section we will provide an overview of how the transient methods are implemented in OpenMOC.

4.4.1 Transient MOC Procedure

The most straight-forward approach in performing the shape function solve involves updating the shape function at constant time step intervals. An illustration of the shape function solve procedure along with the inner amplitude function solve procedure is presented in Figure 4-2.

The amplitude function solve presented in Figure 4-2 utilizes the fully implicit Theta method to approximate the flux derivative between time steps. In Figure 4-2, the steps can be further explained as follows:

1. The steady-state MOC problem is solved using Algorithm 3.

2. The amplitude function is computed by condensing the fine mesh solution from the MOC solve onto the coarse mesh using Equation 2.21. The fine mesh shape is then computed using Equation 4.3. The nonlinear coupling coefficients are computed using Equation 2.37 and the tallied net currents from the last MOC fixed source iteration in **Step 1**.

3. Prolongation of flux amplitude. The TCMFD equations are solved from time t to $t + \Delta t_s$.

3a. The time for the amplitude function solve, t_a, is set to $t + \Delta t_a$.

3b. Interpolate the shape function and nonlinear diffusion coefficients between t and $t + \Delta t_s$ to time t_a using Equation 4.29 and Equation 4.30.

3c. Look up the fine mesh XS at the current temperature and positions as specified in the input or based on feedback. The fine mesh XS are then condensed down to the coarse mesh using Equation 2.21.

3d. Update the precursor concentrations on the fine mesh using Equation 4.24. Condense the precursor concentrations from the fine mesh to the coarse mesh using Equation 4.13.

3e. Solve the TCMFD equations using Algorithm 5 to get the updated amplitude function at time t_a.

3f. Update the temperatures for materials on the fine mesh.

3g. Repeat **Step 3c**.

3h. Repeat **Step 3d**.

3i. If coarse mesh source converged (Equation 4.32) at time t_a, go to **Step 3j**; else, return to **Step 3e**.

3j. If end of outer time step reached, go to **Step 3l**; else, go to **Step 3k**.

3k. Increment t_a by Δt_a. Go to **Step 3b**.

3l. Return to outer loop.

4. The material XS are adjusted to the conditions at time $t + \Delta t_s$. This involves resetting the material cross sections to be at the temperatures and positions of the system at time $t + \Delta t_s$. The time-dependent MOC problem is solved using Algorithm 3 and Algorithm 2, Algorithm 6, or Algorithm 7.

5. The amplitude function is computed by condensing the fine mesh solution from the MOC solve onto the coarse mesh using Equation 2.21. The fine mesh shape is then computed using Equation 4.3. The nonlinear coupling coefficients are

computed using Equation 2.37 and the tallied net currents from the last MOC fixed source iteration in **Step 4**.

6. The TCMFD equations are solved from time t to $t+\Delta t_s$. Repeat TCMFD solve procedure from **Step 3a** to **Step 3l**.

7. Check for the convergence of the fine mesh source at time $t+\Delta t_s$ using Equation 4.31. If converged go to **Step 8**; else, go to **Step 3**.

8. If the end of the simulation has been reached go to **Step 10**; else, go to **Step 9**.

9. Increment time t by Δt_s. Go to **Step 3**.

10. Exit outer loop and perform any postprocessing.

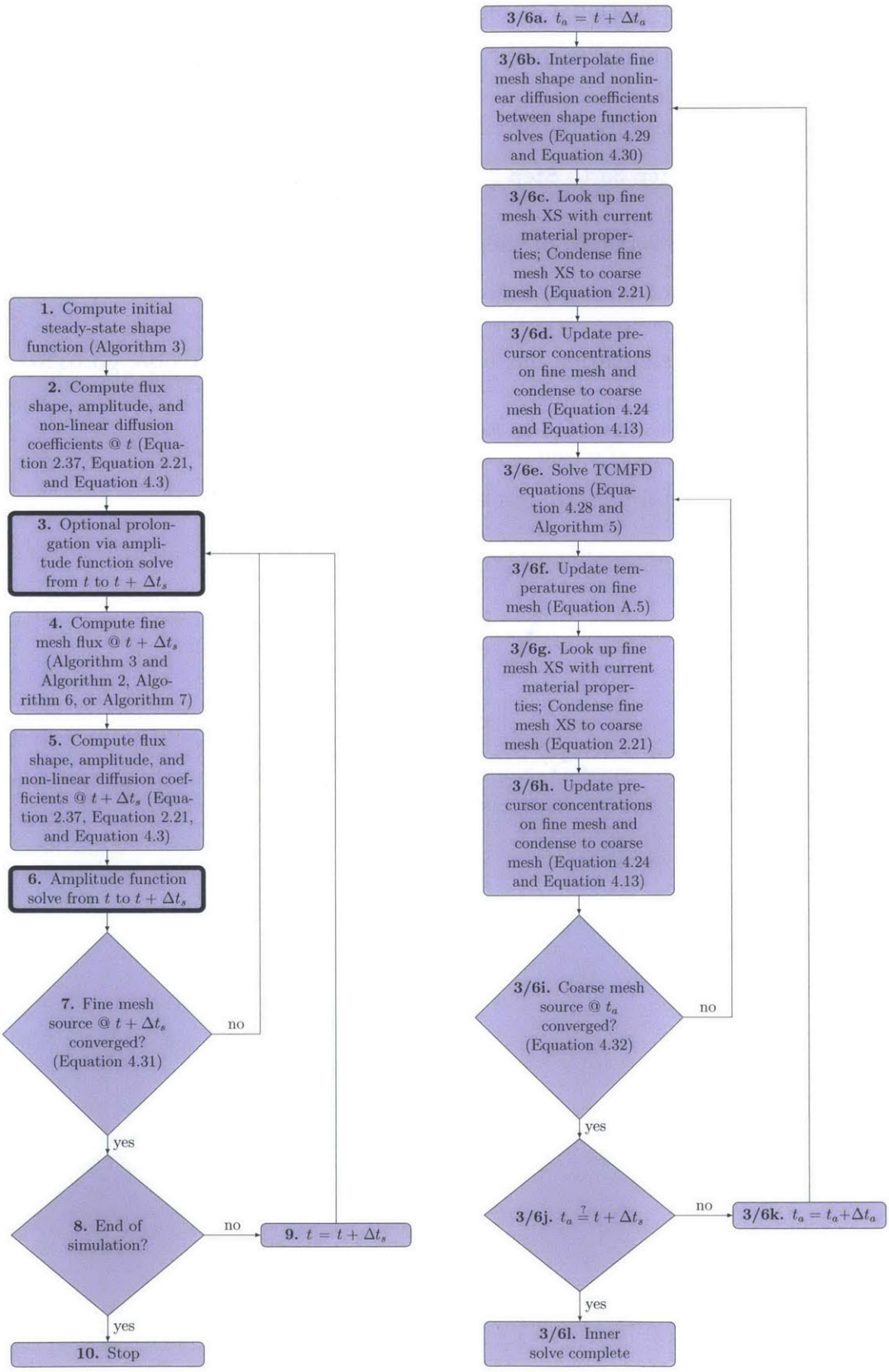

Figure 4-2. The solution procedure for the outer shape function solve (left) and inner amplitude function solve (right) for the transient MOC procedure.

Chapter 5

Transient Results

In this chapter we will use the transient methods described in chapter 4 to solve the transient versions of the LRA and C5G7 benchmarks. These two transient benchmark problems are distinct in that they allow for analysis of both homogeneous and heterogeneous representations of fuel and moderator materials within an assembly. The transients each result in a reactivity insertion of over \$1 due to withdrawal of neutron absorbing materials. The effects of material perturbations, temperature feedback, and delayed neutron precursors all influence the power profile during the transient. The three transient methods, Adiabatic, Theta, and MAF, will be compared based on their runtime performance and sensitivity to simulation parameters including shape function step size, amplitude function step size, and CMFD mesh size. To our knowledge, this analysis will yield the first fine spatial mesh and fine time step solutions to the transient transport versions of the LRA and C5G7 benchmarks. Through this analysis, we identify the key factors influencing the performance of the Adiabatic, Theta, and MAF methods in hopes of gaining intuition on how these methods will scale to large problems and stochastic neutron transport codes.

5.1 Transient LRA Benchmark

The 2D LRA benchmark is a 2-group, quarter-core BWR transient problem that is fully specified in section A.1. The transport version of the LRA benchmark was

originally published in [16] as a test problem for modern transport theory neutronics codes. The geometry is represented with assembly homogenized cross section data and a simple linear absorption cross-section ramp to model the withdrawal of a control blade in one of the regions. The simplicity of the geometry model makes the problem tractable on personal computers and small clusters and therefore a good initial benchmark for new transient neutron transport codes.

5.1.1 Static Control Blade Worth

We begin our analysis by determining the static control blade worth by computing the eigenvalue at the initial and final material states. We compute the control blade worth using two sets of FSR discretizations and track laydowns - the ultrafine FSR discretization and track laydown used to compute the reference solution and the coarse FSR discretization and track laydown identified as producing a sufficiently converged solution in subsection 3.1.1. The simulation parameters and eigenvalues for both cases and both states are presented in Table 5.1 and the initial and final power distributions for the ultrafine reference case are shown in Figure 5-1.

Table 5.1. The simulation parameters and eigenvalues for the initial and final state of the LRA transient transport benchmark

Azimuthal Angles	**Track Spacing (cm)**	**MOC mesh size (cm)**	**State**	k_{eff}	**reactivity ($)**
32	0.05	0.1171875	Initial	0.997213	-
32	0.05	0.1171875	Final	1.017080	3.0711
32	0.05	0.46875	Initial	0.996955	-
32	0.05	0.46875	Final	1.016675	3.0492

It should be noted that the initial eigenvalue for the case with 0.46875 cm MOC mesh is within 10 pcm of the eigenvalue computed in [16] with 32 azimuthal angles, a track spacing 0.04 cm, and a similar mesh size. The control blade reactivity worth was computed using Equation 5.1.

$$\rho_{worth} = \frac{k_{eff}^{final} - k_{eff}^{initial}}{k_{eff}^{initial}} \times \frac{1}{\beta} \tag{5.1}$$

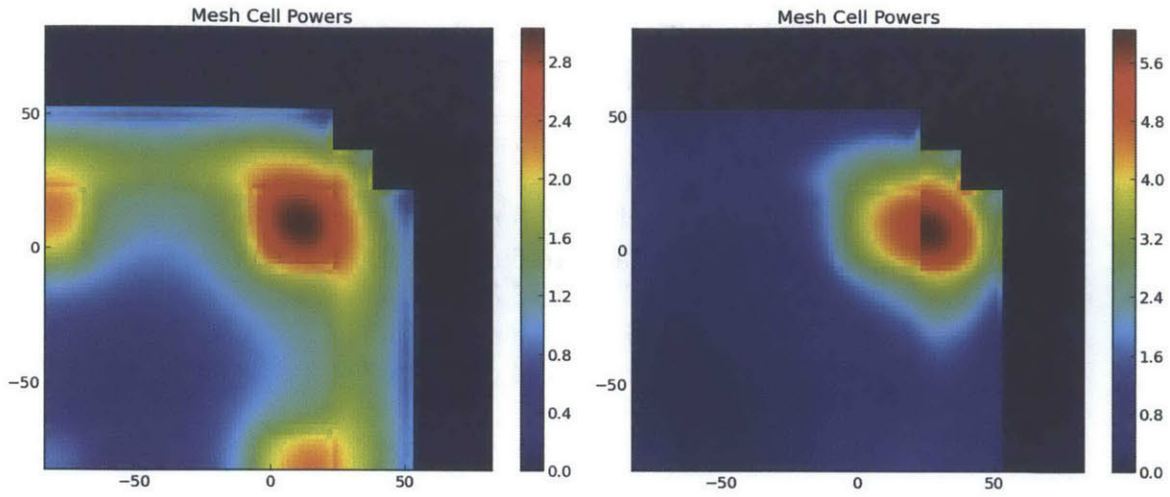

Figure 5-1. Plots of the relative power for the initial (left) and final (right) material states of the LRA benchmark.

Where β is the sum of the delayed neutron fractions, 0.006487. Since the initial state is assumed to be critical, we care much more about the reactivity comparison between the two cases rather than eigenvalues. The control blade worth for our case with a coarse MOC mesh of 0.46875 cm was found to be in good agreement with the ultrafine mesh case, suggesting the coarse MOC mesh should be sufficient to approximate the spatially-converged transient solution.

5.1.2 Reference Solution

To our knowledge there has been no previously reported spatially and temporally converged reference solution for the transport version of the LRA transient benchmark. To generate a reference solution, we performed a simulation with ultrafine shape and amplitude function time steps coupled with a tight convergence on all simulation parameters. The parameters used in all LRA simulations in this chapter are provided in Table 5.2.

The reference solution results and simulation parameters unique to the trial are provided in the Table 5.3. The average core power and peak fuel temperature profile during the transient are presented in Figure 5-2. In subsection 5.1.3 we will investigate the sensitivity of the shape function and amplitude function step sizes for each

Table 5.2. Calculation conditions

Parameter	Value
Number of polar angles	3 (using TY quadrature set)
MOC convergence criteria	1e-6 on RMSD of fission source
TCMFD convergence criteria	1e-8 on RMSD of fission source
CMFD convergence criteria	1e-6 on RMSD of fission source
GS convergence criteria	1e-8 on RMSD of fission source
amp fct convergence criteria	1e-8 on RMSD of fission source
shape fct convergence criteria	1e-4 on RMSD of fission source
CMFD damping factor	0.5
CPU architecture	2 6-core Intel Xeon processors
Threads	12
# of azimuthal angles	32
Track spacing	0.05 cm
FSR size	0.46875 cm × 0.46875 cm

transient method on the convergence of average core power profile.

Table 5.3. Calculation conditions

Parameter	Value
Shape function step size	5e-4 s
Amplitude function step size	5e-5 s
Method	MAF
CMFD mesh size	0.46875 cm
Initial k_{eff}	0.996955
Time @ 1^{st} peak	1.41545 s
Avg. fuel temperature @ 1^{st} peak	440.37 K
Avg. power density @ 1^{st} peak	5513.84 W/cc
Time @ 2^{nd} peak	2.00125 s
Avg. fuel temperature @ 2^{nd} peak	876.85 K
Avg. power density @ 2^{nd} peak	813.46 W/cc
Avg. fuel temperature @ t = 3s	1129.93 K
Avg. power density @ t = 3s	100.71 W/cc
Runtime	18.65 hours

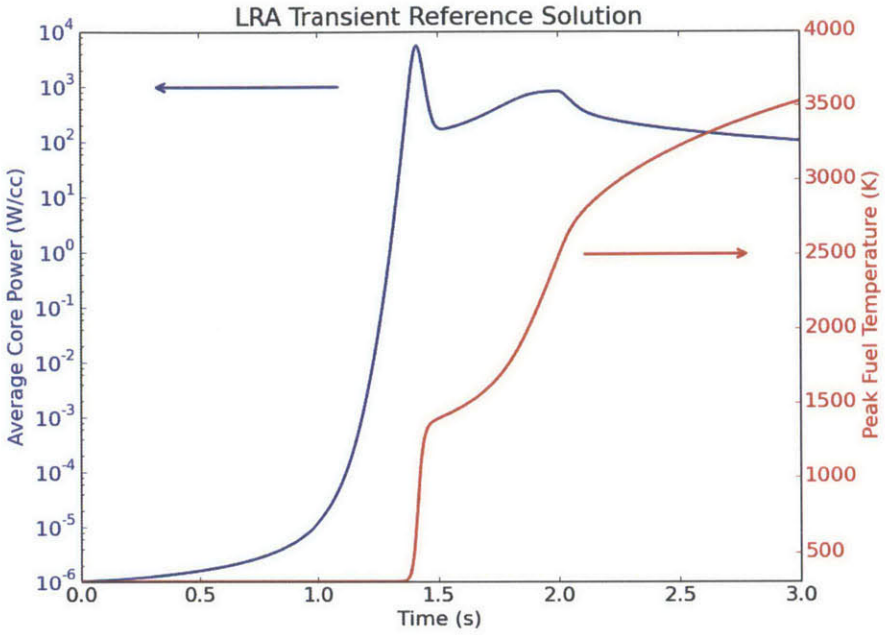

Figure 5-2. Plot of average core power (blue) and peak fuel temperature (red) during the reference LRA transient.

5.1.3 Sensitivity Analysis

In order to evaluate the convergence and performance of the MAF, Theta, and Adiabatic methods in modeling the LRA transient problem, we conducted a series of trials where simulation parameters were adjusted. The simulation parameters considered to be most influential to the solution include the shape function step size, amplitude function step size, and coarse mesh cell size. There are many other solution parameters that will influence the transient solution including the fine mesh discretization, track spacing, number of azimuthal angles, number of polar angles, and convergence of the MOC, CMFD, TCMFD, shape function, and amplitude function during all solves. We ultimately chose to only investigate the effects of the shape function size, amplitude function size, and coarse mesh discretization as these parameters are expected to have a large influence on how the methods scale to larger, more complex problems. Table 5.4 shows the values used for each parameter in their respective sensitivity analyses. In this section, we investigate each parameter individually and seek

to determine the values necessary to produce a converged solution. The reference solution for each analysis was chosen such that only one simulation parameter changed for each to the trials, allowing us to isolate the influence of this variable. Solution convergence is determined by comparing the average core power error percentage and magnitude as a function of time compared to a reference simulation for each analysis. It is important to reiterate that the reference solution for each specific sensitivity study is different; reference solutions were selected such that only one variable (Δt_s, Δt_a, or CMFD mesh size) changed during each study.

Table 5.4. Calculation conditions

Parameter	Values
Shape fct step size (Δt_s in s)	5.0e-4, 1.0e-3, 2.5e-3, 5.0e-3
Amp fct step size (Δt_a in s)	5.0e-5, 1.0e-4, 2.5e-4, 5.0e-4, 1.0e-3
Coarse mesh size (cm)	1.875, 3.75, 7.5, 15

We begin by evaluating the effect of the shape function time step size on the average core power profile convergence. In a series of trials the shape function step size, Δt_s, was adjusted with all other parameters being held constant at the values listed in Table 5.2. Reference solutions were computed for each method with a shape function time step size of 5.0e-4 seconds, an amplitude function step size of 1.0e-4 seconds, and a 0.46875 cm MOC mesh. Figure 5-3 shows the core average power density error percentage and magnitude during the transient for a series of trials with varying shape function time step sizes. Table 5.5 and Table 5.6 provide a summary of other simulation results for these trials.

Figure 5-3 shows that the convergence of the core power is highly sensitive to the shape function step size for the MAF and Theta methods. In particular, the core power experienced the greatest magnitude and percentage error during the time period t=1.3-2.1 seconds. During this period the core power is rapidly rising and doppler feedback is beginning to influence the flux and power profile in the core. It is expected that the error will be most significant during these times as the temperature rise in the fuel from one time step to the next is largest as shown in plot of the peak fuel temperature for the ultrafine mesh reference solution in Figure 5-2. The Adiabatic

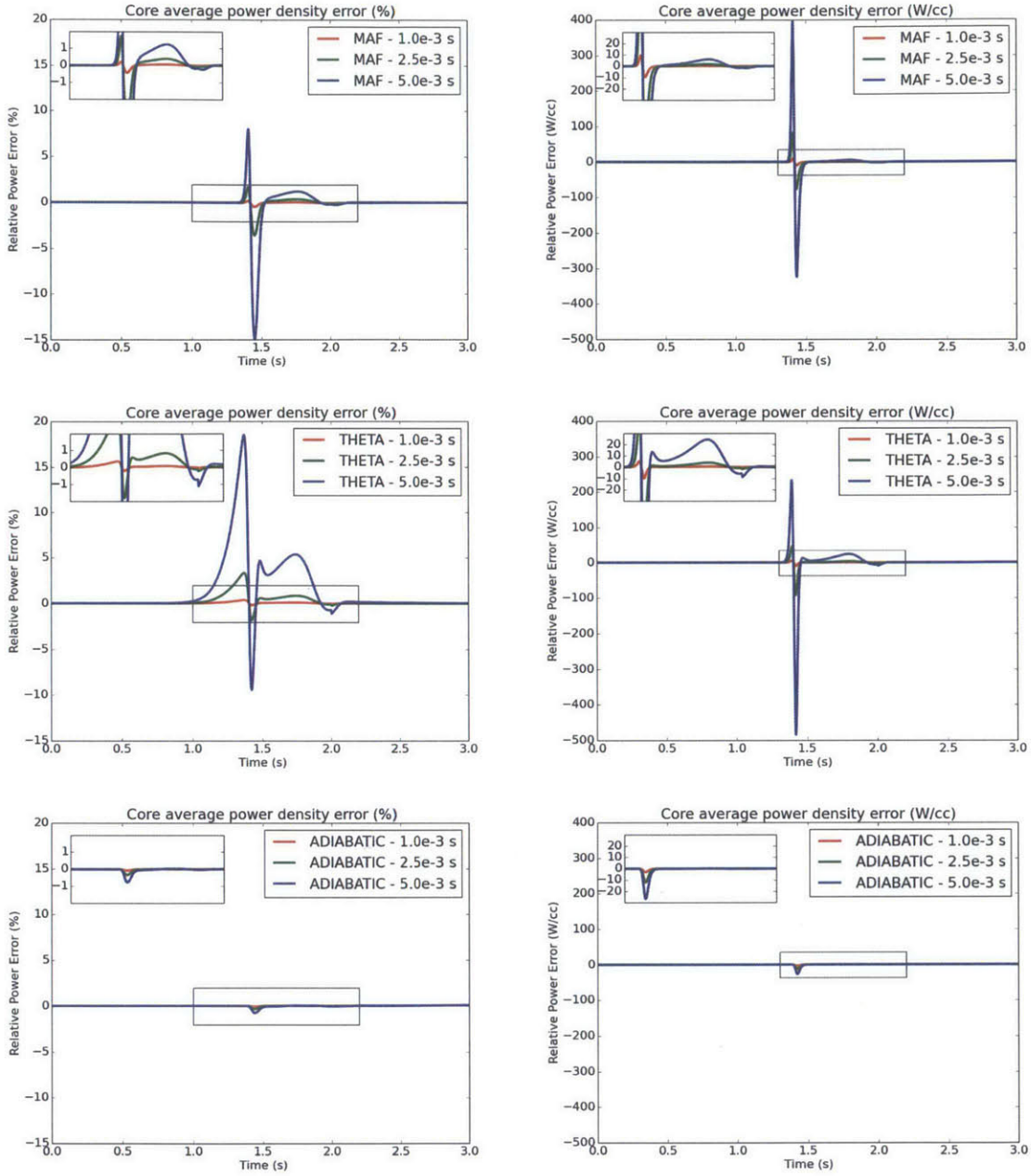

Figure 5-3. Plot of core average power density error percentage (left) and magnitude (right) for the MAF (top), Theta (middle), and Adiabatic (bottom) methods during the LRA transient simulated with different shape function step sizes.

method also experienced the maximum percentage and magnitude error during the t=1.3-2.1 second time frame, but the error was substantially smaller than that seen with the MAF and Theta methods.

Table 5.5 lists the power at various points during the transient, notably during

Table 5.5. LRA - shape function sensitivity analysis results - average core power and power peak time‡

Method	Δt_s (s)	t @ 1^{st} peak (s)	1^{st} Peak Power (W/cc)	t @ 2^{nd} peak (s)	2^{nd} Peak Power (W/cc)	Final Power (W/cc)
*MAF	0.0005	1.4151	5502.47	2.0011	813.29	100.70
MAF	0.0010	1.4151	5508.00	2.0011	813.19	100.70
MAF	0.0025	1.4148	5556.62	2.0011	812.83	100.70
MAF	0.0050	1.4139	5801.08	2.0012	811.81	100.65
*THETA	0.0005	1.4151	5498.77	2.0011	813.26	100.70
THETA	0.0010	1.4151	5490.60	2.0007	813.06	100.70
THETA	0.0025	1.4148	5422.33	2.0000	812.20	100.71
THETA	0.0050	1.4137	5087.80	2.0000	807.39	100.74
*ADIABATIC	0.0005	1.4151	5478.31	2.0011	811.71	100.84
ADIABATIC	0.0010	1.4150	5476.04	2.0011	811.67	100.88
ADIABATIC	0.0025	1.4149	5469.49	2.0011	811.55	100.90
ADIABATIC	0.0050	1.4149	5459.53	2.0011	811.38	100.90

‡ - All trials run with 12 threads on 2 6-core Intel Xeon processors
* - Denotes a reference case

Table 5.6. LRA - shape function sensitivity analysis results - average core temperature and runtime‡

Method	Δt_s (s)	1^{st} Peak Temp (K)	2^{nd} Peak Temp (K)	Final Temp (K)	Runtime (hr)	MOC solve time (hr)
*MAF	0.0005	440.00	876.78	1129.94	13.48	7.54
MAF	0.0010	440.21	876.73	1129.80	10.89	5.45
MAF	0.0025	439.82	876.59	1129.73	9.06	3.18
MAF	0.0050	440.07	876.54	1129.50	7.79	2.05
*THETA	0.0005	439.96	876.78	1129.94	13.51	7.98
THETA	0.0010	440.07	876.34	1129.90	11.49	5.59
THETA	0.0025	438.93	875.52	1129.73	8.72	3.11
THETA	0.0050	435.32	875.39	1129.50	7.71	2.06
*ADIABATIC	0.0005	439.93	876.70	1130.09	82.13	75.85
ADIABATIC	0.0010	439.91	876.57	1130.00	42.73	37.17
ADIABATIC	0.0025	439.19	876.19	1129.62	20.43	15.04
ADIABATIC	0.0050	439.10	875.53	1128.95	13.21	7.55

‡ - All trials run with 12 threads on 2 6-core Intel Xeon processors
* - Denotes a reference case

the first (and largest) power peak. The MAF and Theta methods incurred maximum errors of 5.4% and -7.5% in the average core power for the first power peak, respectively. The Adiabatic method only incurred an error of -0.4%. The average fuel temperature during the power peaks was not very sensitive to the shape function time step size for any of the methods, with all temperatures residing within ~1% of their reference values as shown in Table 5.6.

Next we evaluate the effect of the amplitude function time step size on the core average power density profile convergence. In a series of trials the amplitude function step size, Δt_a, was adjusted with all other parameters being held constant at the

values listed in Table 5.2. Reference solutions for all methods were generated with an amplitude function step size of 5.0e-5 seconds, a shape function time step size of 1.0e-3 seconds, and a 0.46875 cm MOC mesh . Figure 5-4 shows the core average power density error percentage and magnitude during the transient for a series of trials with varying amplitude function time step sizes. Table 5.7 and Table 5.8 provide a summary of other simulation results for these trials.

Table 5.7. LRA - amplitude function sensitivity analysis results - average core power and peak power times‡

Method	Δt_a (s)	t @ 1^{st} peak (s)	1^{st} Peak Power (W/cc)	t @ 2^{nd} peak (s)	2^{nd} Peak Power (W/cc)	Final Power (W/cc)
*MAF	0.00005	1.4154	5519.41	2.00125	813.36	100.71
MAF	0.00010	1.4151	5508.00	2.00110	813.19	100.70
MAF	0.00025	1.4140	5475.83	2.00100	812.61	100.71
MAF	0.00050	1.4125	5442.81	2.00100	811.44	100.71
MAF	0.00100	1.4090	5317.45	2.00100	809.21	100.72
*THETA	0.00005	1.4155	5501.03	2.00070	813.22	100.71
THETA	0.00010	1.4151	5490.60	2.00070	813.06	100.70
THETA	0.00025	1.4140	5461.28	2.00075	812.49	100.71
THETA	0.00050	1.4125	5413.06	2.00050	811.37	100.71
THETA	0.00100	1.4090	5316.70	2.00100	809.19	100.72
*ADIABATIC	0.00005	1.4153	5487.37	2.00130	811.83	100.92
ADIABATIC	0.00010	1.4150	5476.04	2.00110	811.67	100.88
ADIABATIC	0.00025	1.4140	5444.14	2.00100	811.09	100.93
ADIABATIC	0.00050	1.4125	5391.23	2.00100	809.98	100.94
ADIABATIC	0.00100	1.4090	5287.03	2.00100	807.83	100.96

‡ - All trials run with 12 threads on 2 6-core Intel Xeon processors
* - Denotes a reference case

Figure 5-4 shows that the convergence of the core power is sensitive to the amplitude function step size for all methods. The trend where all three methods yield nearly the same sensitivity is expected as the TCMFD solvers are identical for all methods. Furthermore, the amplitude function solve is what determines the core average power in between shape function time steps so therefore increasing the amplitude function time step will increase the temporal truncation error.

While Figure 5-4 shows a significant maximum error of >50% for the core average power during the first power peak for all methods, the core average power at the first power peak from Table 5.7 shows that the peak power error has a maximum of only 3.4-3.7% for each of the methods. The reason for the large peak power error in Figure 5-4 is a shift in peak power to earlier times, which is clearly seen in Table 5.7.

In a series of trials the coarse mesh cell size was adjusted with all other parameters

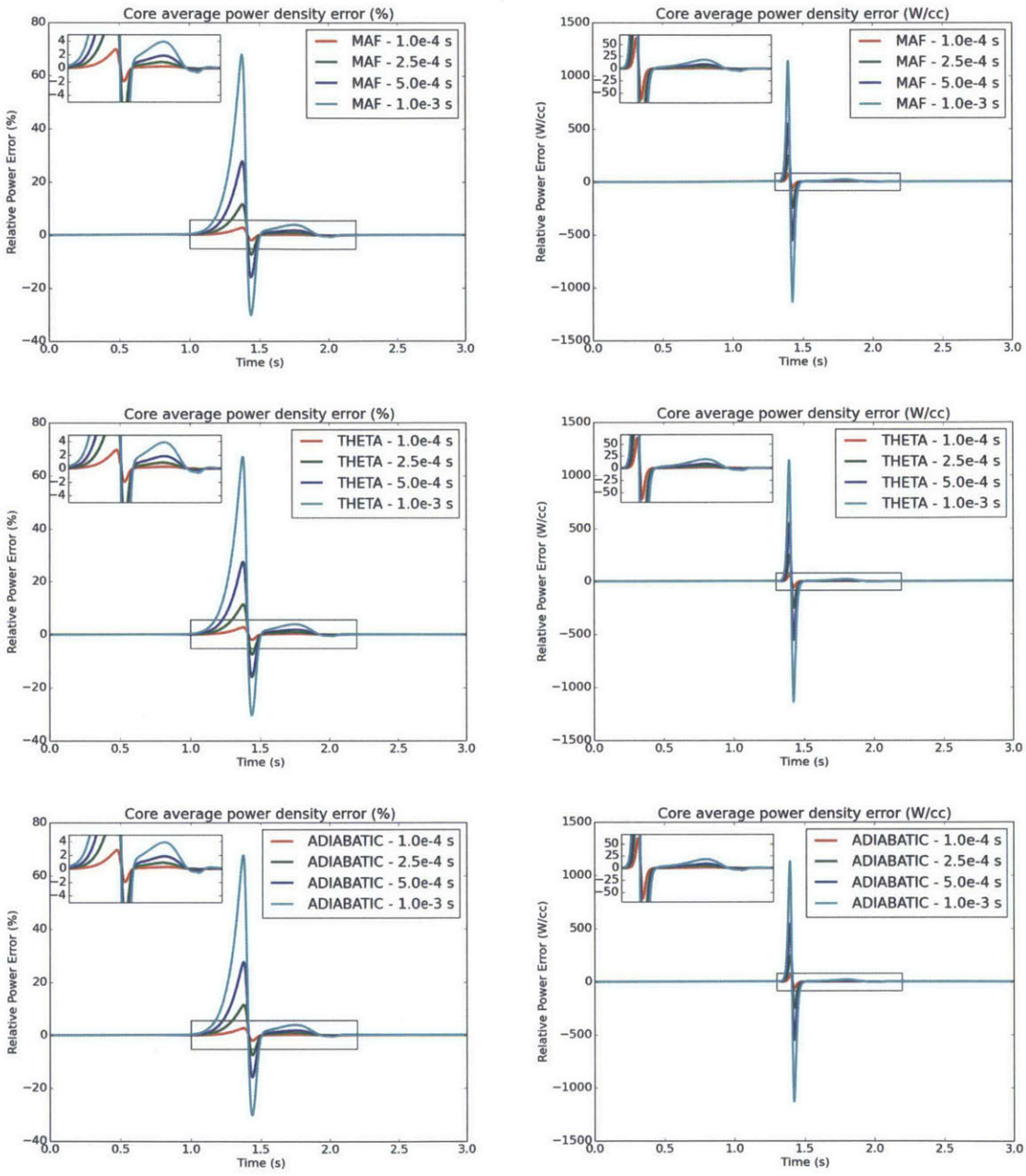

Figure 5-4. Plot of core average power density error percentage (left) and magnitude (right) for the MAF (top), Theta (middle), and Adiabatic (bottom) methods during the LRA transient simulated with different amplitude function step sizes.

being held constant at the values listed in Table 5.2. Reference solutions for each method were generated with an amplitude function step size of 1.0e-4 seconds, a shape function time step size of 1.0e-3 seconds, a coarse mesh cell size of 1.875 cm, and a 0.46875 cm MOC mesh. Figure 5-5 shows the core average power density error

Table 5.8. LRA - amplitude function sensitivity analysis results - average core temperature and runtime‡

Method	Δt_a (s)	1^{st} Peak Temp (K)	2^{nd} Peak Temp (K)	Final Temp (K)	Runtime (hr)	MOC solve time (hr)
*MAF	0.00005	440.26	876.80	1129.88	15.08	5.55
MAF	0.00010	440.21	876.73	1129.80	10.89	5.45
MAF	0.00025	438.69	876.60	1129.88	9.37	5.63
MAF	0.00050	438.36	876.65	1129.89	8.50	5.59
MAF	0.00100	434.55	876.75	1129.89	8.21	5.65
*THETA	0.00005	440.44	876.26	1129.88	15.15	5.51
THETA	0.00010	440.07	876.34	1129.90	11.49	5.59
THETA	0.00025	438.58	876.36	1129.88	9.24	5.52
THETA	0.00050	438.28	876.17	1129.88	8.55	5.59
THETA	0.00100	434.56	876.75	1129.88	8.15	5.65
*ADIABATIC	0.00005	439.96	876.69	1130.02	46.07	36.91
ADIABATIC	0.00010	439.91	876.57	1130.00	42.73	37.17
ADIABATIC	0.00025	439.05	876.44	1130.03	41.10	37.43
ADIABATIC	0.00050	438.70	876.49	1130.03	39.61	36.89
ADIABATIC	0.00100	434.91	876.58	1130.06	39.37	36.99

‡ - All trials run with 12 threads on 2 6-core Intel Xeon processors
* - Denotes a reference case

percentage and magnitude during the transient for a series of trials with varying coarse mesh cell sizes. Table 5.9 and Table 5.10 provide a summary of other simulation results for these trials.

Table 5.9. LRA - coarse mesh sensitivity analysis results - average core power and peak power times‡

Method	Mesh size (cm)	t @ 1^{st} peak (s)	1^{st} Peak Power (W/cc)	t @ 2^{nd} peak (s)	2^{nd} Peak Power (W/cc)	Final Power (W/cc)
*MAF	1.875	1.4151	5508.00	2.0011	813.19	100.70
MAF	3.75	1.4150	5535.90	2.0011	813.89	100.69
MAF	7.5	1.4151	5549.28	2.0011	814.26	100.68
MAF	15	1.4151	5562.46	2.0011	814.52	100.69
*THETA	1.875	1.4151	5490.60	2.0007	813.06	100.70
THETA	3.75	1.4151	5518.13	2.0007	812.79	100.69
THETA	7.5	1.4152	5530.96	2.0006	814.16	100.68
THETA	15	1.4150	5543.64	2.0007	814.42	100.69
*ADIABATIC	1.875	1.4150	5476.04	2.0011	811.67	100.88
ADIABATIC	3.75	1.4161	5480.40	2.0012	813.64	100.84
ADIABATIC	7.5	1.4188	5489.32	2.0012	816.72	100.82
ADIABATIC	15	1.4204	5495.18	2.0012	818.75	100.79

‡ - All trials run with 12 threads on 2 6-core Intel Xeon processors
* - Denotes a reference case

Figure 5-5 shows that the convergence of the core power is relatively insensitive to the coarse mesh discretization for the Theta and MAF methods. The Adiabatic method was sensitive to the coarse mesh size experiencing a maximum error of over 40% for the trial with a 15 cm CMFD mesh. The results presented in Table 5.9 help

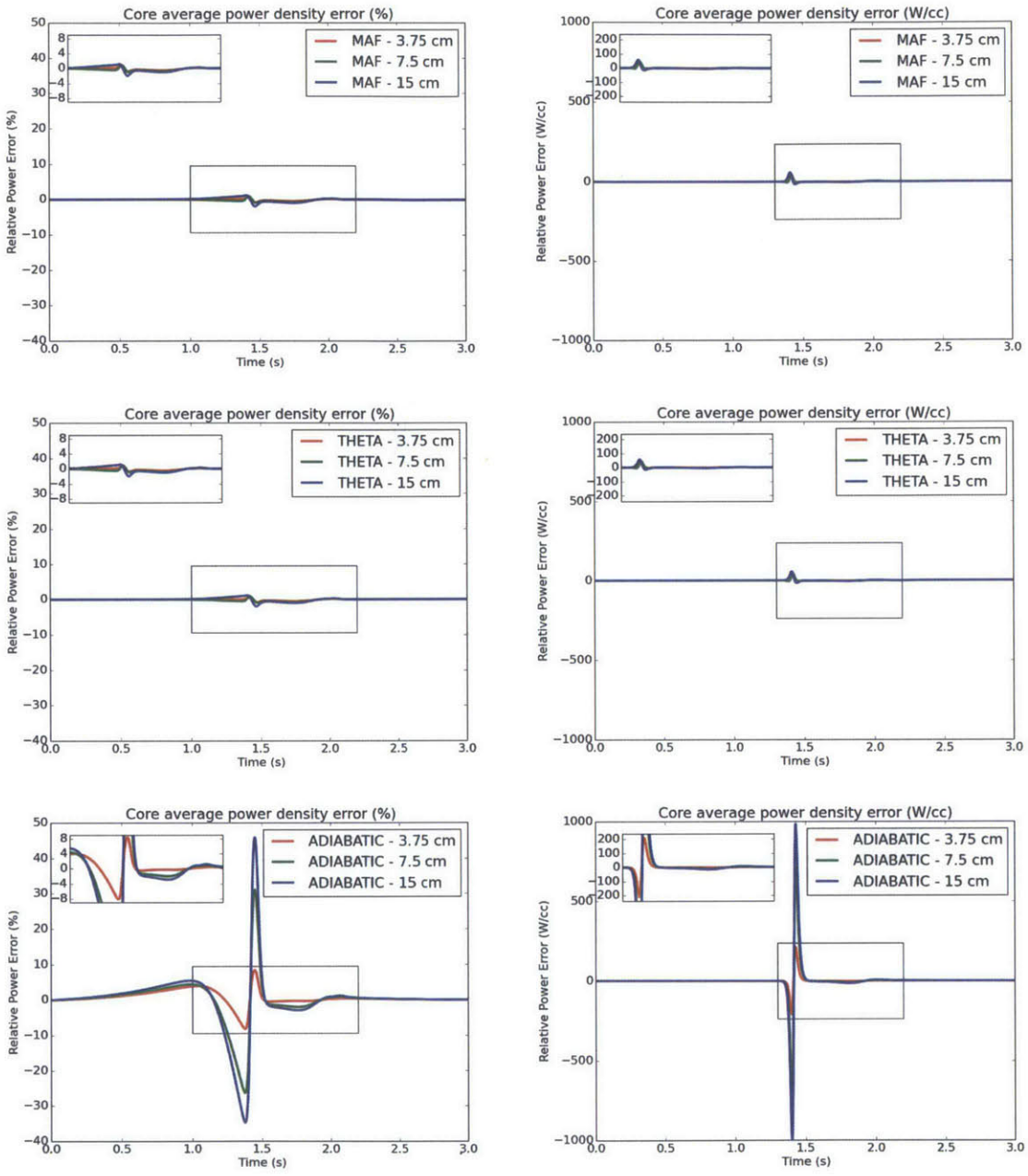

Figure 5-5. Plot of core average power density error percentage (left) and magnitude (right) for the MAF (top), Theta (middle), and Adiabatic (bottom) methods during the LRA transient simulated with different mesh sizes.

to explain these results. From Table 5.9 we see that the first power peak position stays within 0.0001 second of the reference peak position for the MAF and Theta methods, but incurs an error of up to 0.0054 seconds for the Adiabatic method. The peak position error for the Adiabatic method shifts the peak to later times and results

Table 5.10. LRA - coarse mesh sensitivity analysis results - average core temperature and runtime‡

Method	Mesh size (cm)	1^{st} Peak Temp (K)	2^{nd} Peak Temp (K)	Final Temp (K)	Runtime (hr)	MOC solve time (hr)
*MAF	1.875	440.21	876.73	1129.80	10.89	5.45
MAF	3.75	440.07	876.68	1129.86	11.41	7.17
MAF	7.5	440.46	876.62	1129.79	13.92	7.21
MAF	15	441.03	876.76	1129.98	20.50	16.11
*THETA	1.875	440.07	876.34	1129.90	11.49	5.59
THETA	3.75	440.52	876.29	1129.87	11.18	7.02
THETA	7.5	440.94	876.13	1129.81	13.40	9.42
THETA	15	440.88	876.35	1129.99	19.46	14.80
*ADIABATIC	1.875	439.91	876.57	1130.00	42.73	37.17
ADIABATIC	3.75	440.52	877.45	1131.32	33.71	29.91
ADIABATIC	7.5	441.50	877.49	1131.76	57.48	54.06
ADIABATIC	15	442.68	877.81	1132.37	149.53	144.68

‡ - All trials run with 12 threads on 2 6-core Intel Xeon processors
* - Denotes a reference case

in the large error seen in Figure 5-5. It is important to note that Table 5.9 shows that the magnitude of the average core power at the first power peak is relatively constant with a maximum error below 1.0% for all methods and CMFD mesh sizes.

In addition to the core average power error, the runtime performance was also analyzed. The runtime performance for a particular transient simulation can be separated into different components to represent the different procedures taking place. Figure 5-6 shows the runtime breakdown for the shape function reference solutions presented in Figure 5-3 into separate components for the MOC and TCMFD solves. An additional wedge is included for looking up the material cross sections since this procedure takes up a significant amount of time when an implicit procedure is used because it requires all material properties to be reset to a previous state point. The time for the rest of the procedures including convergence checking, computing precursor concentrations, computing cell powers, and other auxiliary functions is represented in the Aux fcts wedge. Figure 5-6 also includes a breakdown of the MOC solve time into separate components for the fixed source iteration, CMFD solve, computation of the source for each FSR, and other auxiliary functions. From Figure 5-6 we see that only 55-60% of the time is spent on the MOC solve for the MAF and Theta methods whereas over 90% of the time is spent on the MOC solve for the Adiabatic method. Furthermore, the MAF and Theta methods required ~84% less time than

the Adiabatic method runtime to solve the problem which is almost entirely due to the additional time required to perform the shape function solve at each time step.

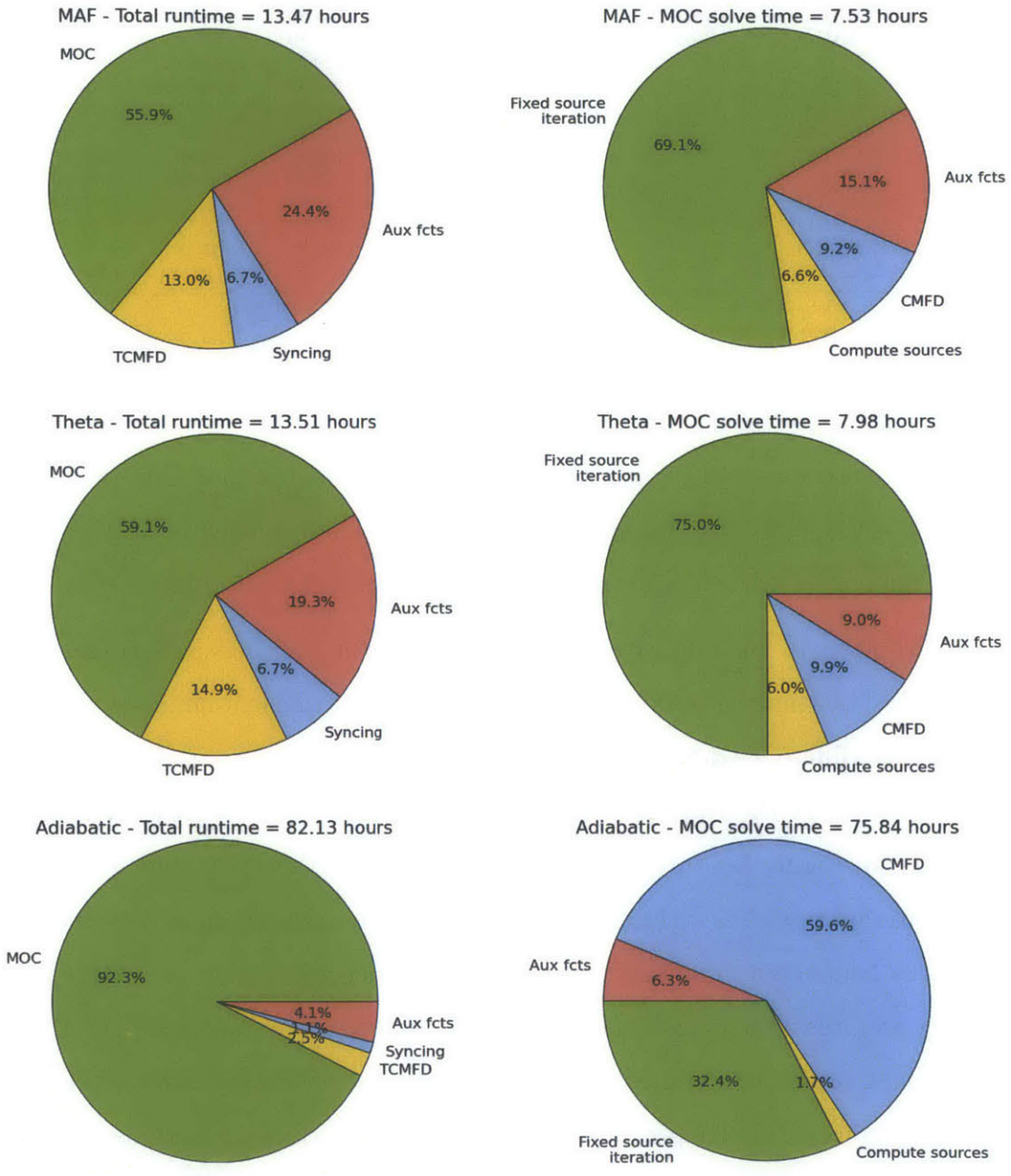

Figure 5-6. Plots that separate the runtime (left) and MOC solve time (right) for LRA trials into various components representing different core processes. The top plots were run with the MAF method, the middle plots with the Theta method, and the bottom plots with the Adiabatic method. All simulations were run with a shape function time step of 5e-4 s and an amplitude function time step of 1e-4 s.

In order to compare trials with an interest in extending these methods to larger

problems, we have analyzed both the total runtime and MOC runtime for all trials presented in the sensitivity analysis. In Figure 5-7 the total runtime and MOC solve time is plotted.

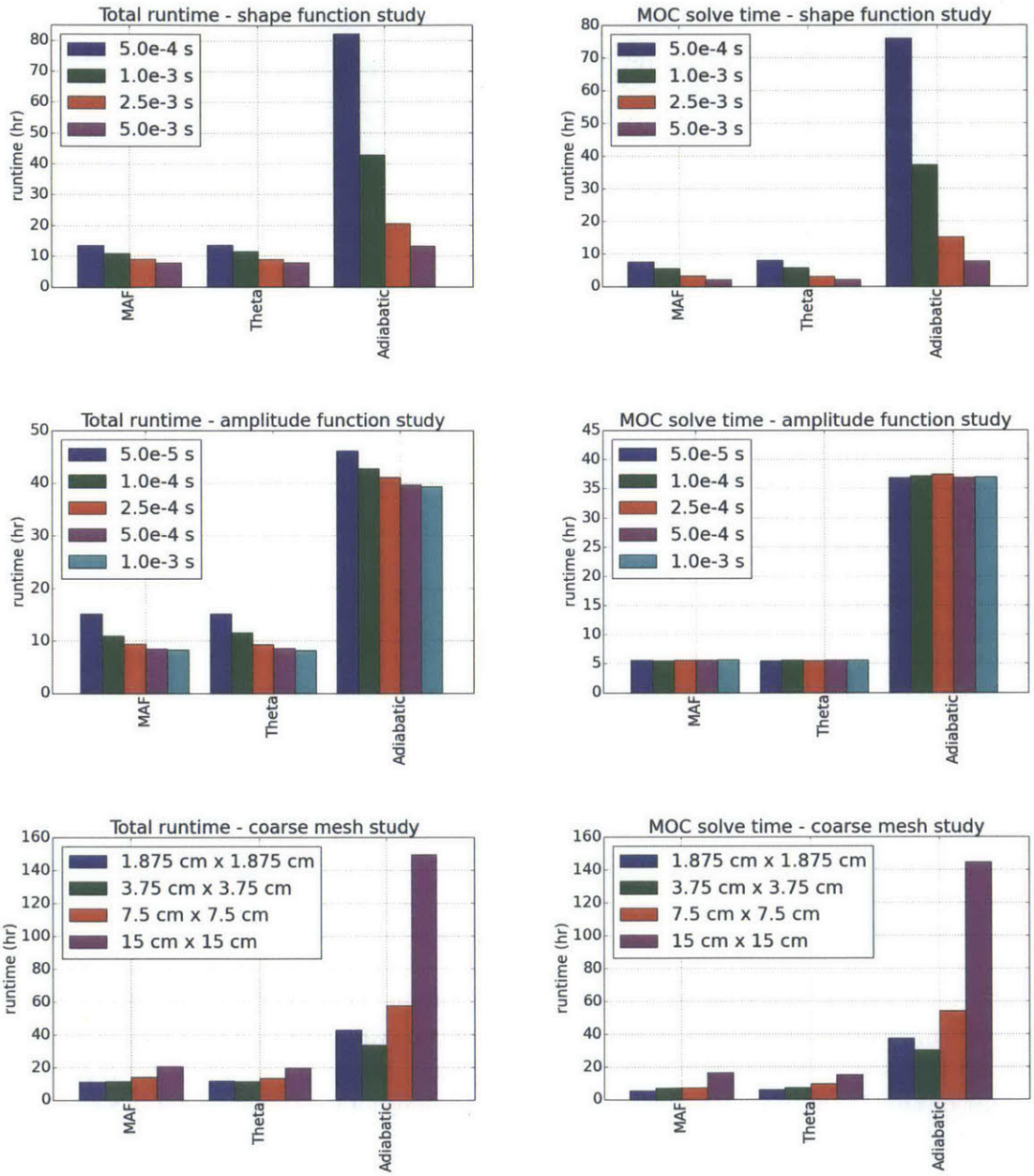

Figure 5-7. Plots that compare the total (left) and MOC solve (right) time for the shape function step size (top), amplitude function step size (middle) and coarse mesh size (bottom) sensitivity studies.

Figure 5-7 highlights the clear performance advantages of the MAF and Theta methods for fine shape function step sizes in solving the LRA benchmark. The reduc-

tion in runtime for these methods is due almost entirely to the reduction in runtime for the MOC solve. The MOC solve time is much shorter in these methods because the eigenvalue problem is reduced to a fixed source problem that can be solved in many fewer iterations as shown in Figure 5-8. Figure 5-7 also demonstrates that the MOC solve time is dependent on the shape function time step size and has little dependence on the amplitude function step size. The runtimes for the coarse mesh solver tend to increase as the mesh is coarsened because it takes more iterations to solve the MOC problems as shown in Table 3.5.

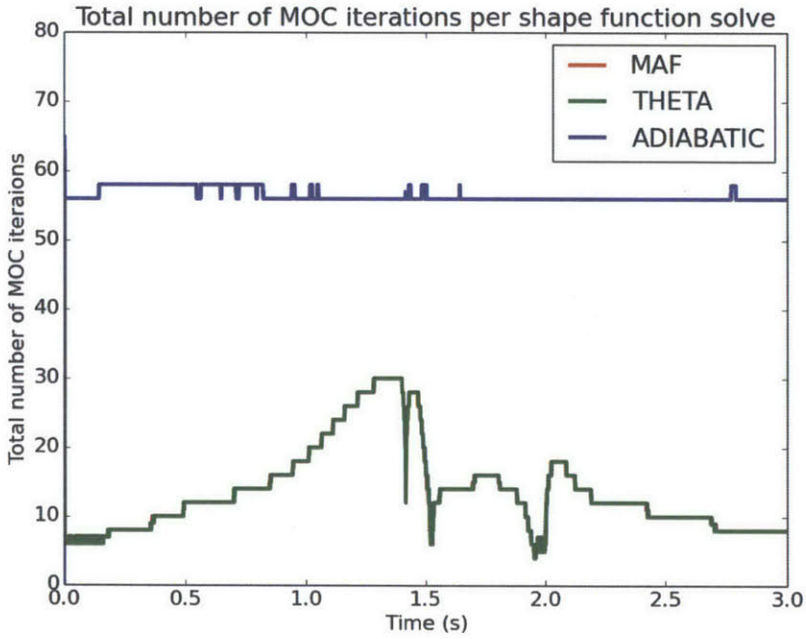

Figure 5-8. Total number of MOC iterations required to solve the MOC problem at each time step. In cases where multiple shape function solves were required to converge the shape function at an outer time step, the total number of iterations is the sum of the MOC iterations required for all solves. The MAF and Theta methods overlap for the majority of the plot.

In summary, the LRA sensitivity study results highlight a few key points:

- **Shape function sensitivity** - The MAF and Theta methods show the highest sensitivity to the shape function time step size with maximum core average power density errors of 5.4% and -7.5% for the trials studied, respectively. The

Adiabatic method incurred a maximum core average power density error of -0.4% for the trials tested, suggesting the Adiabatic method might be able to maintain solution accuracy at larger shape function time step sizes than the MAF and Theta methods for problems similar to the LRA benchmark.

- **Amplitude function sensitivity** - All methods showed some sensitivity in core average power to the amplitude function step size. However, this sensitivity was mainly due to a time-shift in the power profile and not necessarily an increase or decrease in the magnitude of the maximum core average power density.

- **Coarse mesh sensitivity** - The MAF and Theta methods were relatively insensitive to the coarse mesh size whereas the Adiabatic method incurred some error in the core average power density at the first power peak. This error was due almost entirely to a shift in the core average power density to later times and not a shift in the magnitude of the core average power density.

- **Runtime performance** - The MAF and Theta methods required ~84% less runtime compared with the Adiabatic method. This is due to the increased number of fixed source iterations required to compute the shape function at each shape function time step.

5.2 Transient C5G7 Benchmark

The 2D C5G7 benchmark is a 7 group, quarter-core LWR transport problem that is fully specified in section A.2. A transient version of the C5G7 transport benchmark was originally published in [1]. Herein, we have added a doppler feedback model and treated the transient as a ramp withdrawal over 2.0 seconds instead of a prompt withdrawal. The geometry is presented with heterogeneous fuel/clad and water regions cross section data and a simple linear absorption and scattering cross-section ramp to model the withdrawal of the control rods in one assembly. The discretization of this problem makes it more difficult to resolve the spatial flux profile, as shown in subsection 3.2.1.

5.2.1 Static Control Rod Worth

We begin our analysis by determining the static control rod worth by computing the eigenvalue at the initial and final material states. We compute the control rod worth using two sets of FSR discretizations and track laydowns - the ultrafine FSR discretization and track laydown used to compute the reference solution in section 3.2 and the coarse FSR discretization and track laydown identified as producing a sufficiently converged solution in subsection 3.2.1. The simulation parameters and eigenvalues for both cases and both states are presented in Table 5.11 and the initial and final power distributions for the ultrafine reference case are shown in Figure 5-9.

Table 5.11. The simulation parameters and eigenvalues for the initial and final state of the C5G7 transient transport benchmark

Azimuthal Angles	**Track Spacing (cm)**	**# FSRs**	**State**	k_{eff}	**reactivity ($)**
128	0.05	571,856	Initial	1.148122	-
128	0.05	571,856	Final	1.158253	1.3575
64	0.05	235,088	Initial	1.148224	-
64	0.05	235,088	Final	1.158347	1.3563

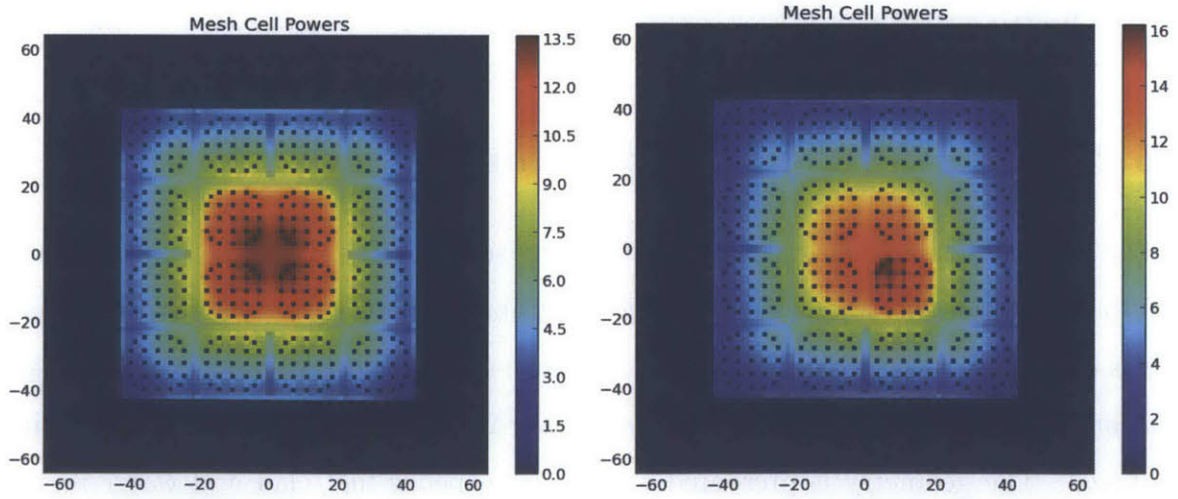

Figure 5-9. Plots of the relative power for the initial (left) and final (right) material states of the C5G7 benchmark.

The initial eigenvalue for the ultrafine mesh case is within 17 pcm of the reference eigenvalue computed in [5]. The control rod worth was computed using Equation 5.1

where β is 0.0065 for this problem. Since the initial state is assumed to be critical, we care much more about the reactivity comparison between the two cases rather than eigenvalues. The control rod worth for our case with a slightly coarser MOC mesh was found to be in very good agreement with the ultrafine mesh case, suggesting the coarse MOC mesh should be sufficient to approximate the spatially converged transient solution.

5.2.2 Reference Solution

To our knowledge there has been no previously reported spatially and temporally converged reference solution for the transport version of the C5G7 transient benchmark. Therefore, we simulated the transient with a fine spatial mesh and fine step sizes in order to generate a reference solution. The parameters used to generate all transient results in the section are provided in Table 5.12.

Table 5.12. Calculation conditions

Parameter	Value
Number of polar angles	3 (using TY quadrature set)
MOC convergence criteria	1e-6 on RMSD of fission source
TCMFD convergence criteria	1e-8 on RMSD of fission source
CMFD convergence criteria	1e-6 on RMSD of fission source
GS convergence criteria	1e-8 on RMSD of fission source
CMFD damping factor	0.5
CPU architecture	2 6-core Intel Xeon processors
Threads	12
# of azimuthal angles	64
Track spacing	0.05 cm
Pin Cell discretization	Scheme 4

The results from the reference simulation are provided in the Table 5.13. The average core power and peak fuel temperature profile during the transient are presented in Figure 5-10.

Table 5.13. Calculation conditions

Parameter	Value
Shape function step size	1e-3 s
Amplitude function step size	1e-4 s
Method	MAF
CMFD mesh size	pin-cell
Initial k_{eff}	1.148224
Time @ peak	1.9796 s
Avg. fuel temperature @ peak	342.45 K
Avg. power density @ peak	1333.64 W/cc
Avg. fuel temperature @ t = 3s	447.77 K
Avg. power density @ t = 3s	34.10 W/cc
Runtime	64.44 hours

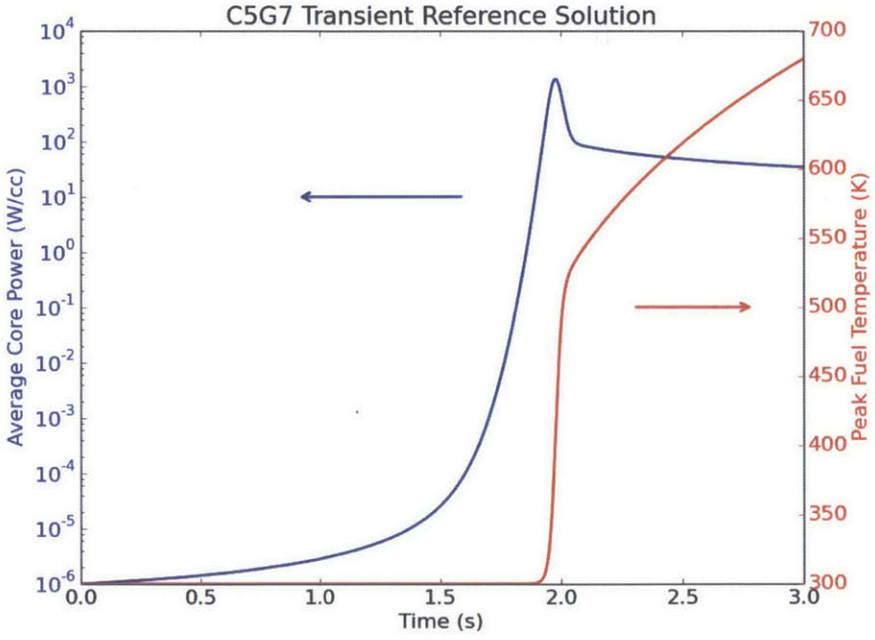

Figure 5-10. Plot of average core power (blue) and average fuel temperature (red) during the reference C5G7 transient.

5.2.3 Sensitivity Analysis

In order to evaluate the convergence and performance of the MAF, Theta, and Adiabatic methods in modeling the C5G7 transient problem, we conducted a series of trials where simulation parameters were adjusted. Table 5.14 shows the values used for each

parameter in their respective sensitivity analyses. In this section, we investigate each parameter individually and seek to determine the influence of each parameter on solution convergence. The reference solution for each analysis was chosen such that only one simulation parameter changed for each to the trials, allowing us to isolate the influence of this variable. Solution convergence is determined by comparing the core average power density error percentage and magnitude as a function of time compared to the reference simulation for each analysis.

Table 5.14. Calculation conditions

Parameter	Values
Shape fct step size (Δt_s in s)	5.0e-4, 1.0e-3, 2.0e-3
Amp fct step size (Δt_a in s)	5.0e-4, 1.0e-3, 2.0e-3
Coarse mesh size	pin-cell and assembly

In all trials two different shape function time step sizes were used. From 0 to 1.5 seconds and 2.2 to 3.0 seconds, where the power profile is slowly changing, the shape function time step size was set to 1.0e-2 seconds. From 1.5 to 2.2 seconds, where the power profile is rapidly changing, the time step size was set to one of the values in Table 5.14.

We begin by evaluating the effect of the shape function time step size on the core average power density profile convergence. In a series of trials the shape function step size, Δt_s, from 1.5 to 2.2 seconds was adjusted with all other parameters being held constant at the values listed in Table 5.12. Reference solutions for each method were generated with a shape function time step size of 5.0e-4 seconds, an amplitude function step size of 5.0e-4 seconds, and pin-cell discretization scheme 4 from Figure 3-8. Figure 5-11 shows the core average power density error percentage and magnitude during the transient for a series of trials with varying shape function time step sizes. Table 5.15 and Table 5.16 provide a summary of other simulation results for these trials.

Figure 5-11 shows that the convergence of the core power is not very sensitive to the shape function step size for any of the methods. However, when the transient step size was increased beyond the noted values, the simulation became unstable for

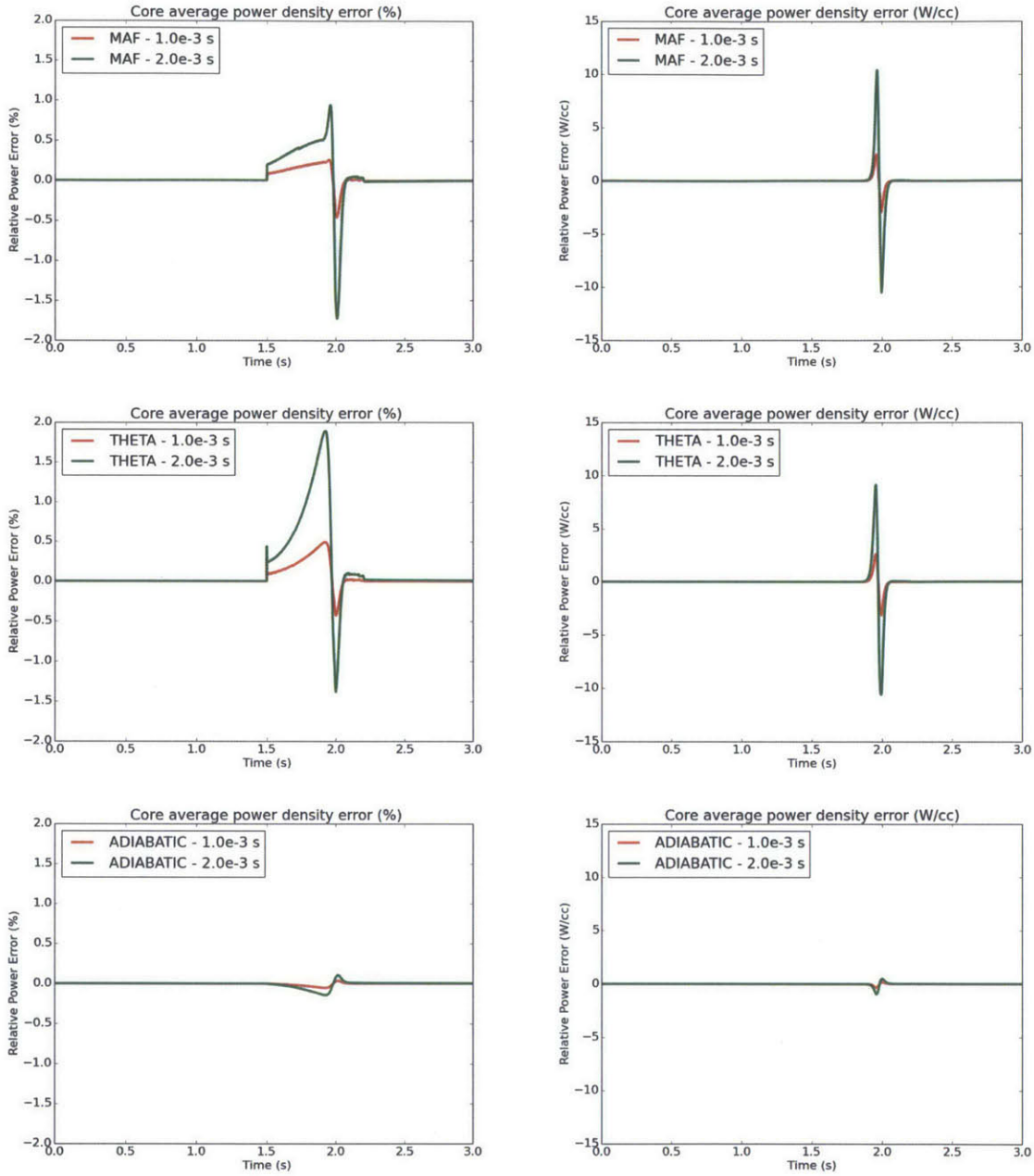

Figure 5-11. Plot of core average power density error percentage (left) and magnitude (right) for the MAF (top), Theta (middle), and Adiabatic (bottom) methods during the C5G7 transient simulated with different shape function step sizes.

the MAF and Theta methods. As was seen for the LRA transient, the Adiabatic method was the least sensitive to shape function time step size in the range tested. The error for all methods is sharply peaked around the core average power peak at ~1.98 s with a maximum core average power density error below 0.5% for all methods

Table 5.15. C5G7 - shape function sensitivity analysis results - average core power and peak power times‡

Method	Δt_s (s)	t @ 1^{st} peak (s)	1^{st} Peak Power (W/cc)	Final Power (W/cc)
*MAF	0.0005	1.9770	1318.28	34.20
MAF	0.0010	1.9770	1319.44	34.20
MAF	0.0020	1.9770	1324.90	34.20
*THETA	0.0005	1.9770	1318.22	34.19
THETA	0.0010	1.9770	1317.70	34.19
THETA	0.0020	1.9770	1314.31	34.20
*ADIABATIC	0.0005	1.9760	1316.70	34.10
ADIABATIC	0.0010	1.9760	1316.52	34.10
ADIABATIC	0.0020	1.9760	1316.21	34.10

‡ - All trials run with 12 threads on 2 6-core Intel Xeon processors
* - Denotes a reference case

Table 5.16. C5G7 - shape function sensitivity analysis results - average core temperature and runtime‡

Method	Δt_s (s)	1^{st} Peak Temp (K)	Final Temp (K)	Runtime (hr)	MOC solve time (hr)
*MAF	0.0005	341.67	447.75	39.85	6.02
MAF	0.0010	341.76	447.74	38.22	4.54
MAF	0.0020	342.00	447.72	37.83	4.46
*THETA	0.0005	341.70	447.74	41.67	6.62
THETA	0.0010	341.80	447.72	38.14	4.57
THETA	0.0020	342.05	447.68	40.73	7.19
*ADIABATIC	0.0005	341.70	447.45	39.07	5.68
ADIABATIC	0.0010	341.69	447.44	37.11	3.83
ADIABATIC	0.0020	341.66	447.43	35.75	2.53

‡ - All trials run with 12 threads on 2 6-core Intel Xeon processors
* - Denotes a reference case

and all trials as shown in Table 5.15. Table 5.16 shows that the temperature at the power peak was nearly constant, staying within 0.4 K of the reference values for all trials.

Next we evaluate the effect of the amplitude function time step size on the core average power density profile convergence. In a series of trials the amplitude function step size, Δt_a, was adjusted with all other parameters being held constant at the values listed in Table 5.12. Reference solutions were generated with an amplitude function step size of 5.0e-4 seconds, a shape function time step size of 2.0e-3 seconds from 1.5 to 2.2 seconds, and scheme 4 pin cell discretization on the MOC mesh. Figure 5-12 shows the core average power density error percentage and magnitude during the transient for a series of trials with varying amplitude function time step sizes. Table 5.17 and Table 5.18 provide a summary of other simulation results for

these trials.

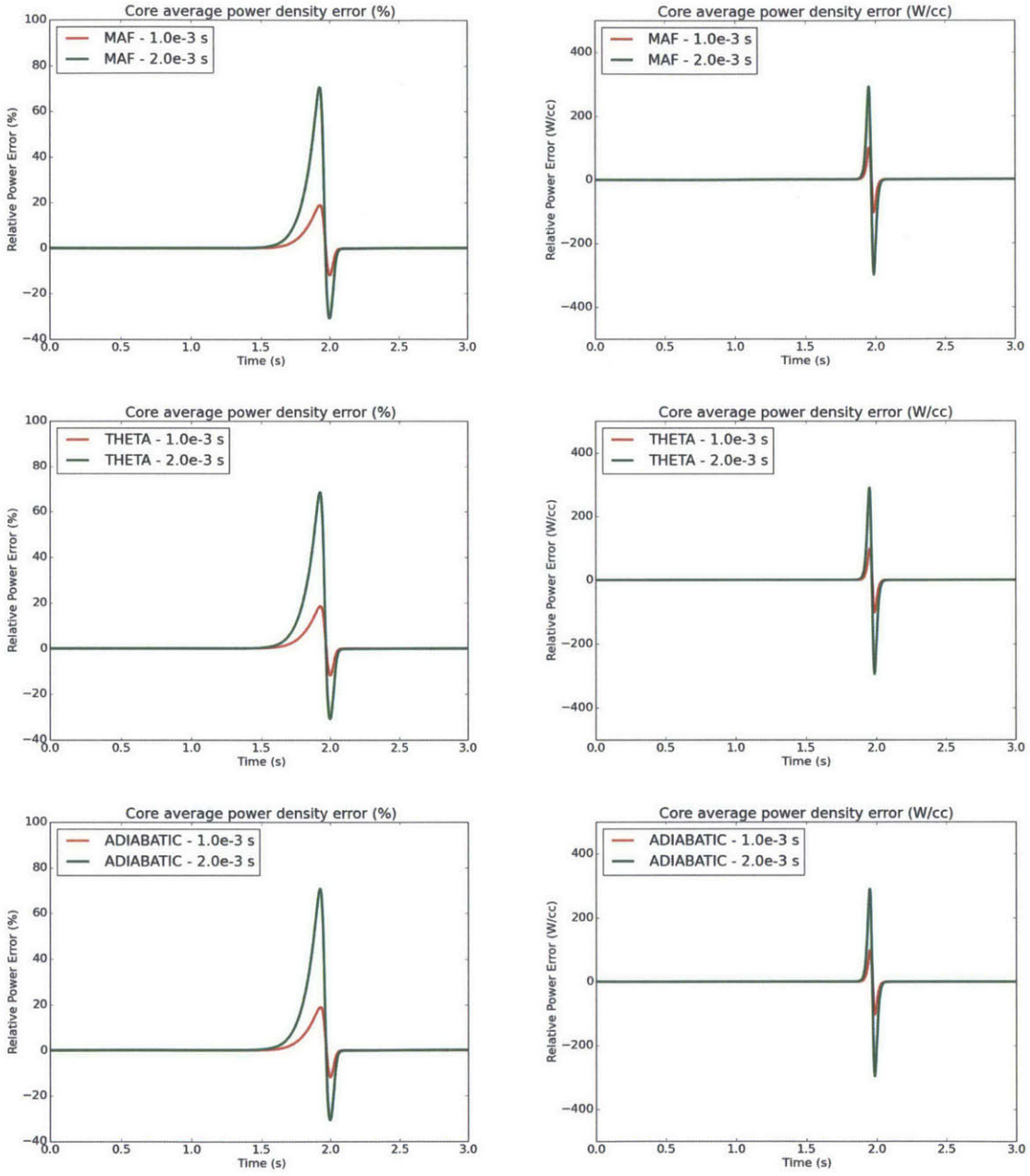

Figure 5-12. Plot of core average power density error percentage (left) and magnitude (right) for the MAF (top), Theta (middle), and Adiabatic (bottom) methods during the C5G7 transient simulated with different amplitude function step sizes.

Figure 5-12 shows that the convergence of the core power is sensitive to the amplitude function step size for all methods, incurring errors of >60% near the power peak. This behavior is very similar to the behavior seen for the sensitivity of the

Table 5.17. C5G7 - amplitude function sensitivity analysis results - average core power and peak power times‡

Method	Δt_a (s)	t @ 1^{st} peak (s)	1^{st} Peak Power (W/cc)	Final Power (W/cc)
*MAF	0.0005	1.9770	1324.90	34.20
MAF	0.0010	1.9740	1306.03	34.20
MAF	0.0020	1.9680	1267.88	34.21
*THETA	0.0005	1.9770	1314.31	34.20
THETA	0.0010	1.9740	1298.63	34.20
THETA	0.0020	1.9680	1266.58	34.21
*ADIABATIC	0.0005	1.9760	1316.21	34.10
ADIABATIC	0.0010	1.9730	1297.52	34.10
ADIABATIC	0.0020	1.9680	1259.56	34.11

‡ - All trials run with 12 threads on 2 6-core Intel Xeon processors
* - Denotes a reference case

Table 5.18. C5G7 - amplitude function sensitivity analysis results - average core temperature and runtime‡

Method	Δt_a (s)	1^{st} Peak Temp (K)	Final Temp (K)	Runtime (hr)	MOC solve time (hr)
*MAF	0.0005	342.00	447.72	37.83	4.46
MAF	0.0010	341.10	447.48	35.31	4.42
MAF	0.0020	339.45	446.98	32.98	4.34
*THETA	0.0005	342.05	447.68	40.73	7.19
THETA	0.0010	341.15	447.44	37.52	6.57
THETA	0.0020	339.49	446.95	33.53	5.00
*ADIABATIC	0.0005	341.66	447.43	35.75	2.53
ADIABATIC	0.0010	340.77	447.19	33.44	2.52
ADIABATIC	0.0020	340.65	446.67	31.04	2.52

‡ - All trials run with 12 threads on 2 6-core Intel Xeon processors
* - Denotes a reference case

amplitude function time step size for the LRA benchmark. Table 5.17 indicates that the error in core average power density at the peak seen in Figure 5-12 is due to both a shift in the power peak to earlier times and a decrease in magnitude of the power peak as the amplitude function step size increases.

The shape and amplitude function step sizes are important to consider in temporally converging the transient solution. In order to spatially converge the solution, it is important to look at both the FSR discretization and coarse mesh cell discretization. In chapter 3 the influence of FSR discretization on the convergence of the initial steady state eigenvalue was investigated. A pin cell discretization with 2 rings in the fuel, 2 rings in the water, and 8 sectors was selected to be used in transient analysis as it produced a sufficiently converged solution to the steady state problem. During the transient solve, both a fine mesh transport and coarse mesh nonlinear diffusion

problem are being solved so its important to consider both the fine mesh (FSR) and coarse mesh discretization in trying to converge the transient solution.

In a series of trials the coarse mesh cell size was adjusted with all other parameters being held constant at the values listed in Table 5.12. Reference solutions were generated with an amplitude function step size of 5.0e-4 seconds, a shape function time step size from 1.5 to 2.2 seconds of 1.0e-3 seconds, a pin-cell coarse mesh, and scheme 4 pin cell discretization on the MOC mesh. Figure 5-13 shows the core average power density error percentage and magnitude during the transient for a series of trials with varying coarse mesh cell sizes. Table 5.19 and Table 5.20 provide a summary of other simulation results for these trials.

Table 5.19. C5G7 - coarse mesh sensitivity analysis results - average core power and peak power times‡

Method	Mesh size	t @ 1^{st} peak (s)	1^{st} Peak Power (W/cc)	Final Power (W/cc)
*MAF	pin-cell	1.9770	1319.44	34.20
MAF	assembly	1.9755	1361.24	34.87
*THETA	pin-cell	1.9770	1317.70	34.19
THETA	assembly	1.9755	1359.41	34.88
*ADIABATIC	pin-cell	1.9760	1316.52	34.10
ADIABATIC	assembly	1.9895	1340.38	34.11

‡ - All trials run with 12 threads on 2 6-core Intel Xeon processors
* - Denotes a reference case

Table 5.20. C5G7 - coarse mesh sensitivity analysis results - average core temperature and runtime‡

Method	Mesh size	1^{st} Peak Temp (K)	Final Temp (K)	Runtime (hr)	MOC solve time (hr)
*MAF	pin-cell	341.76	447.74	38.22	4.54
MAF	assembly	343.45	451.30	42.66	37.69
*THETA	pin-cell	341.80	447.72	38.14	4.57
THETA	assembly	343.48	451.30	41.73	36.68
*ADIABATIC	pin-cell	341.69	447.44	37.11	3.83
ADIABATIC	assembly	343.64	449.01	20.88	15.76

‡ - All trials run with 12 threads on 2 6-core Intel Xeon processors
* - Denotes a reference case

Figure 5-13 shows that the convergence of the core power is sensitive to the coarse mesh discretization for all three methods. While Figure 5-13 shows a large error in core average power density around the power peak, Table 5.19 indicates that the magnitude of the core average power density at the power peak changes by at most

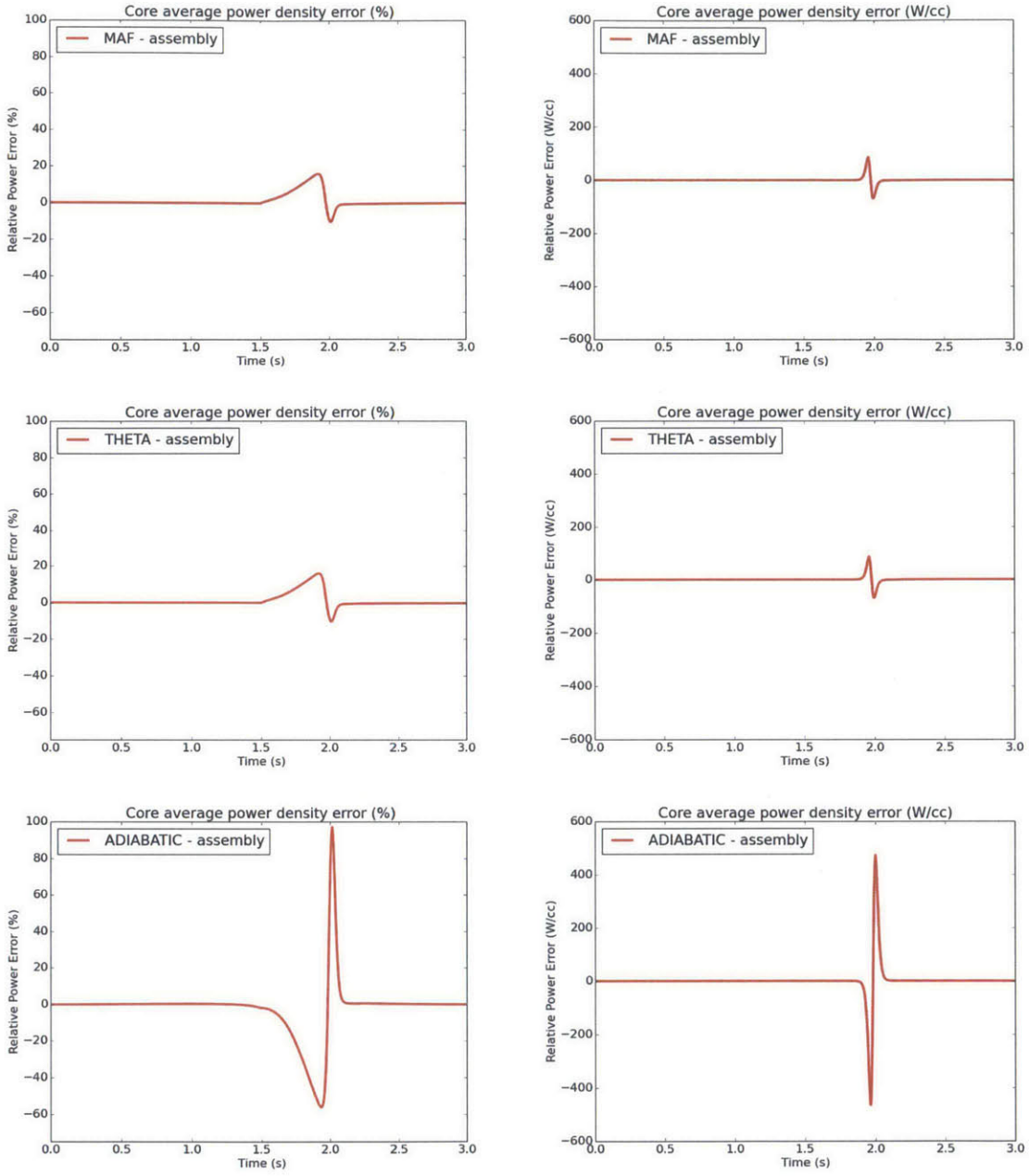

Figure 5-13. Plot of core average power density error percentage (left) and magnitude (right) for the MAF (top), Theta (middle), and Adiabatic (bottom) methods during the C5G7 transient simulated with a coarse mesh on the assembly level.

3.2% for all methods. This suggests that the error encountered in the core average power density for an assembly-wise mesh is mainly due to a shifting of the power peak and not a change in magnitude.

In addition to the core average power density error, the runtime performance

was also analyzed. Figure 5-14 shows the runtime breakdown for the shape function reference solutions presented in Figure 5-11 into separate components for the MOC and TCMFD solves. Figure 5-14 also includes a breakdown of the MOC solve time into separate components for the fixed source iteration, CMFD solve, computation of the source for each FSR, and other auxiliary functions. From Figure 5-14 we see that only ~15% of the time is spent on the MOC solve for all methods whereas ~75% of the time was spent on the TCMFD solve. This is quite different from the behavior seen in the LRA transient simulations and can be attributed to the CMFD/TCMFD problem containing ~4.7× more unknowns for the C5G7 problem. Additionally, the total and MOC runtime was about the same for all three methods. We would expect total runtimes to be similar since they are dominated by the TCMFD solve which does not depend on the transient method used.

In order to compare trials with an interest in extending these methods to larger problems, we have analyzed both the total runtime and MOC runtime for all trials presented in the sensitivity analysis. In Figure 5-15 the total runtime and MOC solve time is plotted.

The results in Figure 5-15 are mixed with no clear performance advantage for any one method. The only discernible trend is a sharp increase the MOC solve time for the assembly level CMFD meshes due to an increase in the number of iterations required to converge the shape at each time step. The MOC solve time is nearly the same for all methods. Figure 5-16 shows that the shape function solve for each method required roughly the same number of MOC fixed source iterations during each MOC solve which would explain the similar solver time. We will note that this behavior was not expected and is noticeably different from the LRA problem results. In is unclear why the Adiabatic method did not require more iterations to converge at each time step, but we postulate that this could be due to the relatively small shift in flux and power shape during the C5G7 transient and the lower dominance ratio of the C5G7 problem.

In summary, the C5G7 sensitivity study results highlight a few key points:

- **Shape function sensitivity** - None of the methods showed significant sensitivity

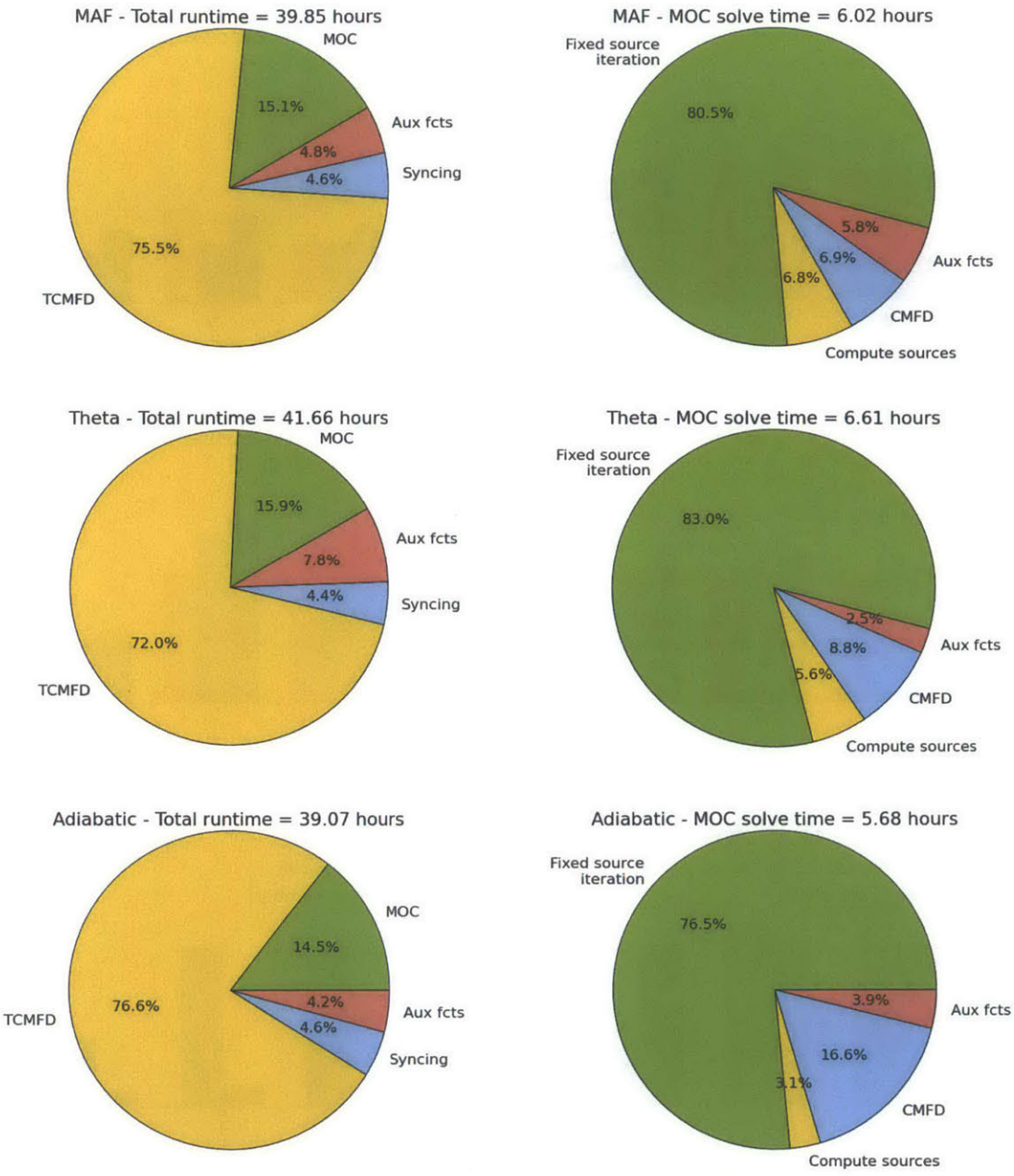

Figure 5-14. Plots that separate the runtime (left) and MOC solve time (right) for C5G7 trials into various components representing different core processes. The top plots were run with the MAF method, the middle plots with the Theta method, and the bottom plots with the Adiabatic method. All simulations were run with a shape function time step of 5.0e-4 s and an amplitude function time step size of 5.0e-4 s.

to the shape function time step size with maximum core average power density errors of below 2.0% for all times during the trials tested. The Adiabatic method incurred the least error, similar to the behavior seen in the LRA benchmark.

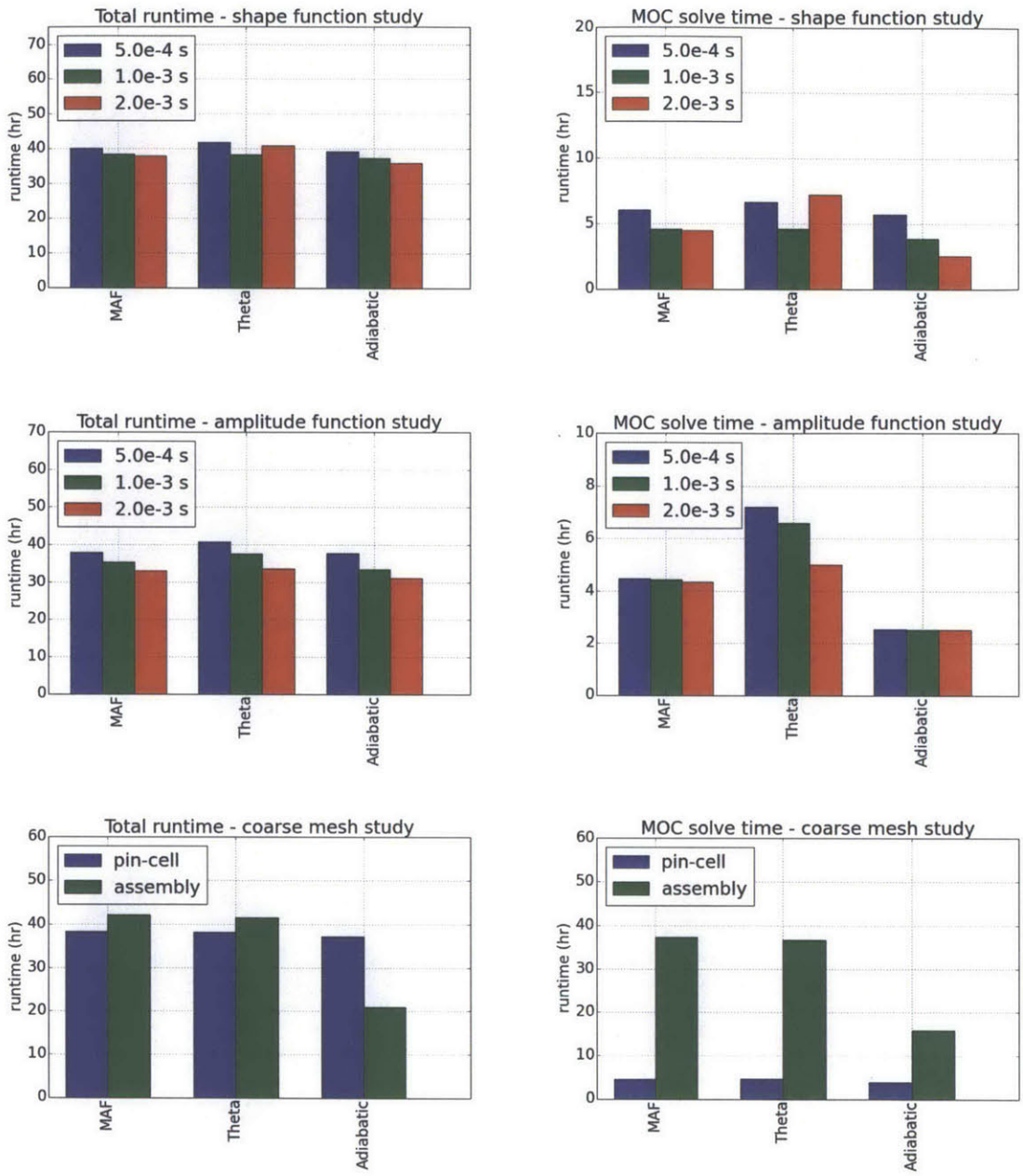

Figure 5-15. Plots that compare the total (left) and MOC solve (right) time for the shape function step size (top), amplitude function step size (middle) and coarse mesh size (bottom) sensitivity studies.

- **Amplitude function sensitivity** - All methods showed some sensitivity in core average power to the amplitude function step size. However, this sensitivity was mainly due to a time-shift in the power profile and not necessarily an increase or decrease in the magnitude of the maximum core average power density.

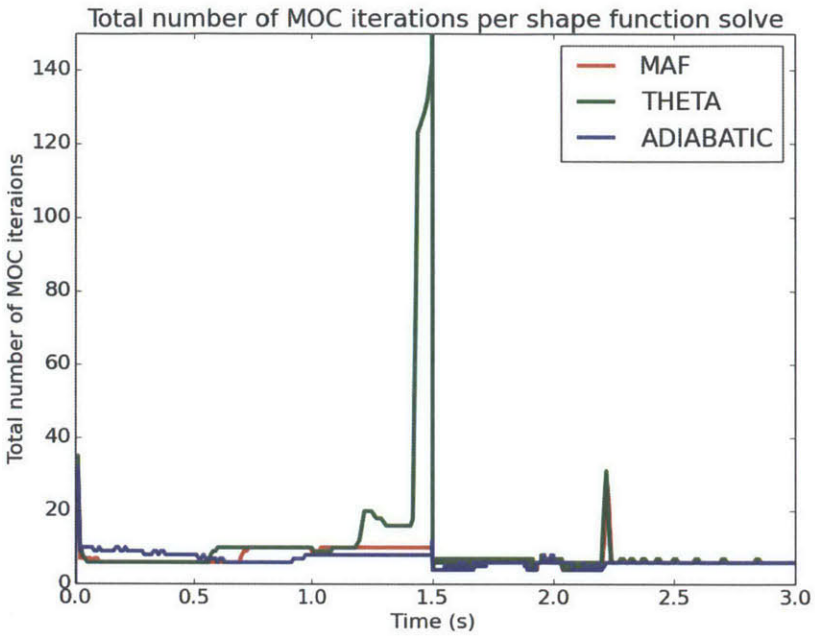

Figure 5-16. Total number of MOC iterations required to solve the MOC problem at each time step. In cases where multiple shape function solves were required to converge the shape function at an outer time step, the total number of iterations is the sum of the MOC iterations required for all solves. The discontinuities at t=1.5s and t=2.2s are due to the transition to and from the fine shape function step size, respectively.

- **Coarse mesh sensitivity** - The MAF and Theta methods were relatively insensitive to the coarse mesh size whereas the Adiabatic method incurred some error in the core average power density at the power peak. This error was due partly to a shift in the core average power density to earlier times and an increase in the magnitude of the core average power density.

- **Runtime performance** - All methods showed similar runtime performance in terms of both total and MOC runtime.

Chapter 6

Summary and Conclusions

6.1 Summary

In this thesis, we have investigated the Adiabatic, Theta, and Multigrid Amplitude Function methods for solving 2D time-dependent neutron transport problems. A sensitivity study was conducted on two 2D transient transport benchmarks. The results highlight the superior runtime performance of the MAF and Theta methods for the LRA problem where the flux and power profile experience a significant shift during the transient. The runtime performance for the C5G7 benchmark, which incurs a less significant flux and power profile shift than the LRA benchmark, was nearly the same for all three methods. The amplitude function time step size was seen to have a similar influence on the convergence of the power profile for all trials. All methods showed a change in position and small change in magnitude of the core average power density peak for both problems. The MAF and Theta methods showed the most sensitivity to the shape function step size, with the Adiabatic method being relatively insensitive to this parameter. The Adiabatic method showed the most sensitivity to the coarse mesh size of all the methods, but the error in core average power density was rather small and due mainly to a shift in the peak core average power density.

6.1.1 Steady State Method of Characteristics

In chapter 2 the steady state Method of Characteristics formulation of the neutron transport equation was derived. A thorough derivation of the CMFD acceleration framework was also included. The initial states of the transient transport versions of the LRA and C5G7 benchmarks were investigated to understand the influence of the number of azimuthal angles, track spacing, and FSR mesh discretization on the convergence of the problem eigenvalue.

We began our analysis by presenting a spatially converged reference solution to the LRA problem by using an ultrafine FSR discretization and track lay down. The reference eigenvalue was computed to be 0.997213. Using the reference eigenvalue, a sensitivity analysis was conducted. The LRA problem was found to be insensitive to the number of azimuthal angles and track spacing. Next, the effect of the FSR discretization was studied by performing a series of trials with different numbers of uniform refinements on the assembly-wise mesh. In order to converge the eigenvalue to within ~25 pcm of the reference solution, an FSR size of 0.46875 cm was required. From this analysis, we selected this mesh size, 32 azimuthal angles, and a track spacing of 0.05 cm to be used in all transient trials. The performance of the CMFD solver was also investigated for different coarse mesh sizes and CMFD group structures. The 3.75 and 1.875 cm coarse meshes were found to provide the most significant speedup of ~45× in runtime and 150-240× in number of MOC iterations. The optimum damping factor was found to be between 0.4 and 0.6. A damping factor of 0.5 was selected for use in transient trials.

Next we performed the same sensitivity analysis on the C5G7 transport benchmark. The reference eigenvalue was computed to be 1.18672 which is +17 pcm off the benchmark value. Using the benchmark eigenvalue, a sensitivity analysis was conducted. The C5G7 problem was found to be sensitive to the number of azimuthal angles, but not sensitive to the track spacing. Next, the effect of the FSR discretization was studied by performing a series of trials with different pin-cell FSR discretizations. In order to converge the eigenvalue to within ~25 pcm of the benchmark solution,

a pin-cell discretization with 2 fuel rings, 2 water rings, and 8 sectors was required. From this analysis, we selected this pin-cell discretization, 64 azimuthal angles, and a track spacing of 0.05 cm to be used in all transient trials. The performance of the CMFD solver was also investigated for different coarse mesh sizes and CMFD group structures. Pin-cell, assembly, and core wise coarse meshes and CMFD group structures with 1, 2, 4, and 7 groups were studied. The pin-wise CMFD provided the most significant speed of 10-20× in runtime and 14-29× in number of MOC iterations. The 2, 4, and 7 energy group structures had similar performance suggesting the CMFD group structure can be coarsened for fine group problems and still achieve reasonable acceleration. The optimum damping factor was found to be between 0.5 and 0.7. A damping factor of 0.6 was selected for use in transient trials.

6.1.2 Time Dependent Method of Characteristics

In chapter 4 the time dependent Method of Characteristics formulation of the neutron transport equation was presented. A derivation of the Adiabatic, fully implicit Theta, and Multigrid Amplitude Function methods were included along with the corresponding modifications of the CMFD acceleration framework required for these methods. The fully implicit Theta method was used to solve for the amplitude function between large shape function time steps. A procedure for implementing the time dependent MOC methods was described and algorithms for solving each of the steps were included.

In chapter 5 the results for transient simulations of the LRA and C5G7 benchmarks were presented. First, the transport version of the LRA transient benchmark was studied. The static control blade worth was computed to be 3.0711$ for an ultrafine mesh case and 3.0492$ for a coarse mesh case. A reference solution was computed using ultrafine shape and amplitude function time steps. The peak core average power density was found to be 5513.84 W/cc at t=1.41545 seconds. Interestingly, this value is very close to the diffusion reference value of 5411 W/cc at 1.436 seconds. A sensitivity analysis was conducted on the shape function time step, amplitude function time step, and coarse mesh size. Table 6.1 provides a qualitative comparison

of the sensitivity of each method to the simulation parameters for the LRA problem.

Table 6.1. LRA - Sensitivity to simulation parameters

	Adiabatic	**Theta**	**MAF**
Shape function time step	low	high	high
Amplitude function time step	moderate	moderate	moderate
Coarse mesh size	moderate	low	low

The goal of our study was to investigate these methods with particular attention to how they will scale to larger, more computationally demanding problems. The sensitivity analysis only provides half of the picture with the other factor being the runtime performance. The Theta and MAF methods have clear performance advantages at small shape function time step sizes due to a decrease in computation time required to solve the shape function at each time step. For the representative cases studied in the performance analysis, the time spent solving the shape function was $\sim 10\times$ greater with the Adiabatic method. This can be attributed to the Theta and MAF methods using a fixed source shape function solve whereas the Adiabatic method requires a full eigenvalue solve. However, the Adiabatic method is much less sensitive to the shape function time step size allowing for solution accuracy to be maintained at much larger time step sizes. This presents a tradeoff - if the time saved from fewer shape function solves with the Adiabatic method is greater than the extra time required per solve, the Adiabatic method would be preferred; otherwise, the Theta or MAF method would be preferred.

Next, the transient version of the C5G7 transport benchmark was studied. The static control rod worth was computed to be 1.3575\$ for an ultrafine mesh case and 1.3563\$ for a coarse mesh case that was used for transient simulations. A reference solution was computed using ultrafine shape and amplitude function time steps. The peak core average power density was found to be 1333.64 W/cc at t=1.9796 seconds. Since this was a contrived problem that we created for this work, no other solutions existed to compare with. It is our hope that future developers will analyze this problem and publish their results so researchers can compare implementations of transient transport methods. A sensitivity analysis was conducted on the shape

function time step, amplitude function time step, and coarse mesh size. Table 6.2 provides a qualitative comparison of the sensitivity of each method to the simulation parameters.

Table 6.2. C5G7 - Sensitivity to simulation parameters

	Adiabatic	**Theta**	**MAF**
Shape function time step	low	low	low
Amplitude function time step	moderate	moderate	moderate
Coarse mesh size	moderate	low	low

In contrast to the LRA problem, the C5G7 flux and power shape does not change significantly during the transient. This characteristic is likely what makes the methods relatively insensitive to the shape function size during the transient. All methods were moderately sensitive to the amplitude function step size with the peak power shifting to earlier times as the step size increased. The magnitude of the peak changed by a maximum value of 6%. All methods showed some sensitivity to coarse mesh size since the spatial resolution of the amplitude is very poor on an assembly level. The Adiabatic method had similar performance to the MAF and Theta methods which is in contrast to the behavior seen for the LRA problem. This difference in behavior is likely due to the shape function solves requiring roughly the same number of MOC iterations per solve. The MOC solve time showed a clear preference for the pin-cell coarse mesh, which is due to the reduced number of iterations to converge the shape function at each time step.

6.2 Conclusions

In summary, we have analyzed, described, implemented, and performed a sensitivity analysis on the Adiabatic, Theta, and MAF methods in 2D Method of Characteristics neutron transport. We have tested these methods on the transient transport versions of the LRA and C5G7 benchmarks. The results show the Adiabatic method is computationally more expensive for high dominance ratio problems that result in large spatial changes in the flux, but is much less sensitive to shape function time step size.

The MAF and Theta methods are preferred when the flux shape is rapidly changing and the shape function time step size is small because of the more computationally efficient fixed source shape function solve. All methods were shown to be sensitive to the amplitude function step size with small values required in order to converge the position of the power profile. In general, it is best to use a CMFD and TCMFD mesh on the pin-cell level where fine spatial flux features can be resolved and used to provide significant speed up for the shape function solve.

6.3 Recommendations for Future Work

In this thesis we have described the Adiabatic, Theta, and MAF methods and provided results for a sensitivity analysis on two small, simplified 2D transient problems. Real reactor analysis of transient scenarios will require the addition of more realistic feedback models, incorporation of thermal hydraulic feedback, thermal expansion of materials, and a more accurate representation of the flux behavior in the axial direction. This work has demonstrated that there are advantages to the Theta and MAF methods in certain scenarios as well as advantages to the Adiabatic method in others. Our recommendations for future work lie in two areas: incorporation of more detailed models to account for spatial and temperature dependence and extension of these and other similar methods to Monte Carlo neutron transport.

The implementation of the Adiabatic, Theta, and MAF methods presented in this thesis makes use of a flux factorization formalism whereby problems are broken up into fine mesh transport and coarse mesh nonlinear diffusion subproblems. The fine mesh transport subproblem exists merely to provide an update to the nonlinear diffusion problem allowing for the flux amplitude, delayed neutron precursors, and temperature feedback to be performed on the coarse mesh. There is a tremendous advantage to this approach as we solve larger problems - the convergence of these parameters can be done using a much more computationally efficient coarse mesh nonlinear diffusion problem. When the Adiabatic, Theta, and MAF methods are expanded to incorporate thermal hydraulic feedback, thermal expansion of materials,

and treatment of flux in the axial direction, it should be done in a way that allows for these characteristics to be converged on the coarse mesh. Ideally thermal hydraulic feedback, thermal expansion of materials, doppler feedback, and convergence of the delayed neutron precursor concentrations will be converged simultaneously on the coarse mesh at each amplitude function time step. Additionally, when performing the shape function update, feedback should be performed and converged after each MOC iteration's CMFD solve. The simplified buckling model for flux treatment in the axial direction will need to be updated with more realistic diffusion or transport theory models. Current codes have begun to implement full 3D MOC neutron transport, but there has been very little transient analysis performed in this area so it is not yet clear what models are needed to treat axial flux behavior.

We also recommend that these methods be extended to other neutron transport methods and in particular Monte Carlo. Implementation in Monte Carlo brings additional computational and statistical challenges. In particular, uncertainty quantification and propagation will need to be considered in analyzing the merit and performance of the methods. The Adiabatic method should be straightforward to implement and could offer the ability to simulate transients with large shape function time steps. Additionally, the frequency transform method should be considered whereby the fine mesh flux temporal derivative term in the transport equation can be approximated using a dynamic frequency that is computed from a coarse mesh solution. In conjunction with implementing the frequency transform method, nonlinear updates to the dynamic frequency during the Monte Carlo solve should be implemented at the same time that nonlinear thermal-hydraulic feedback is performed. In order for Monte Carlo to be a tangible method for transient analysis in the near term, the one thing this is clear is that methods will be required to use large shape function time steps. Flux factorization methods, like the ones studied in this thesis, show promise in being able to achieve this goal.

Our analysis has provided a glance at industrially relevant time-dependent reactor physics problems using neutron transport theory. While time-dependent neutron transport methods have come a long way in the past 7 decades, there is still much

work to be done before fast transport theory codes will be capable of solving large 3D, industrially relevant transient problems.

Appendix A

Test Problems

There do not exist many 2D transient LWR benchmark problems that have been thoroughly studied using transport theory. Numerous diffusion theory transient benchmarks in 2D and 3D exist with the LRA, TWIGL, and LMW benchmarks being some of the most widely studied [20]. Recently, a transport version of the LRA transient diffusion benchmark was proposed [16]. The LRA problem is characterized by a very large change in power ($> 10^{10}\times$) with doppler feedback. These two characteristics make the LRA problem challenging to model accurately and therefore allow us to differentiate between methods. In addition to a transient diffusion benchmark that has been modified into a transport problem, there has been an attempt to design a contrived transient problem from a static transport benchmark. In 2013, a transient problem for the C5G7 static 2D transport benchmark was proposed [1]. The C5G7 benchmark has spatially discretized fuel and moderator regions, which is not the case for the LRA benchmark. This discretization causes the flux shape to be more spatially undulating, which creates additional challenges in homogenization approaches to solving transient problems. Recently, the transport version of the TWIGL transient diffusion benchmark was solved with both studies reporting results that agree quite well [15, 1]. The TWIGL transient benchmark is a 2D unreflected seed-blanket reactor modeled with two energy groups, one delayed precursor group, and assembly-homogenized diffusion coefficients and cross sections. The transient has been modeled with both step and ramp reactivity insertions in one of the seeded regions of the core.

The problem is sufficient to allow spatial and temporal effects of flux changes to be analyzed, but the lack of feedback and a large (> 10x) power excursion makes the TWIGL problem relatively easy to solve with simple models that break down on more complex problems.

Taking into consideration the previous transient transport problems attempted, we have chosen to solve the transport version of the LRA benchmark as presented in [16] and a modified version of the transient C5G7 benchmark as presented in [1].

A.1 2D LRA Benchmark

The 2D LRA benchmark is a 2-group, quarter-core BWR transient problem [4]. The geometry consists of 78 15 cm × 15 cm homogenized fuel assemblies (regions 1-4) surrounded by water cells (region 5) to fill the 165 cm × 165 cm geometry as depicted in Figure A-1. The LRA problem was originally posed as a diffusion problem and has been extensively solved with coarse mesh and nodal diffusion methods [12]. The transport version of the LRA benchmark was originally published in [16] as a test problem for modern transport theory neutronics codes. In our analysis, we will use this version of the benchmark to allow ourselves and others to compare results. The cross sections for the initial state are presented in Table A.1.

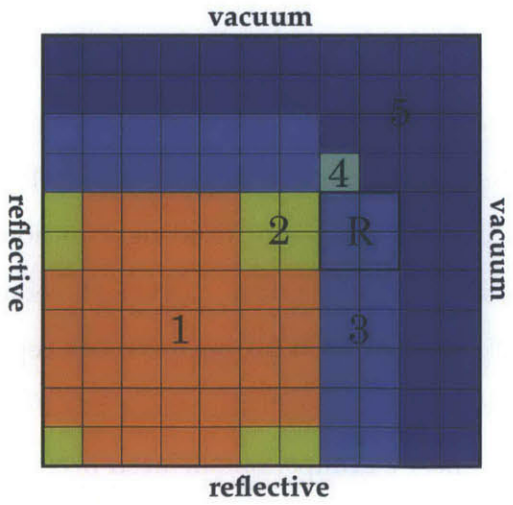

Figure A-1. The LRA benchmark geometry.

Table A.1. Material properties for transport version of 2D LRA benchmark

Region	g	$D\ [cm]$	$\Sigma^{tr}\ [cm^{-1}]$	$\Sigma^{A}\ [cm^{-1}]$	$\nu\Sigma^{F}\ [cm^{-1}]$	$\Sigma^{S}_{g\to 1}\ [cm^{-1}]$	$\Sigma^{S}_{g\to 2}\ [cm^{-1}]$
1	1	1.2550	0.265604	0.008252	0.004602	0.232022	0.025330
	2	0.2110	1.579779	0.100300	0.109100	0.000000	1.479479
2	1	1.2680	0.262881	0.007181	0.004609	0.228030	0.027670
	2	0.1902	1.752541	0.070470	0.086750	0.000000	1.682071
3,R	1	1.2590	0.264760	0.008002	0.004663	0.230588	0.026170
	2	0.2091	1.594134	0.083440	0.102100	0.000000	1.510694
4	1	1.2590	0.264760	0.008002	0.004663	0.230588	0.026170
	2	0.2091	1.594134	0.073324	0.102100	0.000000	1.520810
5	1	1.2570	0.265182	0.000603	0.000000	0.217039	0.047540
	2	0.1592	2.093802	0.019110	0.000000	0.000000	2.074692

Since the LRA benchmark does not specify total and in scattering cross sections, some estimation needs to be performed in order to obtain these terms. The total cross section can be approximated using the diffusion coefficient as shown in Equation A.1.

$$\Sigma^{T}_{g} \approx \Sigma^{tr}_{g} = \frac{1}{3D_g} \tag{A.1}$$

The in scattering cross section is then computed by requiring the absorption and scattering cross sections to equal the total cross section at all times as shown in Equation A.2. During the transient the absorption cross sections will change due to materials perturbation and feedback, so the in-group scattering cross sections will need to be recomputed in order to satisfy Equation A.2.

$$\Sigma^{S}_{g\to g} = \Sigma^{T}_{g} - \Sigma^{A}_{g} - \sum_{\substack{g'=1 \\ g'\neq g}}^{G} \Sigma^{S}_{g\to g'} \tag{A.2}$$

Temperature feedback is modeled with doppler on the fast (group 1) absorption cross sections in the fueled regions with the following equation:

$$\Sigma^{A}_{1}(t) = \Sigma^{A}_{1}(0)\left[1 + \gamma\left(\sqrt{T(t)} - \sqrt{T(0)}\right)\right] \tag{A.3}$$

The control blade drop in region 3R is modeled using a linear approximation on the thermal (group 2) absorption cross section:

$$\Sigma_2^A(t) = \begin{cases} \Sigma_2^A(0) \cdot [1 - (0.0606184 s^{-1} \cdot t)] & \text{if t} \leq 2 \text{ s} \\ \Sigma_2^A(0) \cdot [1 - (0.0606184 s^{-1} \cdot 2s)] & \text{if t} \geq 2 \text{ s} \end{cases} \tag{A.4}$$

The fueled material (regions 1-4) is assumed to heat up adiabatically according to the following equation:

$$\frac{\partial T_{fuel}(r,t)}{\partial t} = \alpha \sum_{g=1}^{G} \Sigma_g^F(r,t)\Phi_g(r,t) \tag{A.5}$$

The average power density, $\bar{P}$, and core temperature, $\bar{T}$, are given by:

$$\bar{P}(t) = \frac{1}{A_{core}} \int_{A_{core}} \kappa \sum_{g=1}^{G} \Sigma_g^F(r,t)\Phi(r,t)dA \tag{A.6}$$

$$\bar{T}(t) = \frac{1}{A_{core}} \int_{A_{core}} T(r,t)dA \tag{A.7}$$

Where A_{core} is the area of the fueled regions. Additional parameters used to solve the problem are listed in Table A.2.

Table A.2. Parameters used to solve LRA transient benchmark problem

Parameter	Symbol	Value
Neutron velocity (group 1)	v_1	3e7 $cm\ s^{-1}$
Neutron velocity (group 2)	v_2	3e5 $cm\ s^{-1}$
Delayed neutron fraction (delay group 1)	β_1	0.0054
Delayed neutron fraction (delay group 2)	β_2	0.001087
Precursor decay constant (delay group 1)	λ_1	0.00654 s^{-1}
Precursor decay constant (delay group 2)	λ_2	1.35 s^{-1}
Axial buckling	B_z^2	1e-4 cm^{-2}
Neutrons per fission	ν	2.43 $n/fission$
Temperature conversion factor	α	3.83e-11 $K\ cm^3$
Doppler feedback constant	γ	3.034e-3 $K^{-0.5}$
Thermal energy per fission	κ	3.204e-11 $J/fission$
Fission emission spectrum (group 1)	χ_1	1.0
Fission emission spectrum (group 2)	χ_2	0.0

The initial average power density in the core is:

$$\bar{P}(0) = \text{1e-6 } W/cm^3 \tag{A.8}$$

The initial temperature of all materials in the core is assumed to be:

$$T(r,0) = 300K \tag{A.9}$$

A.2 2D C5G7 Benchmark

The C5G7 transport benchmark was also studied due to the numerous benchmarking studies that have been performed on the static 2D unrodded case of this problem. This benchmark was presented as a static 2D and 3D, 7 energy group transport theory problem for benchmarking new reactor analysis codes. The C5G7 benchmark maintains heterogeneity between the fuel and moderator regions in a pin-cell, allowing developers to investigate homogenization approaches for lattice physics problems. Recently a transient version of the benchmark was proposed where a control rod drive ejection was simulated [1]. In this thesis, we present a similar transient version of the C5G7 benchmark problem with the incorporation of a thermal feedback model to allow the feedback and delayed neutron precursor effects to be modeled in a heterogeneous problem where a significant reactivity insertion ($> 1\$$) with a commensurate power excursion has occurred. For the thermal feedback, we constructed a doppler feedback model for the epithermal energy regions of the fuel similar to the doppler feedback model used in the LRA benchmark. While the transient version of the C5G7 benchmark is a contrived problem, the geometric specifications are much closer to what would be expected for a real reactor transient and the problem allows us to examine the effects of thermal feedback, flux shape modeling, and delayed neutron precursors in a heterogeneous medium.

The C5G7 problem was developed as a modern benchmark for deterministic neutron transport methods without spatial homogenization. The problem contains sixteen 17 $\times$ 17 pin-cell assemblies surrounded by water with vacuum boundary conditions on all sides. Typically, the problem is presented as a quarter core problem with

reflective boundary conditions on inner surfaces (Figure A-2)

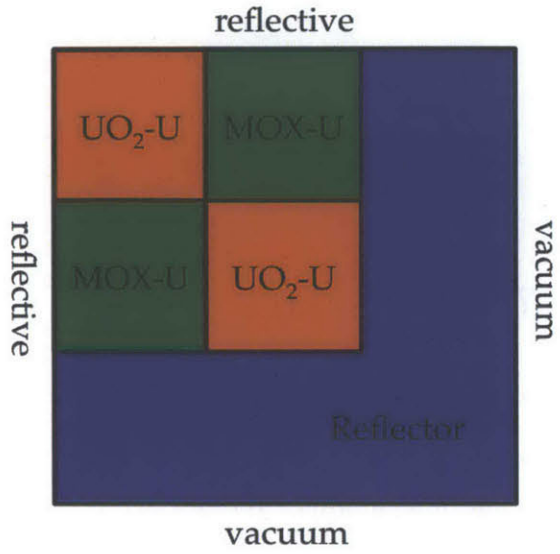

Figure A-2. The C5G7 benchmark problem.

The transport version of the problem is presented in the full sixteen assembly form to model a transient case where quarter core symmetry is not maintained. In our transient form of the benchmark, there is both an initial and final state (Figure A-3). The initial state is composed of the sixteen bundle C5G7 benchmark with each of the inner four assemblies containing control rods in the guide tube positions. In the final state, the control rods have been removed from the southeast rodded UO_2 bundle.

Each bundle consist of a 17 $\times$ 17 array of pins where each pin has a 0.54 cm radius with a pitch of 1.26 cm. Three bundle types, two UO_2 and one MOX, are presented in an initially quarter-core symmetric pattern and specified in Figure A-4. Each assembly contains a pattern of fuel pin-cells of varying enrichments, guide tubes, and fission chambers. Unrodded bundles are denoted as UO_2-U/MOX-U and rodded bundles are denoted UO_2-R.

The C5G7 problem uses 7 group transport corrected, isotropic scattering cross sections generated from a 69 group library using the collision probability code DRAGON. For reference, the cross section data has been included in subsection A.2.1. The control rod cross sections were taken to be the values presented in a previous version

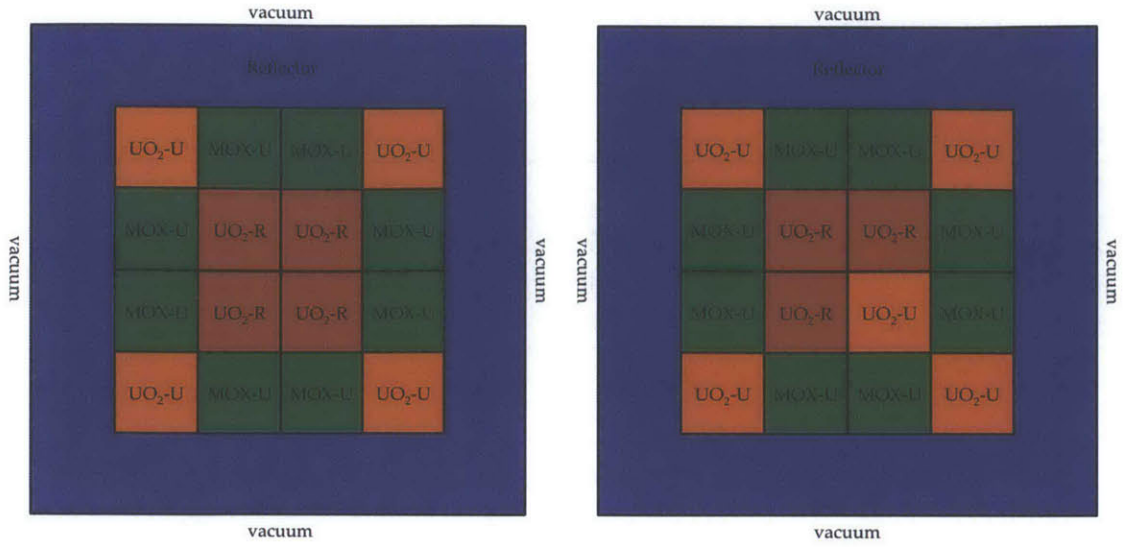

Figure A-3. Assembly layout for the initial state (left) and final state (right) of the transient version of the C5G7 benchmark. The suffix R and U denote rodded and unrodded, respectively.

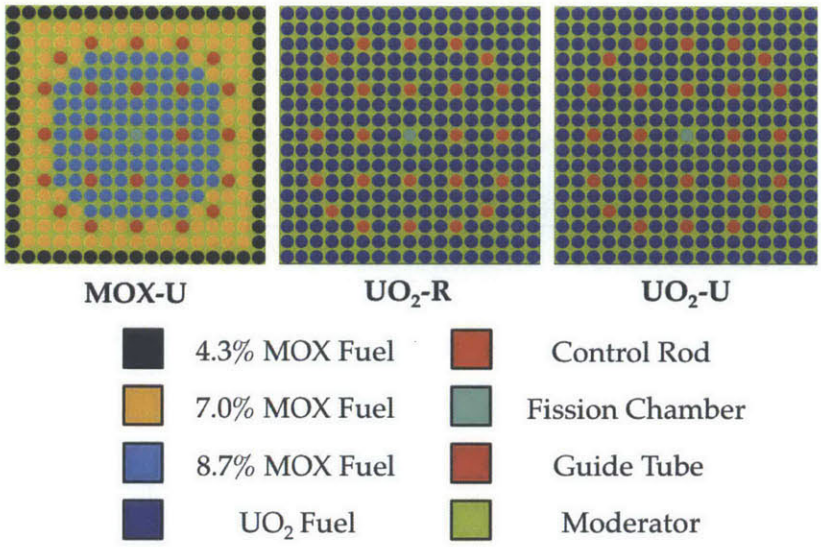

Figure A-4. The pin layouts for the three bundle types in the C5G7 benchmark problem.

of the C5G7 transient problem and not the actual values from the benchmark specification [1]. The C5G7 benchmark does not include kinetics, feedback, or delayed neutron precursor parameters, so representative parameters were adopted from other sources or, in the case of doppler feedback, a model was created. The kinetics and delayed neutron precursor parameters were taken from [1] and shown in Table A.3

and Table A.4.

Table A.3. Energy group structure and transient parameters for transient version of 2D C5G7 benchmark

Group	1	2	3	4	5	6	7
E_{top} (eV)	2e7	1e6	5e5	3e0	6.25e-1	1e-1	2e-2
E_{bottom} (eV)	1e6	5e5	3e0	6.25e-1	1e-1	2e-2	1e-5
v_{midpoint} (cm/s)	4.48e9	1.20e9	6.92e8	1.86e6	8.33e5	3.39e5	1.85e5

Table A.4. Delayed neutron group parameters for transient version of 2D C5G7 benchmark

Group	1	2	3	4	5	6
β	2.470e-4	1.3845e-3	1.222e-3	2.6455e-3	8.32e-4	1.690e-4
λ (s^{-1})	0.0127	0.0317	0.115	0.311	1.40	3.87

Additional parameters used to solve the C5G7 benchmark are included in Table A.5.

Table A.5. Parameters used to solve C5G7 transient benchmark problem

Parameter	**Symbol**	**Value**
Temperature conversion factor	α	3.83e-11 $K\ cm^3$
Thermal energy per fission	κ	3.204e-11 $J/fission$

A feedback model was created based on the assumption of doppler feedback in the epithermal region of fuel materials. We used a model similar to the LRA benchmark feedback model with doppler occurring for the C5G7 problem in group 3 (3 eV to 500 keV) as shown in Equation A.10.

$$\Sigma_3^A(t) = \Sigma_3^A(0)\left[1+\gamma\left(\sqrt{T(t)}-\sqrt{T(0)}\right)\right] \tag{A.10}$$

The doppler feedback coefficients for the fissile materials were set based on the assumption of an average feedback of -3 pcm/K from an initial state at 300K to a final state at 1000K. The cross sections from the benchmark were assumed to be

generated at 300K, giving Equation A.11 as the expression for computing the feedback coefficient.

$$\gamma = \frac{\frac{\Sigma_3^A(T=1000K)}{\Sigma_3^A(T=300K)} - 1}{\sqrt{1000K} - \sqrt{300K}} \tag{A.11}$$

The feedback coefficients were computed by making an infinite lattice pin-cell geometry of each fuel material and running a series of trials where the epithermal absorption cross section was steadily increased. Using the following expression, we were able to compute the reactivity, ρ, of the material as a function of absorption cross section.

$$\rho(\Sigma_3^A) = \frac{k_{inf}(\Sigma_3^A) - k_{inf}(\Sigma_3^A(0))}{k_{inf}(\Sigma_3^A(0))} \tag{A.12}$$

Figure A-5 shows a plot of the reactivity as function of change in Σ_3^A from the base case.

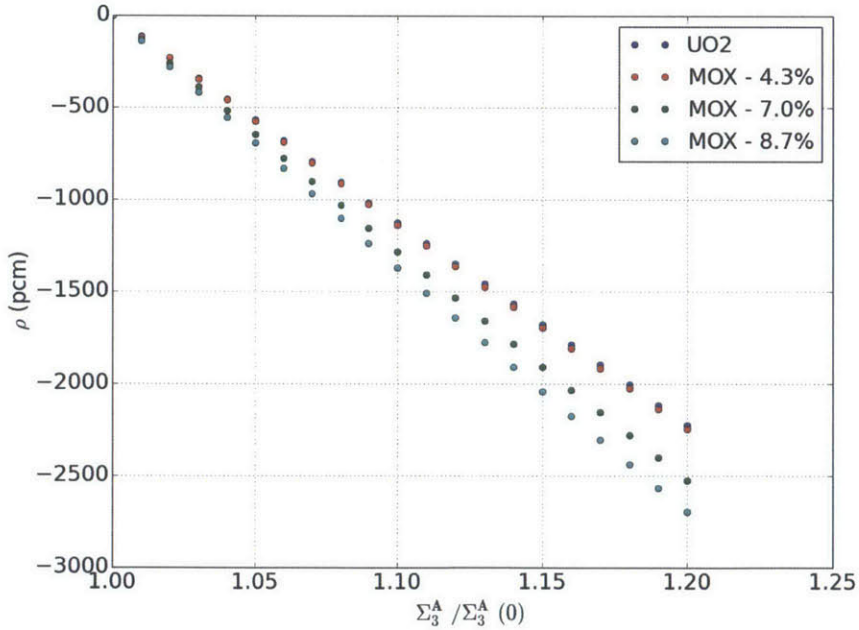

Figure A-5. Reactivity for infinite lattice pin-cell geometries for the various fuel types.

Using Equation A.11 and the data from Figure A-5 the doppler feedback coefficients were computed and are presented in Table A.6.

The doppler feedback coefficient for the fission chamber was assumed to be the

Table A.6. Doppler feedback coefficients for the fissile materials

Material	$\frac{\Sigma_3^A(1000K)}{\Sigma_3^A(300K)}$ @ $\rho = -3pcm/K$	γ ($K^{-0.5}$)
UO_2 Fuel	1.1881	1.315e-02
4.3% MOX Fuel	1.1862	1.302e-02
7.0% MOX Fuel	1.1655	1.157e-02
8.7% MOX Fuel	1.1546	1.081e-02

same as the UO_2 Fuel.

The control rods in the southeast UO_2-R assembly were ejected using a ramp change in material compositions over the course of two seconds. In the ramp reactivity model, the scattering and absorption cross sections of the rodded regions were linearly interpolated between the control rod material and the guide tube material cross sections as shown in Equation A.13. The transport cross sections were adjusted for all fissile materials (including fission chambers) such that they equal the sum of the scattering and absorption cross sections at all times as shown in Equation A.14.

$$\Sigma^{A,S}(t) = \begin{cases} \Sigma^{A,S}_{\text{control rod}}\left(\frac{2-t}{2}\right) + \Sigma^{A,S}_{\text{guide tube}}\left(\frac{t}{2}\right) & \text{if t} \leq 2 \text{ s} \\ \Sigma^{A,S}_{\text{guide tube}} & \text{if t} \geq 2 \text{ s} \end{cases} \tag{A.13}$$

$$\Sigma_g^{tr} = \Sigma_g^A + \sum_{g\prime=1}^{G} \Sigma^S_{g\rightarrow g\prime} \tag{A.14}$$

The initial average power density in the fissile materials is assumed to be:

$$\bar{P}(0) = \text{1e-6 } W/cm^3 \tag{A.15}$$

The initial temperature of all materials in the core is assumed to be:

$$T(r,0) = 300K \tag{A.16}$$

The fissile materials (including the fission chamber) are assumed to heat up adiabatically according to Equation A.17. Thermal hydraulic feedback was not considered during the transient and therefore the moderator temperature and cross sections were

assumed to be constant at their initial values.

$$\frac{\partial T_{fuel}(r,t)}{\partial t} = \alpha \sum_{g}^{G} \Sigma_g^F(r,t)\Phi_g(r,t) \tag{A.17}$$

The average power density, $\bar{P}$, and was taken to be:

$$\bar{P}(t) = \frac{1}{A_{fuel}} \int_{A_{fuel}} \kappa \sum_{g}^{G} \Sigma_g^F(r,t)\Phi(r,t)dA \tag{A.18}$$

Where A_{fuel} is the area of the fissile materials (excluding water around pin cells) in the core.

A.2.1 C5G7 Cross Sections

Table A.7. Control rod macroscopic cross sections

Group	Σ^{tr} $[cm^{-1}]$	Σ^{A} $[cm^{-1}]$	Σ^{F} $[cm^{-1}]$
1	1.3511e−01	6.3068e−04	0.00000e+00
2	3.1185e−01	9.0445e−04	0.00000e+00
3	3.4445e−01	8.6632e−03	0.00000e+00
4	3.4987e−01	4.0822e−02	0.00000e+00
5	3.9204e−01	7.2934e−02	0.00000e+00
6	6.2285e−01	1.0122e−01	0.00000e+00
7	1.2390e+00	1.3875e−01	0.00000e+00

Table A.8. Control rod macroscopic scattering cross sections

	$\Sigma^S_{\rightarrow g}$						
$\Sigma^S_{g\rightarrow}$	1	2	3	4	5	6	7
1	7.6606e−02	5.7603e−02	2.6484e−04	1.3288e−06	1.8578e−08	0.0000e+00	0.00000e+00
2	0.0000e+00	2.6344e−01	4.7260e−02	2.2491e−04	1.7315e−05	2.6888e−06	3.7926e−07
3	0.0000e+00	0.0000e+00	2.4515e−01	8.3229e−02	6.2501e−03	9.7227e−04	1.8510e−04
4	0.0000e+00	0.0000e+00	0.0000e+00	1.2804e−01	1.5327e−01	2.3293e−02	4.4367e−03
5	0.0000e+00	0.0000e+00	0.0000e+00	4.0155e−05	1.1055e−01	1.8649e−01	2.2030e−02
6	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	9.2796e−04	3.0534e−01	2.1537e−01
7	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	4.5166e−02	1.0551e+00

Table A.9. UO_2 fuel-clad macroscopic cross sections

Group	Σ^{tr} $[cm^{-1}]$	Σ^{A} $[cm^{-1}]$	Σ^{F} $[cm^{-1}]$	ν	χ
1	1.77949e−01	8.02480e−03	7.21206e−03	2.78145	5.87910e−01
2	3.29805e−01	3.71740e−03	8.19301e−04	2.47443	4.11760e−01
3	4.80388e−01	2.67690e−02	6.45320e−03	2.43383	3.39060e−04
4	5.54367e−01	9.62360e−02	1.85648e−02	2.43380	1.17610e−07
5	3.11801e−01	3.00200e−02	1.78084e−02	2.43380	0.00000e+00
6	3.95168e−01	1.11260e−01	8.30348e−02	2.43380	0.00000e+00
7	5.64406e−01	2.82780e−01	2.16004e−01	2.43380	0.00000e+00

Table A.10. UO_2 fuel-clad macroscopic scattering cross sections

	$\Sigma^S_{\rightarrow g}$						
$\Sigma^S_{g\rightarrow}$	1	2	3	4	5	6	7
1	1.27537e−01	4.23780e−02	9.43740e−06	5.51630e−09	0.00000e+00	0.00000e+00	0.00000e+00
2	0.00000e+00	3.24456e−01	1.63140e−03	3.14270e−09	0.00000e+00	0.00000e+00	0.00000e+00
3	0.00000e+00	0.00000e+00	4.50940e−01	2.67920e−03	0.00000e+00	0.00000e+00	0.00000e+00
4	0.00000e+00	0.00000e+00	0.00000e+00	4.52565e−01	5.56640e−03	0.00000e+00	0.00000e+00
5	0.00000e+00	0.00000e+00	0.00000e+00	1.25250e−04	2.71401e−01	1.02550e−02	1.00210e−08
6	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	1.29680e−03	2.65802e−01	1.68090e−02
7	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	8.54580e−03	2.73080e−01

Table A.11. 4.3 % MOX fuel-clad macroscopic cross sections

Group	Σ^{tr} $[cm^{-1}]$	Σ^A $[cm^{-1}]$	Σ^F $[cm^{-1}]$	ν	χ
1	1.78731e−01	8.43390e−03	7.62704e−03	2.85209	5.87910e−01
2	3.30849e−01	3.75770e−03	8.76898e−04	2.89099	4.11760e−01
3	4.83772e−01	2.79700e−02	5.69835e−03	2.85486	3.39060e−04
4	5.66922e−01	1.04210e−01	2.28872e−02	2.86073	1.17610e−07
5	4.26227e−01	1.39940e−01	1.07635e−02	2.85447	0.00000e+00
6	6.78997e−01	4.09180e−01	2.32757e−01	2.86415	0.00000e+00
7	6.82852e−01	4.09350e−01	2.48968e−01	2.86780	0.00000e+00

Table A.12. 4.3 % MOX fuel-clad macroscopic scattering cross sections

	$\Sigma^S_{\rightarrow g}$						
$\Sigma^S_{g\rightarrow}$	1	2	3	4	5	6	7
1	1.28876e−01	4.14130e−02	8.22900e−06	5.04050e−09	0.00000e+00	0.00000e+00	0.00000e+00
2	0.00000e+00	3.25452e−01	1.63950e−03	1.59820e−09	0.00000e+00	0.00000e+00	0.00000e+00
3	0.00000e+00	0.00000e+00	4.53188e−01	2.61420e−03	0.00000e+00	0.00000e+00	0.00000e+00
4	0.00000e+00	0.00000e+00	0.00000e+00	4.57173e−01	5.53940e−03	0.00000e+00	0.00000e+00
5	0.00000e+00	0.00000e+00	0.00000e+00	1.60460e−04	2.76814e−01	9.31270e−03	9.16560e−09
6	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	2.00510e−03	2.52962e−01	1.48500e−02
7	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	8.49480e−03	2.65007e−01

Table A.13. 7.0 % MOX fuel-clad macroscopic cross sections

Group	Σ^{tr} $[cm^{-1}]$	Σ^A $[cm^{-1}]$	Σ^F $[cm^{-1}]$	ν	χ
1	1.81323e−01	9.06570e−03	8.25446e−03	2.88498	5.87910e−01
2	3.34368e−01	4.29670e−03	1.32565e−03	2.91079	4.11760e−01
3	4.93785e−01	3.28810e−02	8.42156e−03	2.86574	3.39060e−04
4	5.91216e−01	1.22030e−01	3.28730e−02	2.87063	1.17610e−07
5	4.74198e−01	1.82980e−01	1.59636e−02	2.86714	0.00000e+00
6	8.33601e−01	5.68460e−01	3.23794e−01	2.86658	0.00000e+00
7	8.53603e−01	5.85210e−01	3.62803e−01	2.87539	0.00000e+00

Table A.14. 7.0 % MOX fuel-clad macroscopic scattering cross sections

	$\Sigma^S_{\rightarrow g}$						
$\Sigma^S_{g\rightarrow}$	1	2	3	4	5	6	7
1	1.30457e−01	4.17920e−02	8.51050e−06	5.13290e−09	0.00000e+00	0.00000e+00	0.00000e+00
2	0.00000e+00	3.28428e−01	1.64360e−03	2.20170e−09	0.00000e+00	0.00000e+00	0.00000e+00
3	0.00000e+00	0.00000e+00	4.58371e−01	2.53310e−03	0.00000e+00	0.00000e+00	0.00000e+00
4	0.00000e+00	0.00000e+00	0.00000e+00	4.63709e−01	5.47660e−03	0.00000e+00	0.00000e+00
5	0.00000e+00	0.00000e+00	0.00000e+00	1.76190e−04	2.82313e−01	8.72890e−03	9.00160e−09
6	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	2.27600e−03	2.49751e−01	1.31140e−02
7	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	8.86450e−03	2.59529e−01

Table A.15. 8.7 % MOX fuel-clad macroscopic cross sections

Group	Σ^{tr} $[cm^{-1}]$	Σ^{A} $[cm^{-1}]$	Σ^{F} $[cm^{-1}]$	ν	χ
1	1.83045e−01	9.48620e−03	8.67209e−03	2.90426	5.87910e−01
2	3.36705e−01	4.65560e−03	1.62426e−03	2.91795	4.11760e−01
3	5.00507e−01	3.62400e−02	1.02716e−02	2.86986	3.39060e−04
4	6.06174e−01	1.32720e−01	3.90447e−02	2.87491	1.17610e−07
5	5.02754e−01	2.08400e−01	1.92576e−02	2.87175	0.00000e+00
6	9.21028e−01	6.58700e−01	3.74888e−01	2.86752	0.00000e+00
7	9.55231e−01	6.90170e−01	4.30599e−01	2.87808	0.00000e+00

Table A.16. 8.7 % MOX fuel-clad macroscopic scattering cross sections

	$\Sigma^{S}_{\rightarrow g}$						
$\Sigma^{S}_{g\rightarrow}$	1	2	3	4	5	6	7
1	1.31504e−01	4.20460e−02	8.69720e−06	5.19380e−09	0.00000e+00	0.00000e+00	0.00000e+00
2	0.00000e+00	3.30403e−01	1.64630e−03	2.60060e−09	0.00000e+00	0.00000e+00	0.00000e+00
3	0.00000e+00	0.00000e+00	4.61792e−01	2.47490e−03	0.00000e+00	0.00000e+00	0.00000e+00
4	0.00000e+00	0.00000e+00	0.00000e+00	4.68021e−01	5.43300e−03	0.00000e+00	0.00000e+00
5	0.00000e+00	0.00000e+00	0.00000e+00	1.85970e−04	2.85771e−01	8.39730e−03	8.92800e−09
6	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	2.39160e−03	2.47614e−01	1.23220e−02
7	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	8.96810e−03	2.56093e−01

Table A.17. Fission chamber macroscopic cross sections

Group	Σ^{tr} $[cm^{-1}]$	Σ^{A} $[cm^{-1}]$	Σ^{F} $[cm^{-1}]$	ν	χ
1	1.26032e−01	5.11320e−04	4.79002e−09	2.76283	5.87910e−01
2	2.93160e−01	7.58130e−05	5.82564e−09	2.46239	4.11760e−01
3	2.84250e−01	3.16430e−04	4.63719e−07	2.43380	3.39060e−04
4	2.81020e−01	1.16750e−03	5.24406e−06	2.43380	1.17610e−07
5	3.34460e−01	3.39770e−03	1.45390e−07	2.43380	0.00000e+00
6	5.65640e−01	9.18860e−03	7.14972e−07	2.43380	0.00000e+00
7	1.17214e+00	2.32440e−02	2.08041e−06	2.43380	0.00000e+00

Table A.18. Fission chamber macroscopic scattering cross sections

	$\Sigma^{S}_{\rightarrow g}$						
$\Sigma^{S}_{g\rightarrow}$	1	2	3	4	5	6	7
1	6.61659e−02	5.90700e−02	2.83340e−04	1.46220e−06	2.06420e−08	0.00000e+00	0.00000e+00
2	0.00000e+00	2.40377e−01	5.24350e−02	2.49900e−04	1.92390e−05	2.98750e−06	4.21400e−07
3	0.00000e+00	0.00000e+00	1.83425e−01	9.22880e−02	6.93650e−03	1.07900e−03	2.05430e−04
4	0.00000e+00	0.00000e+00	0.00000e+00	7.90769e−02	1.69990e−01	2.58600e−02	4.92560e−03
5	0.00000e+00	0.00000e+00	0.00000e+00	3.73400e−05	9.97570e−02	2.06790e−01	2.44780e−02
6	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	9.17420e−04	3.16774e−01	2.38760e−01
7	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	4.97930e−02	1.09910e+00

Table A.19. Guide tube macroscopic cross sections

Group	Σ^{tr} $[cm^{-1}]$	Σ^{A} $[cm^{-1}]$	Σ^{F} $[cm^{-1}]$
1	1.26032e−01	5.11320e−04	0.00000e+00
2	2.93160e−01	7.58010e−05	0.00000e+00
3	2.84240e−01	3.15720e−04	0.00000e+00
4	2.80960e−01	1.15820e−03	0.00000e+00
5	3.34440e−01	3.39750e−03	0.00000e+00
6	5.65640e−01	9.18780e−03	0.00000e+00
7	1.17215e+00	2.32420e−02	0.00000e+00

Table A.20. Guide tube macroscopic scattering cross sections

	$\Sigma^S_{\rightarrow g}$						
$\Sigma^S_{g\rightarrow}$	1	2	3	4	5	6	7
1	6.61659e−02	5.90700e−02	2.83340e−04	1.46220e−06	2.06420e−08	0.00000e+00	0.00000e+00
2	0.00000e+00	2.40377e−01	5.24350e−02	2.49900e−04	1.92390e−05	2.98750e−06	4.21400e−07
3	0.00000e+00	0.00000e+00	1.83297e−01	9.23970e−02	6.94460e−03	1.08030e−03	2.05670e−04
4	0.00000e+00	0.00000e+00	0.00000e+00	7.88511e−02	1.70140e−01	2.58810e−02	4.92970e−03
5	0.00000e+00	0.00000e+00	0.00000e+00	3.73330e−05	9.97372e−02	2.06790e−01	2.44780e−02
6	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	9.17260e−04	3.16765e−01	2.38770e−01
7	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	4.97920e−02	1.09912e+00

Table A.21. Moderator macroscopic cross sections

Group	Σ^{tr} $[cm^{-1}]$	Σ^{A} $[cm^{-1}]$	Σ^{F} $[cm^{-1}]$
1	1.59206e−01	6.01050e−04	0.00000e+00
2	4.12970e−01	1.57930e−05	0.00000e+00
3	5.90310e−01	3.37160e−04	0.00000e+00
4	5.84350e−01	1.94060e−03	0.00000e+00
5	7.18000e−01	5.74160e−03	0.00000e+00
6	1.25445e+00	1.50010e−02	0.00000e+00
7	2.65038e+00	3.72390e−02	0.00000e+00

Table A.22. Moderator macroscopic scattering cross sections

	$\Sigma^S_{\rightarrow g}$						
$\Sigma^S_{g\rightarrow}$	1	2	3	4	5	6	7
1	4.44777e−02	1.13400e−01	7.23470e−04	3.74990e−06	5.31840e−08	0.00000e+00	0.00000e+00
2	0.00000e+00	2.82334e−01	1.29940e−01	6.23400e−04	4.80020e−05	7.44860e−06	1.04550e−06
3	0.00000e+00	0.00000e+00	3.45256e−01	2.24570e−01	1.69990e−02	2.64430e−03	5.03440e−04
4	0.00000e+00	0.00000e+00	0.00000e+00	9.10284e−02	4.15510e−01	6.37320e−02	1.21390e−02
5	0.00000e+00	0.00000e+00	0.00000e+00	7.14370e−05	1.39138e−01	5.11820e−01	6.12290e−02
6	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	2.21570e−03	6.99913e−01	5.37320e−01
7	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	1.32440e−01	2.48070e+00

Bibliography

[1] A. Hoffman. A Time-Dependent Method of Characteristics Formulation with Time Derivative Propagation. Ph.D. Thesis, University of Michigan, 2013.

[2] A. Yamamoto. Generalized Coarse-Mesh Rebalance Method for Acceleration of Neutron Transport Calculations. *Journal of Nuclear Science and Engineering*, 151:274–282, 2005.

[3] A. Yamamoto, M. Tabuchi, N. Sugimura, T. Ushio and M. Mori. Derivation of Optimum Polar Angle Quadrature Set for the Method of Characteristics Based on Approximation Error for the Bickley Function. *Journal of Nuclear Science and Engineering*, 44(2):129–136, 2007.

[4] Argonne National Laboratory. Argonne Code Center: Benchmark Problem Book, ANL-7416, Supplement 2. `http://www.corephysics.com/benchmarks/anl7416_benchmark11.pdf`, 1977. [Online; accessed 1/3/2014].

[5] E. E. Lewis, G. Palmiotti, T. A. Taiwo, R. N. Blomquist, M. A. Smith and N. Tsoulfanidis. Benchmark Specifications for Deterministic MOX Fuel Assembly Transport Calculations without Spatial Homogenization. Technical report, Organisation for Economic Co-operation and Development's Nuclear Energy Agency, 2003.

[6] E. Larsen. Infinite medium solutions to the transport equation, s_n discretization schemes, and the diffusion approximation. In *Proc. Mathematics and Computational Methods Applied to Nuclear Science & Engineering.* M&C 2001, 2001. Salt Lake City, UT, September, 2001.

[7] H. G. Joo, J. Y. Cho, J. S. Song, S. Q. Zee, T. J. Downar. Multigroup transient calculation within the framework of a two-group hexagonal cmfd formulation. In *Proc. Mathematics and Computational Methods Applied to Nuclear Science & Engineering.* M&C 2001, 2001. Salt Lake City, UT, September, 2001.

[8] J. R. Askew. A Characteristics Formulation of the Neutron Transport Equation in Complicated Geometries. Technical Report AAEW-M 1108, UK Atomic Energy Establishment, 1972.

[9] J. Y. Cho, H. G. Joo, K. S. Kim and S. Q. Zee. Cell Based CMFD Formulation for Acceleration of Whole-Core Method of Characteristics Calculations. *Journal of the Korean Nuclear Society*, 34(3):250–258, 2002.

[10] J. Y. Cho, H. G. Joo, S. Y. Park, and S. Q. Zee. Consistent Group Collapsing Scheme for multi-group MOC Calculation. In *Proceedings of PHYSOR*, Seoul, South Korea, 2002.

[11] J. Y. Cho, K. S. Kim, C. C. Lee, H. G. Joo, W. S. Yang, T. A. Taiwo, and J. Thomas. Transient Capability for a MOC-Based Whole Core Transport Code DeCART. *Transactions of the American Nuclear Society*, 92(1):721–722, 2005.

[12] K. S. Smith. An Analytical Nodal Method for Solving the Two-group, Multi-dimensional, Static and Transient Neutron Diffusion Equations. M.S. Thesis, Massachusetts Institute of Technology, 1979.

[13] K. S. Smith. Nodal Method Storage Reduction by Non-linear Iteration. *Transactions of the American Nuclear Society*, 44(265), 1983.

[14] K. S. Smith and J. D. Rhodes. Full-Core, 2-D, LWR Core Calculations with CASMO-4E. In *Proceedings of PHYSOR*, Seoul, South Korea, 2002.

[15] T. Endo K. Tsujita and A. Yamamoto. Application of the multigrid amplitude function method for time-dependent transport equation using moc. In *Proc. Mathematics and Computational Methods Applied to Nuclear Science & Engineering.* M&C 2013, 2013. Sun Valley, ID, May 5-9, 2013.

[16] K. Tsujita, T. Endo, A. Yamamoto, Y. Kamiyama, and K. Kirimura. Higher order treatment on temporal derivative of angular flux for time-dependent moc. In *Proc. Mathematics and Computational Methods Applied to Nuclear Science & Engineering.* M&C 2013, 2013. Sun Valley, ID, May 5-9, 2013.

[17] L. Li. A Low Order Acceleration Scheme for Solving the Neutron Transport Equation. M.S. Thesis, Massachusetts Institute of Technology, 2013.

[18] E. E. Lewis and Jr. W. F. Miller, editors. *Computational Methods of Neutron Transport.* John Wiley & Sons, New York, 1984.

[19] R. Ferrer, J. Rhodes and K. Smith. Linear Source Approximation in CASMO5. In *Proceedings of PHYSOR*, Knoxville, TN, USA, 2012.

[20] T. Sutton and B. Aviles. Diffusion theory methods for spatial kinetics calculations. *Progress in Nuclear Energy*, 30(2):119–182, 1996.

[21] W. Boyd. Massively Parallel Algorithms for Method of Characteristics Neutral Particle Transport on Shared Memory Computer Architectures. M.S. Thesis, Massachusetts Institute of Technology, 2014.

[22] W. Boyd, S. C. Shaner, L. Li, B. Forget, K. S. Smith. The openmoc method of characteristics neutral particle transport code. *Annals of Nuclear Energy*, In Press, 2014.

[23] T. Endo Y. Ban and A. Yamamoto. A unified approach for numerical calculation of space-dependent kinetic equation. *Journal of Nuclear Science and Technology*, 49(5):496–515, 2012.

[24] Z. Zhong, T. J. Downar, Y. Xu, M. D. DeHart and K. T. Clarno. Implementation of Two-Level Coarse Mesh Finite Difference Acceleration in an Arbitrary Geometry, Two-Dimensional Discrete Ordinates Transport Method. *Journal of Nuclear Science and Engineering*, 158(3):289–298, 2008.