A LOW ORDER ACCELERATION SCHEME FOR SOLVING THE NEUTRON TRANSPORT EQUATION

by

Lulu Li

B.S., Physics (2011)
University of Illinois at Urbana-Champaign

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

September 2013

Signature of Author . . . .
Lulu Li
Department of Nuclear Science and Engineering
September 2013

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

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

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

A Low Order Acceleration Scheme for Solving the Neutron Transport Equation

by

Lulu Li

Submitted to the Department of Nuclear Science and Engineering
on September 2013, in partial fulfillment of the
requirements for the degree of
Master of Science in Nuclear Science and Engineering

Abstract

The Methods of Characteristics (MOC) is a widely used technique for solving partial differential equations, and has been applied to the neutron transport problems for many years. The MOC method requires many transport iterations to solve large heterogeneous LWR reactor problems with high dominance ratio, and effective acceleration schemes are necessary to make MOC method practical.

Various acceleration methods have been developed using low-order diffusion methods for approximating the scalar flux correction to the high-order scalar flux, and limited work has been performed using a low-order transport solution to accelerate the high-order transport solution.

This work proposes a Low Order Operator (LOO) acceleration scheme for accelerating the transport equation. More specifically, LOO uses a coarsely discretized grid and iteratively solves the low-order system using MOC transport approximations. By conserving the first-order spatial and angular moments, LOO is proposed to capture more angular effects compared with CMFD.

Two variations of the LOO method, together with the CMFD method, are implemented in the OpenMOC framework, which is a 2D MOC solver written to solve the 2D heterogeneous reactor problems. Based on the test cases performed in this work, LOO tends to reduce the number of transport sweeps required compared with the commonly used CMFD acceleration method. LOO also does not rely on under-relaxation as CMFD does to converge typical LWR problems tested in this work. The advantage of LOO over CMFD is more profound for problems with strong angular effects.

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

I would like to express my greatest gratitude towards my thesis advisors Prof. Kord Smith and Prof. Benoit Forget for their mentorship, patience and insights. I am grateful that Prof. Forget took me as an intern when I was a physics undergraduate student during the summer of 2010. He opened up the door to reactor physics for me through pages and pages of notes and sketches and was extremely patient at explaining the same concepts over and over again. I would like to thank both professors for coming up with the OpenMOC project, and for helping me finding the research project that I am passionate about. In particular, the low-order transport operator scheme was formulated by Prof. Smith a couple of years ago, and I am very fortunate to have the opportunity to put it into practices. I benefited tremendously from Prof. Smith and Prof. Forget, both from their lectures offered at MIT and from interactions with them through research. Their doors have always been open to all of their students, and I am extremely grateful for having them as my advisors.

I would also like to thank Dr. Rodolfo Ferrer from Studsvik Scandpower for his mentorship during summer 2013. Dr. Ferrer is very knowledgeable about reactor physics, and he bared with me when I stormed into his office, sometimes multiple times a day, for help and discussion on random questions on transport theory. He suggested solving the low-order scalar fluxes through the balance equation which led to the LOO-balance method presented in this work. Dr. Ferrer has also volunteered to read this thesis, and I owe my gratitude to him for the immeasurable help he has provided.

Other mentors I would like to acknowledge are Dr. Tamer Bahadir and Dr. David Dean for their patience and knowledge on nuclear reactor systems, and for their continuous mentorship beyond the summer that I spent there. I would like to thank Studsvik Scandpower for their generous funding and for their employees' mental and intellectual support.

Last but not least, I would like to thank all my peers and other MIT NSE members who have shared the last two years with me. It was an extreme pleasure to work with Will Boyd and Sam Shaner on the OpenMOC project. I was fortunate to share office with Dr. Koroush Shirvan who has always been a great support and are very generous about sharing his knowledge and understanding on all nuclear matters. Jeremy Roberts, Bryan Herman and Nathan Gibson from our research group have provided guidance, encouragement and help for me. I would not be where I am without any of the people mentioned above.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and Motivation

The Methods of Characteristics (MOC) is a widely used technique for solving partial differential equations, and has been applied to the neutron transport problems for many years. MOC in neutron transport was first proposed by Askew in 1972 [Askew, 1972], and has been implemented in various codes including: the CACTUS module in WIMS [Halsall, 1980], CASMO-4 [Knott et al., 1995], DRAGON [Marleau et al., 1994] and APOLLO2 [Sanchez et al., 2010].

The MOC method uses an iterative approach to solve the neutron transport equation where in each transport iteration (or sweep) the angular and scalar flux are updated based on the neutron sources calculated from the previous iteration's flux. While MOC is an attractive method to solve the neutron transport equation, it requires many iterative transport sweeps to resolve large heterogeneous LWR problems making it impractical for direct implementation. Consequently, various acceleration schemes have been developed to converge the angular fluxes in fewer number of transport sweeps.

Some of the major acceleration schemes include: diffusion synthetic acceleration (DSA), transport synthetic acceleration (TSA), projected discrete ordinates (PDO), coarse-mesh rebalance (CMR), and coarse-mesh finite difference (CMFD) [Adams and Larsen, 2002]. In some literature the term CMFD is used interchangeably with Nonlinear Diffusion Acceleration (NDA) [Knoll et al., 2011b, Park et al., 2011, Willert and Kelley, 2013]. Among the various acceleration schemes, DSA, CMR and CMFD are very popular in practice. All three methods are based on the use of a low-order diffusion solution to approximate the scalar flux correction of the high-order scalar flux, and limited work has been performed using a low-order transport equations for the acceleration method [Grassi, 2007].

In this work, a Low Order Operator (LOO) acceleration scheme is developed by employing a low-order transport solver to conserve the first-order spatial and angular moments. In doing so, LOO has advantages over diffusion-based acceleration schemes in solving problems where transport effects dominate diffusion effects. Two variations of the LOO methods will be compared to the CMFD method for a series of test cases.

1.2 Overview

The approach used to develop the LOO acceleration scheme is based on the following steps:

- Implement an extensible 2D MOC framework called OpenMOC for solving the high-order neutron transport problems using MOC: Chapter 1;
- Implement the classic CMFD acceleration scheme and briefly assess its performance in accelerating the high-order MOC solver: Chapter 2;
- Develop two variations of the LOO acceleration scheme: Chapter 3;
- Present numerical results comparing the LOO acceleration with the CMFD acceleration for a series of test cases ranging from the simple pin cell problems to assembly problems, to C5G7 benchmark problem: Chapter 4.

The rest of this chapter will be devoted to the discussion of the method of characteristics theory and implementation because MOC provides the basic framework for developing the acceleration scheme. The MOC solver will be discussed without going into too much details, as MOC has been implemented by numerous organizations as mentioned in Section 1.1.

Methods for solving the neutron transport equation using MOC will be presented in Section 1.3, and more details of the MOC solver will be provided in Section 1.4, followed by description of the OpenMOC framework in Section 1.5.

1.3 MOC Method

This section derives the Method of Characteristic for solving the neutron transport equation. This section starts by defining the notation in Section 1.3.1 and simplifying the neutron transport equation in Section 1.3.2.

Then the next two sections will be devoted to deriving the outgoing segment angular flux given the incoming segment angular flux (Section 1.3.3), and the scalar flux in each flat source region (Section 1.3.4). As Grassi puts it in [Grassi, 2007], the MOC method consists of two components: prolongating the incoming angular flux to get the outgoing angular flux, and obtaining the angular flux and scalar flux in each region. These two components are the most important concepts in MOC and are treated separately. These formulations will be used in Section 1.4 where detailed MOC implementation will be discussed.

1.3.1 Notation

MOC theory is presented in multiple texts: Ch. 9 in Handbook of Nuclear Engineering [Knott and Yamamoto, 2010], Section 3.10 in Applied Reactor Physics [Hebert, 2009] and Ch. 6 in CASMO-4 Manual [Knott et al., 1995] to name just a few.

Different texts use different notations for variables, thus it is important to start with notations before further discussion of MOC:

1. Subscripts:

    - g: energy group index of the multi-group structure;
    - i: high-order flat source region index;
    - I: low-order coarse mesh cell index;
    - k: segment index; k is typically used together with i and m to represent segments crossing the i-th region at the angle Ω_m;
    - m: angle index, combining 'a' for azimuthal angle index and 'p' for polar angle index. For instance, φ_a is the azimuthal angle, θ_p is the polar angle;

    The cross-section subscripts are conventional:

    - $\Sigma_{s,g' \to g}$: scattering cross-section for group g' to group g;
    - $\Sigma_{t,g}$: total cross-section for group g;
    - $\Sigma_{f,g}$: fission cross-section for group g;
    - ν is the number of neutrons per fission.

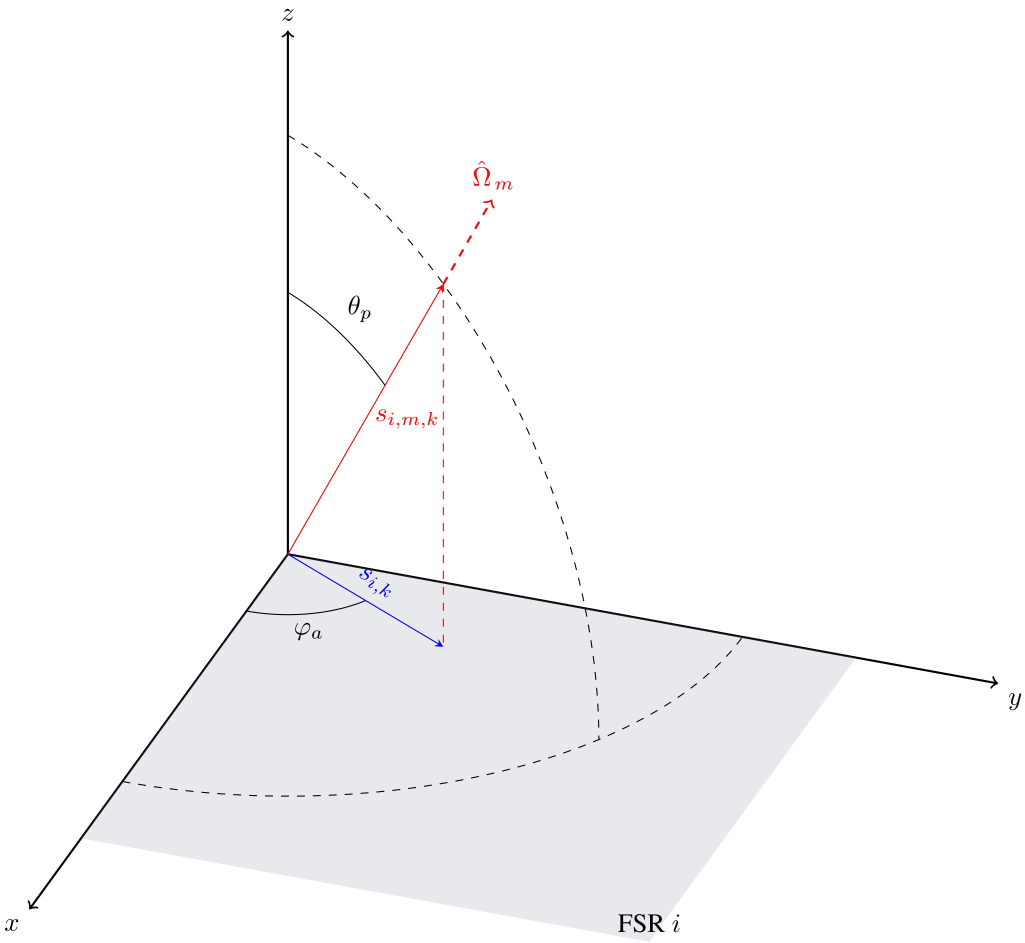

Figure 1-1: Illustration of the i, a, p, m, k Indexes

The subscript indexes are illustrated in Fig. 1-1. In the x-y plane, the shaded area is the i-th FSR, φ_a is the azimuthal angle, and $s_{i,k}$ is the in-plane length of the k-th segment in the i-th FSR. In the 3D domain, θ_p is the polar angle, $\hat{\Omega}_m$ is the solid angle, and $s_{i,m,k}$ is the k-th segment length in the upper angular half-space.

2. Superscripts: the superscripts are used to designate the high-order and low-order iteration number. The convention for iteration number differs among literature, thus Fig. 1-2 is produced to clarify the notation used in this study.

   - (m) for 'm'oc: the iteration number of variables entering the high-order MOC transport sweep [1];
   - $(m+1/2)$: the iteration number for variables computed from the m-th MOC transport sweep;
   - $(m+1)$: the iteration number for variables computed after the prolongation step which becomes the variables that enter the $(m+1)$ transport sweep;
   - (l) for 'l'ow-order: the iteration number of the low-order acceleration scheme;

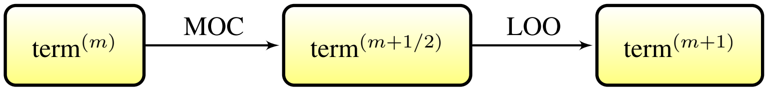

Figure 1-2: Notations for Iteration Indexes

3. Overhead bars: implies averaging. For instance, $\bar{\psi}_{g,i,m,k}$ is the average angular flux over the segment k in Flat Source Region (FSR) i for energy g and angle m.

4. Other variables:

   - δ_m: the average track spacing between tracks of azimuthal angle Ω_m;
   - $s_{i,k}$: in-plane length of the k-th segment in the i-th region;
   - $s_{i,m,k} = \dfrac{s_{i,k}}{\sin\theta_p}$: the length of the k-th segment in the upper angular half-space;
   - $d_{I,d}$: side length of the coarse mesh cell I in d direction (d can be x or y).

5. Conventional variables:

   - ψ: angular flux;
   - ϕ: scalar flux;
   - q and Q are both source terms. $Q_{g,i}$ is the source for region i and energy group g, where as $q_{g,i,m}$ is the angular source of region i, energy group g and direction Ω_m. For isotropic scattering, $q_{g,i,m} = \dfrac{1}{4\pi} Q_{g,i}$.

[1] Unfortunately m is also used in the subscript as index for angle Ω_m, but hopefully it is clear that the angle index is in the subscript, and the MOC iteration number is in the superscript surrounded by brackets

1.3.2 Simplification of the Neutron Transport Equation

The multi-group formulation of the integro-differential form of the time independent neutron transport equation is,

$$\vec{\Omega} \cdot \nabla\psi_g(\vec{r}\,, \vec{\Omega}\,) + \Sigma_{t,g}(\vec{r})\psi_g(\vec{r}\,, \vec{\Omega}\,) = Q_g(\vec{r}) \tag{1.1}$$

where the source term is further expressed in terms of scattering source and fission source,

$$Q_g(\vec{r}) = \sum_{g'} \Sigma_{s,g'\to g}(\vec{r})\phi_{g'}(\vec{r}) + \frac{\chi_g}{k_{\text{eff}}} \sum_{g'} \nu\Sigma_{f,g'}(\vec{r})\phi_{g'}(\vec{r}) \tag{1.2}$$

Assuming isotropic scattering,

$$q_g(\vec{r}\,, \vec{\Omega}\,) = \frac{1}{4\pi}\left(\sum_{g'} \Sigma_{s0,g'\to g}(\vec{r})\phi_{g'}(\vec{r}) + \frac{\chi_g}{k_{\text{eff}}} \sum_{g'} \nu\Sigma_{f,g'}(\vec{r})\phi_{g'}(\vec{r})\right) \tag{1.3}$$

And for any order of scattering anisotropy,

$$\phi_g(\vec{r}) = \int \psi_g(\vec{r}\,, \vec{\Omega}\,)\, \mathrm{d}\vec{\Omega} \tag{1.4}$$

Next we simplify $\vec{\Omega} \cdot \nabla\psi_g(\vec{r}\,, \vec{\Omega}\,)$. In our geometry of interest, a conventional right-handed coordinate system is used. The azimuthal angle is defined in the x-y plane measured counter-clockwise from the x-axis, and the polar angle is measured from the z-axis.

$$\vec{\Omega} = \cos\varphi_a \sin\theta_p \hat{\imath} + \sin\varphi_a \sin\theta_p \hat{\jmath} \tag{1.5}$$

and

$$\nabla\psi_g(\vec{r}\,, \vec{\Omega}\,) = \frac{\partial\psi_g(\vec{r}\,, \vec{\Omega}\,)}{\partial x}\hat{\imath} + \frac{\partial\psi_g(\vec{r}\,, \vec{\Omega}\,)}{\partial y}\hat{\jmath} \tag{1.6}$$

Thus together,

$$\vec{\Omega}\cdot\nabla\psi_g(\vec{r}\,,\vec{\Omega}) = \left(\cos\varphi_a\sin\theta_p\hat{i} + \sin\varphi_a\sin\theta_p\hat{j}\right)\left(\frac{\partial\psi_g(\vec{r}\,,\vec{\Omega})}{\partial x}\hat{i} + \frac{\partial\psi_g(\vec{r}\,,\vec{\Omega})}{\partial y}\hat{j}\right) \tag{1.7}$$

$$= \cos\varphi_a\sin\theta_p\frac{\partial\psi_g(\vec{r}\,,\vec{\Omega})}{\partial x} + \sin\varphi_a\sin\theta_p\frac{\partial\psi_g(\vec{r}\,,\vec{\Omega})}{\partial y} \tag{1.8}$$

Next we note that the distance $\vec{r}$ can be written using the incoming angular flux position $\vec{r}_0$, the in-plane track length $s_{i,k}$ and polar angle θ_p:

$$\vec{r} = \vec{r}_0 + \frac{s_{i,k}}{\sin\theta_p}\,\vec{\Omega} \tag{1.9}$$

Then

$$\frac{\partial\,\vec{r}}{\partial s_{i,k}} = \frac{\partial\left(\vec{r}_0 + \frac{s_{i,k}\,\vec{\Omega}}{\sin\theta_p}\right)}{\partial s_{i,k}} = \frac{1}{\sin\theta_p}\frac{\partial s_{i,k}\,\vec{\Omega}}{\partial s_{i,k}} \tag{1.10}$$

Substituting Eq. 1.5, we have,

$$\frac{\partial x}{\partial s_{i,k}} = \cos\varphi_a\frac{\sin\theta_p}{\sin\theta_p} = \cos\varphi_a \tag{1.11}$$

and

$$\frac{\partial y}{\partial s_{i,k}} = \sin\varphi_a\frac{\sin\theta_p}{\sin\theta_p} = \sin\varphi_a \tag{1.12}$$

Substituting the above two expressions into Eq. 1.8, we get,

$$\vec{\Omega}\cdot\nabla\psi_g(\vec{r}\,,\vec{\Omega}) = \left(\frac{\partial x}{\partial s_{i,k}}\frac{\partial\psi_g(\vec{r}\,,\vec{\Omega})}{\partial x} + \frac{\partial y}{\partial s_{i,k}}\frac{\partial\psi_g(\vec{r}\,,\vec{\Omega})}{\partial y}\right)\sin\theta_p \tag{1.13}$$

$$= \frac{\mathrm{d}\psi_g(s_{i,k},\vec{\Omega})}{\mathrm{d}s_{i,k}}\sin\theta_p \tag{1.14}$$

To further simplify, recall that the k-th segment's length in the upper hemisphere $s_{i,m,k}$ and its in-plane length $s_{i,k}$ has the relation $s_{i,m,k} = \dfrac{s_{i,k}}{\sin\theta_p}$, that is,

$$\frac{\mathrm{d}s_{i,k}}{\mathrm{d}s_{i,m,k}} = \sin\theta_p \tag{1.15}$$

Substituting into Eq. 1.14,

$$\vec{\Omega}\cdot\nabla\psi_g(\vec{r}\,,\vec{\Omega}) = \frac{\mathrm{d}\psi_g(s_{i,k},\vec{\Omega})}{\mathrm{d}s_{i,k}}\frac{\mathrm{d}s_{i,k}}{\mathrm{d}s_{i,m,k}} = \frac{\mathrm{d}\psi_g(s_{i,m,k},\vec{\Omega})}{\mathrm{d}s_{i,m,k}} \tag{1.16}$$

Thus we arrive at the simplified form of the streaming term. We further discretize over angle by defining discrete directions and space by only considering spatially-homogeneous regions with constant source, the neutron transport equation becomes,

$$\frac{\mathrm{d}\psi_{g,i,m,k}}{\mathrm{d}s_{i,m,k}} + \Sigma_{t,g,i}\psi_{g,i,m,k} = q_{g,i,m} \tag{1.17}$$

where $\Sigma_{t,g,i}$ is the transport-corrected total cross-section, and the angular source is,

$$q_{g,i,m} = \frac{1}{4\pi}\left(\sum_{g'}\Sigma_{s0,g'\to g,i}\phi_{g',i} + \frac{\chi_g}{k_{\text{eff}}}\sum_{g'}^{G}\nu\Sigma_{f,g',i}\phi_{g',i}\right) \tag{1.18}$$

Since the isotropic scattering approximation is made in this work,

- The angular source $q_{g,i,m}$ is independent of the angle Ω_m, for simplicity the m subscript will be dropped for the rest of this thesis. Though it is important to keep in mind that even $q_{g,i}$ and $Q_{g,i}$ have the same subscripts, they have different physical meanings and are offset by a factor of 4π.

- The group g' to group g scattering cross-section $\Sigma_{s0,g'\to g,i}$'s subscript $s0$ can be simplified to just s. Thus for the rest of this thesis the term $\Sigma_{s,g'\to g,i}$ will be used.

1.3.3 Derivation of Angular Flux Formulation

Integrating Eq. 1.17 along the track segment $s_{i,m,k}$, the outgoing angular flux ψ_i^+ of group g, angle Ω_m (again Ω_m is at the a-th azimuthal angle and p-th polar angle), k-th segment across flat source region i based on the incoming angular flux ψ_{i-} is:

$$\psi^{+}_{g,i,m,k} = \psi^{-}_{g,i,m,k} e^{-\Sigma_{t,g,i} s_{i,m,k}} + \frac{q_{g,i}}{\Sigma_{t,g,i}} (1 - e^{-\Sigma_{t,g,i} s_{i,m,k}}) \tag{1.19}$$

The track averaged angular flux in group g, angle Ω_m, along k-th segment is computed by integrating Eq. 1.19 over the segment and dividing by the length of the segment $s_{i,m,k}$:

$$\bar{\psi}_{g,i,m,k} = \frac{q_{g,i}}{\Sigma_{t,g,i}} + \frac{\psi^{-}_{g,i,m,k} - \psi^{+}_{g,i,m,k}}{\Sigma_{t,g,i} s_{i,m,k}} \tag{1.20}$$

Defining $\Delta_{g,i,m,k} = \psi^{-}_{g,i,m,k} - \psi^{+}_{g,i,m,k}$, then the average angular flux over the k-th segment becomes,

$$\bar{\psi}_{g,i,m,k} = \frac{q_{g,i}}{\Sigma_{t,g,i}} + \frac{\Delta_{g,i,m,k}}{\Sigma_{g,t,i} s_{i,m,k}} \tag{1.21}$$

Further, the average angular flux in flat source region i for group g and angle Ω_m can be computed by summing Eq. 1.21 over all segments k in flat source region i,

$$\bar{\psi}_{g,i,m} = \frac{\sum\limits_{k} \bar{\psi}_{g,i,m,k} s_{i,k} \delta_m}{\sum\limits_{k} s_{i,k} \delta_m} \tag{1.22}$$

where δ_m is the average track spacing for tracks with the azimuthal angle Ω_m.

1.3.4 Derivation of Scalar Flux Formulation

The scalar flux in group g, region i is computed by integrating the average angular flux in region i over all angle Ω_m,

$$\phi_{g,i} = \int_{4\pi} \bar{\psi}_{g,i}(\Omega_m)\, \mathrm{d}\Omega_m = 4\pi \frac{\sum\limits_{m} \omega_m \bar{\psi}_{g,i,m}}{\sum\limits_{m} \omega_m} \tag{1.23}$$

where the weight associated with each angle Ω_m is further discussed in Section. 1.3.5; for now all that matters is $\sum\limits_{m} \omega_m = 4\pi$, and we can cancel out the denominator of Eq. 1.23 and the 4π term to get,

$$\phi_{g,i} = \sum_m \omega_m \bar{\psi}_{g,i,m} \tag{1.24}$$

The flat source region scalar flux can be simplified by substituting Eq. 1.22 into Eq. 1.24,

$$\begin{aligned}
\phi_{g,i} &= \sum_m \left(\omega_m \frac{\sum_k \bar{\psi}_{g,i,m,k} s_{i,k} \delta_m}{\sum_k s_{i,k} \delta_m} \right) && (1.25) \\
&= \sum_m \frac{1}{\sum_k s_{i,k} \delta_m} \left[\omega_m \delta_m \left(\sum_k \bar{\psi}_{g,i,m,k} s_{i,k} \right) \right] && (1.26)
\end{aligned}$$

Notice that for each flat source region i, its volume V_i can be approximated by $V_{i,m}$ which is computed by adding up all the segments k crossing flat source region i:

$$V_{i,m} = \sum_k s_{i,k} \delta_m \tag{1.27}$$

The $V_{i,m}$ expression used in this thesis is the as-tracked volume for the m-th azimuthal angle. The as-tracked volume for region i can be computed by summing all the $V_{i,m}$ together with the appropriate azimuthal weight:

$$V_i = \sum_a \frac{\omega_a}{2\pi} V_{i,m} \tag{1.28}$$

which approaches the physical volume when infinite segments and angles are employed. With finite segments, V_i differs from the physical volume with some numerical error.

Current implementation does not adjust for the difference between physical volume and as-tracked volume, because the MOC solver requires a dense track lay down to get accurate results, and with that many tracks the volume error becomes negligible.

The scalar flux expression can be further simplified by substituting Eq. 1.21 into Eq. 1.26,

$$\phi_{g,i} = \sum_m \frac{1}{V_{i,m}} \left[\omega_m \delta_m \left(\sum_k \bar{\psi}_{g,i,m,k} s_{i,k} \right) \right] \tag{1.29}$$

$$= \sum_m \frac{1}{V_{i,m}} \left\{ \omega_m \delta_m \sum_k \left[s_{i,k} \left(\frac{q_{g,i}}{\Sigma_{t,i}} + \frac{\Delta_{i,m,k}}{\Sigma_{t,i} s_{i,m,k}} \right) \right] \right\} \tag{1.30}$$

$$= \frac{q_{g,i}}{\Sigma_{t,i}} \sum_m \frac{1}{V_{i,m}} \omega_m \delta_m \sum_k s_{i,k} + \frac{1}{\Sigma_{t,i}} \sum_m \frac{1}{V_{i,m}} \omega_m \delta_m \sum_k \Delta_{i,m,k} \frac{s_{i,k}}{s_{i,m,k}} \tag{1.31}$$

Applying again that $V_{i,m} = \sum_k s_{i,k} \delta_m$, and that $\frac{s_{i,k}}{s_{i,m,k}} = \sin\theta_p$,

$$\phi_{g,i} = \frac{q_{g,i}}{\Sigma_{t,i}} \sum_m \frac{1}{V_{i,m}} \omega_m V_{i,m} + \frac{1}{\Sigma_{t,i}} \sum_m \frac{1}{V_{i,m}} \omega_m \delta_m \sin\theta_p \sum_k \Delta_{i,m,k} \tag{1.32}$$

$$= \frac{q_{g,i}}{\Sigma_{t,i}} + \frac{1}{\Sigma_{t,i}} \sum_m \frac{\omega_m \delta_m}{V_{i,m}} \sin\theta_p \sum_k \Delta_{g,i,m,k} \tag{1.33}$$

1.3.5 Azimuthal and Polar Quadrature

This section discusses the choice of ω_m, which is a product of the azimuthal weight and the polar weight:

$$\omega_m = \omega_a \omega_p \tag{1.34}$$

Theoretically it is possibly to use any sets of angles and weights as long as the weights add up to one. But there are advantages of selecting specific quadrature sets for the azimuthal angles and the polar angles.

1. Azimuthal quadrature set: the simplest azimuthal set is the one that is evenly spaced in the 2D plane. That is, if there are A azimuthal angles, the azimuthal angle spacing is,

$$\overline{\Delta\varphi_a} = \frac{2\pi}{A} \tag{1.35}$$

when the A azimuthal angles are evenly distributed in the 2π angular space. As will be discussed in Section 1.5.3, although evenly spaced azimuthal angles are intended, the desire of cyclic tracking (i.e., tracks that reflect exactly on other tracks) will require some adjustments to be made to the azimuthal angles.

Further, when tracks are laid down, the x- and y-axis should be avoided to prevent error in calculating distances. Thus it is conventional to declare the first azimuthal direction at half the angular spacing above the x-axis, that is,

$$\varphi_a = \left(a - \frac{1}{2}\right) \overline{\Delta\varphi_a} \tag{1.36}$$

for the a-th azimuthal angle, where a starts from 1. The corresponding weight is,

$$\omega_a = \overline{\Delta\varphi_a} = \frac{1}{2}\left(\theta_{a+1} - \varphi_a\right) + \frac{1}{2}\left(\theta_a - \theta_{a-1}\right) \tag{1.37}$$

and note that the azimuthal angles add up to 2π:

$$\sum_a \omega_a = \sum_a \overline{\Delta\varphi_a} = 2\pi \tag{1.38}$$

2. Polar quadrature set: a couple of different polar quadrature sets have been proposed and implemented over the years,

   (a) Equal angles quadrature sets where the polar angles are distributed evenly. In three-dimensional problems, this quadrature set is advantageous since all angles are treated equally [Knott and Yamamoto, 2010];

   (b) Equal weight quadrature set where the polar angles are chosen such that their weights are identical, thus the weight associated with each Ω_m is identical too. This quadrature set emphasizes the polar angles closer to the two-dimensional plane, thus making it more suitable for two-dimensional problem [Knott and Yamamoto, 2010];

   (c) Gauss-Legendre (GL) quadrature set was discovered to be more suitable than the previous methods for two-dimensional problems using more than three polar angles;

   (d) Tabuchi-Yamamoto (TY) optimum quadrature set was recently proposed minimizing the error generated from polar angle discretization when compare with collision probability method. Running MOC solver using the TY set with three polar angles has shown similar accuracy as the same solver using GL quadrature with 12 to 16 polar angles [Yamamoto et al., 2007].

The TY quadrature set is the polar quadrature set of choice for OpenMOC since benchmark results showed that three polar angles are sufficient for typical lattice physics calculation. It is important to notice that polar angle sets proposed in [Yamamoto et al., 2007] is for half of the polar angular space. By symmetry, the other half space has polar weights that add up to unity as well, so together,

$$\sum_p \omega_p = 2.0 \tag{1.39}$$

To recap,

$$\sum_m \omega_m = \sum_a \omega_a \sum_p \omega_p = 4\pi \tag{1.40}$$

1.4 MOC Implementation

This section describe the important components of the MOC scheme. These practices are common among many MOC solvers, and OpenMOC specific features will be deferred to Section 1.5.

1.4.1 Spatial Discretization

Flat Source Region (FSR) is an important concept in MOC where small regions are defined with the approximation that the cross-section and the source are constant within a FSR. Typically a subdivision of material regions is considered a FSR.

For instance, Fig. 1-3 is the FSR map generated by OpenMOC for an assembly consisting of 16 pin cells. In this specific example, each fuel region is subdivided into multiple radial rings, a clad ring with 4 angular sectors, and each radial water ring is subdivided into eight angular sectors.

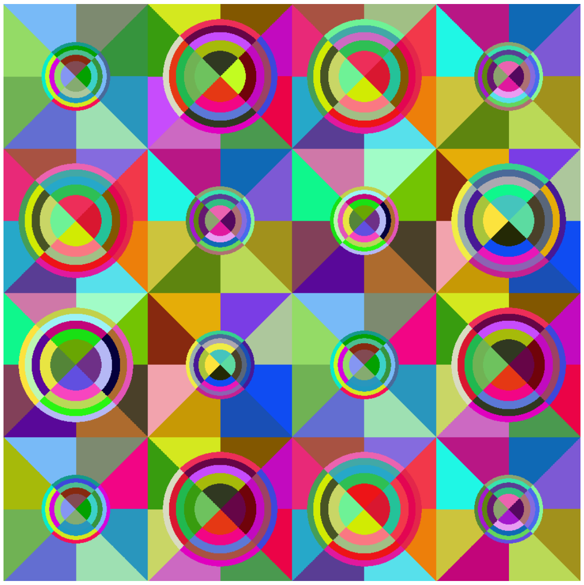

Figure 1-3: Illustration of MOC's FSR Concept

While the FSR constant source approximation is conventional and is used in OpenMOC, recent study by Ferrer has shown that a linear source approximation that computes linear source expansion coefficients from spatial moments is advantageous over the conventional flat source approximation. More specifically, for cases with large flux gradients, the LS approximation can reduce the run time and memory requirement relative to the FS scheme [Ferrer et al., 2012].

1.4.2 Azimuthal and Polar Angular Discretization

Tracks are laid down across the entire geometry with equal spacing for each azimuthal angle. A segment is defined as a fraction of a track that is contained in a FSR. Next each segment is effectively lifted up from the 2D plane into the corresponding polar angle, thus making discretization in both the azimuthal angle and the polar angle. Specific implementation of track and segment generalization differs from one code to another, and more details are given in Section 1.5.3.

The tracks and segments corresponding to the geometry shown in Fig. 1-3 is visualized in Fig. 1-4, where the left plot shows the track laydown, and the right plot shows the segment laydown. A relatively coarse track laydown (track spacing of 0.5cm, 64 azimuthal angles) is used to illustrate the tracks. In a real computation much denser track generation (i.e. smaller track spacings and/or more azimuthal angles) is required to achieve accurate results.

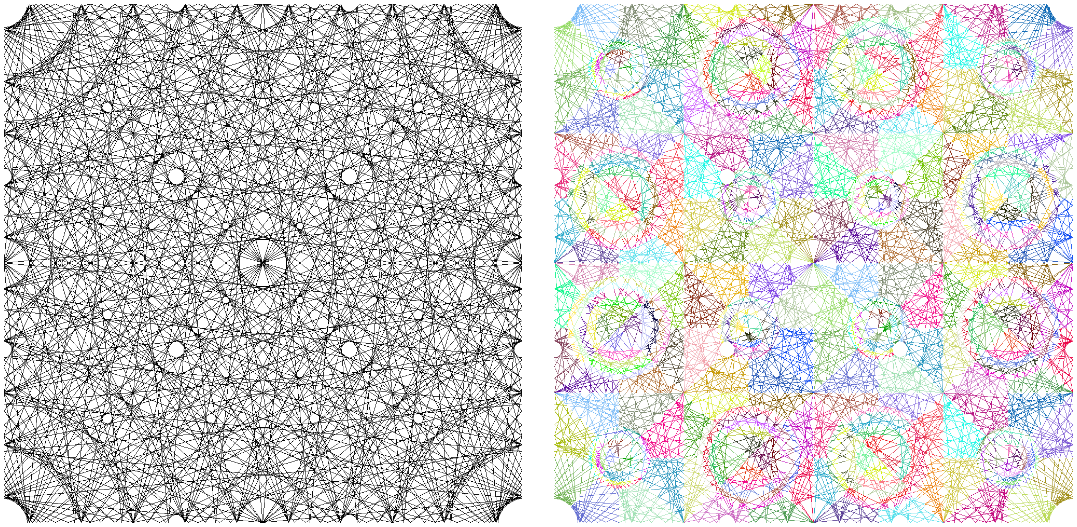

Figure 1-4: Illustration of MOC's Track and Segment Concept

A side note here is that MOC track laydown is similar to that in the Collision Probability (CP) method [Hebert, 2009], thus existing ray tracing routine from CP method may be reused for use in an MOC solver.

1.4.3 Iterative MOC Transport Solve

After setting up the geometry and laying down tracks and segments, an MOC solver can proceed to the transport solve.

1. Before starting the iterative process, initialization is performed using the index $(m = 0)$ to designate that these are the initial conditions of the MOC iterative solve. The FSR scalar fluxes are initialized to 1

$$\phi_{g,i}^{(m=0)} = 1.0 \tag{1.41}$$

and all the angular fluxes on outer reflective boundaries are set to $\dfrac{1}{4\pi}$:

$$\forall i \in S : \psi_{g,i,m,k}^{-,(m=0)} = \frac{1}{4\pi} \tag{1.42}$$

and vacuum boundary fluxes are set to zero.

2. For the (m)-th MOC sweep, five levels of nested loops are performed over all azimuthal angles, tracks, segments (first all segments in the forward directions, then the backward directions), energy groups and polar angles:

   (a) Computes the outgoing angular flux based on the incoming angular flux using Eq. 1.19;

   (b) Accumulates the FSRs' scalar fluxes based on the change in angular flux using Eq. 1.33;

   (c) Pass the outgoing angular flux to the connecting segment.

3. Upon completing the (m)-th MOC sweep, new FSR scalar fluxes are generated, and new FSR sources are computed using Eq. 1.18.

4. Checks for convergence by computing the L2 norm of the relative change in FSR powers. If the pin powers are not converged, the MOC solver proceeds to the $(m + 1)$-th sweep; otherwise exit and proceed to user-requested outputs.

1.5 OpenMOC Framework Overview

An extensible, open-source C++ code called OpenMOC has been developed by Will Boyd, Sam Shaner and the author of this work to solve the Boltzmann neutron transport equation mostly suitable for LWRs.

1.5.1 Flow of Calculations

Fig. 1-5 illustrates the basic flow of OpenMOC, where a yellow block designates a required module of OpenMOC associated with high-order solution, and a blue block designates an acceleration module turned on at users' request. Many details are not shown in this figure, for instance, when acceleration is on, current tallying across mesh surfaces is required during the transport MOC solve module.

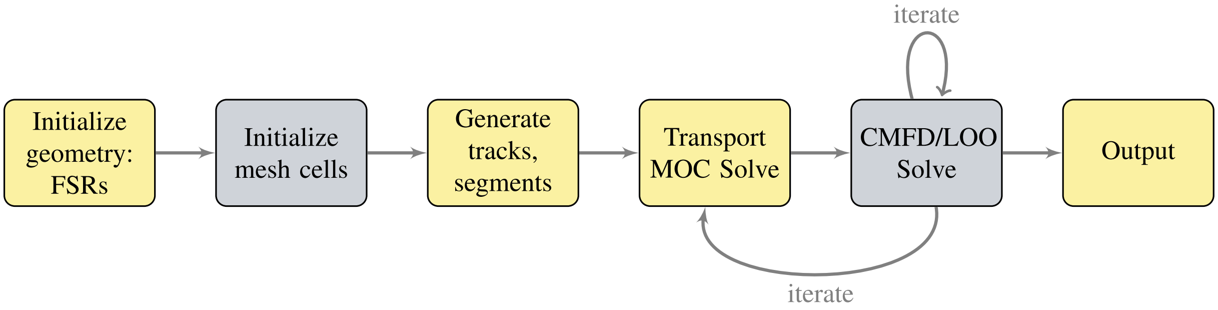

Figure 1-5: OpenMOC Flow of Calculations

1.5.2 XML Geometry Input File and Material Input File

The inputs to OpenMOC are structured as a set of XML (eXtensible Markup Language) files. OpenMOC uses the same nested universes framework as some other transport codes (e.g., MCNP, OpenMC) which allows heterogeneous geometries to be described in a simple manner.

A typical geometry input file includes quadratic surfaces, cells defined from surfaces, and lattices defined from cells or universes. A sample geometry input for a 2x2 pin cell problem is shown in Fig. 1-6: surfaces 1-5 are defined on the bottom of the file using quadratic surface coefficients. Then cells 101-102 are defined using bounding surfaces and filling materials and are assigned universe number 1. Lattice 5 is defined using universe 1. Finally a highest level cell 1 is defined with bounding surfaces 1-4 and is filled with lattice 5.

A typical material input file consists of one block for each material defining the material id, number of energy groups, and common cross-sections like absorption, transport, fission, and scattering etc. A sample material file is shown in Fig. 1-7. If for any reaction type, the number of values provided does not match the number of energy groups defined, an error would be generated in OpenMOC suggesting a user input error has happened.

Upon parsing the geometry input file and the material input file, OpenMOC generates cells filled with the corresponding material, and only unique cells are stored to save memory. During the ray tracing process, routines are called to translate between the global coordinate system used by the ray tracing and the local coordinate system used by each cell. Each FSR has to have its own memory, because even though two FSRs may have the geometrical and material properties, their scalar fluxes are likely different.

Although OpenMOC uses a somewhat general geometry definition scheme, it should be pointed out that there is an alternative that has shown success in CASMO-4 and LANCER02 in terms of reducing execution time. That is, the MOC solver can store pre-defined cell types which avoids the need for calculating the true volume for an arbitrary geometry. Also the flat source region are pre-defined in the solvers to help achieve a reasonable accuracy. Detailed descriptions of such cells can be found in [Knott and Yamamoto, 2010].

```xml
<?xml version="1.0"?>
<!-- 2x2 lattice simple pin cell -->
<geometry>
  <!-- Definition of Cells -->
  <cell id="1" universe="0" fill="5" surfaces="1 -2 3 -4" />
  <cell id="101" universe="1" material="1" surfaces="-5" sectors="4"/>
  <cell id="102" universe="1" material="2" surfaces="5" sectors="8"/>

  <!-- Definition of Lattices -->
  <lattice id="5">
    <type>rectangular</type>
    <dimension>2 2</dimension>
    <width>1.26 1.26</width>
    <universes>
      1 1
      1 1
    </universes>
  </lattice>

  <!-- Definition of Surfaces -->
  <surface id="1" type="x-plane" coeffs="-1.26" boundary="reflective"/>
  <surface id="2" type="x-plane" coeffs="1.26"  boundary="reflective"/>
  <surface id="3" type="y-plane" coeffs="-1.26" boundary="reflective" />
  <surface id="4" type="y-plane" coeffs="1.26"  boundary="reflective"/>
  <surface id="5" type="circle"  coeffs="0.0 0.0 0.54" />
</geometry>
```

Figure 1-6: Sample Geometry Input File

1.5.3 Global Cyclic Tracing Technique

Global tracking means that tracks are laid down for the entire geometry as opposed to a modular ray tracing technique that only lays down tracks for each unique cell and link the tracks on the surface of neighboring cells. Although modular ray tracing reduces memory usage, it places additional constraints on the tracks that can be laid down, thus OpenMOC uses global ray tracing.

Cyclic tracking means that track spacings are chosen such that tracks form closed cycles. More specifically, given a track with an azimuthal angle of φ_a, there exist a complementary track, that is, one with the azimuthal angle $\pi - \varphi_a$ that shares a same point on the boundary. To make sure that such complementary tracks line up, small modifications on evenly spaced azimuthal angles are required. Thus $\overline{\Delta\varphi_a}$ is not identical for every azimuthal angle a, and Eq. 1.37 is used to calculate the azimuthal weight.

The global cyclic tracing method is chosen to make boundary condition treatment easy:

- For reflective boundary condition, the outgoing angular flux of a segment that ends on a boundary surface is passed on to the incoming angular flux of the complementary segment that starts from the exact same spatial point;

- For vacuum boundary condition, the incoming angular flux of any segment originating from the boundary surface is simply set to zero.

In both cases, because cyclic tracking enforces closed cycles, passing angular flux values is exact and straightforward, and there is no need for performing surface-averaging interpolation.

1.5.4 MOC Transport Sweep

In OpenMOC, only locations of the tracks in the $(0, \pi)$ half space are generated and stored, because by the azimuthal angle symmetry, the backward direction of a track $\varphi_a \in (0, \pi)$ represents that of the track of $\varphi_a + \pi$. Thus tracks' geometrical and material properties are shared between the forward and backward characteristic directions. Though since the forward and backward segments have different scalar fluxes, memory needs to be allocated to store scalar fluxes for both directions.

During a transport sweep, it is natural to perform a forward sweep along a track immediately followed by a backward sweep along the same track to take advantage of the memory caching. This bi-directional sweeping method does require storing the boundary angular fluxes for the reflective and the periodic boundary condition. Since the number of boundary angular fluxes is relatively small compared to the total number of angular fluxes, especially for larger geometries, bi-directional sweeping has advantages over mono-directional sweeping.

1.5.5 Visualization

OpenMOC can generate, upon user request, various plots, including:

- Geometry plots color-coded by material types, cell types, or by each unique flat source region to verify user geometry and material inputs;

- Plots of tracks and segments to help debugging track and segment generation;
- Plots of FSR group-wise fluxes, total fluxes, pin powers... to illustrate simulation results;
- Plots of mesh cell homogenized scalar flux, cross-sections, surface net currents, diffusion coefficients... to aid debugging the restriction step when an acceleration scheme is active;
- Plots of mesh cell group-wise scalar flux multiplicative update value to aid debugging the prolongation step when an acceleration scheme is active.

Plots are available in png, tiff, jpg, and pdb formats. In particular, the pdb format allows users to visualize simulations in environments like the VisIt visualization environment.

```
<?xml version="1.0"?>
<materials>

  <!-- 7 Group Cross sections from c5g7 benchmark -->

  <!-- material 1: UO2 fuel -->
  <material id="1" energy="7"
            sigma_a   ="8.024800E-03 3.717400E-03 2.676900E-02 9.623600E-02 3.002000E-02 1.112600E-01 2.827800E-01"
            sigma_t   ="1.779490E-01 3.298050E-01 4.803880E-01 5.543670E-01 3.118010E-01 3.951680E-01 5.644060E-01"
            sigma_f   ="7.212060E-03 8.193010E-04 6.453200E-03 1.856480E-02 1.780840E-02 8.303480E-02 2.160040E-01"
            nu_sigma_f="2.005998E-02 2.027303E-03 1.570599E-02 4.518301E-02 4.334208E-02 2.020901E-01 5.257105E-01"
            chi       ="5.879100E-01 4.117600E-01 3.390600E-04 1.176100E-07 0.000000E+00 0.000000E+00 0.00000E+00"
            sigma_s   ="1.275370E-01 4.237800E-02 9.437400E-06 5.516300E-09 0.000000E+00 0.000000E+00 0.000000E+00
                        0.000000E+00 3.244560E-01 1.631400E-03 3.142700E-09 0.000000E+00 0.000000E+00 0.000000E+00
                        0.000000E+00 0.000000E+00 4.509400E-01 2.679200E-03 0.000000E+00 0.000000E+00 0.000000E+00
                        0.000000E+00 0.000000E+00 0.000000E+00 4.525650E-01 5.566400E-03 0.000000E+00 0.000000E+00
                        0.000000E+00 0.000000E+00 0.000000E+00 1.252500E-04 2.714010E-01 1.025500E-02 1.002100E-08
                        0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 1.296800E-03 2.658020E-01 1.680900E-02
                        0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 8.545800E-03 2.730800E-01"
            />


  <!-- material 2: water -->
  <material id="2" energy="7"
            sigma_a   ="6.010500E-04 1.579300E-05 3.371600E-04 1.940600E-03 5.741600E-03 1.500100E-02 3.723900E-02"
            sigma_t   ="1.592060E-01 4.129700E-01 5.903100E-01 5.843500E-01 7.180000E-01 1.254450E+00 2.650380E+00"
            sigma_f   ="0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00"
            nu_sigma_f="0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00"
            chi       ="0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00"
            sigma_s   ="4.447770E-02 1.134000E-01 7.234700E-04 3.749900E-06 5.318400E-08 0.000000E+00 0.000000E+00
                        0.000000E+00 2.823340E-01 1.299400E-01 6.234000E-04 4.800200E-05 7.448600E-06 1.045500E-06
                        0.000000E+00 0.000000E+00 3.452560E-01 2.245700E-01 1.699900E-02 2.644300E-03 5.034400E-04
                        0.000000E+00 0.000000E+00 0.000000E+00 9.102840E-02 4.155100E-01 6.373200E-02 1.213900E-02
                        0.000000E+00 0.000000E+00 0.000000E+00 7.143700E-05 1.391380E-01 5.118200E-01 6.122900E-02
                        0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 2.215700E-03 6.999130E-01 5.373200E-01
                        0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 1.324400E-01 2.480700E+00"
            />
</materials>
```

Figure 1-7: Sample Material Input File

Chapter 2

CMFD Methodology and Implementation

This chapter starts with an overview of diffusion acceleration methods in Section 2.1 and focuses on the theory of the Coarse Mesh Finite Difference (CMFD) method in Section 2.3. The implementation of a multigroup one-level CMFD method in OpenMOC is introduced and some results are shown in Section 2.5.

2.1 Background and Motivation

OpenMOC's MOC solver provides a framework for solving the neutron transport equation in heterogeneous geometries. Challenges arise when full-core problems are encountered, because these problems typically have high dominance ratios which require thousands of power iterations to converge, making realistic LWR cores computations costly. Numerous acceleration schemes have been proposed and implemented for solving the transport equation. The following subsections are devoted to brief summaries of these acceleration methods.

2.1.1 DSA-type Methods

Diffusion-Synthetic Acceleration (DSA) schemes solve the error function form of the transport equation by applying a diffusion approximation. For instance, in 1D, the error function that is equivalent to the original transport equation is,

$$\mu\frac{\partial\delta\psi^{(m+1/2)}(x,\mu)}{\partial x}+\Sigma_t\delta\psi^{(m+1/2)}(x,\mu)-\frac{\Sigma_s}{2}\int_{-1}^{1}\delta\psi^{(m+1/2)}(x,\mu)\,\mathrm{d}\mu=\frac{\Sigma_s}{2}\left(\phi^{(m+1/2)}(x)-\phi^{(m)}(x)\right) \tag{2.1}$$

where $\delta\psi^{(m+1/2)}(x,\mu)$ is the error in the angular flux at location x with direction μ and at iteration $(m{+}1/2)$. After applying the diffusion approximation, Eq. 2.1 becomes

$$-\frac{\mathrm{d}}{\mathrm{d}x}D\frac{\mathrm{d}}{\mathrm{d}x}\delta\phi^{(m+1/2)}(x)+\Sigma_a\delta\phi^{(m+1/2)}(x)=\Sigma_s\left(\phi^{(m+1/2)}(x)-\phi^{(m)}(x)\right) \tag{2.2}$$

where $\delta\phi^{(m+1/2)}(x)$ is the error in the scalar flux at location x and iteration $(m+1/2)$. This is an easier problem to solve than the original Eq. 2.1. Upon solving Eq. 2.2 for $\delta\phi^{(m+1/2)}$, an additive update is used to prolongate the solution,

$$\phi^{(m+1)}(x)=\phi^{(m+1/2)}(x)+\delta\phi^{(m+1/2)}(x) \tag{2.3}$$

The reason why DSA and DSA-like methods work well is that the transport sweep primarily resolves the high frequency error in scalar flux, and is slow in resolving the low frequency error. With the help of the diffusion solve, the low frequency error reduction is largely improved which makes DSA and DSA-like methods suitable as acceleration schemes. DSA and DSA-like methods have shown great success but with the inconvenience of instability issues. Such issues were resolved by spatially discretizing the diffusion equation in a manner consistent with the spatial discretization of the high-order transport methods, which places constraints on the geometry and discretization that DSA-like methods are applicable to. More importantly, although DSA method is 'unconditionally effective,' it is not 'unconditionally efficient' as the computational cost of the DSA method is often high [Larsen and Morel, 2010].

Compared to DSA and DSA-like methods, CMR and CMFD have the following major advantages:

- The implementation is independent from the discretization schemes, which makes them relatively easier to implement in any transport solver whose coarse mesh may or may not coincide with the fine-mesh discretization;
- They can be implemented as multi-level schemes in problems that have some general rectangular geometry at some level. That is, multiple levels of coarse meshes can be lay down and solved using

geometric multi-grid-like methods to further accelerate the transport solve;

- They can be solved using existing linear algebra packages, which largely reduces the complexity associated with implementing an eigenvalue solver.

In a more recent study in 2012, Larsen compared CMFD with coarse-mesh DSA and concluded the "linearized form of CMFD is algebraically equivalent to a 'coarse mesh DSA' (or CMDSA) method." This is not to say that DSA-like methods work exactly the same as CMFD, because the two classes of methods have different limitations and are applicable to slightly difference problems. Though based on [Larsen and Kelley, 2012], for the common problems that both methods are applicable, numerically "the (linear) CMDSA and (nonlinear) CMFD methods have nearly the same convergence properties" [Larsen and Kelley, 2012].

2.1.2 CMR- and CMFD-type Methods

The Coarse Mesh Rebalance (CMR) and Coarse Mesh Finite Difference (CMFD) methods are presented in the same section because they both differ from traditional diffusion-based methods by conserving the net currents on mesh surfaces. More specifically, CMR relates the current on surface s with the mesh cell averaged flux through a rebalance factor $\alpha_{I,s}$:

$$J_{g,I,s} = \alpha_{g,I,s} \, \bar{\phi}_{g,I} \tag{2.4}$$

CMFD conserves the net current on each surface by introducing a non-linear coupling coefficient $\tilde{D}_{g,I,s}$ for each surface s:

$$J_{g,I,s} = -\,\hat{D}_{g,I,s}\,(\bar{\phi}_{g,I\pm1} - \bar{\phi}_{g,I}) - \tilde{D}_{g,I,s}\,(\bar{\phi}_{g,I\pm1} + \bar{\phi}_{g,I}) \tag{2.5}$$

CMR and CMFD methods take advantage of that the high-order transport sweeps are excellent at resolving local details in the space-angle distribution (i.e., high-frequency error) but lack the ability to quickly reduce the error over a large spatial-angle range (i.e., low-frequency error). These methods correct the averaged scalar flux distribution over a coarse mesh, and compliments the high-order transport solutions by reducing the low-frequency errors.

Some of the major differences between DSA-type methods and CMR-CMFD methods:

- As discussed above, CMR and CMFD can be applied to somewhat arbitrary coarse mesh [Cho and Park, 2003][1], whereas the DSA-type methods require consistency in the spatial discretization of the high-order and the low-order equations.

[1] not truly arbitrary, for instance the thin and thick optical limits should be avoided for CMR as will be discussed later; the 'arbitrary' here merely means that CMR and CMFD types of problem does not require the low-order spatial discretization to be consistent with the high-order one as constrained by the DSA-type methods.

- By introducing non-linearity into the low-order system, CMR and CMFD conserve the net current across each coarse mesh surface, thus producing low-order scalar fluxes identical to the volume weighted coarse mesh scalar fluxes collapsed from the high-order solution when the high-order solution has converged.

- DSA-type methods use diffusion equation to approximate the error function form of the transport equation, and apply an additive prolongation on the high-order solution, whereas CMR and CMFD methods approximate the original form of the transport equation and apply an multiplicative prolongation.

The major drawback in traditional CMR method is that this method converges slowly, if at all, in optical thin or thick mesh problems with large scattering ratios [Cefus and Larsen, 1990]. Relatively speaking, traditional CMFD behaves similarly to DSA in the thin optical thickness limit that they both converge fast, and CMFD behaves similarly to CMR towards the optically thick thickness limit as they both become inefficient or divergent. More details on the convergence comparison between CMR and CMFD, both analytically and numerically, can be found in [Cho and Park, 2003].

In terms of acceleration performance, based on implementation in DeCART, CMR was shown to have a speedup of 5 to 10 in accelerating MOC transport sweeps, and "the CMFD accelerated case is about two or three times faster than the coarse-mesh rebalancing (CMR) accelerated case" [Cho et al., 2002]. Since CMFD is superior to CMR in terms of both stability and performance, it is the choice of low-order diffusion method in OpenMOC.

It is worth mentioning that although traditional CMR and CMFD methods are unstable in the thin and/or thick optical thickness limits, in the last decade unconditional stable versions of CMR and CMFD have been derived. Three recent improvements are discussed, one is an unconditionally stable CMR, the second is an unconditionally stable CMFD, and the third is an unified CMR and CMFD method that can be unconditionally stable:

1. The Coarse-Mesh Angular Dependent Rebalance (CMADR) method was developed by introducing angular dependent rebalance (ADR) factors on coarse mesh surfaces to provide stability. CMADR has been implemented and shown a good improvement on high-order S_N transport solver [Park and Cho, 2004] and on high-order MOC transport solver [Park and Cho, 2008].

2. Similar ideas have been developed for CMFD. For instance, the partial CMFD (p-CMFD) method conserves the partial current on each mesh surface [Cho et al., 2003a]. p-CMFD will be introduced in more details in the next section where we focus on the different variations of CMFD-related schemes.

3. From a slightly different angle, a Generalized Coarse-Mesh Rebalance Method (GCMR) has been proposed to unify the traditional CMR and CMFD methods. It introduces an additional degree of

freedom with an acceleration factor f; when set to different values, the GCMR is analytically equivalent to CMR or CMFD. In addition, for homogeneous system an optimum acceleration factor can be derived and it was shown in 1D that GCMR with this optimal acceleration factor converges faster than CMR and CMFD do and does not exhibit the divergent behavior observed in CMR or CMFD. Unfortunately the derivation of optimum acceleration factor is limited to homogeneous system for now as it is very complicated, if possible at all, to perform such an analytical derivation for fully heterogeneous problem [Yamamoto, 2005]. The topic of GCMR will come up again in Section 2.4.2 as it is related to the introduction of damping factor in CMFD.

2.2 CMFD Variations and Applications

As discussed in Section 2.1, CMFD has various advantages over other low-order diffusion acceleration schemes. To summarize, CMFD offers a balance between spatial discretization options, method stability, and acceleration performance. It has been implemented in production level codes like CASMO-4 and showed a larger than 100 times speedups on large LWR problems [Smith and Rhodes, 2002]. Consequentially, it was chosen as the first acceleration scheme implemented in OpenMOC. The CMFD implementation serves as a basis for the OpenMOC framework and is used as a benchmark tool in developing the low-order transport acceleration method.

CMFD was first proposed by Smith in 1983 for nodal diffusion calculation [Smith, 1983], and has been widely used in accelerating neutron diffusion and transport problems. Since CMFD is applicable to general rectangular geometries, it is easy to generalized to multi-grid. In this work, "one-level" refers to one level of mesh cells, whereas "two-level" refers to have two layers of coarse meshes. There have been a handful of variations:

1. One-node (1-N) vs. two-node (2-N) CMFD formulation [Chao, 2000, Yoon and Joo, 2008, Lee, 2012]. For brevity, the superscript $I\pm$ designates the two surfaces of the I-th mesh cell in each direction. The $I+$ surfaces are the two surfaces in the positive sense of the I-th cell, and $I-$ are the two in the negative sense. For instance, if the x-axis points to the right and the y-axis points to the top, then $J^{+}_{g,I}$ designate the net current on the top or right surface of the I-th cell, and $J^{d-}_{g,I}$ for the bottom or left surface.

   (a) One-node CMFD formulation writes net current in term of the mesh cell scalar flux and the surface flux. For each mesh cell I and every energy group g, there are two equations, one for each surface:

   $$J^{d\pm}_{g,I} = -\hat{D}^{d}_{g,I}\,(\phi^{\pm}_{g,I} \mp \bar{\phi}_{g,I}) - \tilde{D}^{d\pm}_{g,I}\,(\bar{\phi}_{g,I} + \phi^{\pm}_{g,I}) \tag{2.6}$$

   where $J^{d\pm}_{g,I}$ is the net current crossing the $I\pm$ surface in the d direction (where d is x, y in 2D implementation), $\tilde{D}^{d\pm}_{g,I}$ is the non-linear diffusion coefficient (or current correction factor CCF as in [Lee, 2012]) that couples the current cell and its adjacent cell in the d direction, and $\hat{D}^{d}_{g,I}$ is the standard finite-difference form of diffusion coefficient that equals to $\dfrac{2D_I}{h_{I,d}}$ for uniform homogeneous meshes.

   Relative to the 2-N formulation, 1-N version is supposed to be easier to implement and can perform energy condensation during a low-order solve but may need additional spatial sweeps and/or under relaxation to converge [Lee, 2012]. A more recent study focuses on the effect of the dynamic energy condensation and shows that the partial current spectrum update drives the thermal spectrum update the most, while the CCF prolongation drives the fast group spectrum update the most [Lee, 2013].

(b) The original version proposed in [Smith, 1983] for nodal diffusion calculation is the two-node version of CMFD. It relates the currents across a surface using the scalar fluxes in the two adjacent cells:

$$J_{g,I}^{d\pm} = -\,\hat{D}_{g,I}^{d\pm}\,(\bar{\phi}_{g,I\pm1} - \bar{\phi}_{g,I}) - \,\tilde{D}_{g,I}^{d\pm}\,(\bar{\phi}_{g,I\pm1} + \bar{\phi}_{g,I}) \tag{2.7}$$

where $J_{g,I}^{d\pm}$ is the net current on mesh surface $I\pm$ in the d direction, $\hat{D}_{g,I}^{d\pm}$ is the finite difference form of diffusion coefficient coupling the two adjacent mesh cells sharing the same mesh surface in d direction, and $\tilde{D}_{g,I}^{d\pm}$ is the non-linear coupling coefficient.

2-N CMFD method is shown to be more stable than 1-N CMFD and has a faster convergence rate, thus the 2-N CMFD method is the preferred choice in almost all the codes and studies mentioned below.

2. Two-level two-node CMFD has been applied to 2D S_N code NEWT [Zhong et al., 2008].

3. Two-level two-node CMFD has been applied to 2D MOC codes including: CASMO-4 [Smith and Rhodes, 2002], CASMO-5 [Rhodes et al., 2008], DeCART [Cho et al., 2002], PARCS [Downar et al., 2002].

4. Two-level CMFD has been applied to SANM nodal kernels RENUS (Reactor Numerical Simulator). Both the 1-N and 2-N versions are implemented and evaluated, and the 2-N CMFD is considered superb to the 1-N version as the 1-N CMFD requires twice the number of source iterations compared with 2-N CMFD [Yoon and Joo, 2008].

5. Two-node CMFD has been applied to 1D Monte Carlo problem [Wolters et al., 2013] [Willert et al., 2013].

6. Two-node CMFD has been applied to 3D spatial reactor kinetics problem [Shim et al., 2011].

7. Partial-current based CMFD (pCMFD) was proposed by Cho [Cho et al., 2003a] and through Fourier analysis pCMFD was shown to be unconditionally stable [Hong et al., 2008,Hong et al., 2010]. pCMFD has also been applied to 2D/1D Fusion method for 3D whole-core calculations [Cho et al., 2003b]. uCMFD and upCMFD are pCMFD applied to unstructured CMFD and have been implemented in NEWT [Kim and DeHart, 2011].

The CMFD currently implemented in OpenMOC is an one-level two-node multi-group CMFD in its conventional form which conserves the net current across each mesh surface.

2.3 CMFD Formulation in OpenMOC

Section. 2.3.1 through 2.3.5 covers the various components of performing a one-level two-node multi-group CMFD acceleration in OpenMOC.

The notations used for high-order system follows those in Section. 1.3.1, and any additional terms used will be explained accordingly. On a very high level, the steps in CMFD are illustrated in Fig. 2-1.

2.3.1 Coarse Mesh Setup

Before performing any calculation, OpenMOC performs a geometry initialization as illustrated in Fig. 1-5. When CMFD acceleration is on, a mesh cell initialization is performed, where OpenMOC parses the mesh cell level that users request (the default is the pin cell level), constructs mesh cell objects and mesh surface objects.

The I-th coarse mesh cell object contains the following variables [2]:

- Mesh cell ID I;
- Mesh cell width and height $h_{I,x}, h_{I,y}$ generalized as $h_{I,d}$ where d means the x or y direction;
- Mesh cell volume V_I;
- A list of the IDs of the FSRs contained in mesh cell I;
- Mesh cell homogenized quantities: mesh cell homogenized cross-sections $\overline{\Sigma}_{g,xx,I}$ for energy group g and reaction type xx, and mesh cell homogenized diffusion coefficients $\bar{D}_{g,I}$;
- Mesh cell homogenized scalar fluxes $\bar{\phi}_{g,I}$.

where the first four bullet points are computed and stored at the initialization phase because the geometry of the problem does not change throughout the entire solver for a steady state problem, whereas the homogenized cross-sections and fluxes are computed and stored during the CMFD restriction step discussed in Section. 2.3.3.

Coarse mesh surfaces are designated with $I\pm$ for mesh cell I, and a mesh surface object contains the following variables:

- Net current $J^{d\pm}_{g,I}$ across the surfaces in the d direction (where d can be x or y). Since there are two surfaces for each mesh cell in each direction, the $\pm$ superscript specifies whether the surface is on the positive side or the negative side relative to the mesh cell center.

[2] For brevity this is not a complete list.

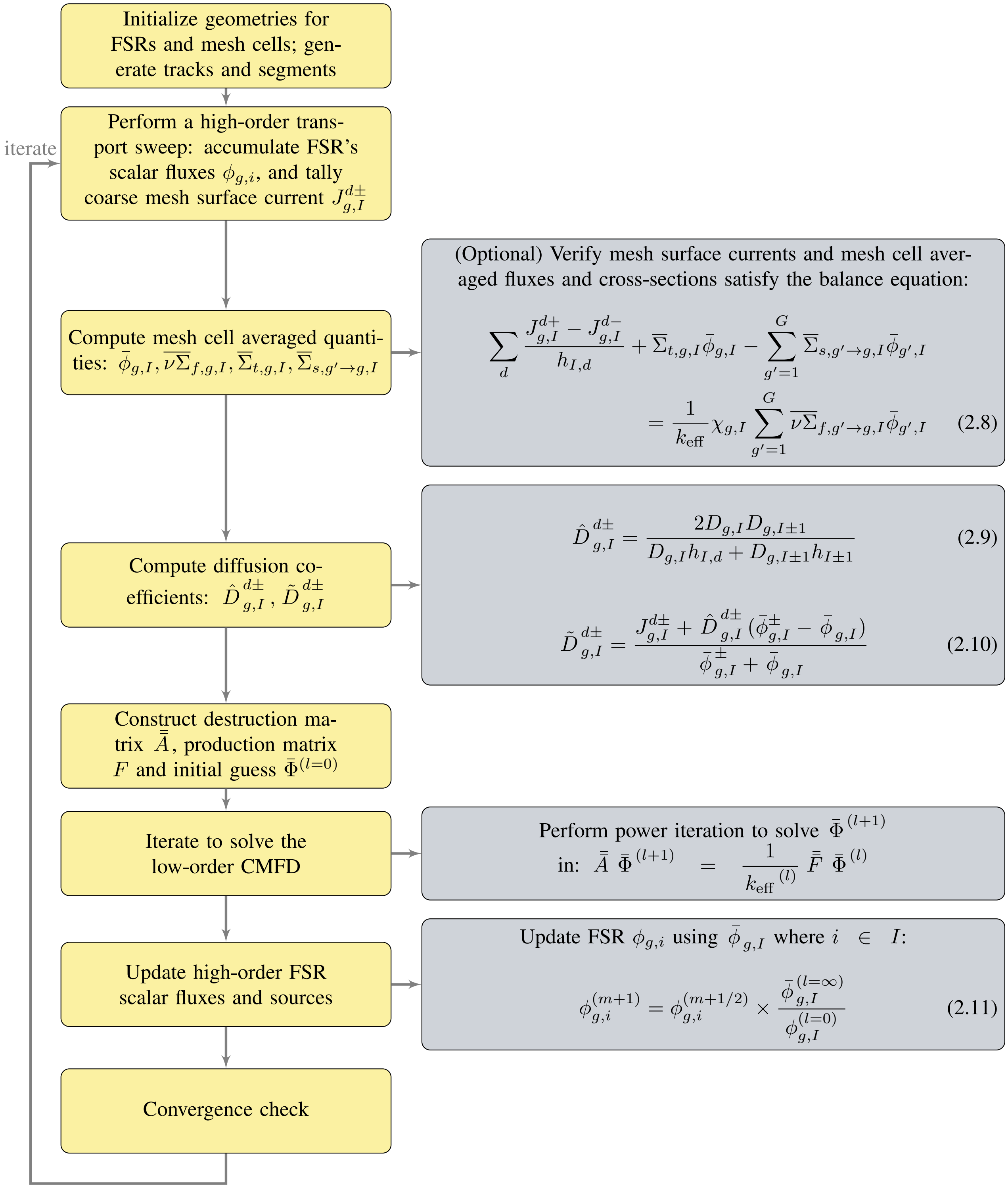
Initialize geometries for FSRs and mesh cells; generate tracks and segments
iterate
Perform a high-order transport sweep: accumulate FSR's scalar fluxes $\phi_{g,i}$, and tally coarse mesh surface current $J^{d\pm}_{g,I}$
Compute mesh cell averaged quantities: $\bar{\phi}_{g,I}, \overline{\nu\Sigma}_{f,g,I}, \overline{\Sigma}_{t,g,I}, \overline{\Sigma}_{s,g'\to g,I}$
(Optional) Verify mesh surface currents and mesh cell averaged fluxes and cross-sections satisfy the balance equation:
$$\sum_d \frac{J^{d+}_{g,I} - J^{d-}_{g,I}}{h_{I,d}} + \overline{\Sigma}_{t,g,I}\bar{\phi}_{g,I} - \sum_{g'=1}^{G} \overline{\Sigma}_{s,g'\to g,I}\bar{\phi}_{g',I} = \frac{1}{k_{\text{eff}}}\chi_{g,I}\sum_{g'=1}^{G}\overline{\nu\Sigma}_{f,g'\to g,I}\bar{\phi}_{g',I} \qquad (2.8)$$
Compute diffusion coefficients: $\hat{D}^{d\pm}_{g,I}$, $\tilde{D}^{d\pm}_{g,I}$
$$\hat{D}^{d\pm}_{g,I} = \frac{2D_{g,I}D_{g,I\pm1}}{D_{g,I}h_{I,d} + D_{g,I\pm1}h_{I\pm1}} \qquad (2.9)$$
$$\tilde{D}^{d\pm}_{g,I} = \frac{J^{d\pm}_{g,I} + \hat{D}^{d\pm}_{g,I}\,(\bar{\phi}^{\pm}_{g,I} - \bar{\phi}_{g,I})}{\bar{\phi}^{\pm}_{g,I} + \bar{\phi}_{g,I}} \qquad (2.10)$$
Construct destruction matrix $\bar{\bar{A}}$, production matrix F and initial guess $\bar{\Phi}^{(l=0)}$
Iterate to solve the low-order CMFD
Perform power iteration to solve $\bar{\Phi}^{(l+1)}$ in: $\bar{\bar{A}}\;\bar{\Phi}^{(l+1)} = \frac{1}{k_{\text{eff}}^{(l)}}\;\bar{\bar{F}}\;\bar{\Phi}^{(l)}$
Update high-order FSR scalar fluxes and sources
Update FSR $\phi_{g,i}$ using $\bar{\phi}_{g,I}$ where $i \in I$:
$$\phi^{(m+1)}_{g,i} = \phi^{(m+1/2)}_{g,i} \times \frac{\bar{\phi}^{(l=\infty)}_{g,I}}{\bar{\phi}^{(l=0)}_{g,I}} \qquad (2.11)$$
Convergence check

Figure 2-1: CMFD Flow of Calculations

- Finite-difference form of diffusion coefficient $\hat{D}_{g,I}^{d\pm}$ that couples the two cells surrounding this surface in the d direction.
- Non-linear diffusion coupling coefficient, or Current Coupling Factor (CCF) in some literature $\tilde{D}_{g,I}^{d\pm}$ that couples the two cells surrounding this surface in the d direction.

During the initialization phase memory is allocated for mesh surface objects and mesh surface objects are linked to the corresponding mesh cell objects. The net current is tallied during the high-order transport sweep, and the two diffusion coefficients are determined during the restriction step of CMFD solver.

2.3.2 CMFD Net Current Tally During Transport Sweep

First an MOC transport sweep as specified in Section 1.4.3 is performed. Using the notation in Multigrid method, this transport sweep is called the 'smoothing' step, as it smooths out the errors in the high frequency component.

During the transport sweep, OpenMOC tallies the net currents across the surfaces of each mesh cell, designated by $J_{g,I}^{d+}$ for the positive side of coarse mesh cell I in the d direction, and $J_{g,I}^{d-}$ for the negative side of I in the d direction. More specifically, when looping through each segment, OpenMOC checks whether the k-th segment ends on any mesh surface; if so, the current is tallied on the $I\pm$-th surface:

$$J_{g,I}^{d\pm} = \frac{\sum\limits_{m} \omega_m \delta_m \sin\theta_p \sum\limits_{k} \psi_{i,m,k}}{\sum\limits_{m} \omega_m \delta_m} \tag{2.12}$$

where the weight on angular flux is $\omega_m \delta_m \sin\theta_p$ which is the same weight associated with Ω_m for FSR scalar flux accumulation as shown in Eq. 1.33. Define the angular weight used in scalar flux accumulation as ω_ϕ, and the weight used in net current accumulation as ω_J, then the former is

$$\omega_\phi = \omega_m \delta_m \sin\theta_p \tag{2.13}$$

where $\omega_m = \omega_a \omega_p$. In computing the angular weight used in net current tally ω_J, the distance between tracks in the direction parallel to the surface $\frac{\delta_m}{\cos\varphi_a}$ should be used instead of δ_m, the normal distance between tracks of the a-th azimuthal angle.

Coincidentally, we need to multiply the above results by $\cos\varphi_a$ to get the current in the direction normal to the surface, thus

$$\omega_J = \omega_m \frac{\delta_m}{\cos\varphi_a} \sin\theta_p \cos\varphi_a = \omega_a \delta_m \sin\theta_p = \omega_\psi \tag{2.14}$$

2.3.3 CMFD Restriction

After an MOC transport sweep – or the smoothing step – is completed, OpenMOC enters the CMFD acceleration step. Like any other Multigrid-like method, a restriction step is needed to transfer the high-order information into the low-order system. In Multigrid theory, the restriction step reduces the local details of the high-order solution as it is fundamentally impossible to preserve all the information from the high-order exactly in the low-order.

The restriction step hides the high frequency behavior and pronounces the low frequency long range oscillation, making it possible for the next step which is the low-order iterative solver to improve the low frequency behavior.

In a CMFD restriction step, the practice differs from classical geometrical multigrid method in that the coarse mesh acceleration methods use as much physics as possible by conserving the fluxes, currents and reaction rates. For simplicity, the subscript is dropped for every term in this section on CMFD restriction, because every term in the restriction step are computed from the (m)-th MOC transport sweep and are thus designated with $(m + 1/2)$.

- $\bar{\phi}_{g,I}$ – the volume-averaged scalar flux for mesh I and energy group g – is computed by performing a volume-weighted summation of all the FSRs that are contained in mesh cell I:

$$\bar{\phi}_{g,I} = \frac{\sum\limits_{i \in I} \phi_{g,i} V_i}{\sum\limits_{i \in I} V_i} \tag{2.15}$$

  In fact, the total volume of each mesh cell V_I is already computed and stored in the geometry initialization step, so there is no need to perform the summation on FSR volumes each time:

$$\bar{\phi}_{g,I} = \frac{\sum\limits_{i \in I} \phi_{g,i} V_i}{V_I} \tag{2.16}$$

- $\bar{\Sigma}_{g,xx,I}$ – the flux-weighted cross-section for reaction type xx, mesh I and energy group g – is calculated by conserving reaction rates,

$$\bar{\Sigma}_{g,xx,I} = \frac{\sum\limits_{i \in I} \Sigma_{g,xx,i} \phi_{g,i} V_i}{\sum\limits_{i \in I} \phi_{g,i} V_i} = \frac{\sum\limits_{i \in I} \Sigma_{g,xx,i} \phi_{g,i} V_i}{\bar{\phi}_{g,I} \, V_I} \tag{2.17}$$

- $\bar{D}_{g,I}$ – the flux-weighted diffusion coefficient for mesh I and energy group g – is calculated in a similar fashion as for the homogenized cross-section:

$$\bar{D}_{g,I} = \frac{\sum\limits_{i\in I} D_{g,i}\phi_{g,i}V_i}{\sum\limits_{i\in I}\phi_{g,i}V_i} = \frac{\sum\limits_{i\in I}\dfrac{\phi_{g,i}V_i}{3\cdot\Sigma_{tr,g,i}}}{\sum\limits_{i\in I}\phi_{g,i}V_i} = \frac{\sum\limits_{i\in I}\dfrac{\phi_{g,i}V_i}{3\cdot\Sigma_{tr,g,i}}}{\bar{\phi}_{g,I}\,V_I} \tag{2.18}$$

Notice it is more appropriate to flux-volume weight $D_{g,i}$, instead of flux-volume weight $\Sigma_{tr,g,i}$ as done in [Cho et al., 2002] [Zhong et al., 2008]. These two implementation produce different results, and the former is more accurate than the latter [Smith, 2012].

- $\hat{D}^{d\pm}_{g,I}$ – the finite difference diffusion coefficient coupling mesh cell I and its neighbors in the d direction – is computed using the mesh cell homogenized diffusion coefficients $\bar{D}_{g,I}$ and mesh cell surface length $h_{I\pm1,d}$ in the d direction:

$$\hat{D}^{d\pm}_{g,I} = \frac{2\,\bar{D}_{g,I}\,\bar{D}_{g,I\pm1}}{\bar{D}_{g,I\pm1}\,h_{I,d} + \bar{D}_{g,I}\,h_{I\pm1,d}} \tag{2.19}$$

- $\tilde{D}^{d\pm}_{g,I}$ – the finite difference nonlinear coupling coefficient – is computed from the $J^{d\pm}_{g,I}$ tallied during MOC sweep based on the following net current equation:

$$J^{d\pm}_{g,I} = \mp\,\hat{D}^{d\pm}_{g,I}\,(\bar{\phi}_{g,I\pm1} - \bar{\phi}_{g,I}) - \tilde{D}^{d\pm}_{g,I}\,(\bar{\phi}_{g,I\pm1} + \bar{\phi}_{g,I}) \tag{2.20}$$

which simplifies to,

$$\tilde{D}^{d\pm}_{g,I} = -\frac{J^{d\pm}_{g,I} \pm \hat{D}^{d\pm}_{g,I}\,(\bar{\phi}_{g,I\pm1} - \bar{\phi}_{g,I})}{\bar{\phi}_{g,I\pm1} + \bar{\phi}_{g,I}} \tag{2.21}$$

For debugging purposes, it is often helpful to verify that $\bar{\phi}_{g,I}$, $J^{d\pm}_{g,I}$, $\overline{\Sigma}_{xx,g,I}$ (for reaction type xx) satisfy the neutron balance equation for each mesh cell I:

$$\sum_d \frac{J^{d+}_{g,I} - J^{d-}_{g,I}}{h_{I,d}} + \overline{\Sigma}_{t,g,I}\bar{\phi}_{g,I} - \sum_{g'=1}^{G}\overline{\Sigma}_{s,g'\to g,I}\bar{\phi}_{g',I} = \frac{1}{k_{\text{eff}}}\chi_{g,I}\sum_{g'=1}^{G}\overline{\nu\Sigma}_{f,g'\to g,I}\bar{\phi}_{g',I} \tag{2.22}$$

Next OpenMOC sets up the destruction matrix $\bar{\bar{A}}$, the source matrix $\bar{\bar{F}}$ and the guess for the coarse mesh scalar flux vector $\bar{\Phi}^{(m+1/2)}$ for the diffusion eigenvalue problem.

- $\bar{\bar{A}}$'s diagonal contains the summation of absorption, out-scattering and out-leakage (from this cell to the four adjacent cells). The off-diagonals contain in-scattering and in-leakage (from the adjacent four cells to this cell);

- $\bar{\bar{F}}$ is the construction matrix contains the fission source terms;
- $\bar{\Phi}$ is the vector containing scalar fluxes for each coarse mesh cell and energy group. The scalar flux vector constructed here is based of the $(m+1/2)$-th order, that is, it is the results coming out the m-th high-order MOC transport sweep. The CMFD iterative solver is initialized using the $(m+1/2)$-th scalar flux results: $\bar{\Phi}^{(l=0)} = \bar{\Phi}^{(m+1/2)}$

The $\bar{\bar{A}}$ and $\bar{\bar{F}}$ matrices resemble standard finite difference multigroup diffusion terms with the exception of the nonlinear coupling coefficients in the destruction matrix (which makes the low-order system consistent with the high-order transport system).

Again keep in mind that every term in this CMFD restriction step has a superscript $(m+1/2)$ omitted for brevity.

2.3.4 CMFD Iterative Solver

Upon constructing $\bar{\bar{A}}$, $\bar{\bar{F}}$, $\bar{\Phi}^{(l=0)} = \bar{\Phi}^{(m+1/2)}$, the generalized eigenvalue problem is of the following type:

$$\bar{\bar{A}}\,\bar{\Phi} = \frac{1}{k_{\text{eff}}}\,\bar{\bar{F}}\,\bar{\Phi} \tag{2.23}$$

with the initial guess $\bar{\Phi}^{(l=0)} = \bar{\Phi}^{(m+1/2)}$ and ${k_{\text{eff}}}^{(l=0)} = {k_{\text{eff}}}^{(m+1/2)}$. These subscripts imply that the initial guess of the low-order system, i.e., the $(l=0)$'th iteration, are those from the high-order solution of the $(m+1/2)$-th iteration.

Typically this nonlinear eigenvalue problem is solved with Source Iteration (SI), where for an initial guess $\bar{\Phi}^{(l=0)}$, ${k_{\text{eff}}}^{(l=0)}$, the source / right-hand-side is computed:

$$\bar{b}^{(l=1)} = \frac{1}{{k_{\text{eff}}}^{(l=0)}}\,\bar{\bar{F}}\,\bar{\Phi}^{(l=0)} \tag{2.24}$$

Then the following linear system is solved for $\bar{\Phi}^{(l=1)}$ and we repeat from Eq. 2.24.

$$\bar{\bar{A}}\,\bar{\Phi}^{(l=1)} = \bar{b}^{(l=1)} \tag{2.25}$$

Various numerical schemes are suitable to solve the system in Eq. 2.25:

- A Gauss-Seidel-type method requires the matrix to possess diagonal dominance. Thus in constructing the matrix $\bar{\bar{A}}$, it is common to invoke the flux-limiting condition [Aragones and Ahnert, 1986]. More specifically, when $\tilde{D}^{d\pm}_{g,I}$ is computed, if its magnitude is larger than that of $\hat{D}^{d\pm}_{g,I}$, the original value of $\hat{D}^{d\pm}_{g,I}$ is ignored, and new $\tilde{D}^{d\pm}_{g,I}$, $\hat{D}^{d\pm}_{g,I}$ are computed from Eq. 2.20 and $\tilde{D}^{d\pm}_{g,I} = \hat{D}^{d\pm}_{g,I}$.

- OpenMOC's CMFD acceleration uses Argonne National Lab's Portable, Extensible Toolkit for Scientific Computation (PETSc) package's [Mathematics and Compute Science Division, 2013]. The Generalized Minimal Residual Method (GMRES) solver is chosen for its good convergence behavior and a canned solver reduces complexity in the implementation.

For the (l)-th order CMFD source iteration, the successive iteration relative change in energy-integrated fission source is used to check for convergence:

$$\epsilon^{(l)} = \sqrt{\frac{1}{I}\sum_{I}\left(\frac{\sum_{g}\overline{\nu\Sigma}_{f,g,I}\phi_{g,I}^{(l)} - \sum_{g}\overline{\nu\Sigma}_{f,g,I}\phi_{g,I}^{(l-1)}}{\sum_{g}\overline{\nu\Sigma}_{f,g,I}\phi_{g,I}^{(l-1)}}\right)^2} \tag{2.26}$$

If $\epsilon^{(l)}$ is smaller than a user-defined threshold, the iterative CMFD solver exits and OpenMOC enters the next step.

2.3.5 CMFD Prolongation

Upon the convergence of the low-order CMFD iterative solver, OpenMOC performs the prolongation step, or interpretation step as addressed by some Multigrid literature.

1. FSR's scalar flux is updated by the ratio of the converged coarse mesh scalar flux over the initial coarse mesh scalar flux that enters the acceleration step:

$$\phi_{g,i}^{(m+1)} = \phi_{g,i}^{(m+1/2)} \times \frac{\bar{\phi}_{g,I}^{(l=\infty)}}{\bar{\phi}_{g,I}^{(l=0)}} \tag{2.27}$$

   where $\phi_{g,i}$ is the FSR scalar flux for the g-th energy group and the i-th FSR, $\bar{\phi}_{g,I}^{(l=0)}$ is the initial mesh cell homogenized scalar flux that enters the CMFD iterative solver, and $\bar{\phi}_{g,I}^{(l=\infty)}$ is the converged mesh cell scalar flux for the g-th energy group and the I-th mesh cell.

2. The boundary angular flux $\psi_{B,g,m}$ for the g-th energy group and Ω_m angle can be updated by the scalar flux ratio as well,

$$\psi_{B,g,m}^{(m+1)} = \psi_{B,g,m}^{(m+1/2)} \times \frac{\bar{\phi}_{g,I}^{(l=\infty)}}{\bar{\phi}_{g,I}^{(l=0)}} \tag{2.28}$$

   $\psi_{B,g,m}$ is the boundary angular flux for the g-th energy group and the m-th angle, and this boundary flux is contained in the I-th mesh cell. $\psi_{B,g,m}^{(m+1/2)}$ is the boundary angular flux coming out of the m-th MOC transport sweep, and $\psi_{B,g,m}^{(m+1)}$ is the one after prolongation.

Alternatively the boundary angular flux can be updated by net current on that surface:

$$\psi_{B,g,m}^{(m+1)} = \psi_{B,g,m}^{(m+1/2)} \times \frac{J_{g,I}^{\pm,(l=\infty)}}{J_{g,I}^{\pm,(l=0)}} \tag{2.29}$$

After the prolongation step, OpenMOC obtains the $(m+1)$-th order results for angular flux and scalar flux. OpenMOC computes the L2 norm of the successive iteration relative change in the energy-integrated FSR fission power:

$$\epsilon^{(m+1)} = \sqrt{\frac{1}{N_i}\sum_i \left(\frac{\sum_g \overline{\nu\Sigma}_{f,g,i}\phi_{g,i}^{(m+1)} - \sum_g \overline{\nu\Sigma}_{f,g,i}\phi_{g,i}^{(m)}}{\sum_g \overline{\nu\Sigma}_{f,g,i}\phi_{g,i}^{(m)}} \right)^2} \tag{2.30}$$

where N_i is the total number of FSRs.

2.4 Convergence Analysis

In low-order acceleration method's development, the limitation of the methods is typically the methods' stability. Numerous efforts have been made to analyze the convergence behavior of methods and to derive unconditionally stable versions of them.

In this section, a preliminary explanation of CMFD's instability issue will be given from iterative method's perspective (Section 2.4.1). Then in Section 2.4.3 a convergence analysis will be performed using the C5G7 benchmark problem [Smith et al., 2003]. The details of the C5G7 benchmark are deferred to Section. 4.4 as this benchmark is used throughout this chapter and the next one.

2.4.1 Picard Iteration and Fixed Point Iteration

The low-order diffusion problem of interest is a generalized eigenvalue problem,

$$\bar{\bar{A}}\ \bar{\Phi} = \frac{1}{k}\ \bar{\bar{F}}\ \bar{\Phi} \tag{2.31}$$

The simplest way to solve such a problem is to use Picard iteration or fixed point iteration or power iteration, where upon an initial guess of $\bar{\Phi}$, an operator is applied repeatably to $\bar{\Phi}$, and this iterative process produce the next solution. In the context of our diffusion problem, a Picard iteration means that the coefficients inside of $\bar{\bar{A}}$ and $\bar{\bar{F}}$ are fixed during the low-order iterative solve, thus fixed operators are being applied to solve the problem.

As pointed out by literature on iterative methods, a Picard iteration is only guaranteed to converge a nonlinear problem when the operator in the system possesses a special property called contractive, which has the following general definition [Berinde, 2002]:

$$d(Tx, Ty) \leq \alpha d(x, y) \text{ for all } x, y \in X \tag{2.32}$$

where (X, d) is a complete metric space, and T is an operator that maps X to X. In our context, a contractive operator means that the difference between successive operated results is smaller than the difference between successive solution.

Research in other context of reactor physics has shown that the instability associated with Picard iteration can be removed by applying a Krasnoselskij iteration in the context of resonance treatment using the discrete generalized multigroup method [Gibson, 2013]. The Krasnoselskij iteration basically modifies a matrix $\bar{\bar{A}}$ to be $(1 - \lambda)\ \bar{\bar{I}} + \lambda\ \bar{\bar{A}}$ where $\bar{\bar{I}}$ is the identity operator, and λ is a constant between 0 and 1. The improved operator is more likely to converge than the original operator.

A recent study by Knoll and others notes the issue with applying a Picard iteration to resolve nonlinearity in the context of nonlinear diffusion acceleration of source iteration, and implements a Jacobian-free Newton-Krylov (JFNK) method to replace the Picard iteration. [Knoll et al., 2011b] and [Knoll et al., 2011a] shows that JFNK has better convergence behavior than Picard iteration. This paper also experiments with the prolongation option and notes that the way coarse grid solutions are transferred to fine grid affect the stability of the problem though it does not go into details. Another study of the same year applies the same idea of combining the nonlinear diffusion acceleration with JFNK on transport eigenvalue problem and shows that NDA-JFNK reduces CPU time compare with NDA-PI method, and this method is called NDA-NCA (Newton-based Nonlinear Criticality Acceleration) [Park et al., 2011] [Park et al., 2012]. [Willert and Kelley, 2013] also shows that the NDA-NCA method reduces the computational time for each low-order solve by a factor of two compare with regular Picard Iteration for a 2D 2-group LRA-BWR benchmark problem with a 165x165 mesh.

To recap, this new class of NDA-NCA method uses a NDA method like the CMFD method to accelerate the scattering source, and further replaces the Picard iteration with Newton-based JFNK method. The Newton-based linearization is reduced by a non-linear elimination. Various different flavors of this method has been tested and generally NDA-NCA can reduce the total computational time and may help resolve the instability issue with traditional NDA method. Although there is not yet a robust analytical study on the implementation of Picard Iteration vs. other iterative methods in the context of NDA method, the numerical experiments from the above work seems to suggest that diverging away from the Picard iteration can relieve some of the convergence difficulty associated with the NDA method.

2.4.2 Introducing the Damping Factor

The natural question to ask next is, since CMFD does not guarantee stability, how does one improve the stability? There are a couple possible options:

1. Perturb the entire matrix as suggested by Krasnoselskij iteration:

$$\bar{\bar{A}} = (1-\lambda)\,\bar{\bar{I}} + \lambda\,\bar{\bar{A}} \tag{2.33}$$

   This method does not take advantage of the properties of the operator and does not conserve any physics, so it is saved as a last resort.

2. Perturb the finite-different diffusion coefficient $\hat{D}$. As shown by Yamamoto, 1-N CMFD and CMR can be unified into one scheme called Generalized Coarse-Mesh Rebalance Method (GCMR) [Yamamoto, 2005]. Through linearized Fourier analysis, the different performance between 1-N CMFD and CMR can be explained by an acceleration factor f that multiplies on $\hat{D}$. In Yamamoto's formulation,

- When $f = 1.0$, GCMR becomes CMFD;
- When $f \approx 0.75\sigma h$, the spectral radius of GCMR is almost identical to that of the CMR method.

Thus by adjusting the value of f, a range of different convergence behavior can be obtained, and some optimum acceleration factor f can be found to stabilize the solution. One limitation of this method is that the optimum acceleration factor f can be analytically derived for simple homogeneous system, and finding such a value may be complicated for a truly heterogeneous system.

Yamamoto suggests that one can start a CMFD acceleration with $f = 1$, and if the convergence becomes unstable, f can be increased to stabilize the solution. When f gets large, the convergence of the CMFD equation becomes slow.

3. Perturb the non-linear coupling coefficient $\tilde{D}$ (or Current Correction Factor CCF as in some literature). The physical intuition behind this choice is that non-linear coupling coefficient is the term most sensitive to the leakage and thus is likely to change most dramatically from iteration to iteration. For instance, [Lee, 2012] places the under-relaxation on the non-linear coupling coefficients [3]:

$$\tilde{D}^{(m+1)} = \eta\, \tilde{D}^{(m+1/2)} + (1-\eta)\, \tilde{D}^{(m+1/2)} \tag{2.34}$$

This study also estimates the error and thus convergence ratio using the rate that the non-linear coupling coefficient is updated:

$$\tilde{D}^{(m)} = \tilde{D}^{(\infty)} + \epsilon^{(m)} \tag{2.35}$$

where $\tilde{D}^{(\infty)}$ is the exact non-linear diffusion coefficient, and $\epsilon^{(m)}$ is the error relative to the exact solution at the (m)-th transport iteration. Consequentially the spectral radius and convergence rate can be approximated as,

$$\rho = \frac{||\epsilon^{(m)} - \epsilon^{(m-1)}||_2}{||\epsilon^{(m-1)} - \epsilon^{(m-2)}||_2} \tag{2.36}$$

The test cases show that an under-relaxation factor on the non-linear coupling coefficient indeed expands the range of stability, and that "[t]he more the under-relaxation, the larger becomes the stability domain" [Lee, 2012].

[3] Notice the notation for non-linear coupling coefficient used in [Lee, 2012] is $\hat{D}$, whereas this thesis uses $\tilde{D}$. Also the index for iteration number in the paper is k where this thesis uses m.

2.4.3 Spectral Radius Evaluation

Analytically a convergence analysis can be performed by introducing a first-order error to the fluxes and performing a Fourier transform to change from the angle-space variables to the wave number λ variables. Then the eigenvalue of the iteration matrix $\omega(\lambda)$ is derived, and the largest value of $|\omega(\lambda)|$ determines the overall convergence behavior of the method. The smaller this number is, the faster the method converges. If $|\omega(\lambda)| \geq 1$ the mode diverges [Larsen and Morel, 2010]. The limit of this analytical analysis is that it is typically only practical for infinitely homogeneous medium with uniform meshes. The math involved in this type of Fourier analysis becomes very complicated for multi-dimensional problems and for methods beyond source iteration [Adams and Larsen, 2002]. Thus for this thesis a numerical evaluation is presented below and an analytical convergence analysis will be left for future work.

Numerically the spectral radius of an iterative method can be approximated using the following equation for the (m)-th iteration:

$$\rho^{(m)} \approx \frac{||\phi^{(m)} - \phi^{(m-1)}||_2}{||\phi^{(m-1)} - \phi^{(m-2)}||_2} \tag{2.37}$$

and the spectral radius of the method is estimated as,

$$\rho = \sup_{-\infty<\lambda<\infty}|\omega(\lambda)| = \lim_{m\to\infty} \frac{||\phi^{(m)} - \phi^{(m-1)}||_2}{||\phi^{(m-1)} - \phi^{(m-2)}||_2} \tag{2.38}$$

Two test cases are chosen to illustrate the numerically computed spectral radii. One is the C5G7 benchmark problem's fuel assemblies modeled with reflective boundary condition. This case is illustrated in Fig. 2-2. The second test case is the C5G7 problem illustrated in Fig. 2-3. These two problems are described in more details in Section 4.4.1 and Section 4.4.2.

The C5G7 fuel assembly problem converges without under-relaxation, and its spectral radius is reduced with a damping factor of 0.7 as illustrated in the top plot of Fig. 2-4. The C5G7 problem has more angular dependency from the water reflectors surrounding the fuel assemblies. When accelerated with CMFD, the C5G7 problem diverges without under-relaxation, and a damping factor brings the asymptotic spectral radius from above 1.0 to below 1.0 as illustrated in the bottom plot of Fig. 2-4.

This same behavior can be observed from the convergence of the L2 norm of the relative change in FSRs' energy-integrated fission source in Fig. 2-5. Again the top plot is for the C5G7's fuel region problem where the damping factor converges the problem faster. The bottom plot is for the C5G7 problem, where with an under-relaxation, the norm decreases monotonically from iteration to iteration, whereas without the under-relaxation the solution diverges.

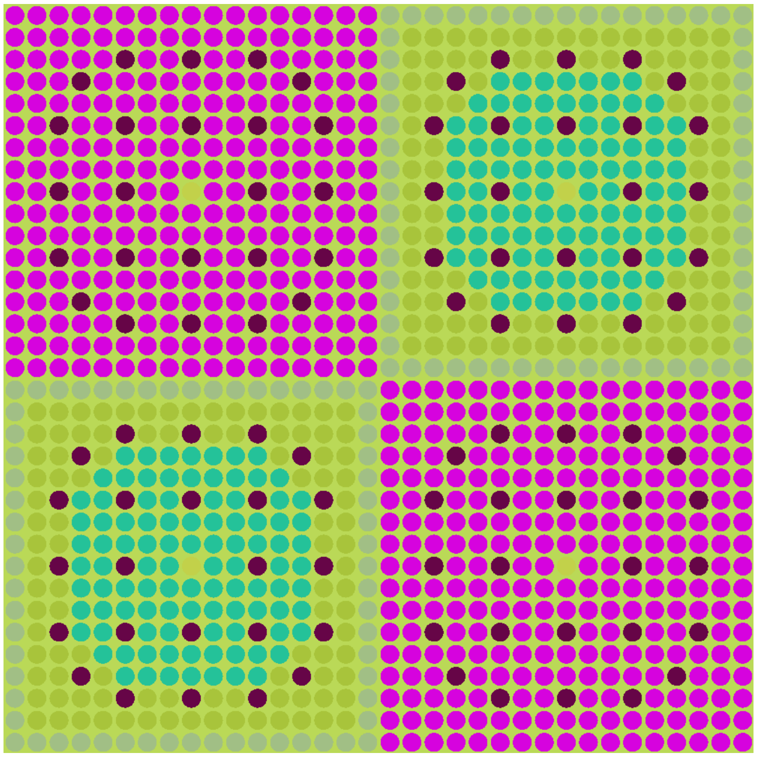

Figure 2-2: Illustration of C5G7 Fuel Assemblies Problem

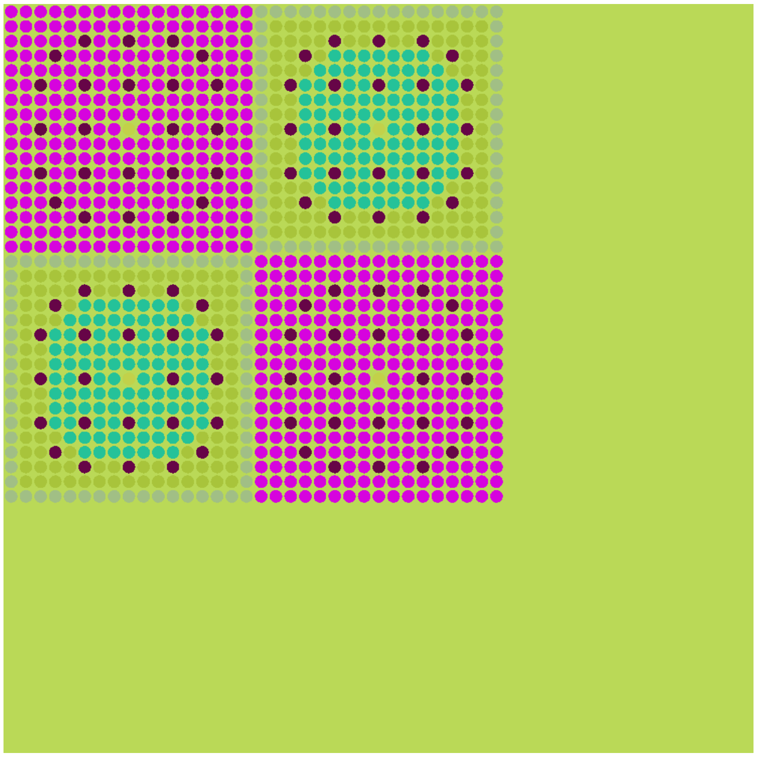

Figure 2-3: Illustration of C5G7 Benchmark Problem

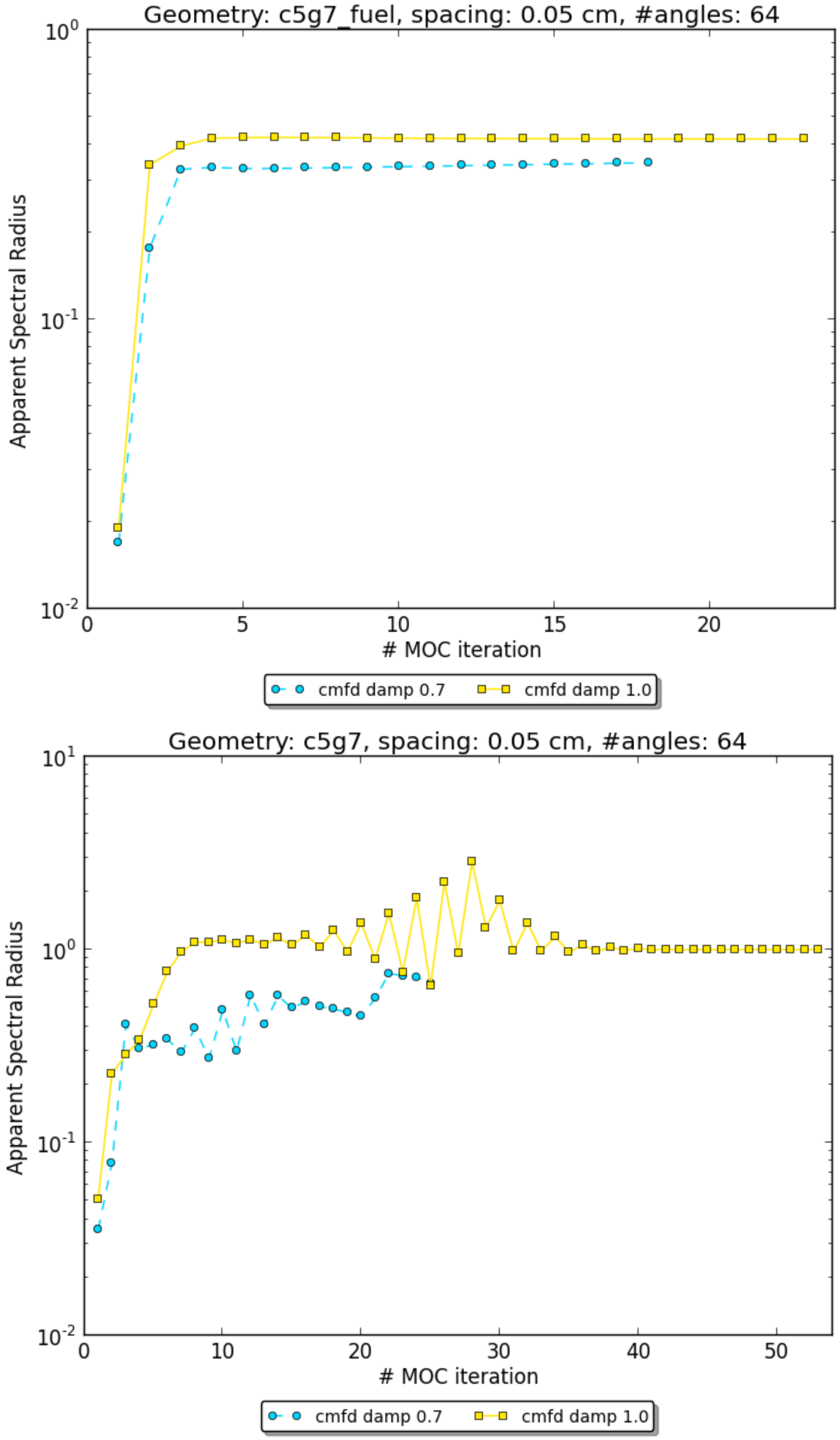

Figure 2-4: Plot of C5G7 Benchmark Problem Numerical Spectral Radii with and without Damping

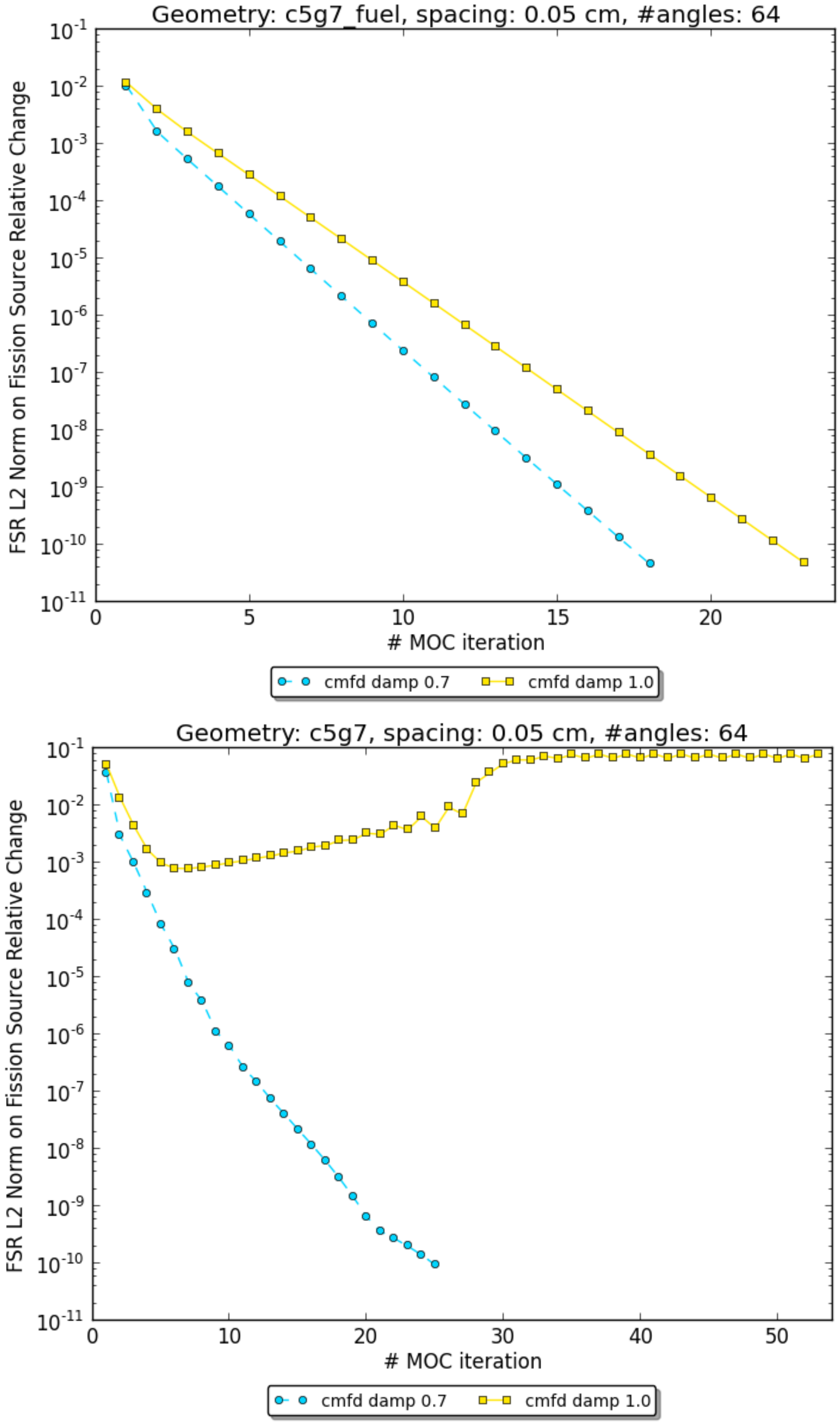

Figure 2-5: Plot of C5G7 Benchmark Problem Convergence Behavior with and without Damping

2.5 Computational Results

In this section computational results are presented to illustrate the effect of CMFD acceleration on the C5G7 benchmark problem. Description of this C5G7 benchmark can be found in Section. 4.4. More results on different test cases will be presented in Chapter 4 where the performance of CMFD and LOO are compared.

2.5.1 CMFD-accelerated vs. Non-accelerated Results

Fig. 2-6 shows the speedup obtained through CMFD acceleration. The y-axis is the L2 norm of the relative change between successive iteration's energy-integrated fission source. The x-axis is the iteration number of the high-order MOC transport sweep. As shown in Fig. 2-6, it takes unaccelerated MOC solver more than 1000 transport sweeps to obtain similar convergence as the MOC solver accelerated with CMFD converges in approximately 30 iterations. As expected, CMFD acceleration dramatically increases the convergence rate of this problem.

In addition, CMFD preserves the high-order heterogeneous transport solution in the sense than when converged, the eigenvalue generated from the low-order CMFD solver agrees with the high-order MOC transport solver.

2.5.2 Damping Factor Comparison

Another topic of interest is the effect of the damping factor on the CMFD method. In this work, a fixed under-relaxation factor is applied to the non-linear coupling coefficient at each transport sweep. For future work a dynamic damping factor will be investigated.

Table. 2.1 shows the number of transport sweeps required to converge the C5G7 benchmark problem with 0.05cm track spacing, 64 azimuthal angles and 3 polar angles to the convergence criteria of 1e-10 on the L2 norm of FSRs' relative change in energy-integrated power. In this case, CMFD method reduces the number of MOC transport sweeps by a factor of 50, making MOC accelerated by CMFD a practical method for large LWR problems. Further the damping factor does not change the converged eigenvalue. The unaccelerated eigenvalue reported in Table 2.1 is slightly different from the CMFD-accelerated results. This is a discrepancy arises because the unaccelerated OpenMOC converges slowly, while the CMFD-accelerated ones have a much faster convergence rate and end up further below the threshold criteria. If the unaccelerated OpenMOC is further converged, its eigenvalue agrees with the CMFD-accelerated ones.

Fig. 2-7 illustrates the convergence behavior of the four damping factors corresponding to Table. 2.1. Based on both the table and the figure, a damping factor around 0.6 and 0.7 appears to be the optimal damping factor which achieves the most rapid convergence.

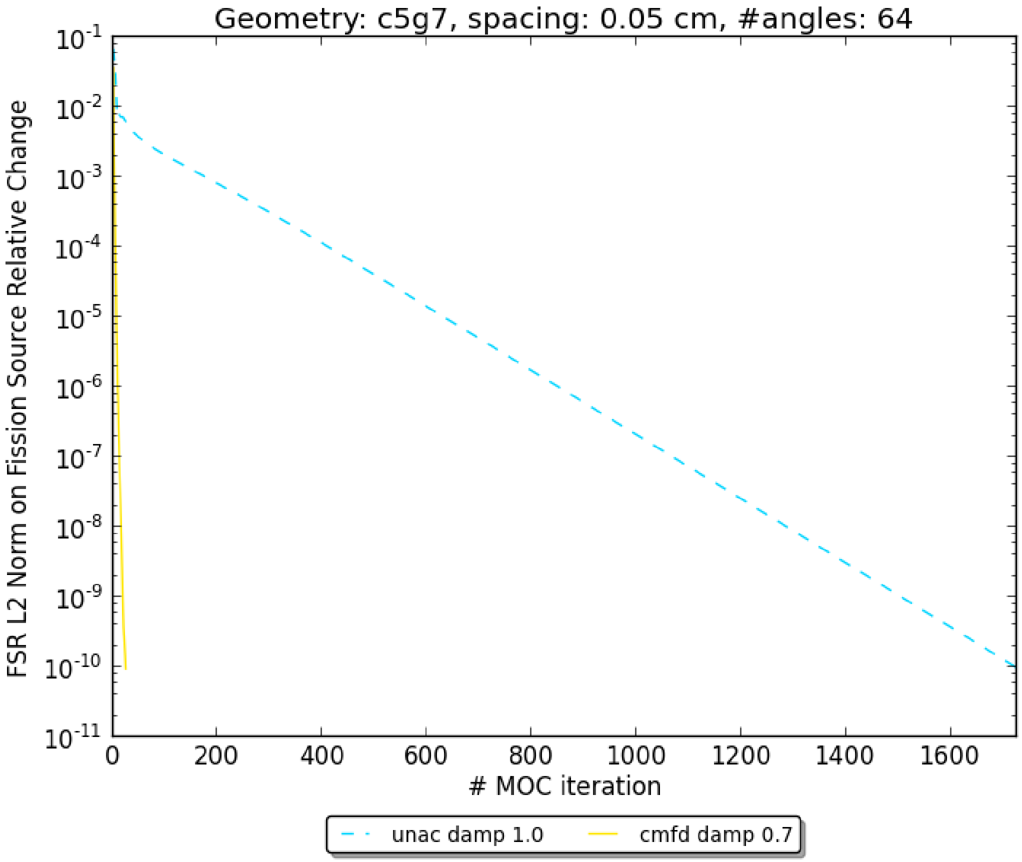

Figure 2-6: Plot of C5G7 Benchmark Problem Convergence Behavior with and without CMFD Acceleration

Method	k_{eff}	# Transport Sweep
No acceleration	1.18730527̲23	1724
CMFD with damping factor of 0.4	1.1873052742	37
CMFD with damping factor of 0.5	1.1873052742	28
CMFD with damping factor of 0.6	1.1873052742	22
CMFD with damping factor of 0.7	1.1873052743	25
CMFD with damping factor of 0.8	1.1873052742	53
CMFD with damping factor of 0.9	–	–
CMFD with damping factor of 1.0	–	–

Table 2.1: CMFD Acceleration Results

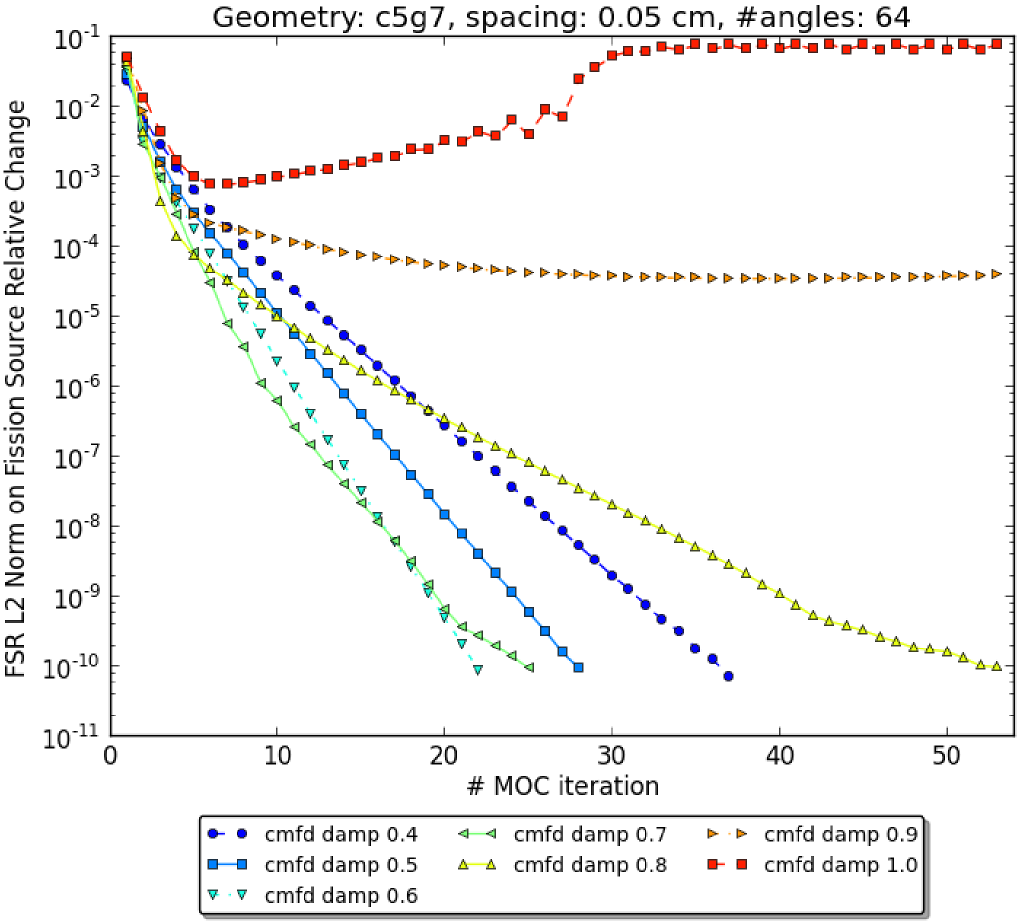

Figure 2-7: Plot of C5G7 Benchmark Problem Convergence Behavior with CMFD Acceleration using Different Damping Factors

Chapter 3

Low Order Transport Methodology

This work proposes a Low Order Operator (LOO) acceleration scheme for the solution of the neutron transport equation using the method of characteristics. By conserving the spatial and angular moments, LOO is anticipated to reduce the number of MOC transport iterations.

3.1 Motivation and Background

As summarized in Section 2.1, various diffusion-based low-order acceleration methods have shown success in accelerating high-order transport methods. Though diffusion-based acceleration methods work well for diffusive problems, e.g., optically thick problem with weak absorption, they can encounter convergence issues in non-diffusive problems. For instance, a simple and elegant theoretical derivation shows that the diffusion approximation is consistent with S_N spatial discretization only if the problem satisfies the optical thick diffusion limit [Larsen and Morel, 2010].

In addition, a diffusion-based method may be slow in converging a transport-dominant problem, e.g., one with strong angular effects. One possible improvement is to introduce a low-order acceleration scheme that is based on the transport equation.

More specifically, LOO is inspired and motivated by the following concepts:

1. As discussed in Chapter 2,

   (a) Coarse-Mesh Angular Dependent Rebalance (CMADR) improves the traditional CMR by introducing an angular dependent re-balancing factor that adds one more degree of freedom;

   (b) CMFD improves the traditional diffusion theory by using a coarse mesh that conserves the net current via the non-linear coupling coefficient $\tilde{D}$;

(c) p-CMFD (partial CMFD) improves traditional CMFD by conserving partial current on each mesh surface by introducing one more degree of freedom;

(d) Generalized Coarse-Mesh Rebalance Method (GCMR) unifies the traditional CMR and CMFD methods by introducing the acceleration factor f which adds one degree of freedom.

This trend suggests that a low-order acceleration scheme performs better by conserving more physical quantities and/or adding more degrees of freedom. Following the same idea, a method that further conserves the flux and current in both space and angle might be advantageous.

2. In 2002, Smith and Rhodes proposed a macro-track transport acceleration method which lays "macro-tracks" directly on top of the "micro-tracks" of the fine grid MOC. Spatially flat equivalent sources are then defined to preserve the outgoing angular fluxes of the macro-tracks. The macro-tracks are not limited by any rectangular meshes and thus are applicable to general unstructured meshes. FSR scalar fluxes need to be updated at each low-order iterative step, and the high-order MOC transport sweep is only performed to generate parameters for the coarse mesh system. This method was implemented in CASMO-4, and "overall speedups of 20-50 relative to unaccelerated MOC power iteration are routinely achieved in large LWR problems" [Smith and Rhodes, 2002]. The macro-track method generally requires more power iterations to meet the same convergence criteria as the CMFD-accelerated MOC iterations. One exception is the cases where CMFD has difficulty converging, then the macro-track acceleration method "is more effective in reducing the number of required MOC transport sweeps" [Smith and Rhodes, 2002].

3. In [Grassi, 2007] a non-linear space-angle multigrid acceleration method was proposed to solve a more coarsely discretized phase-space problem in low-order using MOC method. This method is general for handling unstructured mesh, i.e., that a rectangular acceleration mesh like the one in CMFD is not required. It does not conserve the currents and thus leakage between the low-order and high-order system, and uses a Discontinuity Factor (DF) for each surface to conserve the partial currents. His study concludes that "the correction provided by our acceleration step can be more accurate than that provided by CMFD for a class of reactor transport problems in which diffusion modes are not dominant and transport effects due to higher modes become more important (e.g., accident situations)", however the number of high-order transport sweeps required for his method is larger than that with CMFD on the C5G7MOX benchmark problem [Grassi, 2007].

The goal of this work is to propose a low-order operator MOC transport acceleration with the following properties:

- It conserves the first-order spatial and angular moments in the quadrature space-angle. The goal is to provide better agreement and consistency between the high-order system and the low-order system.

- It is simple and straight-forward to understand and implement.
- It is stable for typical LWR problems;
- It converges typical LWR problems in fewer transport sweeps than CMFD.

Throughout this work, the proposed LOO method will be compared with the CMFD method as the latter is the state-of-the-art low-order acceleration scheme for solving heterogeneous reactor transport problems.

3.2 LOO Fundamentals

The LOO method proposed in this work has the following major differences compared with previous low-order transport methods (the macro-track transport acceleration [Smith and Rhodes, 2002] and the non-linear space-angle multigrid acceleration [Grassi, 2007]):

1. LOO uses rectangular meshes with simple coarse characteristics. Both the macro-track transport acceleration and the non-linear space-angle multigrid acceleration are formulated for general unstructured mesh. The two methods differ in that the former method's macro-track is a summation of micro-track (i.e., the fine grid track lay down for high-order MOC), and the latter method defines macro-region by "fusing together adjacent fine meshes" and employs a set of discrete angles and weights for the low-order.

   Recall that one of the goals of LOO is to generate a low-order system that is simple yet still consistent with the high-order system. LOO achieves this goal by choosing square mesh cells as the coarse grid (although only square mesh cells are implemented, the LOO theory can be easily extended to rectangular mesh cells and will be in the future work).

   The space-angle discretization is confined to the quadrant space-angle domains with equal weights. For instance, the track laydown in coarse mesh cell I is illustrated in Fig. 3-1. This choice of discretization significantly simplifies the coarse grid structure and the restriction and prolongation step.

2. The quadrant source concept is a novelty of this work. To the author's knowledge, there are no low-order methods that allow sources that occupy the same spatial location to have angular dependency. For instance, both the macro-track transport acceleration and the nonlinear space-angle multigrid acceleration assumes source to be flat in each mesh cell.

   In LOO, every cell contains eight source terms, one for each of the eight coarse grid characteristic tracks. Like the nonlinear coupling coefficients $\tilde{D}$ used in CMFD, the quadrature-dependent source terms ensures the conservation of both current and reaction rates when compared to the high-order MOC solution. Because the leakage term is preserved between the high-order system and the low-order system, there is no need for the Discontinuity Factors (DFs) used in the nonlinear space-angle multigrid acceleration method, and this also eliminates some reconstruction steps.

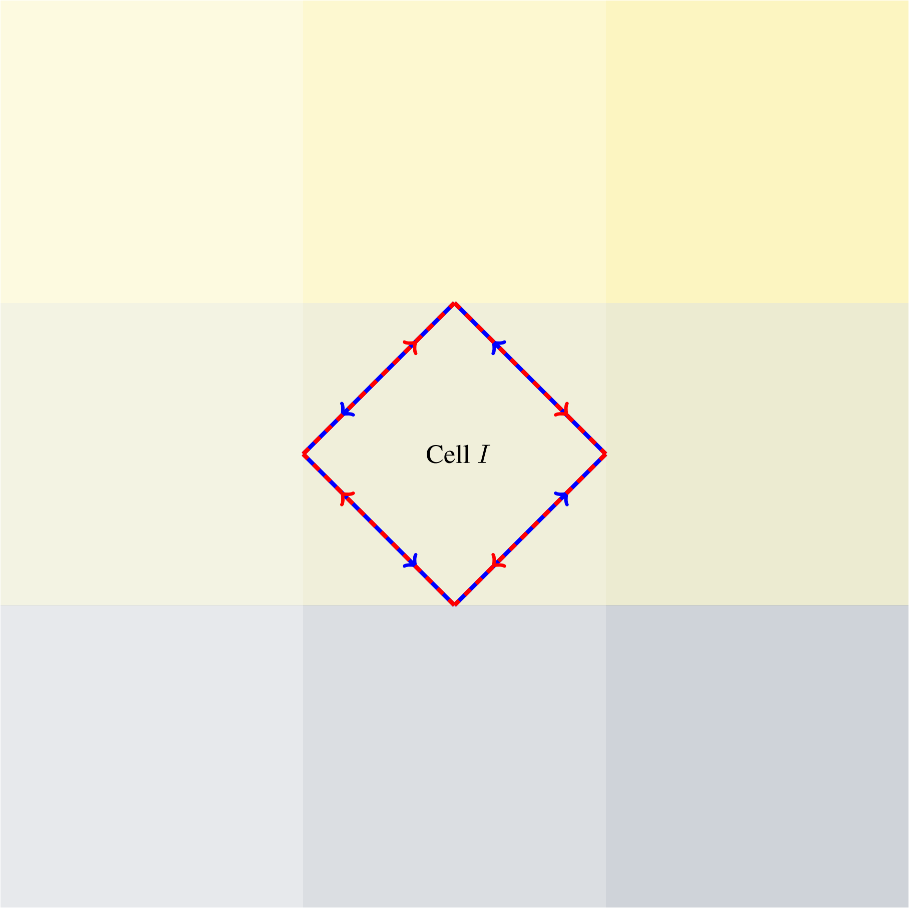

Figure 3-1: Illustration of Coarse Mesh Characteristics In Cell I

3.3 LOO Formulation

In this section detailed description of the LOO methods will be given with special emphasis on the quadrant-integrated concepts. The material is presented in a similar order as in the CMFD method in Section 2.3 for easy comparison between the low-order acceleration schemes. New notations will be introduced, complementing what is previously found in Section 1.3.1.

3.3.1 Flow of Calculations Overview

The basic steps in the LOO-accelerated OpenMOC are outlined in this section with emphasis on the LOO current tallies. Refer to Section 1.4 for more details of the high-order MOC transport sweep. For brevity, any previously presented expressions are referred to by equation number. The detailed LOO-related steps (e.g., restriction, iterative solve, prolongation) are deferred to later sections.

1. Geometry initialization: the initialization of FSRs and generation of tracks and segments remain the same as the unaccelerated OpenMOC. In addition, OpenMOC sets up the coarse mesh cell structure. Conveniently, LOO uses the same square mesh cells as the CMFD method so the coarse grid initialization process is identical to the one presented in Section 2.3.1 for CMFD. The only difference is there are no longer any diffusion coefficients and non-linear coupling coefficients, and the net current variables are replaced by mesh surface-averaged angular-quadrant currents. Additional memory has to be allocated for quadrant fluxes and sources in each coarse mesh cell.

2. Solver initialization: OpenMOC initializes the high-order system with FSR scalar fluxes $\phi_{g,i}^{(m=0)} = 1.0$, geometry boundary angular fluxes $\psi_{g,i,m,k}^{-,(m=0)} = \dfrac{1}{4\pi}$ and $k_{\text{eff}}{}^{(m=0)} = 1.0$.

3. Storing the (m)-th iteration mesh cell averaged source $\bar{q}_{g,I}^{(m)}$: unique to LOO's source computation method, a copy of the mesh cells' source before performing the high-order transport sweep needs to be stored. This is done by simply performing a volume-weighted summation of the FSR sources:

$$\bar{q}_{g,I}^{(m)} = \frac{\sum\limits_{i \in I} \bar{q}_{g,i}^{(m)} V_i}{\sum\limits_{i \in I} V_i} \tag{3.1}$$

4. The (m)-th iteration high-order MOC sweep: for the m-th MOC sweep, OpenMOC performs five levels of nested loops over the azimuthal angles, tracks, segments, energy groups, and polar angles to update the following quantities:

   (a) For each segment in the 3D space, $\psi_{g,i,m,k}^{+}$ is computed from $\psi_{g,i,m,k}^{-}$, $\Sigma_{t,g,i}$, $S_{i,m,k}$, $q_{g,i}$ using Eq. 1.19.

(b) The contribution of each segment is accumulated into the FSR's scalar flux $\phi_{g,i,m,k}$ using Eq. 1.33.

(c) If a segment ends on a mesh surface s, its angular flux is used to accumulate the quadrant current. Similar to Eq. 2.12, OpenMOC tallies the azimuthal and polar angle weighted angular flux projected onto the normal direction of the surface, except it is in the quadrant space-angle $\Omega_{\pm,u}$ in LOO:

$$\hat{J}^{\pm,u}_{g,I,s} = \frac{\sum\limits_{m\in\Omega_{\pm,u}} \omega_m \delta_m \sin\theta_p \sum\limits_k \psi_{g,i,m,k}}{\sum\limits_{m\in\Omega_{\pm,u}} \omega_m \delta_m} \tag{3.2}$$

where the index g is for energy group, I for coarse mesh cell, s for surface. The superscript ± indicates which angular half-space the current is in, and $u = 1, 2$ further distinguishes between the corresponding quadrants (e.g., up or down for the vertical surfaces, and left or right for the horizontal ones) in each angular half-space.

The surface-averaged quadrature-integrated current $\hat{J}^{\pm,u}_{g,I,s}$, or simply "quadrant current", is defined similarly as partial current, except it is defined for four space-angle domains. That is, there are four quadrant currents associated with each surface. By definition, the $\hat{J}^{+,u}_{g,I,s}$ are the two quadrant currents designated by $u = 1, 2$ in the positive half space for the energy group g, coarse mesh cell I, and surface s:

$$\hat{J}^{+,1}_{g,I,s} = \sum_{m\in\Omega_{+,1}} \omega_m \sin\theta_p |\mu_a| \psi_{g,i,m,k} \qquad \hat{J}^{+,2}_{g,I,s} = \sum_{m\in\Omega_{+,2}} \omega_m \sin\theta_p |\mu_a| \psi_{g,i,m,k} \tag{3.3}$$

The $\hat{J}^{-,u}_{g,I,s}$ are the two quadrant currents designated by $U = 1, 2$ in the negative half space for the energy group g, coarse mesh cell I, and surface s:

$$\hat{J}^{-,1}_{g,I,s} = \sum_{m\in\Omega_{-,1}} \omega_m \sin\theta_p |\mu_a| \psi_{g,i,m,k} \qquad \hat{J}^{-,2}_{g,I,s} = \sum_{m\in\Omega_{-,2}} \omega_m \sin\theta_p |\mu_a| \psi_{g,i,m,k} \tag{3.4}$$

The superscript ± designates whether the quadrant current is in the positive or negative half space of the mesh surface with surface normal pointing outwards. The superscript designates whether the characteristic track that the segment belongs to has a φ_a less than $\frac{\pi}{2}$ or greater than $\frac{\pi}{2}$ [1]. This definition for u is chosen such that the forward and backward segments have the same u. See Fig. 3-2 for an illustration.

LOO method chooses to tally quadrant currents instead of quadrant fluxes during the high-order

[1] Recall that in OpenMOC, characteristic tracks are lay down in the $\varphi_a \in [0, \pi)$ only, and the forward and backward segments share the same characteristic track.

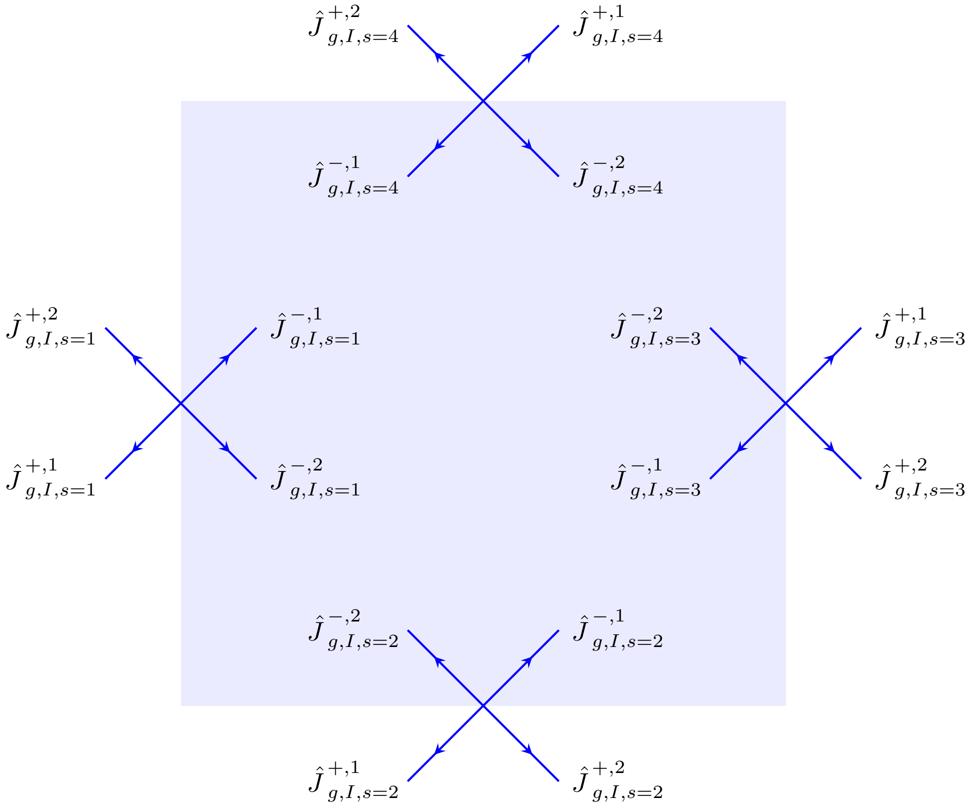
$\hat{J}^{+,2}_{g,I,s=4}$
$\hat{J}^{+,1}_{g,I,s=4}$
$\hat{J}^{-,1}_{g,I,s=4}$
$\hat{J}^{-,2}_{g,I,s=4}$
$\hat{J}^{+,2}_{g,I,s=1}$
$\hat{J}^{-,1}_{g,I,s=1}$
$\hat{J}^{-,2}_{g,I,s=3}$
$\hat{J}^{+,1}_{g,I,s=3}$
$\hat{J}^{+,1}_{g,I,s=1}$
$\hat{J}^{-,2}_{g,I,s=1}$
$\hat{J}^{-,1}_{g,I,s=3}$
$\hat{J}^{+,2}_{g,I,s=3}$
$\hat{J}^{-,2}_{g,I,s=2}$
$\hat{J}^{-,1}_{g,I,s=2}$
$\hat{J}^{+,1}_{g,I,s=2}$
$\hat{J}^{+,2}_{g,I,s=2}$

Figure 3-2: Illustration of Quadrant Current $\hat{J}^{\pm,u}_{g,I,s}$

MOC sweeps for simplicity: as described in Section 2.3.2. When tallying currents, the angular fluxes are weighted by the exact same terms used in accumulating FSR scalar fluxes, thus saving some computational cost by not re-computing new weights every time. Later in the restriction phase, the quadrant currents will be converted into the quadrant fluxes by dividing by a simple constant.

5. Upon completing the smoothing step on the high-order, a LOO restriction step is performed where mesh cell homogenized quantities and quadrant quantities are computed. See Section 3.3.2 for more details.

6. Execute the LOO iterative solver until the low-order convergence criteria is met. See Section 3.3.3 for details.

7. Perform the LOO prolongation to update the FSR's scalar flux and geometry boundary fluxes. See Section 3.3.5 for details. Upon obtaining the new FSR scalar fluxes, new ${k_{\text{eff}}}^{(m+1)}$ is computed, and FSR sources are updated:

$$\bar{q}_{g,I}^{(m+1)} = \sum_{g'} \left[\frac{\chi_{g,I} \nu \Sigma_{f,g',I}}{{k_{\text{eff}}}^{(m+1)}} + \Sigma_{s,g' \to g,I} \right] \phi_{g',I}^{(m+1)} \tag{3.5}$$

8. If the convergence criteria on the high-order solution is not met, return to step 2 and iterate until converged. In OpenMOC current implementation, the convergence criteria is on the L2 norm of successive change in relative energy-integrated fission source as specified in Eq. 2.30.

To summarize, LOO lays down eight representative tracks in each mesh cell. A high-order MOC transport sweep provides the quadrature-integrated fluxes which serve as the incoming and outgoing fluxes of the low-order characteristic tracks. A source term is computed for each track, as well as the quadrant source shape factors. A sequence of low-order transport sweeps are performed, and quadrant fluxes and the cell-averaged scalar fluxes are updated. If not converged, new quadrant sources are computed and the low-order iteration continues until the convergence criteria is met.

3.3.2 LOO Restriction

Similar to CMFD and other physical multi-grid methods, a restriction step is needed to build an initial low-order system based on the high-order system. All variables in the restriction phase are of the $(m + 1/2)$-th iteration. Thus for simplicity the superscript $(m + 1/2)$ is often dropped in this section.

The calculations of the coarse mesh cell volume-averaged scalar fluxes $\bar{\phi}_{g,I}$ and the flux-weighted cross-sections $\overline{\Sigma}_{g,xx,I}$ for reaction type xx are identical as in the CMFD method. See Section 2.3.3. In addition, LOO constructs the following quadrant quantities:

1. The quadrant fluxes $\hat{\psi}^{\pm,u}_{g,I,s}$ are computed based on the tallied quadrant currents $\hat{J}^{\pm,u}_{g,I,s}$ for each mesh I's every surface s.

   Quadrant-integrated angular flux, or simply "quadrant flux", is defined as the angular flux integrated over the quadrature space-angle domain. In LOO, quadrant currents are tallied during the high-order MOC sweeps, and quadrant fluxes are computed from the quadrant currents by simply performing:

   $$\hat{\psi}^{\pm,u}_{g,I,s} = \frac{\hat{J}^{\pm,u}_{g,I,s}}{\cos\left(\frac{\pi}{4}\right)} \tag{3.6}$$

   Since the quadrant current is already projected onto the normal direction, and the representative quadrant flux is considered to be in the center of the quadrature phase-space, there is a $\frac{\pi}{4}$ difference between the representative quadrant flux and the normal direction of the mesh surface. Then to reproduce the quadrant flux consistent with the quadrant current, we divide the quadrant current by the cosine of the difference in angle and obtain Eq. 3.6.

2. The quadrant fluxes $\hat{\psi}^{\pm}_{g,I,q}$ are initialized for each mesh cell I based on the $\hat{\psi}^{\pm,u}_{g,I,s}$ computed previously.

   Both $\hat{\psi}^{\pm,u}_{g,I,s}$ and $\hat{\psi}^{\pm}_{g,I,q}$ are quadrant fluxes. The difference between the two notations is that the former is defined for coarse mesh cell I surface s, and the latter is defined for coarse mesh cell I low-order quadrature track q. In Fig. 3-3, $\hat{\psi}^{\pm,u}_{g,I,s}$ is illustrated in the left plot where there are four quadrant fluxes associated with each mesh surface. Adding up four surfaces, there are 16 $\hat{\psi}^{\pm,u}_{g,I,s}$ associated with coarse mesh cell I and energy group g. In the right plot of Fig. 3-3, the eight low-order characteristic tracks are illustrated, each containing an incoming angular flux designated by the $+$ superscript and an outgoing angular flux designated by the $-$ superscript. Together there are 16 $\hat{\psi}^{\pm}_{g,I,q}$ associated with the a mesh cell and an energy group (two for each of the eight characteristic tracks: $q = 1, \cdots, 8$). Further there is a one-to-one correspondence between the 16 $\hat{\psi}^{\pm,u}_{g,I,s}$ and 16 $\hat{\psi}^{\pm}_{g,I,q}$ for each mesh cell I and energy group g. One tallies $\hat{\psi}^{\pm,u}_{g,I,s}$ during a high order sweep, and uses them to initialize $\hat{\psi}^{\pm}_{g,I,q}$ at the beginning of the LOO method.

   To summarize, $\hat{\psi}^{\pm,u}_{g,I,s}$ exists on the mesh surfaces $s = 1, \cdots, 4$ via the space-angle integration of the high-order solution. Because of the quadrant space-angle discretization chosen in LOO, each one of them is exactly correspond to either the incoming or outgoing angular flux of one of the 8 coarse mesh characteristic tracks.

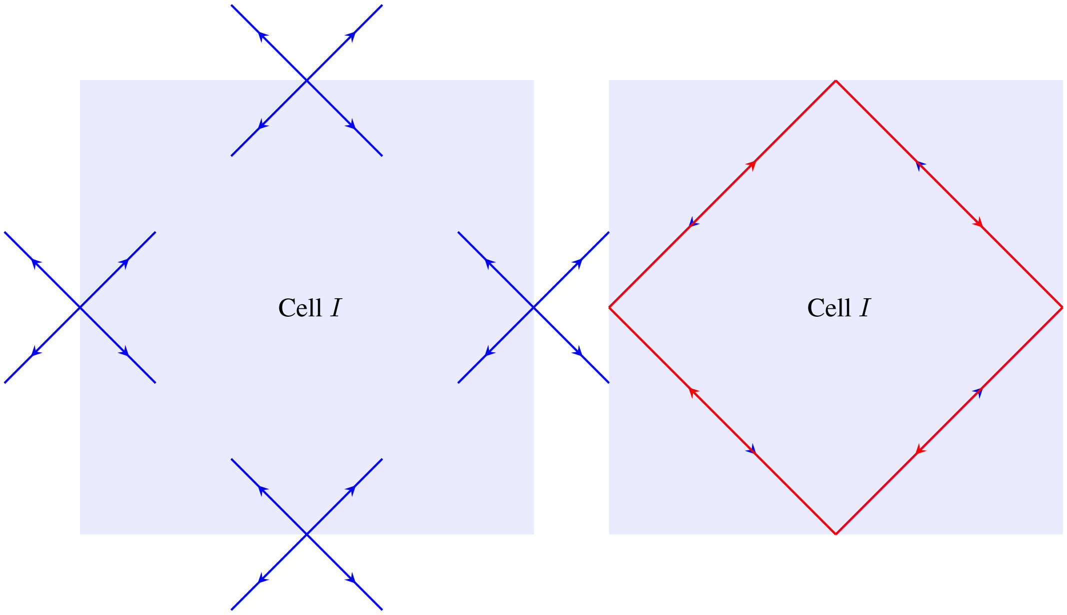

Figure 3-3: Illustration of Quadrant Flux and Coarse Mesh Characteristics

3. The mesh cell averaged sources $\bar{q}_{g,I}$ are computed. The equation for computing $\bar{q}_{g,I}^{(m+1/2)}$ is identical to $\bar{q}_{g,I}^{(m)}$ in Eq. 3.1 except the source terms are of iteration $(m+1/2)$:

$$\bar{q}_{g,I}^{(m+1/2)} = \frac{\sum\limits_{i \in I} \bar{q}_{g,i}^{(m+1/2)} V_i}{\sum\limits_{i \in I} V_i} \tag{3.7}$$

4. The quadrature-integrated sources $\hat{q}_{g,I,q}$, or simply "quadrant sources", are computed for the q-th low-order characteristic track using $\hat{\psi}^{\pm}_{g,I,q}$ which are the incoming and outgoing flux corresponding to the q-th track, the g-th energy group, and the I-th coarse mesh cell.

   Although the subscripts alone are sufficient for distinguishing the high-order and the low-order variables, accents are added for redundancy and clarity: the hats on $\hat{\psi}$, $\hat{q}$, $\hat{S}$ emphasizes that they are unique for every coarse mesh track; the overhead bar on $\overline{\Sigma}$, $\bar{\phi}$ emphasizes that they are mesh cell averaged quantities; and the term $\bar{\hat{\psi}}$ is the coarse mesh track averaged angular flux.

   To compute the q-th quadrature track's source, recall Eq. 1.19 where the outgoing angular flux of a track is written in terms of the incoming angular flux, the track's total cross-section, length and source. The same equation can be written for the low-order characteristics by replacing the high-order's FSR index i with low-order's mesh cell index I, replacing the high-order's angle and segment dependencies

m, k with low-order's quadrature representative track index q, and adding a q index to signify that the sources are track-dependent:

$$\hat{\psi}^{+}_{g,I,q} = \hat{\psi}^{-}_{g,I,q} e^{-\overline{\Sigma}_{t,g,I}\hat{S}_{I,q}} + \frac{\hat{q}_{g,I,q}}{\overline{\Sigma}_{t,g,I}}(1 - e^{-\overline{\Sigma}_{t,g,I}\hat{S}_{I,q}}) \tag{3.8}$$

Eq. 3.8 can be re-written to get $\hat{q}_{g,I,q}$:

$$\hat{q}_{g,I,q} = \frac{\overline{\Sigma}_{t,g,I}}{1 - e^{-\overline{\Sigma}_{t,g,I}\hat{S}_{I,q}}}(\hat{\psi}^{+}_{g,I,q} - \hat{\psi}^{-}_{g,I,q} e^{-\overline{\Sigma}_{t,g,I}\hat{S}_{I,q}}) \tag{3.9}$$

Eq. 3.9 computes an "equivalent" low-order source for each coarse mesh track using the track's incoming and outgoing angular fluxes. These angular fluxes are initialized using high-order quadrature space-angle integrated results, and will be updated at each low-order iteration.

5. For one of the two variations of LOO proposed in this work, the summation of the low-order characteristic-averaged flux in each cell is also computed. Track-averaged quadrant flux for each characteristic: $\bar{\hat{\psi}}_{g,I,q}$ is used to designate the average (thus the overhead bar) of the q-th coarse mesh characteristic in the I-th mesh cell and g-th energy group.

   Recall again in high-order the segment-averaged angular flux is derived in Eq. 1.21. Its low-order equivalent is,

$$\bar{\hat{\psi}}_{g,I,q} = \frac{\hat{q}_{g,I,q}}{\overline{\Sigma}_{t,g,I}} + \frac{\hat{\Delta}_{g,I,q}}{\overline{\Sigma}_{t,g,I}\hat{S}_{I,q}} \tag{3.10}$$

   where $\hat{\Delta}_{g,I,q} = \hat{\psi}^{-}_{g,I,q} - \hat{\psi}^{+}_{g,I,q}$ is the negative change in flux along the q-th low-order characteristic track in the I-th mesh cell at the g-th energy group. Thus the summation of these averaged fluxes in each cell yields,

$$\sum_{q} \bar{\hat{\psi}}^{(m+1/2)}_{g,I,q} = \sum_{q}\left(\frac{\hat{q}_{g,I,q}}{\overline{\Sigma}_{t,g,I}} + \frac{\hat{\Delta}_{g,I,q}}{\overline{\Sigma}_{t,g,I}\hat{S}_{I,q}}\right) \tag{3.11}$$

   This term is stored at the $(m + 1/2)$ iteration level and will be used in updating mesh cell scalar fluxes. More details will be provided in Section 3.3.4.

3.3.3 LOO Iterative Solver

In this section the steps in the LOO iterative solver are described. Recall that the iteration index (m) is for the high-order iteration, and (l) is for the low-order iterations presented in this subsection.

1. Initialize LOO variables with results of the high-order MOC sweep:

$$\bar{\phi}_{g,I}^{(l=0)} = \bar{\phi}_{g,I}^{(m+1/2)} \qquad\qquad k_{\text{eff}}{}^{(l=0)} = k_{\text{eff}}{}^{(m+1/2)} \tag{3.12}$$

2. Compute the mesh cell averaged sources for the (l)-th low-order iteration:

$$\bar{q}_{g,I}^{(l)} = \frac{1}{4\pi}\sum_{g'}\left[\overline{\Sigma}_{s,g'\to g,I}^{(m+1/2)} + \frac{\chi_{g,I}^{(m+1/2)}}{k_{\text{eff}}{}^{(l)}}\,\overline{\nu\Sigma}_{f,g',I}^{(m+1/2)}\right]\bar{\phi}_{g',I}^{(l)} \tag{3.13}$$

3. Compute the quadrant sources for the (l)-th low-order iteration:

$$\hat{q}_{g,I,q}^{(l)} = \frac{\hat{q}_{g,I,q}^{(m+1/2)}}{\bar{q}_{g,I}^{(m)}}\,\bar{q}_{g,I}^{(l)} \tag{3.14}$$

   where $\bar{q}_{g,I}^{(m)}$ is the coarse mesh cell averaged source that goes into the (m)-th high-order MOC sweep computed in Section 3.3.1, $\bar{q}_{g,I}^{(l)}$ is the latest coarse mesh cell averaged source.

4. Perform a low-order MOC iteration sweeping through the coarse characteristic tracks. There are two components of this step:

   (a) Every low-order track $\hat{\psi}_{g,I,q}^{+}$ is computed based on $\hat{\psi}_{g,I,q}^{-}$, $\overline{\Sigma}_{t,g,I}$, $\hat{S}_{I,q}$ and $\hat{q}_{g,I,q}$ using Eq. 3.8.

   (b) Terms are accumulated for updating the mesh cell averaged scalar fluxes. This step is deferred to Section 3.3.4 where the two variations of LOO are discussed.

5. Compute the new cell-averaged scalar flux $\bar{\phi}_{g,I}^{(l+1)}$: there are two variations of how this can be done, and the description will be presented in Section 3.3.4.

6. Calculate the new low-order eigenvalue $k_{\text{eff}}{}^{(l+1)}$:

$$k_{\text{eff}}{}^{(l+1)} = \frac{\sum_I V_I \sum_g \overline{\nu\Sigma}_{f,g,I}^{(m+1/2)} \ \bar{\phi}_{g,I}^{(l+1)}}{\sum_{s\in S_{vac}} J_{g,I,s}^{+,(l+1)} d_{I,s} + \sum_I V_I \sum_g \overline{\Sigma}_{a,g,I}^{(m+1/2)} \ \bar{\phi}_{g,I}^{(l+1)}} \tag{3.15}$$

where $\sum_{s\in S_{vac}} J_{g,I,s}^{+,(l+1)} d_{I,s}$ is the total leakage term by summing all the partial currents leaving a surface s (which is in the positive sense of surface +) if s is on a vacuum boundary S_{vac}.

7. Check whether the low-order fission source meets the convergence criteria. Similar to the high-order MOC convergence criteria as in Eq. 2.30, the low-order system uses the L2 norm of successive iteration relative change in energy-integrated fission source as well. This computes a single number for the entire geometry by accounting for every coarse mesh cell's fission source:

$$\epsilon^{(l+1)} = \sqrt{\frac{1}{N_I}\sum_I \left(\frac{\sum_g \overline{\nu\Sigma}_{f,g,I}\phi_{g,I}^{(l+1)} - \sum_g \overline{\nu\Sigma}_{f,g,I}\phi_{g,I}^{(l)}}{\sum_g \overline{\nu\Sigma}_{f,g,I}\phi_{g,I}^{(l)}} \right)^2} \tag{3.16}$$

where N_I is the total number of coarse mesh cells. The LOO implementation in OpenMOC also monitors a couple of other norms, including the L-infinity norm of the FSR relative change in the energy-integrated fission source and the L2 norm of the coarse mesh relative change in the energy-integrated fission source (see Section 4.1 for some sample results), to assure that the solution is well converged.

If $\epsilon^{(l+1)}$ does not meet the low-order convergence criteria, one repeats from step 2 and iterates on (l) until source convergence. The converged cell-averaged scalar flux is defined as $\bar{\phi}_{g,I}^{(\infty)}$ where the superscript (∞) designates that it is the converged result.

3.3.4 LOO Scalar Flux Calculation

As previously mentioned, this work proposes two different forms of the LOO methods. The two variations only differ in how they compute the mesh cell averaged scalar fluxes in the low-order and share the same routines for all other steps. Each variation is described below in two steps: the term accumulated during the low-order transport sweep, and how $\bar{\phi}_{g,I}^{(l+1)}$ is computed based on the accumulated terms.

1. The first variation is called the "LOO-psi" method because it uses the ratio of $\sum_q \bar{\hat{\psi}}_{g,I,q}$ as a multiplicative update for the mesh cell averaged scalar flux $\bar{\phi}_{g,I}$.

   (a) Accumulation during the (l)-th low-order transport sweep: the term of interest is the summation of the eight coarse tracks' averaged angular flux $\sum_q \bar{\hat{\psi}}^{(l)}_{g,I,q}$. It is accumulated in a similar fashion as Eq. 3.11:

$$\sum_q \bar{\hat{\psi}}^{(l+1)}_{g,I,q} = \sum_q \left(\frac{\hat{q}^{(l)}_{g,I,q}}{\bar{\Sigma}^{(m+1/2)}_{t,g,I}} + \frac{\hat{\Delta}^{(l)}_{g,I,q}}{\bar{\Sigma}^{(m+1/2)}_{t,g,I}\ \hat{S}_{I,q}} \right) \tag{3.17}$$

   (b) Calculation of $\bar{\phi}_{g,I}$: upon completing the low-order transport sweep, the new $(l+1)$-th iteration mesh cell averaged scalar flux $\bar{\phi}^{(l+1)}_{g,I}$ can be approximated from the new $\sum_q \bar{\hat{\psi}}^{(l+1)}_{g,I,q}$:

$$\bar{\phi}^{(l+1)}_{g,I} = \frac{\bar{\phi}^{(m+1/2)}_{g,I}}{\sum_q \bar{\hat{\psi}}^{(m+1/2)}_{g,I,q}} \sum_q \bar{\hat{\psi}}^{(l+1)}_{g,I,q} \tag{3.18}$$

   where $\frac{\bar{\phi}^{(m+1/2)}_{g,I}}{\sum_q \bar{\hat{\psi}}^{(m+1/2)}_{g,I,q}}$ is effectively a shape factor that describes $\bar{\phi}_{g,I}$'s relative magnitude to $\sum_q \bar{\hat{\psi}}_{g,I,q}$. Thus upon obtaining the updated $\sum_q \bar{\hat{\psi}}_{g,I,q}\, j^{(l+1)}$, one multiplies it by the shape factor and gets the updated mesh cell averaged scalar flux $\bar{\phi}^{(l+1)}_{g,I}$.

   Another way to interpret Eq. 3.18 is to consider the term:

$$\frac{\sum_q \bar{\hat{\psi}}^{(l+1)}_{g,I,q}}{\sum_q \bar{\hat{\psi}}^{(m+1/2)}_{g,I,q}} \tag{3.19}$$

   as a multiplicative update to the coarse mesh cell averaged scalar flux.

2. The second variation is called the "LOO-balance" method because it uses a neutron balance equation in each coarse mesh cell to produce the updated mesh cell averaged scalar flux.

   (a) Accumulation during the (l)-th low-order transport sweep: the term of interest here is the surface-integrated net leakage term for each cell, $\sum_{s} d_s J_{g,I,s}$ where d_s is the length of surface s. It is desired to accumulate the surface-integrated currents instead of the surface-averaged ones because in the most general case, the four surfaces of the mesh cell may not have the same side length.

Recall that during the high-order MOC transport sweep, the quadrant current is accumulated using Eq. 3.2. A similar expression can be written in low-order for the net current:

$$J_{g,I,s}^{(l+1)} = \frac{\sum_{q_s,\pm} \hat{\omega}\ \hat{\delta} \sin\theta_p \left(\pm\ \hat{\psi}_{g,I,q_s}^{\pm,(l+1)}\right)}{\sum_{q_s,\pm} \hat{\omega}\ \hat{\delta}} \tag{3.20}$$

where the current on a surface s is computed by accumulating a subset of tracks q_s that intersect the surface s with its incoming (-) or outgoing (+) angular flux. The $\hat{\omega}$, $\hat{\delta}$ notations are used since all representative low-order characteristic tracks are laid down with the same azimuthal angular spacing and spatial track spacing. There is a ± sign in front of the $\hat{\psi}_{g,I,q_s}^{\pm}$ term to take into account that the direction of the positive and negative partial currents. Recall that partial currents, and by extension quadrant currents, are scalar quantities with an assumed positive direction. The net current is typically computed by subtracting the negative partial current from the positive partial current. Specifically to the LOO method, the outgoing angular flux $\hat{\psi}_{g,I,q_s}^{+}$ always leaves the mesh cell thus contributing positively to the net current on that surface, and the incoming angular flux $\hat{\psi}_{g,I,q_s}^{-}$ always leaves the mesh cell contributing negatively to a different mesh surface. Thus to consider both the incoming and outgoing fluxes' contributions the ± sign in Eq. 3.20 is necessary.

To compute the surface-integrated net current on surface s in mesh cell I, the denominator of Eq. 3.20 is dropped:

$$d_s J_{g,I,s}^{(l+1)} = \sum_{q_s,\pm} \hat{\omega}\ \hat{\delta} \sin\theta_p \left(\pm\ \hat{\psi}_{g,I,q_s}^{\pm,(l+1)}\right) \tag{3.21}$$

To obtain the surface-integrated leakage in mesh cell I, the contributions of the four mesh surfaces are summed. With the simplified coarse mesh characteristic tracks, every quadrant flux will be accounted for. That is, the contribution of each and every quadrant track's incoming and outgoing fluxes should be accounted for in the cell net leakage expression:

$$\sum_s d_s J^{(l+1)}_{g,I,s} = \sum_q \hat{\omega}\ \hat{\delta}\,\sin\theta_p \left(\pm\ \hat{\psi}^{\pm,(l+1)}_{g,I,q}\right) = \hat{\omega}\ \hat{\delta}\,\sin\theta_p \sum_q \sum_\pm \left(\pm\ \hat{\psi}^{\pm,(l+1)}_{g,I,q}\right) \tag{3.22}$$

Notice that

$$\sum_\pm \left(\pm\ \hat{\psi}^{\pm,(l+1)}_{g,I,q}\right) = \hat{\psi}^{+,(l+1)}_{g,I,q} - \hat{\psi}^{-,(l+1)}_{g,I,q} = -\ \hat{\Delta}^{(l+1)}_{g,I,q} \tag{3.23}$$

That is, we can replace the inner summation in Eq. 3.22 with the simple notation for the change in flux along a low-order characteristic track. Then,

$$\sum_s d_s J^{(l+1)}_{g,I,s} = \hat{\omega}\ \hat{\delta}\,\sin\theta_p \sum_q \hat{\Delta}^{(l+1)}_{g,I,q} \tag{3.24}$$

(b) Calculation of $\bar{\phi}_{g,I}$: upon obtaining the surface-integrated net leakage term $\sum_s d_s J^{(l+1)}_{g,I,s}$ using Eq. 3.24, a neutron balance equation for mesh cell I, energy group g can be constructed blow:

$$\sum_s d_s J^{(l+1)}_{g,I,s} + \Sigma^{(m+1/2)}_{t,g,I}\ \bar{\phi}^{(l+1)}_{g,I}\, V_I = 4\pi\ \bar{q}^{(l+1)}_{g,I}\, V_I \tag{3.25}$$

Thus the mesh cell averaged scalar flux computed from this iteration is,

$$\bar{\phi}^{(l+1)}_{g,I} = \frac{4\pi\ \bar{q}^{(l+1)}_{g,I}\, V_I - \sum_s d_s J^{(l+1)}_{g,I,s}}{\Sigma^{(m+1/2)}_{t,g,I} V_I} \tag{3.26}$$

The two implementations both produce results that are consistent with a converged high-order solution. That is, when the high-order solution is converged, both low-order alternatives observe no changes[2] in low-order quantities after one low-order iteration: $\hat{q}^{(l=1)}_{g,I,q} = \hat{q}^{(m+1/2)}_{g,I,q}$, $\hat{\psi}^{(l=1)}_{g,I,q} = \hat{\psi}^{(m+1/2)}_{g,I,q}$, $\bar{\phi}^{(l=1)}_{g,I} = \bar{\phi}^{(l=1)}_{g,I}$. Thus the low-order iterative solver will exit after one iteration, producing no updates.

See Section 4 for computational results of LOO-balance and LOO-psi accelerations compared with CMFD.

[2] The "no changes" here means up to the precision of the high-order converged solution. For instance, if the high-order is converged to 1e-8 and passes the terms into the low-order, the low-order quantities' relative changes are less than 1e-8.

3.3.5 LOO Prolongation

Recall that at the end of Section 3.3.3, a converged solution $\bar{\phi}_{g,I}^{(\infty)}$ is generated for each mesh cell I and every energy group g. In this section, a prolongation step is performed to pass information from the coarse grid to the fine grid.

1. The FSR scalar flux update is the same as in the CMFD method. That is, the FSR scalar fluxes computed from the high-order transport sweep is multiplied by the ratio of the cell-averaged scalar flux before and after the LOO steps:

$$\phi_{g,I}^{(m+1)} = \phi_{g,I}^{(m+1/2)} \frac{\bar{\phi}_{g,I}^{(l=\infty)}}{\bar{\phi}_{g,I}^{(m+1/2)}} \tag{3.27}$$

2. Each fine grid boundary angular flux is updated by the quadrant flux of the quadrature space-angle that it belongs to. That is,

$$\psi_{B,g,m}^{(m+1)} = \psi_{B,g,m}^{(m+1/2)} \times \frac{\hat{\psi}_{g,I,s}^{\pm,u,(l=\infty)}}{\hat{\psi}_{g,I,s}^{\pm,u,(m+1/2)}} \tag{3.28}$$

   where s is the boundary surface that the boundary flux $\psi_{B,g,m}$ intersects, and the $\pm, u$ describes the quadrature space-angle that $\psi_{B,g,m}$ belongs to.

Upon performing the prolongation, the $(m+1)$-th results are computed for scalar fluxes and angular fluxes, and the L2 norm as in Eq. 2.30 is computed for convergence monitoring.

3.4 Corner Track Treatment

When coarse mesh cells are imposed on the fine-grid characteristic tracks, some segments go through the corners of the coarse mesh cells. Since the segments' angular fluxes have to be accumulated on the coarse mesh cell surfaces, one cannot leave out the ones going through coarse mesh cell corners.

Though the physical locations of the corner tracks are not actually moved, the angular fluxes are tallied on the neighboring coarse mesh cell surfaces in a manner that resembles moving the segments slightly to avoid going through the corners. Two important qualities need to be preserved in treating these corner tracks:

1. The forward and the backward segments' angular fluxes have to be accumulated on the same mesh cell surfaces such that the forward and backward segments are not be separated spatially.

2. On the reflective exterior boundary surfaces, the complementary segments (e.g., the pairs that form cyclic tracking) have to be tallied on the same coarse mesh cell surfaces. Otherwise, the balance in each coarse mesh cell will not be preserved.

For tracks crossing an interior corner, the treatment is reasonably simple. As illustrated in the left image of Fig. 3-4, a forward corner segment designated by a solid blue line leaves the bottom right coarse mesh cell. Its value is split into two halves. The top half, designated by the dashed red line, is added to the top surface of the current coarse mesh cell (i.e., the bottom right cell) and to the left surface of the coarse mesh cell above the current one (i.e., the top right cell). The bottom half, designated by the dashed green line, is added to the left surface of the current coarse mesh cell. It is also tallied on the top surface of the coarse mesh cell left to the current one (i.e., the bottom left cell). Similar treatment is done for the backward direction as illustrated in the right image of Fig. 3-4.

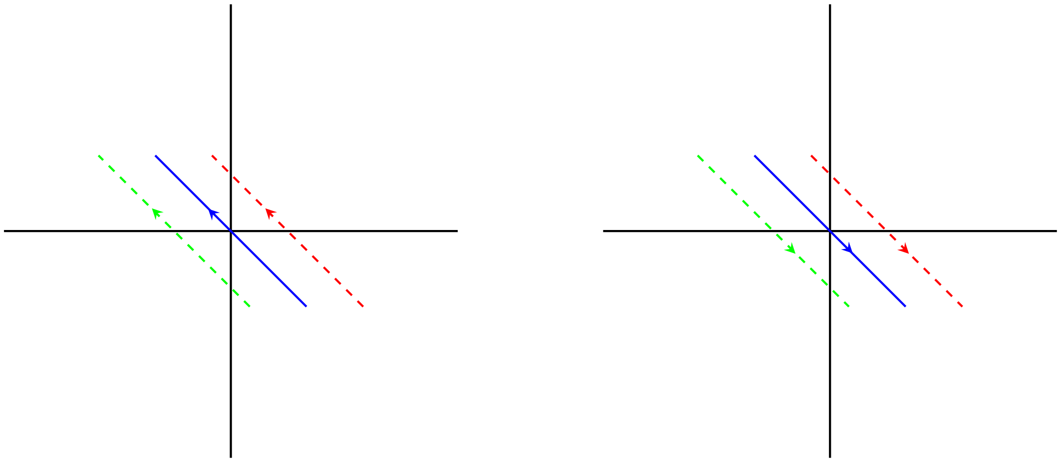

Figure 3-4: Illustration of Corner Splitting Method For Interior Cells

Fig. 3-5 illustrates the treatment for tracks crossing an exterior corner. In the left image, the solid blue line designates a corner segment ending on a reflective boundary. Again the segment is split into halves, and in this case, the treatment of the top half designated by dashed red line remains the same as in the interior corner case. The bottom half designated by dashed green line is tallied on the left surface of the current coarse mesh cell. It is also tallied on the top surface of the current coarse mesh cell, but in a different quadrature, to resemble that the dashed green line is reflected from the exterior boundary. A similar operation is performed for the backward segment illustrated in the right image of Fig. 3-5.

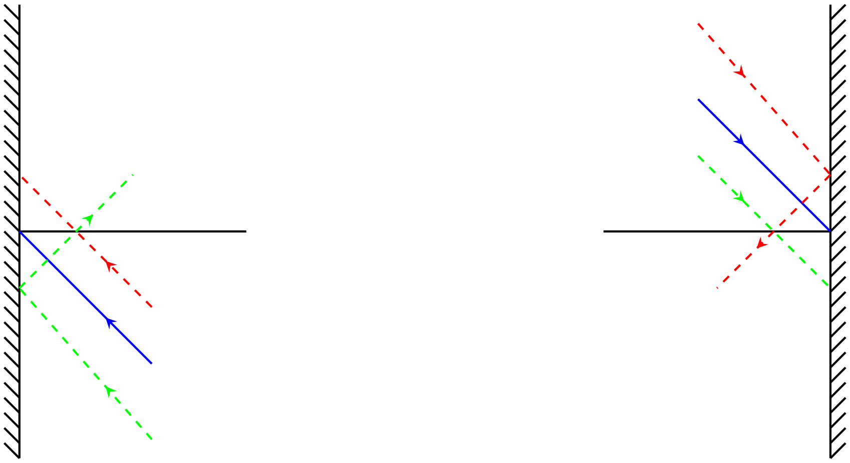

Figure 3-5: Illustration of Corner Splitting Method For Exterior Cells

To put this corner treatment to test, a 2x2 pin cell problem is created with a pin cell length of 1.26cm and is filled with a homogeneous material. OpenMOC is ran with 0.29cm track spacing and eight azimuthal angles to create the track laydown illustrated in Fig. 3-6. Notice in the bottom left pin cell designated by a different color, for the left and bottom coarse mesh surfaces, there are 12 tracks crossing the surfaces, and one track crossing a corner. Whereas for right and the top coarse mesh surfaces, there are 10 tracks crossing the surfaces, and three crossing the corners. With this track laydown, if the corner splitting is performed incorrectly, the tallied currents will differ.

The partial currents tallied in OpenMOC are plotted in Fig. 3-7 where the value next to each surface is the partial current summed over seven energy groups on that coarse mesh cell surface. The tallied partial currents are identical on every coarse mesh surface which is expected for this homogeneous material test problem.

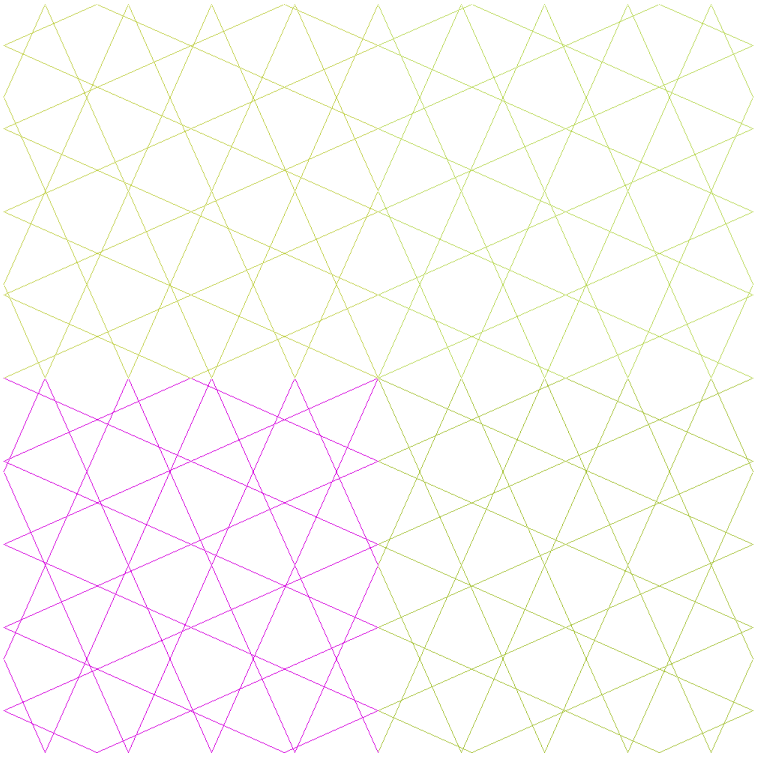

Figure 3-6: Plot of Track Laydown For A Homogeneous Problem

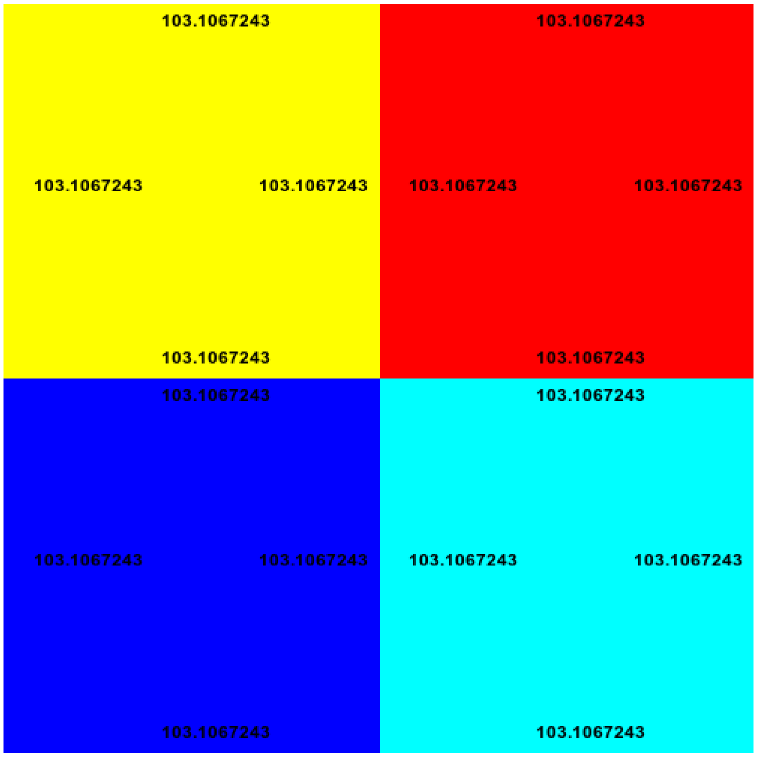

Figure 3-7: Plot of Partial Currents For A Homogeneous Problem

3.5 The Choice For the First Acceleration Step

In this section the accuracy of the early iterations is addressed. While it is entirely valid to perform LOO accelerations after every high-order transport sweep, during the early high-order iterations, especially the first one, the leakage passed to the low-order system is far from the true solution. To facilitate LOO, the following measures are taken:

1. The first high-order transport solution is accelerated by a low-order finite difference diffusion solve instead of LOO. The rational behind this choice is that the very first high-order MOC sweep produces a very poor approximation of the flux shape and the leakage between cells. LOO is sensitive to the leakage terms, and if the neutron transport effect generated by the high-order MOC is not accurate, there is no reason to perform a LOO acceleration at the first iteration. A finite difference diffusion solve is better at getting a reasonable flux shape.

2. In the prolongation step of the first low-order finite difference diffusion acceleration, the FSR scalar fluxes are updated normally using a multiplicative factor similar to CMFD and LOO:

$$\phi_{g,i}^{(m=1)} = \phi_{g,i}^{(m=1/2)} \times \frac{\bar{\phi}_{g,I}^{(l=\infty)}}{\bar{\phi}_{g,I}^{(l=0)}} \tag{3.29}$$

   But for the boundary angular flux, the previous solution from the high-order MOC sweep $\psi_{B,g,m}^{(m=1/2)}$ is not used, and the new boundary flux is computed solely based on the finite difference diffusion solve's scalar flux solution:

$$\psi_{B,g,m}^{(m=1)} = \frac{\phi_{g,I}^{(l=\infty)}}{4\pi} \tag{3.30}$$

   That is, every reflective boundary flux is set to one over four pi that of the scalar flux of the mesh cell under the assumption of isotropic angular flux. The point is, if almost nothing is known about the leakage yet, OpenMOC is better off assuming isotropic angular flux.

To summarize, the angular fluxes computed from the first transport sweep are only used to construct the low-order finite difference diffusion solve (more specifically the scalar flux spectrum and spatial distribution is used to generate energy-dependent cross-section). The angular fluxes computed from the first transport sweep is replaced by the ones computed from the diffusion solve scalar flux solution by assuming isotropic flux. Then OpenMOC proceeds to perform regular transport sweeps followed by LOO acceleration as described in Section 3.3.

3.6 High-Order Boundary Sweep

To further reduce the number of high-order MOC transport sweep, OpenMOC can perform, upon users' request, a high-order boundary sweep before running a high-order transport solve. The high-order boundary sweep is one that update the high-order boundary angular fluxes but not the FSR scalar fluxes or the FSR sources.

The purpose of this feature is purely to improve the boundary fluxes before running a real high-order transport sweep followed by a low-order acceleration. For an example, a 8x8 pin cell problem is created with UO2 fuel pins and MOX fuel pins arranged in a checker board pattern (see Section 4.2 for more details). This test case is ran with and without a high-order boundary sweep using LOO-psi accelerated OpenMOC. The result is plotted in Fig. 3-8, and as expected a high-order boundary sweep reduces the number of full high-order transport sweep.

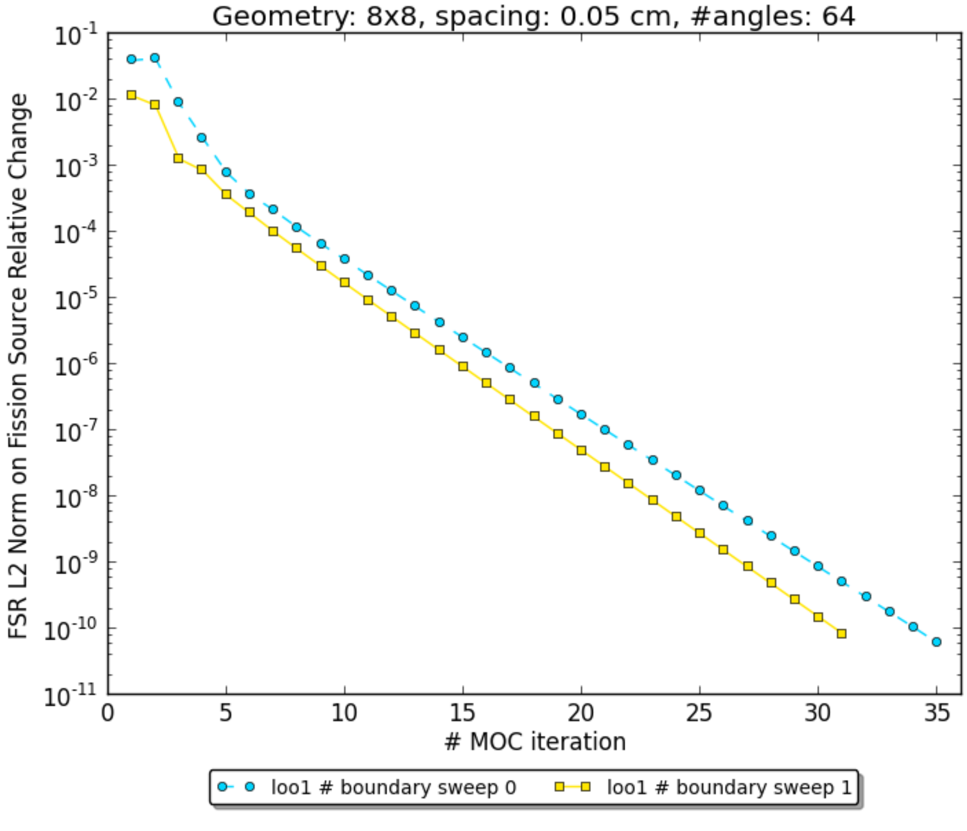

Figure 3-8: Effects of High-order Boundary Sweep

The drawback of this feature is that it is still a high-order sweep and is expensive in the current implementation. But there is room for improvement: during a long characteristic sweep (that is, one that starts from one exterior geometry boundary and ends on another), if the FSR sources' contribution to the outgoing angular fluxes are accumulated, then one can isolate the contributions from the incoming angular fluxes and the contribution from the FSR sources. Thus it is possible to perform a long characteristic sweep without looping through the segments along the long characteristic one at a time. This method would significantly reduces the computational cost of the high-order boundary sweep, making it an attractive feature for an MOC solver. The implementation of this long characteristic sweep is proposed for future work.

Chapter 4

Computational Results of CMFD and LOO

In this section numerical results are presented for three types of cases. An overview of the cases and the OpenMOC configurations will be given in Section 4.1, followed by more detailed descriptions and results for each case.

4.1 Overview of Test Cases and Configurations

Three types of test problems will be presented in this chapter:

1. "Pin cell problems": simple pin cells arranged in a checker board pattern with reflective boundary conditions. Two variations will be presented, one with four pin cells, and the other with sixty-four pin cells.

2. "UO2 lattice problems": 17x17 assemblies with UO2 fuel pins, guide tubes and water cells. Two variations will be presented, one with a regular lattice configuration, and the other with boundary water cells.

3. "C5G7 benchmark problems": two variations will be presented, one is the original 2D C5G7 benchmark problem, and the other is the fuel regions of the C5G7 benchmark problem.

The following configurations are used in all the test cases:

- The high-order MOC track spacing (the normal distance between parallel tracks of the same azimuthal angle): 0.05cm.

- The high-order MOC number of azimuthal angles: 64.
- The high-order MOC number of polar angles: 3.
- The high-order MOC transport iteration convergence criteria: 1e-10 on L2 norm of FSR successive iteration relative change in energy-integrated fission sources, as in Eq. 2.30.
- The low-order MOC transport iteration convergence criteria: 1e-10 on L2 norm of mesh cell successive iteration relative change in energy-integrated fission sources, as in Eq. 3.16.
- The high-order MOC boundary sweep: one high-order transport sweep without updating the source is performed before running the regular high-order transport sweep. This was described in details in Section 3.6.
- The first transport sweep is accelerated by a low-order finite-difference coarse mesh diffusion solve. See Section 3.5.
- Damping factor: the CMFD method uses a fixed damping factor of 0.7 on the non-linear coupling coefficients, and the LOO methods use no under-relaxation.

Before proceeding to the test cases and results, it is worthy to briefly discuss the various measurements of the convergence rate. Using the 2D C5G7 benchmark problem as an example[1],

- Fig. 4-1 plots the L2 norm of the coarse mesh cell relative change in the energy-integrated fission power calculated by Eq. 3.16.
- Fig. 4-2 plots the L-infinity norm of the FSR relative change in the energy-integrated fission power.
- Fig. 4-3 plots the L2 norm of the FSR relative change in the energy-integrated fission power calculated by Eq. 2.30.

The above norms, together with the relative change in eigenvalues, are monitored during an OpenMOC simulation to assure a well converged solution. For the test cases shown, the three norms demonstrate very similar patterns. As a result, only the L2 norm of FSR relative fission power change will be presented in this work to illustrate the convergence behavior of OpenMOC accelerated by CMFD and by the two variations of LOO.

[1] These plots are generated using coarse FSRs. For instance, the fuel pin is not subdivided into rings and sections as discussed in Section 4.4. Fine FSRs are required to get an accurate solution. For the purpose of illustrating the convergence behavior, coarse FSRs are sufficient for saving some computational time.

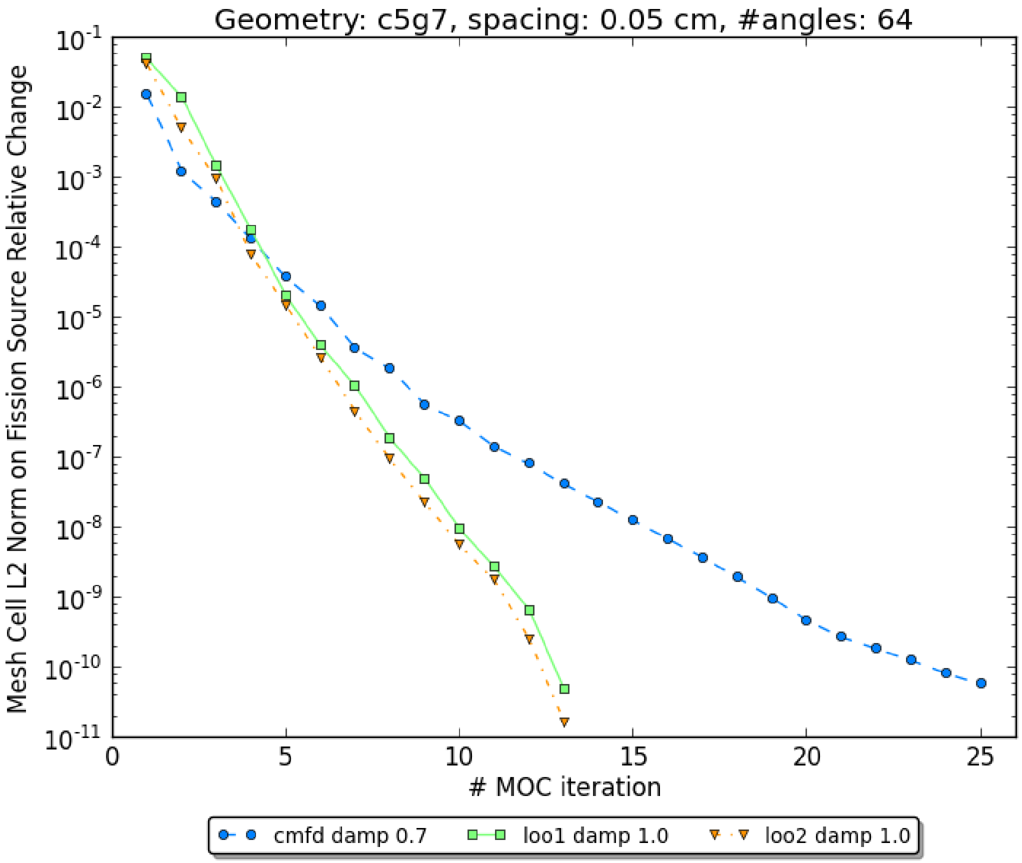

Figure 4-1: Plot of C5G7 Benchmark Problem Coarse Mesh Cell L2 Norm

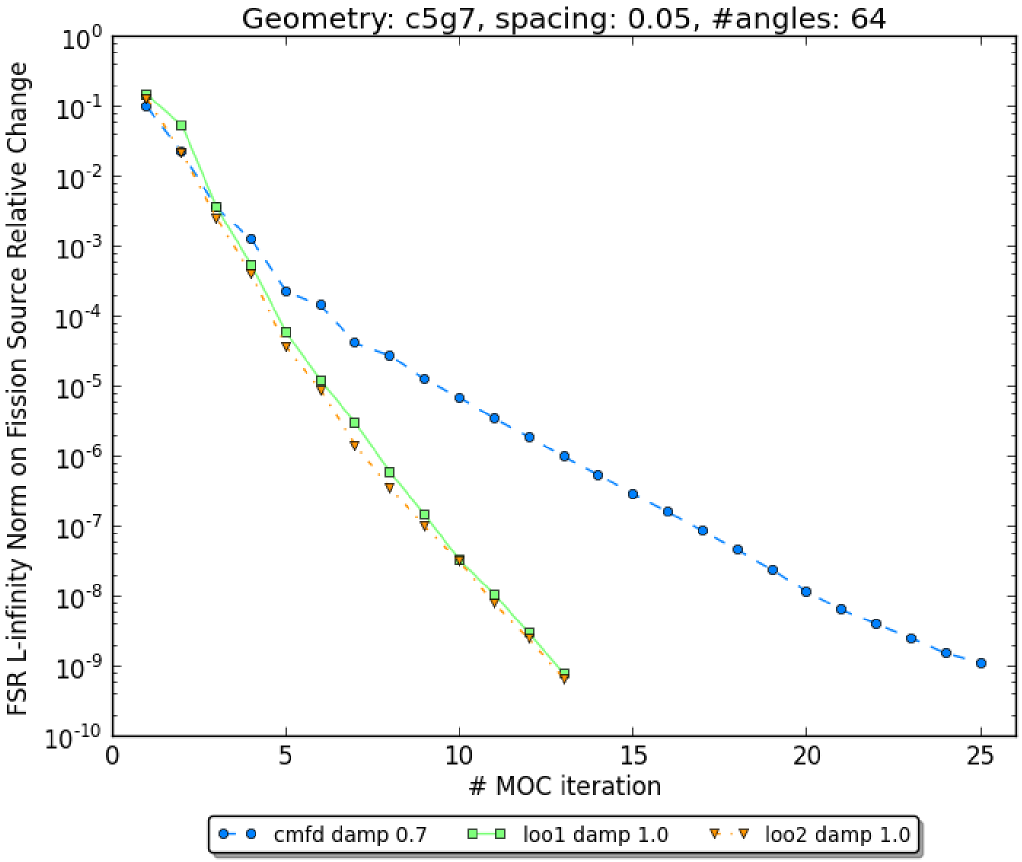

Figure 4-2: Plot of C5G7 Benchmark Problem FSR L-infinity Norm

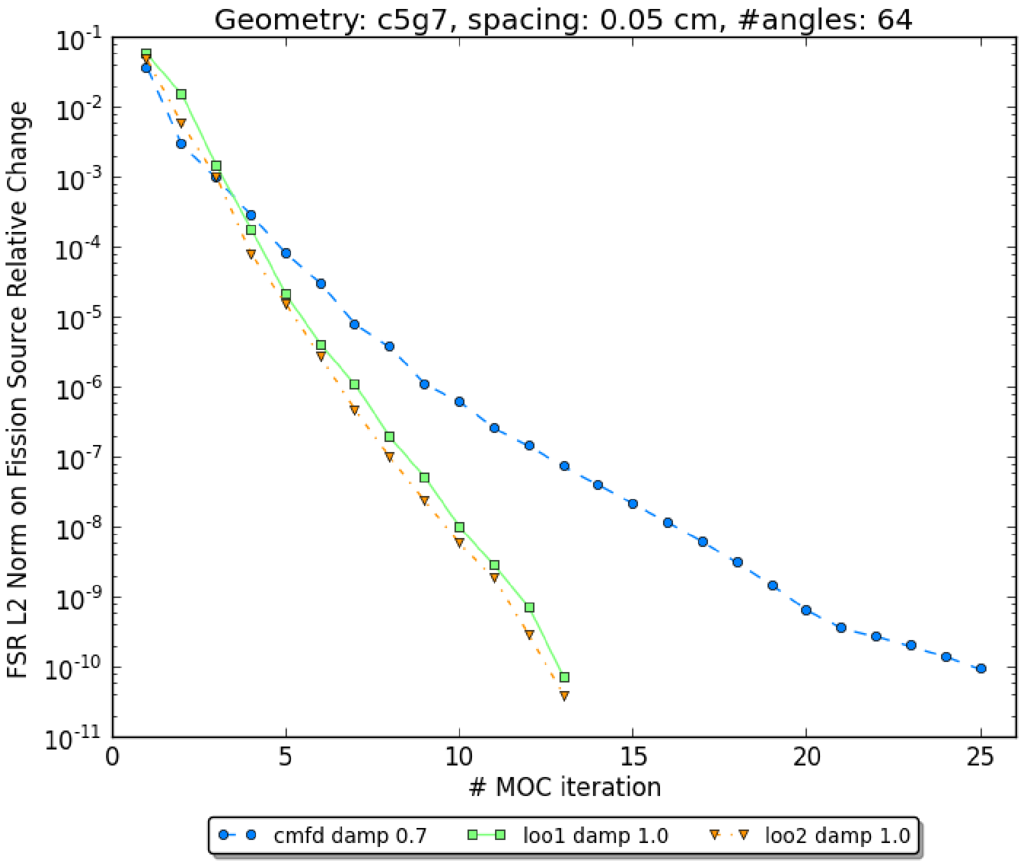

Figure 4-3: Plot of C5G7 Benchmark Problem FSR L2 Norm

4.2 Pin Cell Problems

The pin cell problems have pin pitches of 1.26cm and fuel radii of 0.54cm. The top left and bottom right fuel pins are made of UO2 fuel, and the top right and bottom left fuel pins are made of 4.3% enriched MOX fuel. The pin cell problems use the seven energy group cross-sections of the C5G7 benchmark problem [Smith et al., 2003]. The boundary conditions on the exterior surfaces are reflective.

The 2x2 pin cell case is shown in Fig. 4-4, where the left plot is an illustration of the material distribution. The right plot is an illustration of the FSR distribution, where each fuel pin is subdivided into four FSRs, and each water region is subdivided into eight FSRs for simulation. The 8x8 pin cell case is shown in Fig. 4-5.

The converged eigenvalues and number of transport sweeps taken to solve the 2x2 pin cell problem are shown in Table 4.1. Notice the unaccelerated k_{eff} is slightly different from the accelerated results. If we further converge the unaccelerated OpenMOC to 1e-11 convergence criteria, the k_{eff} values agree. The computational results for the 8x8 pin cell are in Table 4.2, and one should note that the slight difference in the 2x2 case's eigenvalues and the 8x8 case's eigenvalues arises from a slightly different track laydown.

Method	k_{eff}	# Transport Sweep
No acceleration	1.2165594396	117
CMFD	1.2165594495	32
LOO-psi	1.2165594495	31
LOO-balance	1.2165594495	31

Table 4.1: 2x2 Pin Cell Problem Computational Results

Method	k_{eff}	# Transport Sweep
No acceleration	1.2162482381	85
CMFD	1.2162482413	32
LOO-psi	1.2162482413	31
LOO-balance	1.2162482413	31

Table 4.2: 8x8 Pin Cell Problem Computational Results

The convergence behaviors of these two pin cell problems are plotted in Fig. 4-6 and Fig. 4-7. It is observed that LOO-accelerated OpenMOC converges slightly faster than CMFD-accelerated OpenMOC.

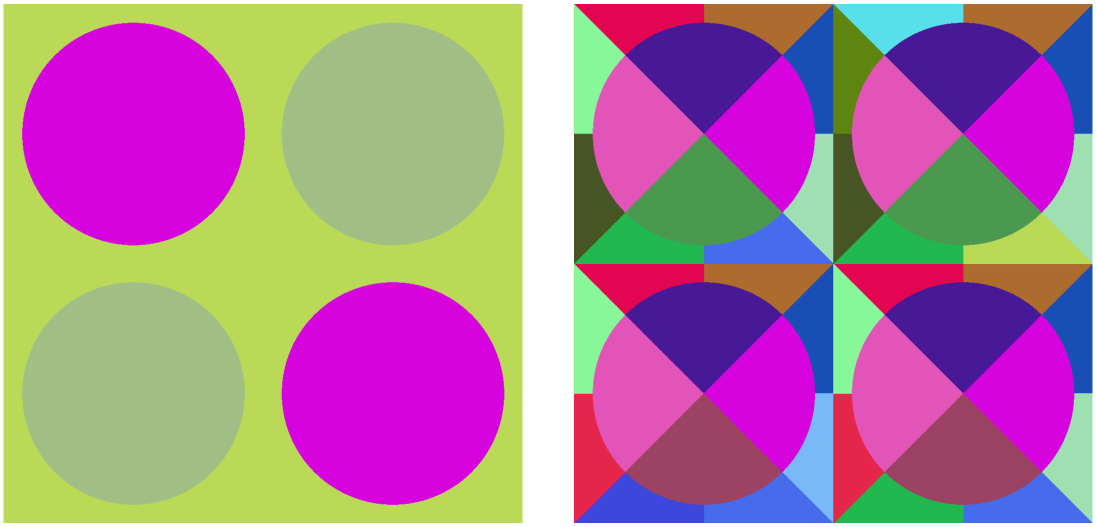

Figure 4-4: Illustration of 2x2 Pin Cell Problem

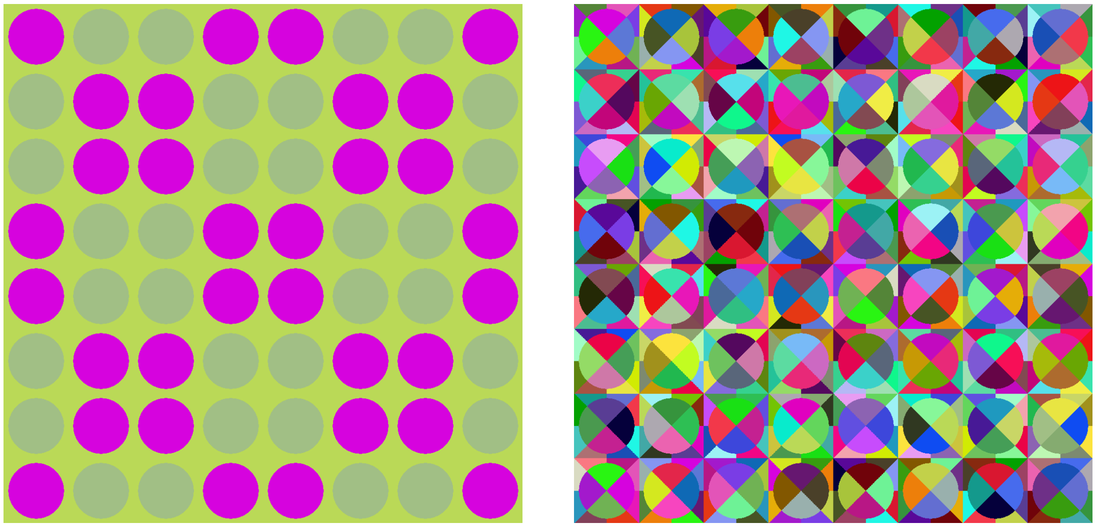

Figure 4-5: Illustration of 8x8 Pin Cell Problem

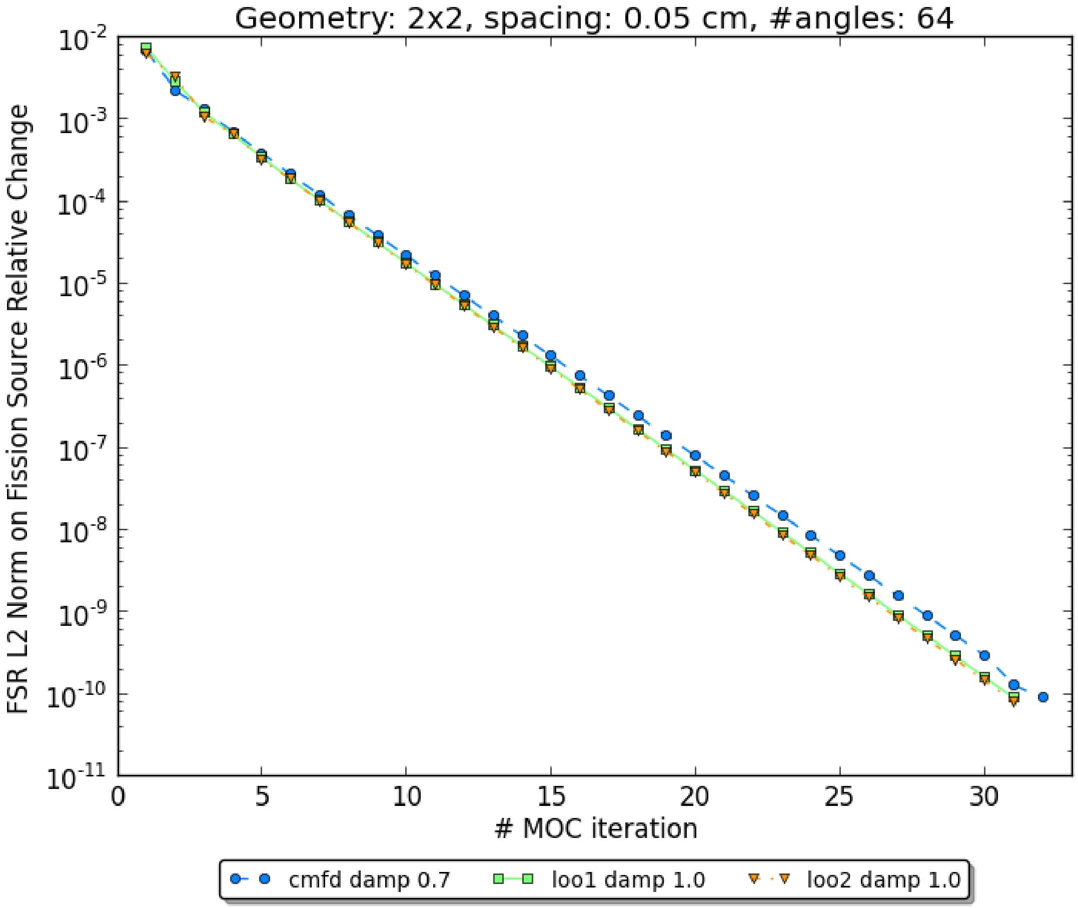

Figure 4-6: Plot of 2x2 Pin Cell Problem Convergence Behavior

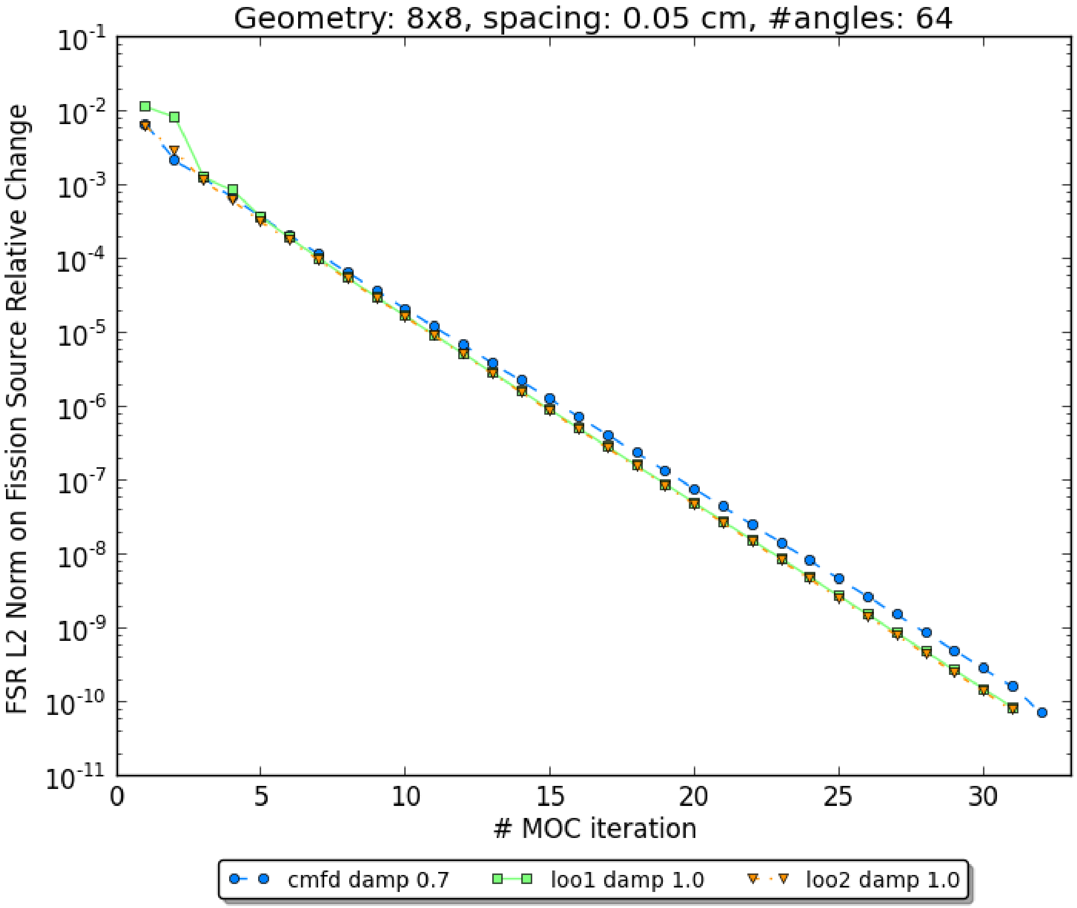

Figure 4-7: Plot of 8x8 Pin Cell Problem Convergence Behavior

4.3 UO2 Lattice Problems

The UO2 lattice problem consists of a single assembly with 17 by 17 pin cells. Each pin cell has a 1.26cm pitch and a pin radius of 0.54cm. It is the UO2 assembly from the C5G7 benchmark problem with UO2 fuel pins, guide tube pins, and water cells [Smith et al., 2003]. This geometry is illustrated in the left plot of Fig. 4-8.

A variation of the UO2 lattice problem is the UO2 leakage problem, where four columns and four rows of pin cells are replaced by water cells, which is illustrated in the right plot of Fig. 4-8. This variation is produced to create transport and anisotropic leakage effect in the problem.

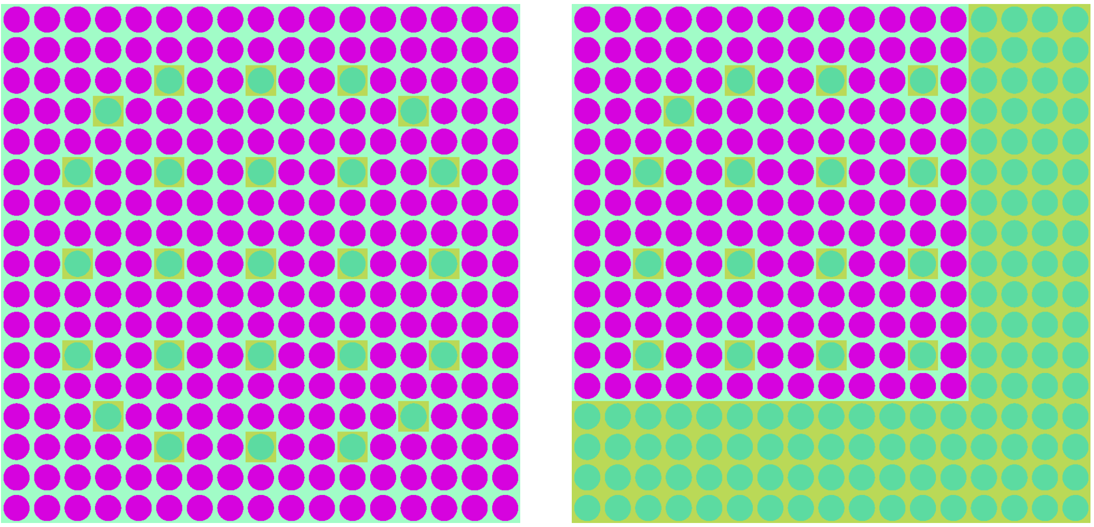

Figure 4-8: Illustration of UO2 Lattice Problems

The computational results for the UO2 lattice problem and the UO2 leakage problem are tabulated in Table 4.3 and Table 4.4. For the UO2 leakage lattice problem, the discrepancy between the unaccelerated and the accelerated eigenvalues can be resolved by a tighter convergence of the unaccelerated result.

Method	k_{eff}	# Transport Sweep
No acceleration	1.3350595975	254
CMFD	1.3350595974	17
LOO-psi	1.3350595974	13
LOO-balance	1.3350595975	13

Table 4.3: UO2 Lattice Problem Computational Results

Method	k_{eff}	# Transport Sweep
No acceleration	1.0958206670	637
CMFD	1.0958206653	19
LOO-psi	1.0958206653	14
LOO-balance	1.0958206653	13

Table 4.4: UO2 Leakage Lattice Problem Computational Results

Observations from these two cases include that:

- LOO-psi and LOO-balance methods show a similar convergence rate which is steeper than the CMFD's convergence rate.
- LOO methods show more advantage in the UO2 leakage lattice problem than in the UO2 lattice problem. This is expected because the UO2 leakage lattice problem has more transport effect than the UO2 lattice problem.

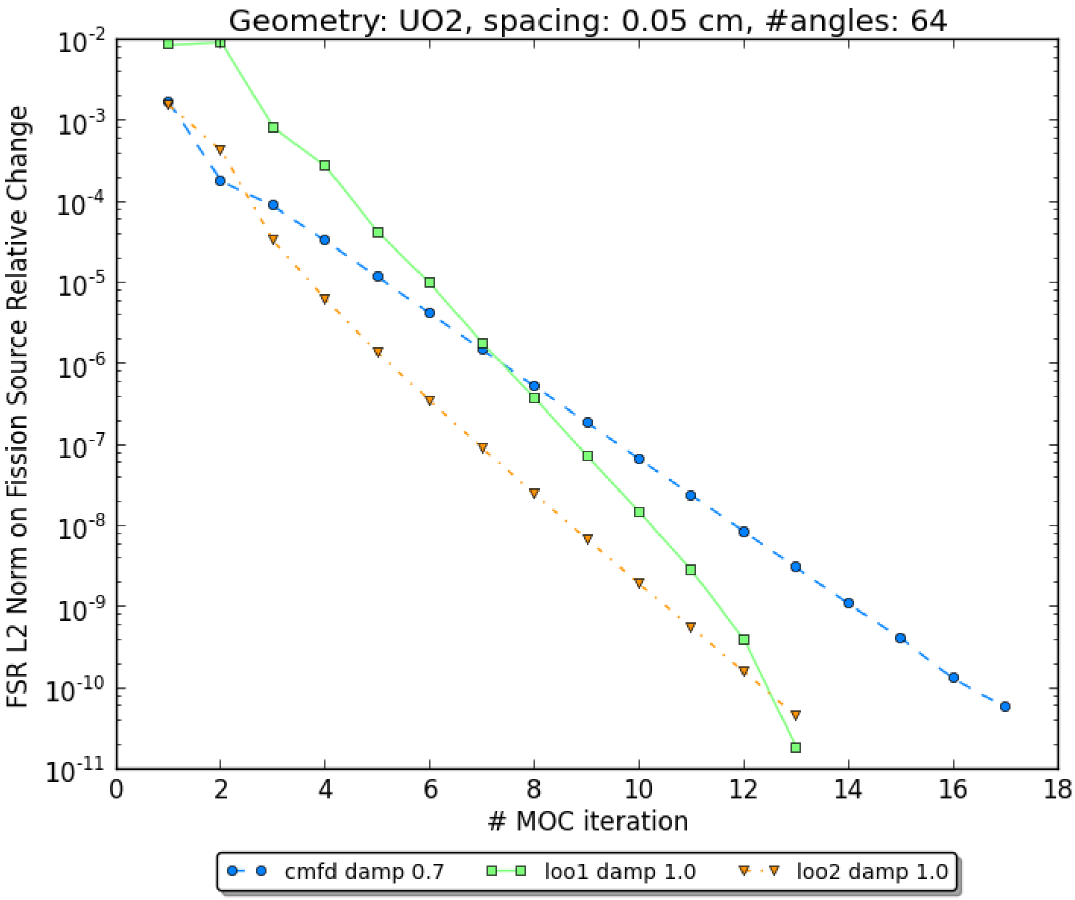

Figure 4-9: Plot of UO2 Lattice Problem Convergence Behavior

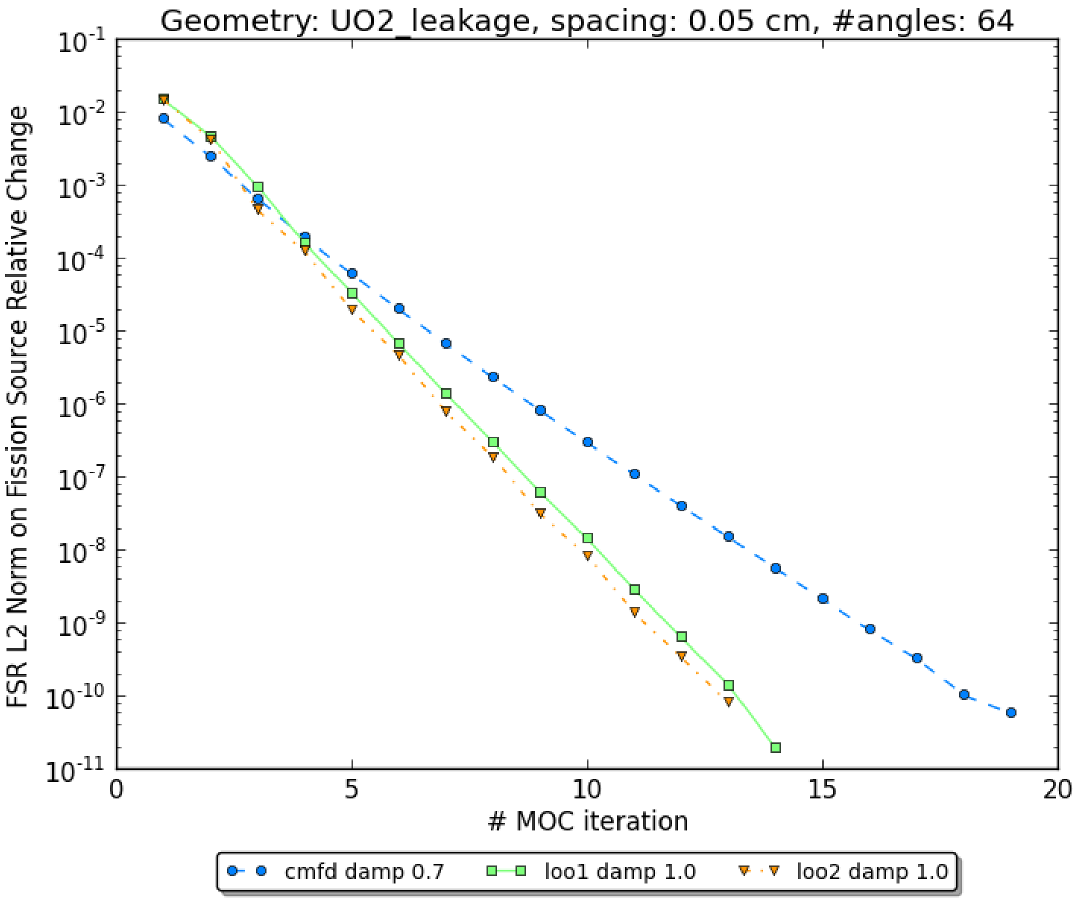

Figure 4-10: Plot of UO2 Leakage Lattice Problem Convergence Behavior

4.4 C5G7 Benchmark Problems

This section is based on the 2D C5G7 benchmark problem. OECD's Nuclear Energy Agency developed this benchmark problem in 2001 for comparing and validating deterministic transport methods without spatial homogenization. The benchmark provides seven group cross-sections for a 2D and a 3D problem. The 2D problem has four assemblies surrounded by water. The bottom and right boundary of the geometry is reflective, and the other two are vacuum. Each fuel assembly consists of 17 by 17 pin cells, where a pin cell has a pitch of 1.26cm and a fuel pin radius of 0.54cm. The assemblies on the top left and bottom right positions are UO2 fuel, and the remaining two assemblies contains MOX fuel. Twenty codes' solutions were available for the 2-D benchmark problem and were compared to the reference Monte Carlo solution [Smith et al., 2003].

The material distribution of the C5G7 benchmark problem is illustrated in Fig. 2-3. A variation of the problem where only the four assemblies are modeled with reflective boundary conditions is given in Fig. 2-2.

4.4.1 C5G7 Fuel Region Problem

Using 0.05cm tracking spacing, 64 azimuthal angles and 3 polar angles, OpenMOC generates 1156 mesh cells, 69912 tracks and 2786368 segments for the C5G7 fuel region problem and its convergence behavior is the top plot in Fig. 4-11.

Method	k_{eff}	# Transport Sweep
CMFD	1.2620476364	18
LOO-psi	1.2620475276	14
LOO-balance	1.2620476594	14

Table 4.5: C5G7 Fuel Region Benchmark Problem Computational Results, Coarse FSRs

If each fuel pin is subdivided into four FSRs, and the water region is subdivided into eight FSRs, then OpenMOC generates 4632160 segments for the C5G7 fuel region problem, and its convergence behavior is the bottom plot in Fig. 4-11. Its results are listed below in Table 4.6. Unfortunately it seems that when more FSRs are present in each mesh, the advantages of LOO over CMFD are reduced.

Method	k_{eff}	# Transport Sweep
CMFD	1.2622868181	33
LOO-psi	1.2622866999	32
LOO-balance	1.2622868428	32

Table 4.6: C5G7 Fuel Region Benchmark Problem Computational Results, Dense FSRs

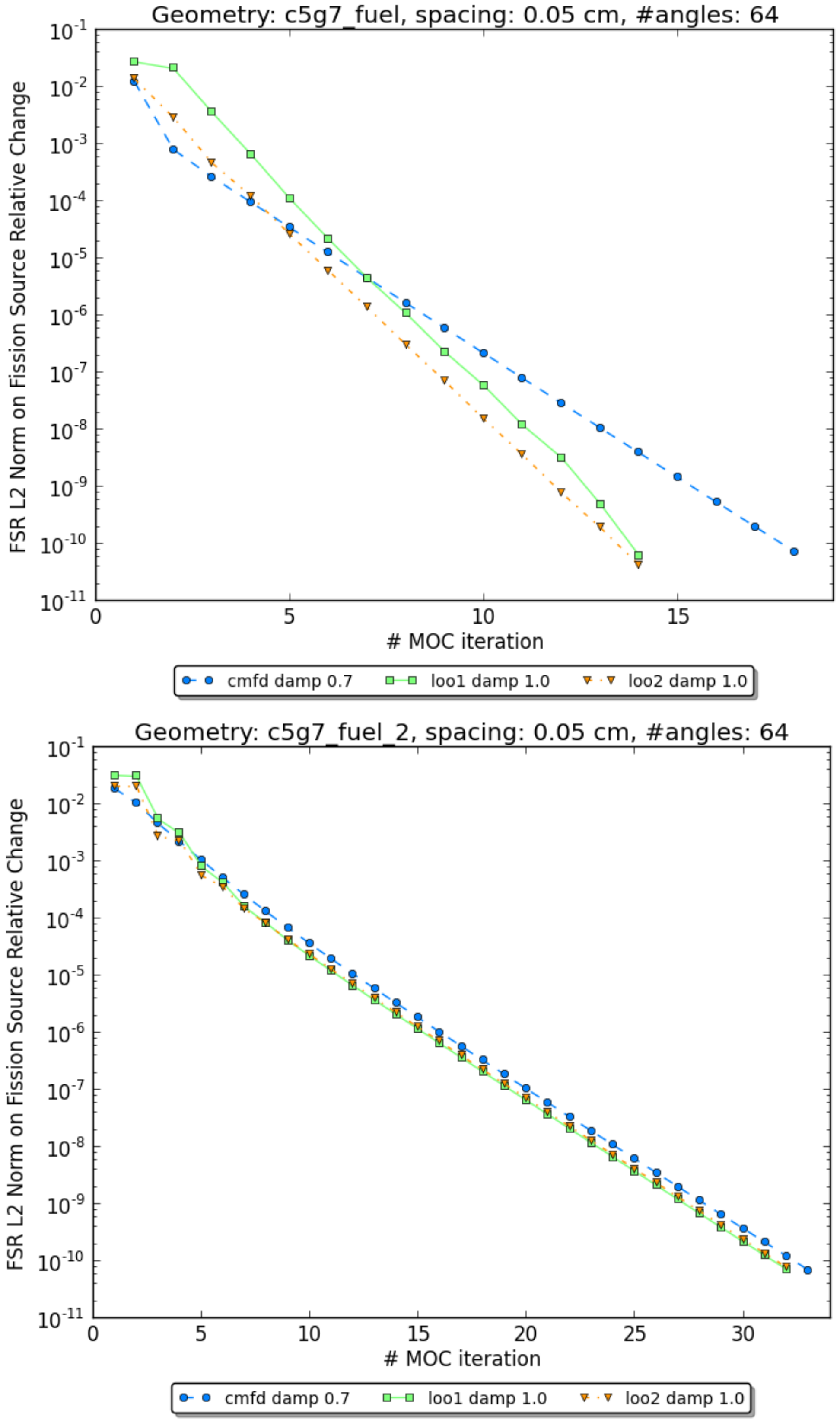

Figure 4-11: Plot of C5G7 Fuel Region Problem Convergence Behavior

4.4.2 C5G7 Benchmark Problem

For the C5G7 benchmark problem, OpenMOC generates 2601 mesh cells, 104840 tracks and 4272264 segments. The computational results are shown in Table 4.7 and the convergence rate is shown in the top plot of Fig. 4-12.

Method	k_{eff}	# Transport Sweep
CMFD	1.2620476364	18
LOO-psi	1.2620475276	14
LOO-balance	1.2620476594	14

Table 4.7: C5G7 Benchmark Problem Computational Results

As we further divide the FSRs, again LOO's advantages over CMFD are decreased as can been seen in the bottom plot of Fig. 4-12 and in Table 4.8.

Method	k_{eff}	# Transport Sweep
CMFD	1.1869180225	28
LOO-psi	1.1869014320	25
LOO-balance	1.1869351083	26

Table 4.8: C5G7 Benchmark Problem with Fine FSRs Computational Results

Similar to the comparison between the UO2 lattice problem and the UO2 leakage problem, these two C5G7 benchmark problems demonstrate that the two LOO variations exhibit very similar convergence behavior. In general LOO converges faster than CMFD, especially for problems with large leakage (e.g., UO2 leakage problem compared to the UO2 lattice problem, C5G7 problem compared to the C5G7 problem with fuel region only). When more FSRs are added to each mesh cell, LOO's advantage in convergence rate decreases.

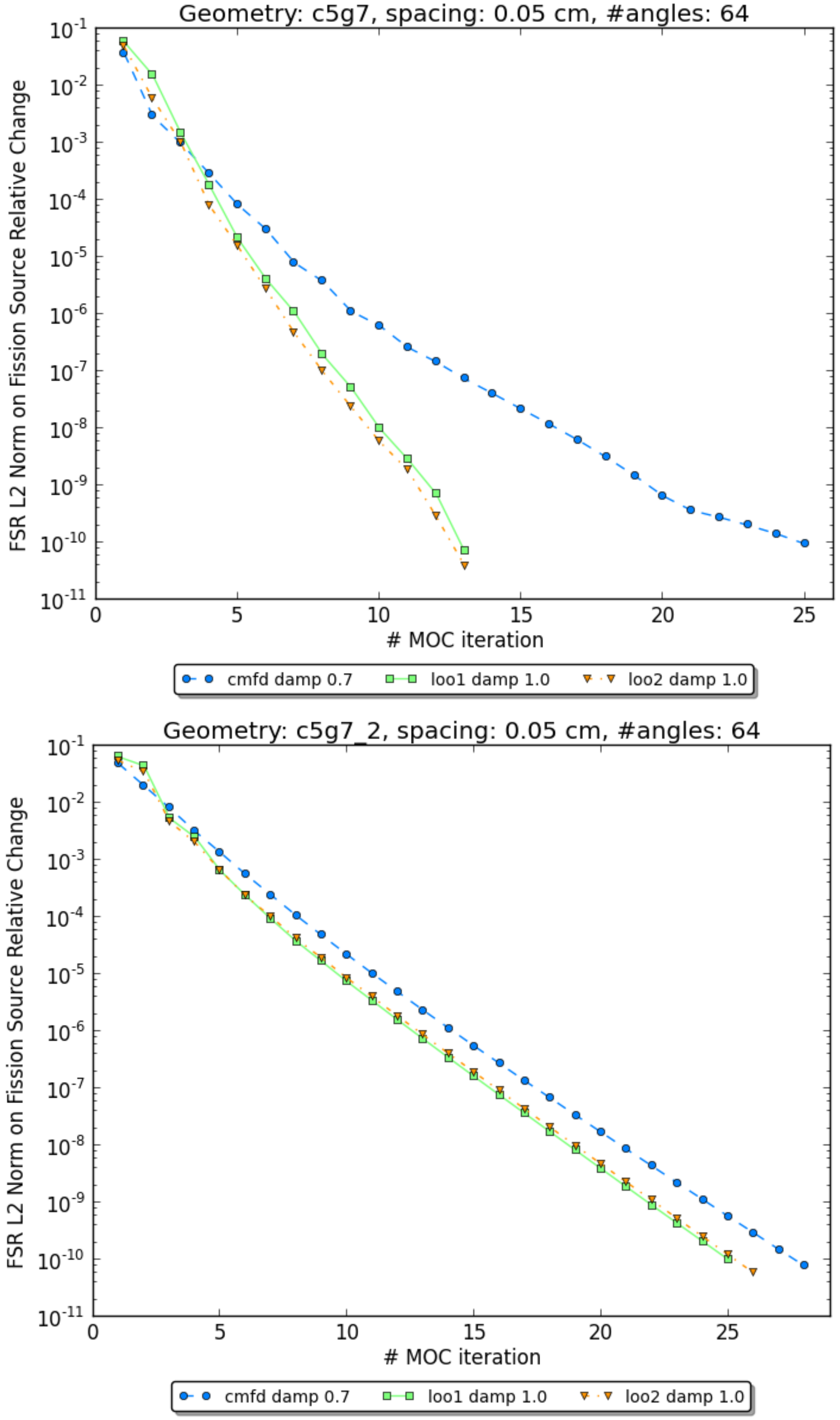

Figure 4-12: Plot of C5G7 Problem Convergence Behavior

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This work proposes LOO, a low-order transport acceleration scheme for accelerating the transport equation using MOC, to address the challenges of slow convergence in solving large heterogeneous reactor problems with high dominance ratio. The current state-of-the-art acceleration scheme is the CMFD method, which has been shown to reduce the number of high-order transport sweeps by more than a factor of 100 on large LWR problems [Smith and Rhodes, 2002]. The goal of LOO is to further reduce the number of high-order transport sweeps required, and to demonstrate improved stability behavior comparing with CMFD.

An MOC solver called OpenMOC was written to solve the 2D heterogeneous reactor problems and it serves as the framework for the CMFD and LOO methods. Details of the MOC method and OpenMOC implementation are given in Section 1. Section 2 presents the derivation and implementation of the CMFD acceleration method in the OpenMOC framework. The convergence issues in CMFD are briefly discussed with numerical results.

LOO starts by generating a coarsely discretized grid. Then it iteratively solves the low-order system using MOC transport approximations. The key is that the flux and current in the quadrature angle-space is preserved. Two variations of LOO are proposed in Section 3. These two flavors of the LOO method are tested against CMFD for three types of sample test problems in Section 4.

Based on the numerical experimental results, current implementation of LOO shows two major advantages over CMFD:

1. LOO does not rely on under-relaxation to converge typical LWR problems tested in this work, whereas CMFD needs a damping factor to converge problems like the C5G7 benchmark problem.

2. LOO typically reduces the number of required MOC high-order transport iterations to solve a LWR

problem to a certain convergence criteria compared with CMFD. This advantage is more profound for problems with strong angular effects (e.g., with large leakages). Examples include the UO2 leakage problem compared with the regular UO2 lattice problem (Section 4.3), and the C5G7 benchmark problem compared with the C5G7 problem's fuel regions only (Section 4.4.2 and Section 4.4.1). However LOO's convergence rate advantage over CMFD's diminishes as the number of FSRs per mesh cell increases.

To summarize, this work presents some preliminary results on accelerating transport solve with a low-order MOC solve on a coarsely discretized grid. This work shows potential benefits of LOO in terms of convergence rate and stability compared with the state-of-the-art CMFD acceleration method.

5.2 Future Work

This work proposes two variations of LOO and there is much room for improvement:

- The reason that LOO does not perform as well when coarse mesh cells are divided into more FSRs as shown in Section 4.3 and Section 4.4 is not clear. Since this has not been reported by other research on CMFD acceleration, the behavior seen here must be resolved.

- LOO's stability is analyzed numerically in this work, and more analytical and numerical stability analysis comparing CMFD and LOO will be helpful in understanding the convergence behavior of LOO.

- LOO will be tested on BWR benchmark problems which typically have larger leakage effects. This will also require extension of the square acceleration mesh to the more general rectangular mesh.

- Currently OpenMOC performs a high-order transport sweep to converge the boundary fluxes without updating FSRs' fluxes and sources. This operation can be either improved to be less costly, or it will be replaced by proper boundary flux prolongation schemes from the low-order solution.

- This work presents one type of low-order transport acceleration method with two slight variations. The difference between the two variations and the reason behind why they behave differently is not understood well.

- Efficient methods for iteratively converging the LOO equations have not yet been studied, as the focus in this work was to determine if the LOO methods could reduce the number of MOC sweeps. Clearly the efficiency of the LOO iteration is also important.

- Application to accelerate the convergence of linear-source versions of MOC should also be studied, as the LOO methods can preserve spatial shapes within acceleration regions, whereas CMFD cannot.

- LOO can clearly also be applied in a multi-level scheme, and this should be tested for full-core LWR problems.
- Extensions to 3D appear to be straightforward, but effort to implement this approach should be tested.

Bibliography

[Adams and Larsen, 2002] Adams, M. L. and Larsen, E. W. (2002). Fast iterative methods for discrete-ordinates particle transport calculations. *Prog. Nucl. Eng.*, 40(1):3–159.

[Aragones and Ahnert, 1986] Aragones, J. M. and Ahnert, C. (1986). A linear discontinuous finite difference formulation for synthetic coarse-mesh few-group diffusion calculations. *Nuclear Science and Engineering*, 94:309–322.

[Askew, 1972] Askew, J. R. (1972). A characteristics formulation of the neutron transport equation in complicated geometries. Technical Report AEEW-M 1108, United Kingdom Atomic Energy Establishment, Winifrith.

[Berinde, 2002] Berinde, V. (2002). Iterative approximation on fixed points for pseudo-contractive operators. In *Seminar on Fixed Point Theory Clui-Napoca*, volume 3, pages 209–216.

[Cefus and Larsen, 1990] Cefus, C. R. and Larsen, E. W. (1990). Stability analysis of coarse-mesh rebalance. *Nucl. Sci. Eng.*, 105(31).

[Chao, 2000] Chao, Y. A. (2000). Coarse mesh finite difference methods and applications. In *Proc. PHYSOR*, Pittsburgh, PA.

[Cho et al., 2002] Cho, J. Y. et al. (2002). Cell based cmfd formulation for acceleration of whole-core method of characteristics calculations. *J. Korean Nucl. Soc.*, 34(250). not printed out.

[Cho et al., 2003a] Cho, N. Z. et al. (2003a). On a new acceleration method for 3d whole core trasport calculations. In *Proc. Annual Meeting of Atomic Energy Society of Japan*, Sasebo, Japan. cannot access this paper.

[Cho et al., 2003b] Cho, N. Z., Lee, G. S., and Park, C. J. (2003b). Partial current-based cmfd acceleration of the 2d/1d fusion method for 3d whole-core transport calculations. *Trans. Am. Nucl. Soc.*, 88(594).

[Cho and Park, 2003] Cho, N. Z. and Park, C. J. (2003). A comparison of coarse mesh rebalance (cmr) and coarse mesh finite difference (cmfd) acceleration methods for the neutron transport calculations. Technical Report NURAPT-2002-02, Nuclear Reactor Analysis and Particle Transport Laboratory, Korea Advanced Institute of Science and Technology.

[Downar et al., 2002] Downar, T. J. et al. (2002). Parcs: Purdue advanced reactor core simulator. In *Proc. PHYSOR 2002*, Seoul, Korea.

[Ferrer et al., 2012] Ferrer, R., Rhodes, J., and Smith, K. S. (2012). Linear source approximation in casmo5. In *Proc. PHYSOR*.

[Gibson, 2013] Gibson, N. A. (2013). Resonance treatment using the discrete generalized multigroup method.

[Grassi, 2007] Grassi, G. (2007). A nonlinear space-angle multigrid acceleration for the method of characteristics in unstructured meshes. *Nucl. Sci. Eng.*, 155:208–222.

[Halsall, 1980] Halsall, M. J. (1980). Cactus, a characteristics solution to the neutron transport equations in complicated geometries. Technical report, United Kingdom Atomic Energy Authority.

[Hebert, 2009] Hebert, A. (2009). *Applied Reactor Physics*. Presses Internationales Polytechnique.

[Hong et al., 2008] Hong, S. G. et al. (2008). On the convergence of the rebalance methods for transport equation for eigenvalue problems. In *Proc. ANS Reactor Physics Topical Meeting, PHYSOR*, Interlaken, Switzerland.

[Hong et al., 2010] Hong, S. G., Kim, K. S., and Song, J. S. (2010). Fourier convergence analysis of the rebalance methods for discrete ordinates transport equations in eigenvalue problems. *Nucl. Sci. Eng.*, 164:33–52.

[Kim and DeHart, 2011] Kim, K.-S. and DeHart, M. D. (2011). Unstructured partial- and net-current based coarse mesh finite difference acceleration applied to the extended step characteristics method in newt. *Annals of Nuclear Engineering*, 38:527–534.

[Knoll et al., 2011a] Knoll, D. A., Park, H., and Newman, C. (2011a). Acceleration of k-eigenvalue / criticality calculations using the jacobian-free newton-krylov method. *Nucl. Sci. Eng.*, 167:133–140. need to print out this paper.

[Knoll et al., 2011b] Knoll, D. A., Park, H., and Smith, K. S. (2011b). Application of the jacobian-free newton-krylov method to nonlinear acceleration of transport source iteration in slab geometry. *Nucl. Sci. Eng.*, 167:122–132.

[Knott et al., 1995] Knott, D., Forssen, B. H., and Edenius, M. (1995). *CASMO-4 Methodology Manual*. Sutdsvik of America. SOA-95/02.

[Knott and Yamamoto, 2010] Knott, D. and Yamamoto, A. (2010). Lattice physics computations. In Cacuci, D. G., editor, *Handbook of Nuclear Engineering*, chapter 9, pages 913–1239. Springer US.

[Larsen and Kelley, 2012] Larsen, E. W. and Kelley, B. W. (2012). Cmfd and coarse-mesh dsa. In *Proc. PHYSOR*, Knoxwille, Tennessee, USA.

[Larsen and Morel, 2010] Larsen, E. W. and Morel, J. E. (2010). Advances in discrete-ordinates methodology. In Azmy, Y. and Sartori, E., editors, *Nuclear Computational Science: A Century in Review*, chapter 1, pages 1–84. Springer Netherlands.

[Lee, 2012] Lee, D. (2012). Convergence analysis of coarse mesh finite difference method applied to two-group three-dimensional neutron diffusion problem. *Journal of Nuclear Science and Technology*, 49(9):926–936.

[Lee, 2013] Lee, D. (2013). Impact of dynamic condensation of energy groups on convergence behavior of one-node cmfd method for neutron diffusion problem. *Nucl. Sci. Eng.*, 174:300–317.

[Marleau et al., 1994] Marleau, G., Roy, R., and Hebert, A. (1994). *DRAGON: A Collision Probability Transport Code for Cell and Supercell Calculations*. Institut de genie nuclearire, Ecole Polytechnique de Montreal. IGE-157.

[Mathematics and Compute Science Division, 2013] Mathematics and Compute Science Division, A. N. L. (2013). PETSc Users Manual, Revision 3.4. `http://www.mcs.anl.gov/petsc/petsc-current/docs/manual.pdf`.

[Park et al., 2011] Park, H., Knoll, D. A., and Newman, C. K. (2011). Nonlinear acceleration of transport criticality problems. In *Proceedings of International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering (M&C 2011)*, Rio de Janeiro, RJ, Brazil. Latin American Section (LAS) / American Nuclear Society (ANS).

[Park et al., 2012] Park, H., Knoll, D. A., and Newman, C. K. (2012). Nonlinear acceleration of transport criticality problems. *Nucl. Sci. Eng.*, 171:1–14. need to print out this paper.

[Park and Cho, 2004] Park, Y. R. and Cho, N. Z. (2004). Coarse-mesh angular dependent rebalance acceleration of the discrete ordinates transport calculations. *Nucl. Sci. Eng.*, 148:355–373.

[Park and Cho, 2008] Park, Y. R. and Cho, N. Z. (2008). Coarse-mesh angular dependent rebalance acceleration of the method of characteristics in x-y geometry. *Nucl. Sci. Eng.*, 158:154–163.

[Rhodes et al., 2008] Rhodes, J., Lee, D., and Smith, K. S. (2008). *CASMO-5 / CASMO-5M – A Fuel Assembly Burnup Program.*

[Sanchez et al., 2010] Sanchez, R., Zmijarevic, I., et al. (2010). Apollo2 year 2010. *Journal of Nuclear Engineering and Technology*, 42(5):474–499.

[Shim et al., 2011] Shim, C. B. et al. (2011). Application of backward differentiation formulation to spatial reactor kinetics calculation with adaptive time step control. *Journal of Nuclear Engineering and Technology*, 43(6).

[Smith, 1983] Smith, K. S. (1983). Nodal method storage reduction by non-linear iteration. *Trans. Am. Nucl. Soc.*, 44(265).

[Smith, 2012] Smith, K. S. (2012). 22.211 nuclear reactor physics lecture 10.

[Smith and Rhodes, 2002] Smith, K. S. and Rhodes, J. D. (2002). Full core, 2d lwr core calculation with casmo-4e. In *Proc. PHYSOR*, Seoul, Korea.

[Smith et al., 2003] Smith, M. A., Lewis, E. E., and Na, B.-C. (2003). Benchmark on deterministic transport calculations without spatial homogenisation: A 2-d/3-d mox fuel assembly benchmark. Technical report, Nuclear Energy Agency Organisation for Economic Co-Operation and Development.

[Willert and Kelley, 2013] Willert, J. A. and Kelley, C. T. (2013). Efficient solutions to the nda-nca low-order eigenvalue problem. In *Proceedings of International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Enginering (M&C 2013)*, Sun Valley, ID.

[Willert et al., 2013] Willert, J. A., Kelley, C. T., Knoll, D. A., and Park, H. (2013). A hybrid approach to the neutron transport k-eigenvalue problem using nda-based algorithms. In *Proceedings of International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering (M&C 2013)*, Sun Valley, ID. American Nuclear Society.

[Wolters et al., 2013] Wolters, E. R., Larsen, E. W., and Martin, W. R. (2013). Hybrid monte carlo-cmfd methods for accelerating fission source convergence. *Nucl. Sci. Eng.*, 174:286–299.

[Yamamoto, 2005] Yamamoto, A. (2005). Generalized coarse-mesh rebalance method for acceleration of neutron transport calculations. *Nucl. Sci. Eng.*, 151:274–282.

[Yamamoto et al., 2007] Yamamoto, A. et al. (2007). 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.

[Yoon and Joo, 2008] Yoon, J. I. and Joo, H. G. (2008). Two-level coarse mesh finite difference formulation with multigroup source expansion nodal kernels. *Journal of Nuclear Science and Technology*, 45(7):668–682.

[Zhong et al., 2008] Zhong, Z. et al. (2008). Implementation of two-level coarse mesh finite difference acceleration in an arbitrary geometry, two-dimensional discrete ordinates transport method. *Nucl. Sci. Eng.*, 158(289).