Advanced Response Matrix Methods for Full Core Analysis

by

Jeremy Alyn Roberts

B.S./M.S. University of Wisconsin - Madison (2009)

Submitted to the Department of Nuclear Science and Engineering in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Nuclear Science and Engineering at the MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2014

© Massachusetts Institute of Technology 2014. All rights reserved.
Advanced Response Matrix Methods for Full Core Analysis

by

Jeremy Alyn Roberts

Submitted to the Department of Nuclear Science and Engineering
on September 23, 2013 in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Nuclear Science and Engineering

Abstract

Modeling full reactor cores with high fidelity transport methods is a difficult task, requiring the largest computers available today. This thesis presents work on an alternative approach using the eigenvalue response matrix method (ERMM). The basic idea of ERMM is to decompose a reactor spatially into local "nodes." Each node represents an independent fixed source transport problem, and the nodes are linked via approximate boundary conditions to reconstruct the global problem using potentially many fewer explicit unknowns than a direct fine mesh solution.

This thesis addresses several outstanding issues related to the ERMM based on deterministic transport. In particular, advanced transport solvers were studied for application to the relatively small and frequently repeated problems characteristic of response function generation. This includes development of preconditioners based on diffusion for use in multigroup Krylov linear solvers. These new solver combinations are up to an order of magnitude faster than competing algorithms.

Additionally, orthogonal bases for space, angle, and energy variables were investigated. For the spatial variable, a new basis set that incorporates a shape function characteristic of pin assemblies was found to reduce significantly the error in representing boundary currents. For the angular variable, it was shown that bases that conserve the partial current at a boundary perform very well, particularly for low orders. For the deterministic transport used in this work, such bases require use of specialized angular quadratures. In the energy variable, it was found that an orthogonal basis constructed using a representative energy spectrum provides an accurate alternative to few group calculations.

Finally, a parallel ERMM code Serment was developed, incorporating the transport and basis development along with several new algorithms for solving the response matrix equations, including variants of Picard iteration, Steffensen's method, and Newton's method. Based on results from several benchmark models, it was found that an accelerated Picard iteration provides the best performance, but Newton's method may be more robust. Furthermore, initial scoping studies demonstrated good scaling on an \( \mathcal{O}(100) \) processor machine.

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

I want first to express my sincere gratitude to my advisor, Prof. Ben Forget, for his advice and guidance over the years. This work was influenced significantly by his own graduate work, and the freedom I've had to explore the topic and others has been an essential and much appreciated part of my time here.

I would also like to thank my thesis reader, Prof. Kord Smith. The practical, real world experience he brings to the classroom (and elsewhere) is invaluable, serving as a constant check against the somewhat more abstract mathematical approach I undertook for this research.

Additionally, Prof. Michael Driscoll and Prof. Emilio Baglietto have my appreciation for serving as an additional committee member and chair, respectively, and for providing valuable suggestions to improve the present (and future) work.

My time and education would not have been the same without the many fruitful discussions I've shared with friends at the office. I'll miss the morning walk-by chat with Bryan, the office chats (about violins, 100+ digits of π, etc.) with Nathan, the many chats with Matt where I had to remember why class Foo did that one thing that one way\(^1\), the (one-way) chats (rants, really) with Gene, the chats where I promised Paul that I would actually contribute to OpenMC (on the radar, I swear), the nerdy transport chats with Lulu, the greedy chats with Carl, the homework chats (venting sessions) with 106 students, and any of the many other chats I've been fortunate to have.

At home, the chats involved lots of neutrons going in and out of the box, but were just as fruitful, and for that I owe my wife Sam all the love in the world. I might also add I owe it to little Felix-to-be for providing a very big impetus to complete this work sooner than later.

I would also like to acknowledge the Department of Energy and its Nuclear Energy University Program for the graduate fellowship that partially supported the first three years of this work, giving me flexibility to pursue a topic of interest to me.

\(^1\)Just kidding, of course, since the documentation is 100% complete and unit tests abound.
## Contents

1 **Introduction and Background** ........................................... 15  
   1.1 The Eigenvalue Response Matrix Method .......................... 16  
      1.1.1 Neutron Balance in a Node ............................... 16  
      1.1.2 Global Neutron Balance .................................. 18  
   1.2 Survey of ERMM Methods ........................................... 18  
      1.2.1 Diffusion-based Methods ................................... 18  
      1.2.2 Transport-based Methods .................................. 19  
   1.3 Major Challenge .................................................. 21  
   1.4 Goals .......................................................... 22  
      1.4.1 Deterministic Transport Algorithms ....................... 22  
      1.4.2 Conservative Expansions for Deterministic Transport Responses .... 23  
      1.4.3 Solving the Eigenvalue Response Matrix Equations ........... 23  
      1.4.4 Parallel Solution of the Eigenvalue Response Matrix Equation .. 24  

2 **Transport Methods — Discretizations and Solvers** ............ 25  
   2.1 The Neutron Transport Equation ................................. 26  
      2.1.1 Discrete Ordinates Method ................................ 26  
      2.1.2 Method of Characteristics ................................ 29  
      2.1.3 Diffusion Approximation .................................. 33  
   2.2 Operator Notation ............................................... 39  
      2.2.1 Angular Flux Equations .................................... 40  
      2.2.2 Moments Equation .......................................... 41  
      2.2.3 Boundary Conditions ....................................... 43
5.3 Angular Expansion ........................................ 84
  5.3.1 Conservative Expansions ............................. 87
  5.3.2 Numerical Examples ................................ 91
5.4 Energy Expansions ........................................ 97
  5.4.1 Mixed Group/Order Expansions ....................... 97
  5.4.2 Low Order Energy Representations .................... 99
  5.4.3 A Numerical Example ................................ 100
5.5 Summary .................................................... 103

6 Analysis of a Fixed Point Iteration ..................... 106
  6.1 Fixed Point Iteration ................................... 106
  6.2 Analytic Model ........................................... 108
  6.3 Convergence of ERMM ...................................... 111
  6.4 Numerical Studies .......................................... 117
    6.4.1 One Group Homogeneous Model ....................... 117
    6.4.2 Two Group Homogeneous Model ....................... 119
    6.4.3 Applications to Heterogeneous Models ............. 120
  6.5 Summary .................................................... 122

7 Accelerating \( \lambda \)- and \( k \)-Convergence .......... 125
  7.1 Solving the Inner Eigenvalue Problem .................. 125
    7.1.1 Power Iteration ..................................... 125
    7.1.2 Krylov Subspace Methods ............................ 127
  7.2 Accelerating the Fixed Point Iteration .................. 128
    7.2.1 Regula Falsi and Related Methods ................... 128
    7.2.2 Steffensen's Method .................................. 129
  7.3 Solving ERME's via Newton's Method ...................... 131
    7.3.1 Newton's Method ..................................... 132
    7.3.2 Inexact Newton and JFNK ............................ 133
    7.3.3 Preconditioning JFNK ................................ 133

6
8 Comparison of ERME Solvers

8.1 Diffusion Benchmarks .......................... 135
  8.1.1 Tolerances and Errors ..................... 136
  8.1.2 Orders and Accuracy ...................... 137
  8.1.3 Inner Solver Comparison ................... 139
  8.1.4 Outer Solver Comparison ................... 140
  8.1.5 Comments ................................. 144

8.2 2-D C5G7 .......................................... 146
  8.2.1 Orders and Accuracy ...................... 146

8.3 3-D Takeda ........................................ 148
  8.3.1 Order Convergence ......................... 148
  8.3.2 Solver Comparison .......................... 150

8.4 Summary ......................................... 151

9 Parallel Algorithms for ERME Solvers

9.1 A Parallel Decomposition ...................... 152

9.2 Implementation .................................. 154

9.3 Numerical Results .............................. 154
  9.3.1 A Response-Dominated Problem ............. 154
  9.3.2 A Balance-Dominated Problem ............... 156

9.4 Comments ....................................... 156

10 Conclusion and Future Work

10.1 Summary ....................................... 159

10.2 Broader Implications .......................... 160

10.3 Future Work .................................... 161
  10.3.1 Linear and Nonlinear Coarse Mesh Preconditioners .... 161
  10.3.2 Sparse Approximate Inverse for Preconditioning .... 162
  10.3.3 Advanced Space and Energy Bases ............... 163
  10.3.4 MOC Responses ............................. 164
  10.3.5 Large Scale Parallel Computing .............. 164
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>E2</td>
<td>Numerical Results</td>
<td>189</td>
</tr>
<tr>
<td>E3</td>
<td>Discussion</td>
<td>190</td>
</tr>
<tr>
<td>E4</td>
<td>Detran Quadratures</td>
<td>190</td>
</tr>
</tbody>
</table>
List of Figures

2-1 Discrete ordinates discretization in 2-D Cartesian coordinates. The shaded region denotes cell (i, j). ................................................. 27
2-2 MOC tracking of a 2-D Cartesian mesh using Gauss-Legendre points. . . . . 32
2-3 Sample code for construction of loss matrix. ........................................... 39

4-1 Three group transport operator sparsity pattern. ................................. 65
4-2 Spectrum of A and the DSA-preconditioned spectrum. ......................... 66
4-3 Spectrum of A and the CMDSA-preconditioned spectrum. .................... 67
4-4 Spectrum of A and the CMDSA+S-preconditioned spectrum. ............... 68
4-5 Spectrum of A and the TC+CMDSA+S-preconditioned spectrum. ......... 69
4-6 Seven group C5G7 assembly. ......................................................... 70
4-7 Comparison of flux and pin fission rate error norms as a function of successive flux difference norm. ......................................................... 72
4-8 Ratio of pin fission rate error norm to flux error norm as a function of successive flux difference norm. ......................................................... 73

5-1 Fast and thermal group partial currents. ........................................... 82
5-2 Fast and thermal eigenspectra of UO₂-MOX minicore. .......................... 82
5-3 DLP and mDLP basis sets for 17 × 17 UO₂ assembly. .......................... 83
5-4 Comparison of fast group expansion and errors. .................................. 84
5-5 Comparison of thermal group expansion and errors. ............................ 85
5-6 Slab reactor schematics. ............................................................... 93
5-7 Absolute relative error in eigenvalue and maximum absolute relative error in fission density as a function of polar order for several expansions. 94
5-8 2-D Quarter Assembly ........................................... 95
5-9 Absolute relative error in eigenvalue, and average and maximum absolute relative error in fission density as a function of angle order for several expansions. .................................................... 96
5-10 Three region, mixed energy slab. ........................................ 98
5-11 1-D pin cell lattice schematic. ........................................ 101
5-12 Critical spectra of the full lattice and the UO$_2$ and MOX pins. ... 101
5-13 Comparison of DLP and modified DLP energy bases. Each basis is orthonormal. ..................................................... 104
5-14 Comparison of eigenvalue errors for various energy bases. ........ 105
5-15 Comparison of pin fission rate errors for various energy bases. ... 105

6-1 Dependence of asymptotic error constants on $\Delta$ for several slab widths. 118
6-2 Dependence of asymptotic error constants on node width $\Delta$. The solid line represents 1-D diffusion, and the markers represent 2-D diffusion. The dashed line corresponds to 1-D transport. ......................... 119
6-3 Dependence of asymptotic error constants on node width $\Delta$. The dashed line represents the two group case. The solid line represents the same problem homogenized to one group using the reference two group solution. ..................................................... 121
6-4 Two group and one group IAEA convergence. .......................... 122
6-5 Three group and one group assembly convergence. ..................... 123

7-1 Representative current eigenspectra for the IAEA problem. In both cases the maximum modulus along the imaginary axis is $e \approx 0.54$. ........ 127

8-1 Absolute relative eigenvalue error for the 2-D IAEA problem as a function of residual norm tolerance for several spatial orders. .......... 137
8-2 Maximum absolute relative assembly power error for the 2-D IAEA problem as a function of residual norm tolerance for several spatial orders. 138
8-3 Absolute relative eigenvalue error as a function of spatial order. .... 139
8-4 Maximum absolute relative assembly power error as a function of spatial order.

8-5 Comparison of Picard acceleration schemes for the 2-D IAEA problem (solid lines) and Koeberg problem (dashed lines).

8-6 Takeda problem absolute relative eigenvalue error as a function of angular order for several spatial orders. The solid lines indicate the conservative basis, while the dashed and dashed-dot lines indicate the DLP basis used to expand the angular flux $\psi$ and current $J$, respectively.

8-7 Takeda problem absolute relative nodal power error as a function of angular order for several spatial orders. The solid lines indicate the conservative basis, while the dashed and dashed-dot lines indicate the DLP basis used to expand the angular flux $\psi$ and current $J$, respectively.

9-1 Schematic of a parallel decomposition for the eigenvalue response matrix method.

9-2 Speedup of 2-D C5G7 problem with zeroth order expansions. The single process time was 8091 seconds over 5 $k$-evaluations.

9-3 Speedup of 3-D IAEA problem with fourth order spatial expansions. The number of local groups is equal to the process count except for several cases labeled by “nlg.”

C-1 Simplified BWR bundle.

C-2 2-D C5G7 core schematic.

C-3 C5G7 pin cell schematic.

D-1 Core power for DD and diffusion cases with 15 cm mesh.

D-2 Relative core power error due to scalar flux approximation.

D-3 Relative assembly error in DD using only scalar flux at 1.496 sec.

D-4 Detran speedup on four core machine.

D-5 Detran parallel efficiency on four core machine.
List of Tables

4.1 Spectrum diagnostic information .................................. 67
4.2 Sweeps and timings for a C5G7 assembly with $\tau = 10^{-4}$ ........ 74
4.3 Sweeps and timings for a C5G7 assembly with $\tau = 10^{-6}$ ........ 75
4.4 Sweeps and timings for a C5G7 assembly with $\tau = 10^{-8}$ ......... 76

5.1 Fast group error norms of DLP and mDLP .......................... 86
5.2 Thermal group error norms of DLP and mDLP ....................... 86
5.3 Two-group cross sections (cm$^{-1}$). ................................... 92

8.1 Picard inner solver comparison for diffusion problems. ............... 141
8.2 Newton solver ILU preconditioner comparison for diffusion problems. 143
8.3 Outer solver comparison for diffusion problems. ....................... 145
8.4 2-D C5G7 order convergence. All errors in %, with reference results from Detran on same mesh. ........................................... 147
8.5 Outer solver comparison for 2-D C5G7 problem with first order expansion. Picard with exponential extrapolation fails due to the nonconservative spatial basis, i.e. $\lambda \neq 1$. ................................. 148
8.6 Outer solver comparison for 3-D Takeda problem with second order spatial expansion and third order polar and azimuthal angle expansions. .. 151

C.1 Three group BWR fuel cross sections. .............................. 170
C.2 Three group BWR moderator cross sections. ........................ 171
C.3 Detran calculation results for eigenvalue and pin power. Here, $p_i$ represents the power of the $i$th pin. The MCNP errors are the stochastic uncertainties, while the Detran errors are relative to the MCNP values. 173

C.4 Detran calculation results for pin power distribution. Here, $p_i$ is the power of the $i$th pin, $e_i$ is the error $p_i - p_i^{\text{ref}}$, and $\bar{p}$ is a region average. 174

C.5 Detran calculation results for eigenvalue and pin power. Here, $p_i$ represents the power of the $i$th pin. The MCNP errors are the stochastic uncertainties, while the Detran errors are relative to the MCNP values. 174

D.1 Summary of transient results. 180

E1 3-, 4-, and 5-point trigonometric Gaussian quadrature rules for polar integration over the range $\mu \in [0,1]$. 188

E2 Absolute errors in 1-D Jacobi moments 191

E3 Absolute errors in 2-D Chebyshev moments 192

E4 Absolute errors in 3-D Chebyshev moments 193

E5 Detran polar quadratures. 194

E6 Detran azimuthal quadratures. 194

E7 Other Detran quadratures. 194
Chapter 1

Introduction and Background

Fundamental to reactor modeling is analysis of the steady-state balance of neutrons, described concisely as

\[ T\phi(\bar{\rho}) = \frac{1}{k} F\phi(\bar{\rho}) \]

where the operator \( T \) describes transport processes, \( F \) describes neutron generation, \( \phi \) is the neutron flux, \( \bar{\rho} \) represents the relevant phase space, and \( k \) is the eigenvalue, the ratio of the number of neutrons in successive generations.

For the past several decades, full core analyses for light water reactors (LWR) have been performed using relatively low fidelity nodal methods based on clever homogenization of phase-space with proven success. However, for more aggressive fuel loadings (including MOX) and longer cycle lengths in existing LWR’s, these methods are becoming less applicable, and for new, highly heterogeneous reactor designs, even less so. While advances in production nodal codes including use of generalized multigroup SP\(_3\) transport with subassembly resolution address several important issues [7], there likely is limited room for further improvement of the underlying approach.

Consequently, a move toward full core analysis techniques that can leverage the high fidelity methods typically used for smaller problems is desired. One such approach is the response matrix method, which is based on a spatial decomposition of the global problem of Eq. 1.1 into local fixed source problems connected by approximate boundary conditions.
1.1 The Eigenvalue Response Matrix Method

The response matrix method (RMM) has been used in various forms since the early 1960's [65]. Using the terminology of Lindahl and Weiss [43], the method can be formulated using explicit volume flux responses, called the “source” RMM, or by using current responses that include fission implicitly and hence are functions of $k$, known as the “direct” RMM. While both methods are used in various nodal methods, the former is more widespread; the focus of this work is on the latter, which shall be referred to as the eigenvalue response matrix method (ERMM) for clarity.

Various formulations of ERMM have been proposed since its first use. Here, we describe a rather general approach based on expansions of the boundary conditions that couple subvolumes of the global problem, a formalism introduced as early as the work of Lindahl [42] and studied more recently by several authors [46, 56, 58].

1.1.1 Neutron Balance in a Node

Suppose the global problem of Eq. 1.1 is defined over a volume $V$. Then a local homogeneous problem can be defined over a subvolume $V_i$ subject to

$${\bf T}\phi(\bar{\rho}_i) = \frac{1}{k}{\bf F}\phi(\bar{\rho}_i), \quad (1.2)$$

and

$$J_{\text{local}}(\bar{\rho}_{is}) = J_{\text{global}}(\bar{\rho}_{is}), \quad (1.3)$$

where $J_{\text{local}}(\bar{\rho}_{is})$ is a function of the incident boundary flux, typically the partial current, which quantifies net flows through a surface.

To represent the local problem numerically, an orthogonal basis, $P_n$, over the relevant phase space is defined

$$P_n(\bar{\rho}_{is}), \quad n = 0, 1, \ldots N \quad (1.4)$$
subject to
\[ \int P_m(\tilde{\rho}_{is})P_n(\tilde{\rho}_{is}) = \delta_{mn} d\rho_{is}. \]  
(1.5)

A response equation is defined
\[ T\phi_i^{ms}(\tilde{\rho}_i) = \frac{1}{k} F\phi_i^{ms}(\tilde{\rho}_i) \]
subject to
\[ J^\text{local}(\tilde{\rho}_{is}) = P_m(\tilde{\rho}_{is}). \]  
(1.7)

The resulting outgoing currents \( J_+(\tilde{\rho}_{is}) \) are used to define response functions
\[ r_i^{ms} = \int P_n(\tilde{\rho}_{is}) J^m_j(\tilde{\rho}_{is}) d\rho_{is}. \]  
(1.8)

The quantity \( r_i^{ms} \) has a simple physical interpretation: it is the \( m'th \) order response out of surface \( s' \) due to a unit incident \( m'th \) order condition on surface \( s \) of subvolume \( i \).

The incident and outgoing currents are expressed as truncated expansions using the same basis
\[ J_{is}^\pm(\tilde{\rho}_{is}) \approx \sum_{n=0}^{N} j^n_{is} P_n(\tilde{\rho}_{is}). \]  
(1.9)

where
\[ j^n_{is} = \int P_n(\tilde{\rho}_{is}) J^n_{is} d\rho_{is}. \]  
(1.10)

These coefficients are then represented in vector form as
\[ J_{i+} = (j^0_{i1+} j^1_{i1+} \cdots j^0_{i2+} j^1_{i2+} \cdots j^N_{iS_2})^T, \]  
(1.11)

and using these together with Eq. 1.8 yields the nodal balance equation
\[ J_{i+} = \left[ \begin{array}{c} j^0_{i1+} \\ j^1_{i1+} \\ \vdots \end{array} \right] = \left[ \begin{array}{cccc} r_{01}^{01} & r_{01}^{11} & \cdots & j_j^0 \\ r_{11}^{01} & r_{11}^{11} & \cdots & j_j^1 \\ \vdots & \vdots & \ddots & \vdots \end{array} \right] \left[ \begin{array}{c} j_j^0 \\ j_j^1 \\ \vdots \end{array} \right] = R_{ij} J_{i-}. \]  
(1.12)
1.1.2 Global Neutron Balance

Global balance is defined by the eigenvalue response matrix equation

\[ MR(k)J_\lambda = \lambda J_\lambda , \]  

where \( R \) is the block diagonal response matrix of \( R_i \), \( J_\lambda \) are vectors containing all incident current coefficients, \( M = M^T \) is the connectivity matrix that redirects outgoing responses as incident responses of neighbors, superscript \( T \) represents the matrix transpose, and \( \lambda \) is the current eigenvalue. If the response matrix \( R \) is conservative (i.e. it strictly maintains neutron balance),

\[ \lim_{k \to k^*} \lambda = 1 , \]  

where \( k^* \) is the true eigenvalue. For nonconservative response expansions, the deviation of \( \lambda \) from unity measures the discontinuities introduced across node boundaries and may be used to evaluate the accuracy of the expansions used (with respect to an infinite expansion).

1.2 Survey of ERMM Methods

1.2.1 Diffusion-based Methods

The method defined by Eqs. 1.2-1.12 has its roots in the work of Shimizu et al. [64, 65], which represents what appears to be the first work on response matrix methods. While independent from it, the authors acknowledge a connection between their work and the earlier and more general theory of invariant imbedding as developed by Bellman et al. [8]. This initial work was based on 1-D diffusion in slab geometry. Aoki and Shimizu extended the approach to two dimensions, using a linear approximation in space to represent boundary currents [5]. A fundamental shortcoming of this early work is what seems to be an assumed value (unity) of the \( k \)-eigenvalue when evaluating responses. Since \( k \) is typically around unity for nuclear reactors, the errors
observed are only tens of pcm, which in general is pretty good, but in their case may be deceptively small. In the later 2-D analysis, the results observed compared favorably to fine mesh diffusion calculations.

Weiss and Lindahl generalized the method by considering arbitrarily high order expansions of the boundary currents in Legendre polynomials [78], while also accounting for \( k \) explicitly. Lindahl further studied expansions of the current, comparing Legendre expansions to an approach that divides the boundary in several segments in which the current is taken to be flat [42]. A more complete overview of these approaches can be found in the review by Lindahl and Weiss [43].

These diffusion-based methods all rely on semi-analytic solutions to the diffusion equation and hence require homogeneous nodes. In the initial scoping studies performed for the present thesis, diffusion-based responses using discretized operators were examined [56]. By numerically integrating the diffusion equation, heterogeneous nodes are treated naturally, though no diffusion models having heterogeneous nodes were studied.

\section{Transport-based Methods}

In addition to methods based on diffusion theory, work was also done to use transport theory in generating responses. Pryor \textit{et al.} used what might be now called a hybrid approach, employing Monte Carlo coupled with the collision probability method to generate responses [54, 53, 66]. That work is unique in its definition of the response matrix \( R(k) \). Considering again Eq. 1.2, the solution \( \phi \) (omitting indices) can be expressed as

\[ \phi = \phi_0 + \frac{1}{k^n} \phi_1 + \frac{1}{k^{2^n}} \phi_2 + \ldots, \]

(1.15)

where \( \phi_i \) is the flux for the \( i \)th neutron generation due to fission. Correspondingly, we can define the associated currents and responses, resulting in

\[ R(k) = R_0 + \frac{1}{k} R_1 + \frac{1}{k^2} R_2 + \ldots. \]

(1.16)
The authors claim this series can be truncated in as few as three terms by estimating the analytical sum, though it is not clear with what accuracy [66]. Note that when no fissile material is present, \( \phi_i = 0 \) for \( i > 0 \), and so \( R(k) = R_0 \).

A somewhat similar approach was taken by Moriwaki et al. [45] in which Monte Carlo was used to generate responses, nominally for application to full assemblies for full core analyses. Their method decomposes the response matrix into four physically distinct components: transmission of incident neutrons from one surface to another surface \((T)\), escape of neutrons born in the volume out of a surface \((L)\), production of neutrons in the volume due to neutrons born in the volume \((A)\), and production of neutrons in the volume due to neutrons entering a surface \((S)\). If we neglect all indices but the surface, a current response can be expressed as

\[

r_{is}^t = T_{is}^t + \frac{1}{k} S_{is} (L_{ir} + \frac{1}{k} A_{ir} (L_{ir} + \frac{1}{k} A_{ir} (L_{ir} + \cdots ))) + \cdots.

\]

Like Eq. 1.16, this infinite sum represents the contributions of each generation to the total response. The matrices \( T, L, A, \) and \( S \) are precomputed, and the full matrix \( R \) is computed on-the-fly by iteration. In the actual implementation, the volume-dependent responses are actually unique for each pin in an assembly. Additionally, spatial segmentation was used on boundaries, but angular dependence was neglected.

The more recent extension of Ishii et al. addressed the limitation by including angular segmentation, increasing the achievable fidelity. However, the resulting amount of data required is quite significant, since the responses are then dependent on spatial segment, angular segment, energy group, and for volume responses, unique pins. For this reason, it seems as though the approach contained in Eq. 1.16 might be more economical, as no volume-dependent responses are required. Of course, obtaining pin reaction rates would require such responses to be available but would preclude their use in solving Eq. 1.13.

Other related work has been development of the incident flux expansion method [33, 46]. The initial work by Ilas and Rahnema focused on a variational approach using a basis consisting of Green's functions for each variable using one-dimensional discrete
ordinates [33]. Mosher and Rahnema extended the method to two dimensions, again using discrete ordinates, and used discrete Legendre polynomials for space and angle expansions. Additionally, they introduced a nonvariational variant that is equivalent to ERMM, though without explicit construction of matrices. Forget and Rahnema further extended this nonvariational approach to three dimensions using Monte Carlo, with continuous Legendre polynomials in space and angle [20]. In all cases, the responses were precomputed as functions of the $k$-eigenvalue, and linear interpolation was used to compute responses during global analysis.

### 1.3 Major Challenge

To apply ERMM even to realistic steady state analyses including feedback effects entails several challenges, the chief of which is the sheer number of responses functions and hence transport solves required. Of course, these response functions are entirely independent for a given state and $k$, and so parallelization is a natural part of the solution.

In the most recent work, responses were pre-computed as a function of $k$ and interpolated as needed. In many cases, clever use of symmetry can further minimize the required data. For benchmark problems, this is sensible, but as the effects of thermal feedback are included, each node becomes unique. As such, precomputation of responses would require dependence on several variables, in addition to $k$, be included in some manner. There seems at this time no completely satisfactory way to parameterize a response function database for accurate steady state analyses. The problem is further exacerbated if burnup is included for cycle analyses. Recent work has attempted to parameterize the responses for steady state analysis of cold critical experiments [31]. While the results are promising (sub-percent errors on pin fission rates—though which error and what norm used are not clearly noted), the problem assessed is not entirely representative of the problems of interest here.

Consequently, the major thrust of this thesis focuses on ERMM implementations suitable for on-the-fly generation of response functions, as this seems at present to be
the only meaningful manner in which to apply ERMM. Of course, any successes achieve in this context would be readily applicable for generation of response databases, should an adequate parameterization scheme be developed in the future.

1.4 Goals

The primary goal of this work is to develop a response matrix method that can efficiently leverage high fidelity deterministic transport methods for solving large scale reactor eigenvalue problems on a variety of computer architectures. A secondary goal is to produce transport algorithms tailored to the relatively small problems associated with response function generation. To support these goals, a number of specific subtopics are addressed, each of which is described as follows.

1.4.1 Deterministic Transport Algorithms

For the response matrix method to be viable, response functions must be computed efficiently. Each response function represents by itself a comparatively small fixed source transport problem subject to an incident boundary flux on one surface and vacuum on the remaining surfaces. Most recent work on transport algorithms has addressed very large models, with a particular emphasis on acceleration techniques for large scale parallel computing.

In this work, algorithm development focuses exclusively on the smaller transport problems associated with response generation. The various transport discretizations used and the algorithms to solve the resulting equations are discussed in Chapter 2. Chapter 3 extends the discussion with development of preconditioners (effectively, accelerators) for application to Krylov linear solvers for the transport equation. A detailed numerical comparison of several solvers and preconditioners is provided in Chapter 4.
1.4.2 Conservative Expansions for Deterministic Transport Responses

As this work applies deterministic transport methods to compute response functions, the associated integrals over the boundary phase space become weighted sums based on numerical quadrature. For the angular variable in particular, the orthogonal basis and quadrature used must be consistent to maintain a "conservative" expansion. Such expansions maintain neutron balance and typically lead to more accurate results with fewer expansion terms. A study of basis sets and associated quadrature sets for a variety of transport approximations is discussed in Chapter 5.

1.4.3 Solving the Eigenvalue Response Matrix Equations

Once the responses defining Eq. 1.13 are defined, an iterative scheme is required to solve for the boundary coefficients and eigenvalue. The most common approach historically has been to use the power method to solve the global balance equation for $J$, with a corresponding $k$-update based on neutron balance, leading to a fixed point iteration. Within this fixed point iteration, more efficient schemes can be used to solve the global balance equation. However, in practice the most expensive part of the computation is likely to be the response functions. If these are computed on-the-fly (that is, for each new $k$), a method that minimizes the number of $k$ evaluations is highly desirable.

Solution of the response matrix equations via fixed point iteration, including an analysis of the convergence of the iteration, is discussed in Chapter 6. In Chapter 7, efficient solvers for the inner $\lambda$ eigenvalue problem are discussed. Additionally, methods to minimize $k$ evaluations are developed, including acceleration of the fixed point iteration via Steffensen's method and solution of the response matrix equations via inexact Newton methods. Chapter 8 provides a numerical study of the the methods of Chapter 7 as applied to several benchmark problems.
1.4.4 Parallel Solution of the Eigenvalue Response Matrix Equation

Ultimately, to solve large models via the response matrix method requires large scale computing. Because the response functions are independent, parallelization is relatively straightforward and may be the most "embarrassingly parallel" scheme available for deterministic methods.

In this work, two levels of parallelism are incorporated explicitly. The first (and natural) level is produced via the geometric domain decomposition of the global model. The second level is based on distribution of responses within a subdomain to a subset of processes. A potential third level exists if the underlying transport solves can also be parallelized. A detailed analysis of the parallel response matrix algorithm is presented along with preliminary numerical results in Chapter 9.
Chapter 2

Transport Methods — Discretizations and Solvers

All variants of the response matrix method are based on decomposing a global fine mesh problem into several local fine mesh problems linked through approximate boundary conditions computed using surface-to-surface and possibly volume-to-surface expansions. Hence, for the local fine mesh problems, a transport method must be used.

In this chapter, several transport approximations are reviewed. In particular, the discrete ordinates ($S_N$) method and method of characteristics (MOC) are developed for application to reactor models. Additionally, the diffusion approximation is also presented. While the fine mesh local problems can also be solved using stochastic (i.e. Monte Carlo) methods, the focus in this work is on deterministic methods and ultimately on the use of the response matrix method as a way to apply such methods to a much larger domain than would be otherwise possible.

In addition to describing the various phase space discretizations, several approaches to solving the resulting algebraic equations are discussed, and particular attention is paid to those methods relevant to the fixed boundary source problems used to compute response functions.
2.1 The Neutron Transport Equation

The time-independent multigroup transport equation for a fixed source is

\[ f_g(\vec{r}, \Omega) + \Sigma_{t,g}(\vec{r})\psi_g(\vec{r}, \Omega) + \sum_{g'=1}^{G} \chi_{g,g'}(\vec{r})\phi_{g'}(\vec{r}, \Omega') \]

\[ + \frac{1}{4\pi k} \sum_{g'=1}^{G} v\Sigma_{f,g'}(\vec{r})\phi_{g'}(\vec{r}, \Omega') + s_g(\vec{r}, \Omega), \quad g = 1 \ldots G, \]

where

\[ \phi_g = \int_{4\pi} d\Omega \psi_g(\vec{r}, \Omega) \]

and the notation is standard [16]. Isotropic scattering has been assumed for clarity of presentation, though this is not in general a requirement. The \( k \)-eigenvalue of Eq. 2.1 is set to unity for a typical multiplying problem, while for the case of response generation, \( k \) is a variable parameter fixed for a given solve.

In the next few sections, methods to discretize Eq. 2.1 are presented. The description goes into some detail to facilitate the development of response function generation given in Chapter 5. In particular, the discrete ordinates method and method of characteristics for discretizing the transport equation are presented. Additionally, a discretization scheme for the diffusion equation is also presented.

2.1.1 Discrete Ordinates Method

The discrete ordinates method [40], often called the \( S_N \) method, is based on discretization of the angular domain into discrete directions for which the transport equation may be solved exactly in simple geometries. For realistic problems, the spatial domain is discretized using one of several methods.

Consider the one group transport equation

\[ \hat{\Omega} \cdot \nabla \psi(\vec{r}, \Omega) + \Sigma_i(\vec{r})\psi(\vec{r}, \Omega) = q(\vec{r}, \Omega), \]
where the emission density $q$ contains all scattering, fission, and external sources. To consider this in more detail, we assume a 2-D Cartesian coordinate system. This provides a slightly more complete set of details than would 1-D without the verbiage of 3-D. In 2-D Cartesian coordinates, Eq. 2.3 becomes

$$\mu \frac{\partial \psi}{\partial x} + \eta \frac{\partial \psi}{\partial y} + \Sigma_i(x, y)\psi(x, y, \mu, \eta) = q(x, y, \mu, \eta),$$

(2.4)

where $\mu$ and $\eta$ are the directional cosines with respect to the $x$ and $y$ axes, respectively.

In the discrete ordinates method, a discrete set of angles $\Omega_n$, $n = 1 \ldots N$ is selected, resulting in

$$\mu_n \frac{\partial \psi_n}{\partial x} + \eta_n \frac{\partial \psi_n}{\partial y} + \Sigma_i(x, y)\psi_n(x, y) = q_n(x, y),$$

(2.5)

where, for example, $\psi_n(x, y) \equiv \psi(x, y, \Omega_n)$.

Numerous advanced schemes have been developed for discretizing the $S_N$ equations in space; the reader is encouraged to see the rather thorough review of Larsen and Morel [38]. However, to incorporate that body of work is beyond the present scope. Rather, we use a simple finite volume approach. The geometry is discretized in $x$ and $y$ as illustrated in Figure 2-1.
In each region, the material and source are assumed to be constant. Integrating Eq. 2.5 over the rectangular element \([x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}] \times [y_{j-\frac{1}{2}}, y_{j+\frac{1}{2}}]\) yields

\[
\Delta_j \mu_n (\psi_{i+\frac{1}{2}, j, n} - \psi_{i-\frac{1}{2}, j, n}) + \Delta_i \eta_n (\psi_{i, j+\frac{1}{2}, n} - \psi_{i, j-\frac{1}{2}, n}) + \Delta_i \Delta_j \Sigma_{tij} \psi_{ijn} = \Delta_i \Delta_j \psi_{ijn},
\]

where \(\Delta_i = x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}, \Delta_j = y_{j+\frac{1}{2}} - y_{j-\frac{1}{2}},\)

\[
\psi_{ijn} = \frac{1}{\Delta_i \Delta_j} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} dx \int_{y_{j-\frac{1}{2}}}^{y_{j+\frac{1}{2}}} dy \psi_n(x, y)
\]

and, for example,

\[
\psi_{i+\frac{1}{2}, j, n} = \frac{1}{\Delta_j} \int_{y_{j-\frac{1}{2}}}^{y_{j+\frac{1}{2}}} dy \psi_n(x_{i+\frac{1}{2}}, y).
\]

To close the equations, the cell center flux must be related to the cell edge fluxes. The simplest scheme is to define the cell center flux as the average of each pair of opposing edge fluxes,

\[
\psi_{ijn} = \frac{1}{2} (\psi_{i+\frac{1}{2}, j, n} + \psi_{i-\frac{1}{2}, j, n}) = \frac{1}{2} (\psi_{i, j+\frac{1}{2}, n} + \psi_{i, j-\frac{1}{2}, n}),
\]

which is the well-known “diamond difference” scheme [40]. Assuming neutron travel into the first octant \((\mu, \eta > 0)\), we have

\[
\psi_{ijn} = \frac{\frac{2}{\Delta_i} \psi_{i, j-\frac{1}{2}, n} + \frac{2}{\Delta_j} \psi_{i+\frac{1}{2}, j, n}}{\Sigma_{tij} + \frac{2}{\Delta_i} \eta_n + \frac{2}{\Delta_j} \mu_n}
\]

\[
\psi_{i+\frac{1}{2}, j, n} = 2\psi_{ijn} - \psi_{i-\frac{1}{2}, j, n}
\]

\[
\psi_{i, j+\frac{1}{2}, n} = 2\psi_{ijn} - \psi_{i, j-\frac{1}{2}, n},
\]

with similar relations for the remaining octants. The last two relations describe a sweeping action, taking known fluxes from a left or bottom boundary and propagating the information across the cell in the direction of travel. Extensions to 1-D and 3-D are straightforward.
The only unspecified term is the emission density. Because a discrete set of angles has been selected, the integrals of the continuous emission density

\[ q(x, y, \Omega) = \frac{\Sigma_i(x, y) + \frac{1}{k} \Sigma_f(x, y)}{4\pi} \int_{4\pi} \psi(x, y, \Omega) d\Omega + s(x, y, \mu, \eta) \quad (2.11) \]

become weighted sums, so that

\[ q_n(x, y) = \frac{\Sigma_i(x, y) + \frac{1}{k} \Sigma_f(x, y)}{4\pi} \sum_{n'=1}^N w_{n'} \psi_n'(x, y) + s_n(x, y). \quad (2.12) \]

Selecting the quadrature points and weights is a nontrivial task. For 1-D problems, the Gauss-Legendre quadrature is by far the most common, since it can accurately integrate the Legendre moments necessary for anisotropic scattering with comparatively few points. In 2-D and 3-D, the choice is less clear. The level symmetric quadrature has been popular, since it can integrate relevant spherical harmonic moments (needed for anisotropic scattering in higher dimensions). However, other quadratures may be better suited for a particular task, and as discussed in Chapter 5, quite specialized quadratures may be needed for optimal use within the response matrix method.

### 2.1.2 Method of Characteristics

The method of characteristics (MOC) is an alternative scheme for discretizing Eq. 2.1. The basic idea of MOC is to integrate the transport equation along a characteristic path of neutron travel from one global boundary to another. The obvious benefit of such an approach is that arbitrary geometries can be easily handled. Only MOC as applied to 2-D geometries is considered in this work.

**The Characteristic Form of the Transport Equation**

Consider again Eq. 2.3. The streaming term \( \hat{\Omega} \cdot \nabla \psi \) is just the rate of change of \( \psi \) along the direction of travel, i.e. along the characteristic. This is particularly clear in Cartesian coordinates. Letting \( \vec{r} = \vec{r}_0 + s\hat{\Omega} \), where \( s \) is the distance traveled from an
initial point $\vec{r}_0$, we have

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

$$= \mu \frac{\partial}{\partial x} + \eta \frac{\partial}{\partial y} + \xi \frac{\partial}{\partial z}$$

$$= \vec{\Omega} \cdot \nabla .$$

We can therefore cast Eq. 2.3 in the form

$$\frac{d}{ds} \psi(\vec{r}_0 + s\vec{\Omega}, \vec{\Omega}) + \Sigma_t(\vec{r}_0 + s\vec{\Omega}) = q(\vec{r}_0 + s\vec{\Omega}, \vec{\Omega}),$$

(2.14)

which is the characteristic form of the transport equation.

To solve Eq. 2.14 numerically via MOC, the simplest approach is to divide the domain into flat source regions (FSR) in which materials, the scalar flux, and the source are all assumed constant. Tracks are defined across the geometry, from one exterior point to another. Tracks are comprised of segments, portions of the track that exist within a single flat source region.

Equation 2.14 can then be integrated directly along each track segment, beginning with a global boundary and propagating outgoing angular fluxes of one segment to the next. For a single segment of length $L$ and incident flux of $\psi^-$, integrating Eq. 2.14 exactly yields an outgoing flux

$$\psi^+ = \psi^- e^{-\Sigma_t L} + \frac{q}{\Sigma_t} \left( 1 - e^{-\Sigma_t L} \right),$$

(2.15)

and average flux

$$\bar{\psi} = \frac{\psi^-}{\Sigma_t L} \left( 1 - e^{-\Sigma_t L} \right) + \frac{Q}{\Sigma_t^2 L} \left( \Sigma_t L + e^{-\Sigma_t L} - 1 \right).$$

(2.16)

This is the so-called “step characteristic” method, which can also readily be applied to the 1-D $S_n$ method. As for the $S_n$ method, higher order discretization schemes have been developed for MOC (see e.g. Refs. [19] and [39]), but for the purposes of this work, the step characteristic scheme is sufficient.
The average scalar fluxes $\bar{\phi}_i$ in the $i$th FSR are weighted sums of the average segment angular fluxes $\bar{\psi}_{is}$, or

$$V_i \bar{\phi}_i = \sum_l w_l \sum_m w_m \sum_{s=i} \bar{\psi}_s L_s \delta_s. \quad (2.17)$$

where $l$ indexes polar angle, $m$ the azimuth, and $s$ a segment, and where $L_s$ is the segment length, $\delta_s$ is the segment width (possibly constant for $m$), and $V_i$ is the volume of region $i$. The emission density is defined in terms of these average FSR fluxes.

Again, similar to the $S_n$ method, an angular quadrature is required to define the azimuthal distribution tracks and the polar angles over which Eq. 2.14 is integrated. For the polar dependence, a segment length $L$ represents the length of the segment in the $xy$-plane, $L_{xy}$, divided by the sine of the angle with respect to the polar ($z$) axis, or $L = L_{xy} / \sin \theta$.

**MOC Tracking**

Probably the most common tracking procedure is to employ evenly-distributed azimuthal angles with a polar angle quadrature based on accurate integration of the Bickley-Naylor functions together with evenly-spaced tracks of some fixed spacing. Frequently, cyclic tracking is enforced, meaning that each track ends exactly where its reflection begins [37]. This perturbs the azimuthal spacing somewhat, but for large enough numbers of tracks, the difference is small.

For this work, a different approach is taken. Any azimuthal and polar quadratures can be used. For each azimuth, tracks are defined with starting points based on Gauss-Legendre quadrature. This allows for highly accurate integration of boundary fluxes over space, a desirable feature for response generation.

Typically, track spacing is an input parameter. Use of uneven spacing complicates this somewhat. Rather, suppose we limit the track spacing to some maximum value, $\delta_{\text{max}}$, and distribute an odd number, $n$, of Gauss-Legendre points be distributed over the total track width. The maximum weight $w_{\text{max}}$ corresponds to the central point. While
there is an analytic form for $w_{\text{max}}$ as a function of $n$, a simple rational approximation is

$$w_{\text{max}} \approx \frac{1.585}{n + 0.5851}.$$  \hspace{1cm} (2.18)

Setting this equal to the allowable spacing and rounding up to the nearest integer gives

$$n = \left\lceil \frac{1.585}{\delta_{\text{max}}} - 0.5851 \right\rceil.$$  \hspace{1cm} (2.19)

Note, only the horizontal or vertical boundary can have a single set of Gauss-Legendre points. For the remaining surfaces, two sets of points are generated due to overlap. However, this still maintains accurate piece-wise integration over the entire boundary. Figure 2-2 shows an example tracking for one quadrant, singling out a FSR of interest to highlight several quantities. Due to symmetry, only the first two quadrants need to be tracked in practice.

Because the end point of one track generally does not coincide with the origin of another track in the reflected direction, reflecting boundary conditions require interpolation. However, for response generation, this is not an issue.
MOC Geometry

Because MOC discretization is based on following tracks across the domain, the geometry is not constrained to a mesh of any type. One of the most flexible ways to represent arbitrary models is through use of constructive solid geometry (CSG).

The basic approach of CSG is to combine so-called primitives using the union, intersection, and difference operators. Primitives typically include basic volumes such as cuboids or ellipsoids. However, these primitives can actually be further divided into primitive surfaces. In this work, all CSG models are based on this latter approach.

2.1.3 Diffusion Approximation

As an alternative to a full treatment of the transport equation, we can use the simpler diffusion approximation. We present a diffusion discretization here for two reasons. First, the diffusion approximation is central to the preconditioners developed below for accelerating transport solves. Second, several well-known diffusion benchmarks that provide further tests of the response matrix method.

If we take the angular flux of Eq. 2.1 to be at most linearly anisotropic, we arrive at the multigroup diffusion equation

\[-\nabla D_g(\vec{r})\nabla \psi_g(\vec{r}) + \Sigma_{rg}(\vec{r}) \psi_g(\vec{r}) = \sum_{g'=1, g' \neq g}^{G} \Sigma_{g\rightarrow g'}(\vec{r}) \psi_{g'}(\vec{r})
\]

\[+ \frac{\chi_g(\vec{r})}{k} \sum_{g'=1}^{c} \nu \Sigma_{fg'}(\vec{r}) \psi_{g'}(\vec{r}) + S_g(\vec{r}), \quad g = 1 \ldots G. \tag{2.20}\]

Similar to the our approach for the S\textsubscript{N} equations, we discretize the diffusion equation using a mesh-centered finite volume approach, assuming that cell materials and sources are constant [27]. Consider the one group diffusion equation

\[-\nabla D(\vec{r})\nabla \phi(\vec{r}) + \Sigma_r(\vec{r}) \phi(\vec{r}) = Q(\vec{r}). \tag{2.21}\]
Integrating Eq. 2.21 over the volume $V_{ijk}$, one obtains

$$
\int_{x_{i-1/2}}^{x_{i+1/2}} dx \int_{y_{j-1/2}}^{y_{j+1/2}} dy \int_{z_{k-1/2}}^{z_{k+1/2}} dz \left\{ -\nabla D(\vec{r}) \nabla \phi(\vec{r}) + \Sigma_{r}(x, y, z) \phi(x, y, z) \right\}
$$

$$
= \int_{x_{i-1/2}}^{x_{i+1/2}} dx \int_{y_{j-1/2}}^{y_{j+1/2}} dy \int_{z_{k-1/2}}^{z_{k+1/2}} dz Q(x, y, z),
$$

or

$$
-D_{ijk} \left\{ \Delta_{yj} \Delta_{zk} \left( \phi_x(x_{i+1/2}, y, z) - \phi_x(x_{i-1/2}, y, z) \right) 
+ \Delta_{xk} \Delta_{yk} \left( \phi_y(x, y_{j+1/2}, z) - \phi_y(x, y_{j-1/2}, z) \right) 
+ \Delta_{zk} \Delta_{xj} \left( \phi_z(x, y, z_{k+1/2}) - \phi_z(x, y, z_{k-1/2}) \right) 
+ \Delta_{xk} \Delta_{yj} \sum_{r_{ijk}} \phi_{ijk} = \Delta_{x} \Delta_{y} \Delta_{z} Q_{ijk},
\right.
$$

where the cell-centered flux and source are defined as

$$
\phi_{ijk} = \frac{1}{\Delta_{x}} \frac{1}{\Delta_{y}} \frac{1}{\Delta_{z}} \int_{x_{i-1/2}}^{x_{i+1/2}} dx \int_{y_{j-1/2}}^{y_{j+1/2}} dy \int_{z_{k-1/2}}^{z_{k+1/2}} dz \phi(x, y, z),
$$

the cell emission density is

$$
Q_{ijk} = \frac{1}{\Delta_{x}} \frac{1}{\Delta_{y}} \frac{1}{\Delta_{z}} \int_{x_{i-1/2}}^{x_{i+1/2}} dx \int_{y_{j-1/2}}^{y_{j+1/2}} dy \int_{z_{k-1/2}}^{z_{k+1/2}} dz Q(x, y, z),
$$

and, for example, $\phi_{x}(x_{i+1/2})$ is the derivative of $\phi$ with respect to $x$, averaged over $y$ and $z$, and evaluated at $x = x_{i+1/2}$.

To evaluate the partial derivatives $\phi_x$, $\phi_y$, and $\phi_z$ in Eq. 2.23, we employ Taylor expansions. For the $x$-directed terms, we have

$$
\phi(x_{i-1}, y_j, z_k) \approx \phi(x_{i-1/2}, y_j, z_k) - \frac{\Delta x_{i-1}}{2} \phi_x(x_{i-1/2}, y_j, z_k)
$$

$$
\phi(x_{i}, y_j, z_k) \approx \phi(x_{i-1/2}, y_j, z_k) + \frac{\Delta x_{i}}{2} \phi_x(x_{i-1/2}, y_j, z_k)
$$

(2.26)
\( \phi(x_{i+1}, y_j, z_k) \approx \phi(x_{i+1/2}, y_j, z_k) - \frac{\Delta x_{i+1}}{2} \phi_x(x_{i-1/2}, y_j, z_k) \)
\( \phi(x_i, y_j, z_k) \approx \phi(x_{i+1/2}, y_j, z_k) + \frac{\Delta x_i}{2} \phi_x(x_{i-1/2}, y_j, z_k) \),

and similarly for the \(y\)- and \(z\)-directed terms. By continuity of net current, we must have
\[
D_{i-1,jk} \phi_x(x_{i-1/2}, y_j, z_k) = D_{ijz} \phi_x(x_{i+1/2}, y_j, z_k).
\]

Multiplying Eqs. 2.26 by \(D_{i-1,jk}/\Delta x_{i-1}\) and \(D_{ijk}/\Delta x_i\) respectively yields
\[
\frac{D_{i-1,jk}}{\Delta x_{i-1}} \phi(x_{i-1}, y_j, z_k) = \frac{D_{i-1,jk}}{\Delta x_{i-1}} \phi(x_{i-1/2}, y_j, z_k) - \frac{D_{i-1,jk}}{2} \phi(x_{i-1/2}, y_j, z_k)
\]
\[
\frac{D_{ijk}}{\Delta x_i} \phi(x_i, y_j, z_k) = \frac{D_{ijk}}{\Delta x_i} \phi(x_{i-1/2}, y_j, z_k) + \frac{D_{ijk}}{2} \phi(x_{i-1/2}, y_j, z_k),
\]

and adding the resulting equations and rearranging yields
\[
\phi(x_{i-1/2}, y_j, z_k) = \frac{D_{i-1,jk} \phi(x_{i-1/2}, y_j, z_k) + D_{ijk} \Delta x_{i-1}}{D_{i-1,jk} \Delta x_i + D_{ijk} \Delta x_{i-1}}.
\]

Substituting this into the second of Eqs. 2.26 gives
\[
\phi_x(x_{i-1/2}, y_j, z_k) = \frac{2}{\Delta x_i} \left( \phi_{ijk} - \frac{D_{i-1,jk} \phi_{i-1,jk} \Delta x_i + D_{ijk} \phi_{ijk} \Delta x_{i-1}}{D_{i-1,jk} \Delta x_i + D_{ijk} \Delta x_{i-1}} \right)
\]
\[
= \frac{2}{\Delta x_i} \left( \frac{D_{i-1,jk} \phi_{ijk} \Delta x_i + D_{ijk} \phi_{i-1,jk} \Delta x_{i-1}}{D_{i-1,jk} \Delta x_i + D_{ijk} \Delta x_{i-1}} \right)
\]
or
\[
\phi_x(x_{i-1/2}, y_j, z_k) = 2D_{i-1,jk} \left( \frac{\phi_{ijk} - \phi_{i-1,jk}}{\Delta x_i D_{i-1,jk} + \Delta x_{i-1} D_{ijk}} \right),
\]
and similarly
\[
\phi_x(x_{i+1/2}, y_j, z_k) = 2D_{i+1,jk} \left( \frac{\phi_{i+1,jk} - \phi_{ijk}}{\Delta x_i D_{i+1,jk} + \Delta x_{i+1} D_{ijk}} \right).
\]

These equations and the equivalents for \(y\) and \(z\) go back into Eq. 2.23 to obtain a set of internal equations.
Substituting the Eqs. 2.32 and 2.33 into Eq. 2.23 yields

\[
-D_{ijk} \left\{ \Delta_{y_j} \Delta_{z_j} \left[ 2D_{i+1,jk} \frac{\phi_{i+1,j} - \phi_{i,j}}{\Delta x_i D_{i+1,jk} + \Delta x_i + 1 D_{ijk}} \right] + 2D_{i-1,jk} \frac{\phi_{i-1,j} - \phi_{i,j}}{\Delta x_i D_{i-1,jk} + \Delta x_i - 1 D_{ijk}} \right\} + \Delta x_i \Delta x_j \Delta z_k \sum_{r,ijk} \phi_{ijk} = \Delta x_i \Delta x_j \Delta z_k Q_{ijk},
\]

where

\[
Q_{ij} = \sum_{g'=1}^{G} \sum_{g' \neq g} \Sigma_{ij} \psi_{i j}^g \psi_{i j}^{g'} + \chi_{k} \sum_{g'=1}^{G} \Sigma_{f,ij} \phi_{ij}^{g'}.
\]

and where \(g\) is the group index that has been suppressed in our development thus far. Equation 2.34 represents the balance of neutrons in the cell \((i, j, k)\).

We can further simplify the notation. Let us define a coupling coefficient

\[
\bar{D}_{i+1/2,jk} \equiv \frac{2D_{i+1,jk} D_{ijk}}{\Delta x_i D_{i+1,jk} + \Delta x_i + 1 D_{ijk}},
\]

with similar coefficients for each direction. Then we can rewrite Eq. 2.34 as

\[
\bar{D}_{i+1/2,jk} \left( \phi_{ijk} - \phi_{i+1,j,k} \right) + \bar{D}_{i-1/2,jk} \left( \phi_{ijk} - \phi_{i-1,j,k} \right) + \bar{D}_{i+1/2,k} \left( \phi_{ijk} - \phi_{i,j+1,k} \right) + \bar{D}_{i-1/2,k} \left( \phi_{ijk} - \phi_{i,j-1,k} \right) + \bar{D}_{i,j+1/2,k} \left( \phi_{ijk} - \phi_{i,j,k+1} \right) + \bar{D}_{i,j,k-1/2} \left( \phi_{ijk} - \phi_{i,j,k-1} \right) + \sum_{r,ijk} \phi_{ijk} = Q_{ijk}.
\]

Note that each term on the left with a coupling coefficient represents the net leakage from a surface divided by the area of that surface.
Boundary Equations

At the boundaries, we employ the albedo condition [27]

\[
\frac{1}{2} D(x, y, z) \nabla \phi(x, y, z) \cdot \hat{n}(x, y, z) + \frac{1 - \beta(x, y, z)}{1 + \beta(x, y, z)} \phi(x, y, z) = J^-(x, y, z),
\]

(2.38)

where \( \hat{n} \) is the outward normal, \( \beta \) describes the albedo condition (\( \beta = 0 \) for vacuum or incident current, and \( \beta = 1 \) for reflection). Additionally, we explicitly include a possible incident partial current \( J^+ \), for which \( \beta = 0 \). For reference, the outgoing partial current is simply

\[
J^+(x, y, z) = -\frac{1}{2} D(x, y, z) \nabla \phi(x, y, z) \cdot \hat{n}(x, y, z) + \frac{1 - \beta(x, y, z)}{1 + \beta(x, y, z)} \phi(x, y, z),
\]

(2.39)

In general, we have six global surfaces, though there can be more depending on whether jagged edges are modeled. However, at a cell level, there are six faces, some of which may be part of a global surface. As an example, we consider the west global boundary:

\[
\text{west boundary: } -\frac{1}{2} D_{1jk} \phi_x(x_{1/2}, y_j, z_k) + \frac{1 - \beta}{4(1 + \beta)} \phi_{1/2,jk} = J^\text{west}_{jk}. \]

(2.40)

Because this expression contains \( \phi \) at the edge, we again need to employ Taylor expansions. For the west boundary, note that

\[
\phi_{1jk} \approx \phi_{1/2,jk} + \frac{\Delta x_i}{2} \phi_x(x_{1/2}, y_j, z_k),
\]

(2.41)

and we rearrange to get

\[
\phi_{1/2,jk} = \phi_{1jk} - \frac{\Delta x_i}{2} \phi_x(x_{1/2}, y_j, z_k).
\]

(2.42)
Placing this into the albedo condition yields

\[ J_{jk}^{\text{west}} = -\frac{1}{2}D_{1jk} \phi_x(x_{1/2},y_j,z_k) + \frac{11 - \beta}{4} \Delta_x \left( \phi_{1jk} - \frac{\Delta_x}{2} \phi_x(x_{1/2},y_j,z_k) \right) \]  

(2.43)

Solving for \( \phi_x(x_{1/2},y_j,z_k) \) gives

\[ \phi_x(x_{1/2},y_j,z_k) = \frac{2(1 - \beta)\phi_{1jk} - 8(1 + \beta)J_{jk}^{\text{west}}}{4(1 + \beta)D_{1jk} + \Delta_x(1 - \beta)} . \]  

(2.44)

For the east, we similarly find

\[ \phi_x(x_{1+1/2},y_j,z_k) = -\frac{2(1 - \beta)\phi_{1jk} - 8(1 + \beta)J_{jk}^{\text{east}}}{4(1 + \beta)D_{1jk} + \Delta_x(1 - \beta)} , \]  

(2.45)

and likewise for the other surfaces. Each of these is placed into the proper partial derivative of Eq. 2.23. For example, the leakage contribution on the left hand side of Eq. 2.37 due to leakage from the west boundary is transformed as follows\(^1\):

\[ \frac{\tilde{D}_{1/2,jk}}{\Delta_x} (\phi_{0jk} - \phi_{1jk}) \rightarrow \frac{2D_{1jk}(1 - \beta)\phi_{1jk}}{4(1 + \beta)D_{1jk} + \Delta_x(1 - \beta)\Delta_x} . \]  

(2.46)

Notice that the \( J^{\text{east}} \) term is not present; rather, we shift that to the right hand side as an external source because that term is not a coefficient for any value of the flux. Hence, that source term becomes

\[ Q_{1jk}^{\text{boundary}} = \frac{8D_{1jk}(1 + \beta)J_{jk}^{\text{west}}}{4(1 + \beta)D_{1jk} + \Delta_x(1 - \beta)\Delta_x} . \]  

(2.47)

**Constructing the Diffusion Matrix**

The equations for cell balance derived heretofore can be combined into a linear system for the cell fluxes. Generically, we can express this system as

\[ L\phi = Q , \]  

(2.48)

\(^1\)Ignore the fact that \( \phi_{0jk} \) doesn't exist!
L = zeros((N, N))  # N x N loss matrix
Q_b = zeros(N)  # Boundary source of size N, if applicable
rendims = [[1, 2], [0, 2], [0, 1]]  # For a given dim, provides remaining dims

for n in range(0, N):
    ijk = cell.to_ijk(n)
    w = [dx[ijk[0]], dy[ijk[1]], dz[ijk[2]]]

    for dim in range(0, 3):
        bound = [0, Nxyz[dim]]
        for dir in range(0, 2):
            if ijk[dim] == bound[dir]:
                side = 3 * dim + dir  # Index of this side (0-5)
                B = beta[side]  # Albedo for this side
                L[n][n] += L[n][n] + 2 * Q_b[n] * (1 - B) / ((4 * (1 + B) * D[n] + w[dir] * (1 - B)) * w[dir])

    else:
        ijk_m = ijk
        ijk_m[dim] = ijk_m[dim] + (-1)**dir
        m = cell.from_ijk(ijk_m)
        w_m = [dx[ijk_m[0]], dy[ijk_m[1]], dz[ijk_m[2]]]
        D_tilde = 2 * D[n] * D[m] / ((w[dir] * D[m] + w_m[dir] * D[n]) * w[dir])
        L[m][n] = -D_tilde / w[dir]
        L[n][m] = L[m][n] + D_tilde / w[dir]

L[n][n] += Sigma_r[n]

Figure 2-3: Sample code for construction of loss matrix.

where L is the so-called diffusion loss operator. While construction of the diffusion matrix is straightforward, we have witnessed (and, admittedly, created ourselves) extremely inelegant matrix construction routines. A particularly elegant solution inspired by Bryan Herman is given in Figure 2-3 using Pythonic notation.

2.2 Operator Notation

So far, three basic models for discretizing the transport equations have been discussed. The latter model, neutron diffusion, leads to a very straightforward system of linear equations for which standard techniques readily apply. The $S_N$ and MOC transport schemes also lead to a set of linear equations. However, casting these equations in a standard form for solution by generic solvers is less obvious. Because our goal is ultimately to apply generic solvers to the equations, it is useful here first to provide a general representation of the equations.
2.2.1 Angular Flux Equations

Before proceeding, we adopt an operator notation following Larsen and Morel [38] to simplify somewhat the exposition to follow. Recall that the scalar flux is defined

\[ \phi(\vec{r}) = \int_{4\pi} d\Omega \psi(\vec{r}, \Omega) \]
\[ \approx \sum_{n=1}^{N} w_n \psi_n(\vec{r}). \]  

Introducing a discrete-to-moment operator \( D \), we enforce

\[ \phi = D\psi, \] \hspace{1cm} (2.50)

where space and angle indices are implicit. Moreover, we introduce a moment-to-discrete operator \( M \), defined

\[ \psi = M\phi, \] \hspace{1cm} (2.51)

where in general \( D \neq M^{-1} \). Finally, defining the operator

\[ L(\cdot) \equiv (\hat{\Omega} \cdot \nabla + \Sigma_t(\vec{r})) (\cdot), \]
\hspace{1cm} (2.52)

the one group discretized transport equation becomes

\[ L\psi = MS\phi + q, \] \hspace{1cm} (2.53)

where \( S \) is the scattering operator (which can include fission). For the case of isotropic scattering in one group, we have \( M = \frac{1}{4\pi} \) while \( S = \Sigma_t(\vec{r}) \).

For the multigroup problem, this notation generalizes to

\[ L_g \psi_g = M \sum_{g'=1}^{G} \left( S_{gg'} + \frac{1}{k} X_{g'g} \right) \psi_{g'} + q_g, \] \hspace{1cm} (2.54)

where the fission source has been explicitly represented. Here, \( X \) represents the fission spectrum \( \chi \) in operator form.
2.2.2 Moments Equation

**$S_N$ in Operator Form**

Equations 2.53 and 2.54 represent equations in which the unknown is the angular flux $\psi$. In general, and particularly for reactor physics, our interest is in computing reaction rates for which only the scalar flux is needed. Consequently, in practice the angular flux is rarely stored explicitly, but is rather computed on-the-fly during a sweep through the space-angle grid. This can be represented explicitly by manipulating the equations to be functions only of the scalar flux (and higher order flux moments if scattering is anisotropic).

To illustrate, consider Eq. 2.53. Multiplying through by the space-angle transport sweep operator $T = DL^{-1}$, we have

\[ D\psi = DL^{-1}MS\phi + DL^{-1}q. \]  \hfill (2.55)

Recalling $\phi = D\psi$ and rearranging yields

\[ (I - TMS)\phi = Tq, \]  \hfill (2.56)

or

\[ A_{WG}\phi = b, \]  \hfill (2.57)

where

\[ A_{WG} \equiv I - TMS \]  \hfill (2.58)

is the within-group transport operator and

\[ b = Tq \]  \hfill (2.59)

represents the uncollided flux moments. Eq. 2.56 represents a linear equation in standard form for the scalar flux moments $\phi$. 
The operator $DL^{-1}$ has a very specific physical interpretation. The inverse $L^{-1}$ represents the set of space-angle sweeps from one global boundary to another. The additional factor $D$ implies that along the space-angle sweep, $\phi$ is updated, thus eliminating the need to store $\psi$. This is exactly how $SN$ and MOC solvers have traditionally been implemented.

The extension to multigroup yields a similar form

$$
(I - T_{MG} M_{MG} \left( S_{MG} + \frac{1}{k} X_{MG} F_{MG}^T \right)) \phi = T_{MG} q_{MG},
$$

(2.60)

where $T_{MG}$ and $M_{MG}$ are block diagonal operators having blocks $T_g$ and $M$, respectively, $X_{MG}$, $F_{MG}$, and $q_{MG}$ are the column vectors having elements $X_g$, $F_g$, and $q_g$ respectively, and the multigroup scattering operator is defined

$$
S_{MG} = \begin{pmatrix}
S_{11} & \cdots & S_{1G} \\
\vdots & \ddots & \vdots \\
S_{G1} & \cdots & S_{GG}
\end{pmatrix}.
$$

(2.61)

Similar to the within-group problem, we define the multigroup transport operator to be

$$
A_{MG} = I - T_{MG} M_{MG} \left( S_{MG} + \frac{1}{k} X_{MG} F_{MG}^T \right).
$$

(2.62)

MOC in Operator Form

While the operators and equations apply directly the $SN$ method, slight changes must be made to represent MOC. In MOC, we iterate on the quantities averaged over a cell, specifically the scalar flux moments $\phi_{cell}$. However, the angular fluxes are computed for each track segment, and each cell can have several tracks crossing it. The average angular flux on each track segment in a given cell is used to generate the average scalar flux.

This can be cast in operator form for a one group problem as

$$
L \psi_{seg} = PMS \phi_{cell} + Pq_{cell},
$$

(2.63)
where \( q_{\text{cell}} \) is the cell-wise external source and \( P \) is a cell-to-segment operator, "P" denoting prolongation, though not quite in the multigrid sense.

We can also define a segment-to-cell operator (restriction) \( R \) so that

\[
\psi_{\text{cell}} = R\psi_{\text{seg}}, \tag{2.64}
\]

and

\[
\phi_{\text{cell}} = DR\psi_{\text{seg}}. \tag{2.65}
\]

Finally, the moments equation akin to Eq. 2.56 is

\[
\left( I - DR\psi_{\text{seg}} \right) \phi_{\text{cell}} = DRL^{-1}Pq_{\text{cell}}. \tag{2.66}
\]

In the context of MOC, the operator

\[
A_{\text{WG-MOC}} = I - DRL^{-1}PMS \tag{2.67}
\]

is implied when discussing the within-group transport operator, and similarly for the multigroup variant.

2.2.3 Boundary Conditions

So far, we have neglected to treat boundary conditions. In particular, reflecting (or periodic, or white) boundary conditions create an additional set of unknowns, namely the incident (or exiting) boundary fluxes. In theory, treating these conditions is straightforward, but we refrain from further discussion for two reasons. First, adding boundary unknowns complicates presentation of the algorithms discussed below. Second, and more importantly, response function calculations use vacuum conditions only, and so treatment of reflecting boundaries is outside our scope. A more detailed discussion can be found in the work of Warsa et al. [77].
2.3 Transport Solvers

Consider the fixed source multigroup transport problem represented by

\[ A_{MG} \phi = b. \]  \hspace{1cm} (2.68)

The traditional method for solving the multigroup equations is via a nested iteration in which a series of within-group equations is solved for each group, with the scattering (and possibly) fission sources updated between each group solve. More modern treatments view the equations as a complete set to be solved simultaneously.

2.3.1 Gauss-Seidel

The Gauss-Seidel method has long been used to solve the multigroup equations. For standard problems, the method first solves the fast (within-)group equation. The updated fast group flux can then be used to define the in-scatter source for the next group, and so on. Because this algorithm really implies inversion of each group block (as defined in Eq. 2.60), the method is more accurately described as block Gauss-Seidel in energy.

Algorithm 1 describes a basic implementation of the method. Historically, convergence of the method is based on the difference of successive fluxes, either as an absolute norm as indicated or a norm of the relative point-wise difference. Assessing convergence in this way can be misleading. The difference between successive iterates can be much smaller than the difference between an iterate and the solution, sometimes resulting in premature convergence. Section 4.2.2 provides a numerical comparison of various convergence criteria.

For problems in which there is no upscatter and no fission, Gauss-Seidel is an essentially exact scheme assuming the within-group equations are solved exactly. However, for cases with upscatter or fission, extra "upscatter" iterations are required, and in some cases, the convergence of Gauss-Seidel becomes prohibitively slow.
**Data:** initial guess for group fluxes $\phi^{(0)}$, external source $q$, eigenvalue $k$, tolerance $\tau$, maximum number of iterations $N$

**Result:** converged group fluxes $\phi$

$n = 1$

while $\tau > \epsilon$ and $n < N$ do

for $g$ from 1 to $G$ do

// Compute in-scatter and in-fission sources

$$b_g = T_g \left( M \sum_{g' = 1}^{G} \left( S_{gg'} + \frac{1}{k} X_g F_{gg'} \right) \phi_{g'}^{(n-1)} + q_g \right)$$

// Solve the within-group problem

$$\phi_g^{(n)} = A_g^{-1} b_g$$

end

$\tau = ||\phi^{(n)} - \phi^{(n-1)}||$

$n = n + 1$

end

**Algorithm 1:** Gauss-Seidel Algorithm for the Multigroup Transport Equation

### 2.3.2 Source Iteration

The standard method for solving the within-group transport equation has been source iteration. The basic idea is a simple one: given an external source (including in-scatter and fission), we guess the flux, compute the within-group scattering source, solve for a new flux, and repeat until converged. Mathematically, source iteration is defined by the process

$$\phi^{(n)} = TMS\phi^{(n-1)} + Tq. \quad (2.69)$$

However, recall that Richardson iteration for the system $Ax = b$ is just

$$x^{(n)} = (I - A)x^{(n-1)} + b. \quad (2.70)$$

Since $I - A_{wg} = TMS$, we see that source iteration is equivalent to Richardson iteration.

Source iteration has a particularly intuitive physical interpretation. Suppose our initial guess is zero. The resulting flux is just the right hand side, or the uncollided flux. Performing one iteration adds the contribution of neutrons having undergone a single collision. A second iteration includes the twice-collided neutrons, and so on. For systems in which neutrons undergo many scattering events, the physics suggests the process can be painfully slow to converge. The math verifies the physics [38]: the
error \|\phi^{(n)} - \phi^{(n-1)}\| of the infinite medium one group problem goes as the scattering ratio \( c = \frac{\Sigma_t}{\Sigma_i} \), meaning that increased scattering leads to slower convergence.

### 2.3.3 Krylov Solvers

Because of the limitations of the Gauss-Seidel and source iteration schemes, much work has been done to apply modern linear solvers to transport problems. One of the most successful class of solvers studied consists of Krylov subspace methods.

**Overview**

One can categorize linear (and eigenvalue) solvers as being *stationary* or *nonstationary*. Stationary methods produce updated solutions using only a single previous solution, and the Gauss-Seidel and Richardson methods are both stationary. Other well-known examples include the Jacobi and successive over-relaxation (SOR) methods. On the other hand, nonstationary methods produce a solution based on two or more previous iterates (or the information used to create those iterates).

Krylov methods are nonstationary methods that rely on construction of a so-called *Krylov subspace* of dimension \( n \), defined for an \( m \times m \) operator \( A \) as

\[
\mathcal{K}(n, x_0) = \text{span}\{x_0, Ax_0, A^2x_0, \ldots, A^{m-1}x_0\},
\]

for some initial, possibly random, vector \( x_0 \). The main idea of Krylov subspace methods is to find \( x \in \mathcal{K}(m, x_0) \) that "best" solves the system of interest, be it an eigenproblem or linear system, where it is assumed that \( m \ll n \).

Working with \( \mathcal{K}(n, x_0) \) directly is difficult numerically, since repeated application of \( A \) sends the initial vector \( x_0 \) into the same direction, namely that of the dominant eigenvector of \( A \). Hence, the basis must be orthogonalized. The canonical approach for nonsymmetric operators is Arnoldi's method, which by successive application of the modified Gram-Schmidt process yields the Arnoldi decomposition

\[
AV = VH + fe_m^T,
\]

(2.72)
where $V \in \mathbb{R}^{m \times n}$ consists of orthonormal columns, $H \in \mathbb{R}^{n \times n}$ is an upper Hessenberg matrix, $e_n$ is the $n$-vector of all zeros except a one in the $n$th location, and $f$ is the residual, which is orthogonal to the columns of $V$.

The most popular Krylov method for nonsymmetric linear systems (such as the transport equations above) is GMRES [61]. The basic idea of GMRES is straightforward: the $n$th step of GMRES produces a $n \times n$ Hessenberg matrix and the corresponding basis $V$, and the approximate solution $x_n$ is found by minimizing the residual norm $\|r\|_2 = \|Ax_n - b\|_2$ for $x_t \in \mathcal{K}(n, x_0)$. In other words, $x_n$ must be in the column space of $\mathcal{K}(n, x_0)$, mathematically expressed as $x_n = Vy$, where $y$ satisfies

$$
\|AVy - b\|_2 = \|V^TAx_n - V^Tb\|
= \|Hy - V^Tb\|.
$$

The last equation shows that GMRES can be thought of as finding the best solution $x_n \in \mathcal{K}(n, x_0)$ in a least-squares sense.

**Preconditioning**

While Krylov methods are generally more robust than the classical stationary methods, their performance can be improved significantly via **preconditioning**. A preconditioner $M$ is an operator whose inverse satisfies $M^{-1} \approx A^{-1}$ in some sense and is relatively inexpensive to apply.

A left preconditioned linear system is

$$
M^{-1}Ax = M^{-1}b
$$

(2.74)

while a right preconditioned system is

$$
AM^{-1}y = b
$$

(2.75)
with $x = M^{-1}y$. The left preconditioned residual differs from the original residual but may be a better metric for convergence. The right preconditioned system preserves the original residual.

A preconditioner typically leads to a clustering of eigenvalues. As an extreme example, suppose that $M = A$. The preconditioned operator is then $AM^{-1} = I$, for which all the eigenvalues are unity. Of course, to apply $M^{-1}$ in this case represents solving the original problem. While preconditioners cannot in general be expected to yield a set of eigenvalues equal to unity, any clustering typically improves convergence. Often, even pushing eigenvalues away from the origin tends to improve convergence [38]. Chapter 3 provides a relatively thorough development of several diffusion-based preconditioners for the transport equation.

**Krylov Methods for the Transport Problems**

Krylov solvers have been used to solve both the within-group [77] and multigroup [18] transport equations. For the multigroup equations in particular, the independent nature of the group-wise blocks makes parallelization in energy much more straightforward than for Gauss-Seidel.

For problems in which there is no fission and upscatter is limited to a subset of thermal groups, it is possible to solve the downscatter groups via Gauss-Seidel and to use a Krylov method on the thermal block of Eq. 2.60. Doing so can yield improved efficiency for some problems [18]. However, when fission is included, as is often true for response function generation, there is always thermal-to-fast coupling, and solving the full system via a Krylov method is to be preferred.

**2.4 Summary**

In this chapter, several transport approximations were described, and several solvers for the resulting equations were discussed. For the rest of this work, the analysis is limited to use of the discrete ordinates method and mesh-centered diffusion, both on Cartesian grids. In the next chapter, the discussion of Krylov solvers is extended with
development of several diffusion-based preconditioners, while Chapter 4 provides a numerical study of the solvers and preconditioners for application to multiplying fixed source problems.
Chapter 3

Transport Methods — Preconditioners

Chapter 2 provided a short survey of trends in transport discretizations and solvers applicable to the fixed source multiplying problems relevant to response matrix methods. As noted, Krylov methods have quickly become important tools for a variety of transport problems, but to be successful, they require adequate preconditioning. This chapter details several preconditioners relevant to transport calculations for reactor physics, and specifically the comparatively small problems characteristics of response matrix methods.

3.1 Diffusion Synthetic Acceleration

Frequently, the most successful preconditioners are those that are based on a priori knowledge of the physics or structure of the problem. This has long been the case for accelerating transport problems, for which early techniques enforced balance over coarse regions in space, angle, and energy.

A related idea that has remained an important innovation is to apply a simple, low order approximation, invariably based on diffusion, to provide an efficient update or correction to an unconverged transport solution. Here, we provide a brief sketch of one diffusion-based scheme known as diffusion synthetic acceleration that has long been used but only recently as a preconditioner. For simplicity, the development is given for the one group case, following the excellent treatment of Larsen and Morel [38].
Suppose we perform one source iteration, so that

$$\phi^{n+\frac{1}{2}} = \text{TMS}\phi^n + Tq.$$  \hfill (3.1)

where \( S \) is assumed to contain both the scatter and fission within-group terms. Note the half index. We subtract this equation from Eq. 2.56 to get

$$\left(\mathbf{I} - \text{TMS}\right)\phi - \phi^{n+\frac{1}{2}} = -\text{TMS}\phi^n,$$  \hfill (3.2)

and adding \( \text{TMS}\phi^{n+\frac{1}{2}} \) to both sides and rearranging,

$$\varepsilon = \left(\mathbf{I} - \text{TMS}\right)^{-1}\text{TMS}\left(\phi^{n+\frac{1}{2}} - \phi^n\right)$$  \hfill (3.3)

where \( \varepsilon \) is the error. Note the error satisfies the transport equation

$$(\Omega \cdot \nabla + \Sigma)\varepsilon - \frac{\Sigma_e}{4\pi} \varepsilon = \frac{\nu}{4\pi},$$  \hfill (3.4)

where

$$\varepsilon = \int_{4\pi} d\Omega \varepsilon.$$  \hfill (3.5)

Solving the error equation is just as expensive as solving the original transport equation. As an alternative, we can use the diffusion approximation. In this case, we have

$$(-\nabla \cdot D \nabla + \Sigma - \Sigma_e)\varepsilon = \nu,$$  \hfill (3.6)

or in operator form,

$$\varepsilon = C^{-1} \nu = C^{-1}\text{TMS}(\phi^{n+\frac{1}{2}} - \phi^n).$$  \hfill (3.7)
This leads to the update

\[ \phi \approx \phi^{n+1} = \phi^{n+\frac{1}{2}} + C^{-1}S(\phi^{n+\frac{1}{2}} - \phi^n) \]

\[ = (I + C^{-1}S)\phi^{n+\frac{1}{2}} - C^{-1}S\phi^n \]

\[ = \left( I - (I + C^{-1}S)(I - TMS) \right)\phi^n + (I + C^{-1}S)Tq \]

\[ = (I - P_{WG-DSA}^{-1}A)\phi^n + P_{WG-DSA}^{-1}Tq, \]

which is the preconditioned source iteration (i.e. source iteration plus diffusion synthetic acceleration), and where

\[ P_{WG-DSA}^{-1} = (I + C^{-1}S) \]

is the within-group diffusion preconditioning process. In standard left-preconditioner form, we have

\[ P_{WG-DSA}^{-1}A = P_{WG-DSA}^{-1}Tq. \]

Previous work indicates diffusion preconditioning coupled with a Krylov solver is less restrictive with respect to discretization and more effective for realistic problems [77].

It is natural to extend the diffusion preconditioner to multigroup problems, yielding the process

\[ P_{MG-DSA}^{-1} \equiv I + C^{-1} \left( S + XF^T \right), \]

where \( C \) is the multigroup diffusion operator defined block-wise as

\[ C_{gg'} \equiv \delta_{gg'} \left( \nabla \cdot D_g(\vec{r})\nabla + \Sigma_{tg}(\vec{r}) \right) - \Sigma_{sgg'}(\vec{r}) - \chi_g \nu \Sigma_{fg'}(\vec{r}). \]

An initial review of the literature yielded no application of multigroup diffusion as a preconditioner for multigroup transport problems. Related work addressed acceleration of outer Gauss-Seidel upscatter iterations [3] and fission iterations based on the rank one fission operator \((XF^T)\) [44], but all using an equivalent one group formulation. In general, the full multigroup diffusion problem is itself expensive for large
problems (an issue addressed below), and this may explain why it has not been used extensively.

3.2 Coarse Mesh Diffusion Preconditioning

Preconditioning the multigroup equations with diffusion can lead to very large diffusion operators, and for many problems, the cost of constructing and, more importantly, applying the preconditioner becomes prohibitive. As an alternative, we investigate the use of coarse mesh diffusion preconditioners. Coarse mesh diffusion operators have long been central to acceleration techniques in reactor analysis, a chief example being the nonlinear diffusion acceleration scheme developed by Smith [70] for nodal diffusion methods and later extended to transport methods [72].

While no results could be found in the literature describing use of coarse mesh diffusion as a preconditioner, more general coarse mesh schemes in space, angle, and energy have been of substantial recent interest, particularly for multigrid preconditioning. Multigrid methods, like DSA (itself a two-grid method in angle), use a coarse grid solution to improve a fine grid solution by way of restriction (essentially averaging) and prolongation (essentially interpolation) operations. The idea is that slowly-varying, slowly-converging error modes become highly oscillatory, quickly-converging modes on the coarse mesh. While a complete description of multigrid methods is outside the present scope, the following sections describe implementation of a two-grid diffusion preconditioner. For a more complete overview of multigrid methods, the reader would be best served by one of the standard reviews available, e.g. Ref. [11].

3.2.1 A Spatial Two-Grid Multigroup Diffusion Preconditioner

In this work, we apply a two-grid spatial scheme to the diffusion preconditioner. Recent work suggests that multigrid methods in the energy variable can work very well [26, 68]. While a slightly different context, nonlinear diffusion acceleration also typically uses a coarse energy mesh to great effect. However, our initial studies using a
coarsened energy variable within a diffusion preconditioner suggests that the simultaneous restriction of space, angle, and energy may have inherent difficulties.

The coarse mesh diffusion preconditioner is a natural extension to Eq. 3.11 and is defined as

$$ P^{-1}_{MG-DSA} \equiv I + pC_H^{-1}r \left( S + XF^T \right), \quad (3.13) $$

where $p$ and $r$ are the prolongation and restriction operators, respectively, and $C_H$ is the diffusion operator defined on the coarse spatial mesh.

Coarse Mesh Operator Homogenization

To define $C_H$, cross sections must be homogenized over the fine mesh. The most physically-sound approach for producing averaged cross sections is to use flux-weighting in a manner that preserves reaction rates. While a variety of such homogenization techniques exist, we use a rather conventional homogenization scheme that is simple to apply in preconditioning. For the particular case of group constant generation via an assembly-level lattice physics solver, the results of the scheme are referred to as assembly homogenized cross sections [71].

To illustrate, suppose we need the total cross section $\Sigma_e$ averaged over coarse mesh cell $j$. Denoting the fine mesh flux in cell $i \in j$ to be $\phi_i$, we have

$$ \Sigma_{e,j} = \frac{\sum_{i \in j} V_i \phi_i \Sigma_{e,i}}{\sum_{i \in j} V_i \phi_i}, \quad (3.14) $$

where $V_i$ is the fine mesh cell volume. If we define the coarse mesh flux to be the volume average

$$ \phi_j = \frac{\sum_{i \in j} V_i \phi_i}{V_j}, \quad (3.15) $$

where

$$ V_j = \sum_{i \in j} V_i, \quad (3.16) $$
then the total interaction rate in coarse cell $j$ is

$$
\sum_{i \in j} V_i \Sigma_{i,j} \phi_i = \Sigma_j \sum_{i \in j} V_i \phi_i = V_j \rho_j \Sigma_{i,j}.
$$

(3.17)

Hence, the averaged quantities preserve the integrated reaction rate associated with the given fine mesh flux. All group constants, including the diffusion coefficient $D$, can be generated in this way.

The obvious problem with this approach is that the fine mesh flux $\phi_i$ is not known. The simplest approximation is to assume a constant flux, leading to volume-weighted cross sections. For preconditioning, this is a suitable approximation, since conserving true reaction rates is not a prerequisite. Unlike certain nonlinear acceleration techniques (e.g., CMFD) that rely on conservation to provide an integral form of the solution at each step, preconditioning—a linear process—only seeks to provide an additive improvement.

However, that certainly does not preclude use of more accurate flux shapes to achieve better results. In typical lattice physics applications, group constants are found by solving the transport equation in a pincell or assembly subject to reflecting conditions and using the resulting spectrum for weighting. For preconditioning, a simple pincell homogenization scheme could be used, in which 2-D pincell problems approximating parts of the full problem are solved, and the resulting region-wise volume-averaged spectra are used to produce homogenized materials in the appropriate cells. Analysis of this approach would be a natural extension to the present work.

One might note that the current flux iterate is freely available for use in homogenization; however, because the flux and homogenization process would change from step to step, the entire process would become nonlinear, and its application within the context of standard Krylov linear solvers would be suspect. Its use in solvers allowing variable preconditioners (e.g., FGMRES [59]) would be valuable future research.
Restriction

To restrict the fine mesh input vector for application of the inverse coarse mesh diffusion operator, a simple spatial average is used. As an example, consider a 1-D problem discretized into 6 fine meshes. The coarse meshing process is based on a level parameter $l$ that defines the number of fine meshes per coarse mesh. Suppose $l = 2$, leading to the fine-to-coarse mapping [0, 0, 1, 1, 2, 2]. The restriction operator is defined

$$
\mathbf{r} = 
\begin{pmatrix}
  v_0 & v_1 & 0 & 0 & 0 & 0 \\
  0 & 0 & v_2 & v_3 & 0 & 0 \\
  0 & 0 & 0 & 0 & v_4 & v_5
\end{pmatrix},
\quad (3.18)
$$

where

$$
v_i = \frac{V_i}{V_j}, \quad \text{for fine mesh } i \text{ in coarse mesh } j, \quad (3.19)
$$

and hence $\sum_{i \in j} v_i = 1$.

Prolongation

To prolong the coarse mesh result back to the fine mesh, the coarse mesh value is distributed on the fine grid based on the (approximate) flux $\bar{\phi}$ used to produce the coarse mesh diffusion operator. Given a coarse mesh value $\phi_j$, the prolonged flux is defined as

$$
\phi_{i \in j} = \phi_j \frac{l \bar{\phi}_i}{\sum_{i \in j} \bar{\phi}_i}. \quad (3.20)
$$
For the example above, suppose each coarse region is assumed to have a fine mesh flux shape of \( \phi = [a, b] \), where \( a + b = 1 \). The prolongation operator is defined

\[
p = \begin{pmatrix}
2a & 0 & 0 \\
2b & 0 & 0 \\
0 & 2a & 0 \\
0 & 2b & 0 \\
0 & 0 & 2a \\
0 & 0 & 2b
\end{pmatrix}.
\] (3.21)

In the constant flux approximation \( (a = b = 0.5) \), multiplication of \( r \in \mathbb{R}^{n\times m} \) by \( p \in \mathbb{R}^{m\times n} \) yields the identity matrix \( I \in \mathbb{R}^{n\times n} \).

**Smoothing**

While the coarse mesh scheme described so far represents a complete preconditioner, it can be significantly improved by a smoothing operation. The motivation for smoothing is that the coarse mesh solve damps low frequency error modes of the fine mesh problem but not high frequency modes. In more physical terms, a coarse mesh solve can be expected to get the gross shape right, but not the finer details. A smoothing operator uses a few iterations of a classical scheme like Richardson, Jacobi, or Gauss-Seidel with the fine mesh operator. In many multigrid algorithms, a smoother is also often used before the coarse mesh solve to ensure the error is “smooth” before restriction; however, in this case the joint scattering and fission operator \( S + \frac{1}{k}XFT \) precedes the coarse mesh solve and can be expected to provide a suitable input vector.

For the coarse mesh diffusion preconditioner, smoothing requires the action of the fine mesh diffusion matrix. While the cost of producing this matrix may be somewhat large, that cost is much smaller than inverting the operator in a fine mesh preconditioner (and much, much smaller than application of the transport operator). The smoothing process used is the weighted Jacobi method, which for the generic problem
\( Ax = b \) is defined by the process

\[
x^{(n+1)} = -\omega A_D^{-1}(A_L + A_U)x^{(n)} + (1 - \omega)A_D^{-1}b,
\]

(3.22)

where \( A_D \) represents the diagonal of \( A \), \( A_L \) and \( A_U \) represent the strictly lower and strictly upper triangular parts of \( A \), and \( \omega \) is the weighting parameter. For completely solving the linear system, \( \omega = 1 \) is ideal (with \( \omega > 1 \) potentially yielding instability); however, for the purposes of damping high frequency errors, the optimum is typically smaller, with special 1-D and 2-D model cases having optima of \( \omega = \frac{2}{3} \) and \( \omega = \frac{4}{5} \), respectively [60]. Algorithm 2 provides a basic outline of the coarse mesh diffusion preconditioner.

**Data:** Input vector \( v_0 \), number of smoothing iterations \( N \), weighting factor \( \omega \)

**Result:** Preconditioned vector \( v \)

// Apply the scattering and fission operator
\( x = (S + \frac{1}{k}XFT)v_0 \)

// Apply restriction
\( x_H = rx \)

// Solve the coarse mesh diffusion problem
\( y_H = C_H^{-1}x_H \)

// Apply prolongation
\( z = py_H \)

// Optionally apply smoothing
for \( i \) from 1 to \( N \) do
    \( z = -\omega C_D^{-1}(C_L + C_U)z + (1 - \omega)C_D^{-1}x \)
end

// Compute output vector
\( v_1 = v_0 + z \)

Algorithm 2: Two-Grid Coarse Mesh Multigroup Diffusion Preconditioner

### 3.3 Transport-Corrected Diffusion Preconditioners

#### 3.3.1 Explicit Versus Matrix-Free Transport Operators

A final technique developed is to form a preconditioner that contains information from the transport operator rather than relying solely on the diffusion approximation. In
theory, forming the transport operator defined by Eq. 2.62 explicitly is possible. With an explicit operator, various approximate factorizations become natural preconditioner options. However, constructing such operators implies working with matrices with sizes proportional to the number of angles, at least implicitly, as part of the construction process. This can quickly lead to huge memory requirements, a complicated construction process, or both.

Construction of explicit transport operators have been studied for for the discrete ordinates method [49] and for the method of characteristics [82]. The former work treated the angular flux directly, and the analysis was limited to relatively small multi-group 1-D problems. The latter work noted the cost of constructing the transport operator was exceedingly large and developed a marginally successful parallel scheme limited to the within-group equations.

In the present work, the transport operator is always a “matrix free” operator, meaning that the action $y \leftarrow Ax$ is done by functions and not explicit matrix operations. Hence, while the action of $A$ is available, $A$ itself is not, so preconditioners based on approximate factorizations are not directly applicable. Ultimately, we restrict any preconditioner using a transport operator to be based on actions only.

### 3.3.2 Newton-based Correction

As an alternative, suppose we have a matrix inverse $M^{-1}$ that represents an initial approximation of $A^{-1} \in \mathbb{R}^{m \times m}$. For application to transport problems, we let $M$ be the diffusion preconditioner $P_{MG-DSA}$ or its coarse mesh equivalent. Our goal is to improve this preconditioning process using the (possibly approximate) action of $A^{-1}$.

To do this, we apply a method attributed by various sources to Shulz or Hotelling and Bodewig [63, 32], but in fact which represents application of Newton’s method for finding a matrix inverse. For brevity, we refer to it as the Newton-Shulz method. While a more rigorous analysis of the method can be found elsewhere [32], it is possible to motivate it by considering the simple problem $ax = 1$ following Ref. [13]. Given $a$, we seek $x$. One way to view this is as the nonlinear problem $f(x) = 1/x - a = 0$. Then
\[ f'(x) = -1/x^2, \] and Newton's method (see Section 7.3) gives the process

\[ x_n = x_{n-1} + \frac{1}{x_{n-1}^2} \left( \frac{1}{x_{n-1}} - a \right) \]

\[ = 2x_{n-1} - x_{n-1}a x_{n-1}. \]  

(3.23)

For the more general case of matrices, this suggests the process

\[ X = X(2I - AX), \]

(3.24)

where \( X \approx A^{-1} \).

As is always the case for Newton's method, convergence to the solution depends on the initial guess being sufficiently close to the solution. By using \( P_{\text{MG-DSA}} \) as the initial guess, convergence has always been achieved in our studies. Hence, a simple one step transport-corrected diffusion preconditioning process is defined by

\[ P_{\text{TC-MG-DSA}}^{-1} = P_{\text{MG-DSA}}^{-1}(2I - A_{\text{MG}}P_{\text{MG-DSA}}^{-1}). \]  

(3.25)

While Eq. 3.25 represents an improved preconditioner, it adds the cost of an additional space-angle-energy sweep, the number of which we are ultimately trying to minimize. As an alternative, we can substitute

\[ A_{\text{MG}} \approx \tilde{A}_{\text{MG}}, \]

(3.26)

where the tilde indicates some approximation. Here, we let that imply a coarser angular quadrature. While this appears at first to be a multigrid method, note that the (approximate) transport operator is not being inverted but rather is being used to improve the inversion of the diffusion preconditioner. In the terminology of Newton methods, using the approximate operator results in an approximate Jacobian.

Because only a few iterations of Newton's method are needed to converge given an appropriate initial guess, using the approximate operator \( \tilde{A}_{\text{MG}} \) will lead to its approximate inverse and not that of \( A_{\text{MG}} \). However, \( \tilde{A}_{\text{MG}}^{-1} \) should be even closer to \( A_{\text{MG}}^{-1} \) than
the original diffusion preconditioning process, and so application of the full transport operator can be expected to yield a much more valuable improvement if used in one final Newton iteration.

Algorithm 3 provides the complete preconditioning process for application to an input vector based on the recursive application of the Newton-Schulz method defined in Algorithm 4. The true operator is only applied once in the last step, but the number of actions of $\tilde{A}$ grows exponentially with the number of iterations: one for $k = 2$, three for $k = 3$, seven for $k = 4$, and so on. Hence, the cost of applying the preconditioner is likely to become excessive for more than two or three iterations unless the approximate operator is significantly less expensive than the true operator.

**Algorithm 3:** Transport-Corrected Diffusion Preconditioner

**Data:** transport operator $A$, approximate transport operator $\tilde{A}$, diffusion preconditioner $P_{\text{MGDP}}$, number of corrections $k$, input vector $x$

**Result:** output vector $y$

```plaintext
y ← Newton-Shulz(\tilde{A}, P_0, k - 1, x)
y ← Ay
y ← 2x - y
y ← P_0^{-1}y
```

**Algorithm 4:** Newton-Shulz

**Data:** operator $A$, initial preconditioner $P_0$, number of corrections $k$, input vector $x$

**Result:** output vector $y$

```plaintext
if $k > 1$ then
    // Apply earlier steps recursively
    y ← Newton-Shulz(A, P_0, k - 1, x)
y ← 2x - Ay
    y ← Newton-Shulz(A, P_0, k - 1, y)
else
    // Apply the initial preconditioner
    y ← (2I - AP_0^{-1})x
    y ← P_0^{-1}y
end
```

61
3.4 Summary

This chapter has presented the classic diffusion synthetic acceleration technique and its multigroup analog in the context of preconditioning for iterative transport solvers. Additionally, a coarse mesh variant was proposed based on two spatial grids and optional smoothing that represents in theory a much less computationally-intensive process than the fine mesh preconditioner. To incorporate more of the transport operator, the Newton-Shulz method was used to form a transport-corrected diffusion preconditioner that has the potential to yield better preconditioners than diffusion alone can provide.

The importance of preconditioning cannot be overstated. Because the reactor models of interest could require hundreds or thousands of individual responses, it is of paramount importance to reduce the cost of each response. Our approach is to couple the Krylov methods of Chapter 2 to the preconditioners developed in this chapter, and as the results of Chapter 4 demonstrate, this yields a very efficient scheme.
Chapter 4

Transport Methods — Results

In this chapter, the theoretical development of Chapters 2 and 3 is applied to representative test problems. The goal is to provide a thorough comparison of several linear solver options and associated preconditioners, ultimately yielding a recommended approach for computing responses as applied to the remainder of this work and for pursuing future work. The methods assessed have been implemented in Detran, a multithreaded, C++ transport code described more completely in Appendix D.

4.1 Spectral Properties

As recently noted by Hamilton, comparatively little work has been done to examine the spectrum of discretized transport operators [26]. Such analysis is largely unuseful for studying the behavior of classical algorithms, such as the power method for eigenvalue problems or Gauss-Seidel or source iteration for fixed source problems, since these are governed primarily by the dominant modes that are reasonably easy to analyze. Moreover, a detailed spectral analysis of an operator usually entails working with dense linear algebra, and for all but the most trivial problems, the operators in question have been far too large to manipulate directly.

On modern computers, memory and processors have improved to the point where it is possible to create an explicit representation of the transport operator and the various preconditioning operators. Having a full spectral analysis can help provide an
understanding of why Krylov methods work very well in many cases but fail in others, and how various preconditioners transform the spectrum in a way rendering problems easier to solve.

4.1.1 Test Problem Description

As an illustrative example, we consider a 9 × 9 square pin BWR bundle with an exaggerated water gap using three group cross sections with isotropic scattering, described in Appendix C. The $S_N$ method is used in two dimensions with a 6 × 6 Chebyshev-DP$_N$ product quadrature. To minimize the number of unknowns, the pins are discretized coarsely using a 3 × 3 mesh. While this is a small problem, it does represent a typical (albeit coarsely modeled) node for responses.

Figure 4-1 shows the sparsity pattern of the full transport operator—and it is certainly not a sparse operator. The structure of the full operator clearly shows the structure of the group-to-group scattering and fission matrices. Distinct columns arise in the upper right because only the fueled cells contribute to those blocks, while in the lower left, the columns are due to the fact that the fuel cross section has no $3 \leftarrow 1$ scattering. The total size of the operator is 3267, and the number of nonzeros is 8382033, about 78% dense. In practice, such high density would make explicit operators impractical. However, recall that all the operators used to create $A$ are inherently sparse, even if never constructed. By going from a $\psi$-based representation to one for $\phi$, we have essentially gone from a typically sparse differential representation to a characteristically dense integral representation.

4.1.2 Spectral Analysis of Multigroup Preconditioners

To obtain a preliminary understanding of the preconditioners developed in Chapter 4, the complete eigenspectrum of $A$ and $AM^{-1}$ was computed, where the preconditioner $M$ is the multigroup DSA (DSA), coarse mesh DSA (CMDSA), or transport-corrected CMDSA (TC+CMDSA). For CMDSA, two cases were investigated, one with no smoothing, and one with three smoothing iterations and $\omega = 0.7$ (denoted by CMDSA+S).
Figure 4-1: Three group transport operator sparsity pattern.

A full pin cell was selected as the coarse mesh, and a spatially flat spectrum was used for homogenization. TC+CMDSA used three steps of an approximate transport operator based on a $2 \times 2$ product quadrature with CMDSA+S as the initial preconditioner. Initially, a $1 \times 1$ quadrature (one angle per octant) was used, but this led to instability.

Figures 4-2 through 4-5 show the spectrum of $A_{MG}$ with each of the preconditioned spectra computed. The $A_{MG}$ spectrum observed here is complex, in contrast to the real spectrum observed by Hamilton for a somewhat similar problem (few group fuel-water lattice with isotropic scattering). The difference could be due to inclusion of the fission term in our operator or the discretization used (DD as opposed to a characteristic discretization).

The various preconditioners all have the effect of condensing the eigenvalues to a smaller region. This is evident both in the spectrum plots and the standard deviation of the eigenvalues, listed in Table 4.1. DSA and CMDSA yield spectral shifts that are
quite similar, but CMDSA leaves more of the spectrum below about 0.8 on the real axis. By adding smoothing, CMDSA+S shifts the spectrum toward unity and provides fairly distinct clustering. At the same time, it adds a comparatively significant complex component, but keep in mind the Re(\lambda) and Im(\lambda) scales are not the same. Adding transport correction to CMDSA+S leads to a spectrum that visually appears to be nearly ideal: all values are tightly clustered away from the origin, in this case centering fairly close to unity.

Table 4.1 also includes the Frobenius norm of the residual matrix \( R = I - AM^{-1} \) for each preconditioner, which gives a quantitative measure of how close a given preconditioning process is to the inverse of the operator of interest. In other words, it quantifies how close the resulting spectrum might be to unity, as we can assess visually from the plots. Additionally, the table includes the number of iterations of a non-restarted, right-preconditioned GMRES for convergence to \( 10^{-12} \) on the relative norm. In general, a preconditioner is successful (at least spectrally), if it reduces all of the parameters listed in Table 4.1. In practice, the reduction in iterations given a reduction in the Frobenius norm or variance of the spectrum may be far from a linear relationship.

![Figure 4-2: Spectrum of A and the DSA-preconditioned spectrum.](image-url)
4.2 Comparison of Solvers and Preconditioners

4.2.1 Problem Description

For a preliminary performance assessment of solvers and preconditioners for fixed source multiplying transport problems, we consider a single UO$_2$ assembly based on the seven group 2-D C5G7 benchmark. Figure 4-6 shows the geometry where the ma-
Figure 4-4: Spectrum of $A$ and the CMDSA+S-preconditioned spectrum.

terial indices map onto the benchmark materials; see the benchmark documentation for more. This problem represents a fairly typical node for 2-D response matrix analysis.

The basic model consists of the discrete ordinates method with an $8 \times 8$ angle-per-octant Chebyshev-DP$_N$ product quadrature. The diamond-difference discretization is used in space, and pins are represented with a volume-conserving $7 \times 7$ discretization. The source is a uniform isotropic boundary source on the left side in the fast group. For the fission treatment, a value of $k_{\text{eff}} = 1$ is assumed.

Three basic solver combinations were studied: Gauss-Seidel (GS) for the multi-group equations with source iteration (SI) for the within-group equations, GS with GMRES(30) for the within-group equations, and GMRES(30) for the multigroup equations. As a reminder, GMRES($N$) refers to GMRES with a restart of $N$.

For SI, two cases were considered. The first used a maximum of 20 inner iterations, while the second used a maximum of just one iteration, yielding what has been referred to as the “flattened” transport operator [24]. For GMRES applied to the within-group equations, DSA preconditioning was applied using either LU or ILU-preconditioned
GMRES(30) to invert the diffusion operator. Using LU leads to an “exact” inversion, while the GMRES solver was set with a tolerance of 0.0001 on the relative residual norm, yielding an inexact inversion. Additionally, ILU was applied using two levels of factorization, with all other remaining parameters set to their PETSc default values.

For GMRES applied to the multigroup equations, several preconditioners were investigated. A full fine mesh MG-D SA was studied, using the same LU and GMRES(30)+ILU settings as the within-group DSA preconditioner. Additionally, MG-CMDSA was tested with and without smoothing. Homogenization is performed over the pin cell with a flat spectrum. The smoothed version uses three iterations and $\omega = 0.7$. The same smoothed MG-CMDSA was used as the initial preconditioner for the transport corrected preconditioner (TC-CMDSA). Three updates with the coarse operator were used followed by one update using the full transport operator.

Because the typical convergence criteria employed for SI and GS differ from that of GMRES (i.e. criteria based on successive flux differences versus a residual norm), a consistent criterion is used throughout. Using a tightly converged solution as a ref-
Figure 4-6: Seven group C5G7 assembly.

ence, the tolerance of each method is set so that the method produces a solution satisfying \( \|x - x_{\text{ref}}\|_2/\|x_{\text{ref}}\|_2 < \tau \), where \( x \) is a vector containing the full multigroup flux and \( \tau \) is the desired "true" tolerance.

4.2.2 Numerical Results

In this study, the number of space-angle sweeps and the total wall time were used as metrics for performance. A space-angle transport sweep, denoted by the operator \( T = DL^{-1} \) in Chapter 2, represents in most cases the single most computationally-intensive part of the iterative schemes analyzed. This is especially true when the number of angles is large, rendering the essentially space- and energy-dependent expenses related to source construction insignificant. However, as the overhead of Krylov solvers or their preconditioners grows, the sweeps alone fail to tell the whole story, and the computational time remains the deciding factor.
Comparison of Norms

Each of the methods was tested using a “true” tolerance of $10^{-4}$, $10^{-6}$, and $10^{-8}$. A tolerance on the multigroup flux directly, while easy to implement, is not necessarily easy to interpret physically. In reality, the analyst cares about reaction rates, and in this case, pin powers becomes a relevant metric. Here, pin power errors are measured by the the maximum relative pin fission rate error, defined

$$\epsilon_{\text{pin}} = \max_{p \in \text{pins}} \left| \frac{d_p - d_{p,\text{ref}}}{d_{p,\text{ref}}} \right|,$$

(4.1)

where $d_p$ represents the total fission rate in any pin. To help correlate errors on the flux to errors on pin powers, Figure 4-7 provides both as a function of the successive flux tolerance as used in GS+SI(20), while Figure 4-8 shows the ratio of the norms as a function of GS+SI(20) tolerance. From the first figure, it is apparent that the flux norm is numerically less conservative than the pin-based norm; that is, given an actual error in the flux, the actual error in the pin fission rates is larger. Figure 4-8 shows that the ratio of the norms is not monotonic, but in the limit, the ratio seems to approach an asymptotic value just under two. This is likely problem specific, but serves to illustrate that error norms for one quantity may not be entirely applicable to another quantity. Similarly, the norm of the successive flux difference, used as the GS convergence criterion, is even less conservative by an additional order of magnitude. This fact is sometimes overlooked, and tolerances that seem appropriately tight can in fact lead to premature convergence. This makes use of a consistent metric for comparison crucial.

Comparison of Within-Group Solvers

Tables 4.2-4.4 provide the number of sweeps and execution time for each solver. The results support our general observation that converging inners leads to wasted effort in the classical GS+SI scheme, with the flattened scheme becoming more effective with a tighter tolerance. A similar position has long been held by Kord Smith in the context of
Figure 4-7: Comparison of flux and pin fission rate error norms as a function of successive flux difference norm.

power iteration for eigenvalue problems, in that case meaning forgoing any iteration on the scattering source, multigroup or otherwise.

The results also show that GMRES affords a fair improvement over SI when nested inside GS with respect to both sweeps and time, but diminishingly so for tighter tolerances. The effect is essentially the same as observed for SI(20) and SI(1); sweeps on the inners are simply not as valuable as outers. However, because sweeps for GMRES on the within-group problem are comparatively more effective than SI sweeps, the degradation is not as great.

Applying DSA preconditioning to the within-group problem further reduces the time to just under half the time of GS+SI(20), with a similar decrease in sweep count, irrespective of the preconditioner implementation. However, the ILU-based implementation is slightly more expensive, meaning that the small, one group diffusion problem representing the DSA process is easily inverted directly via LU elimination. This would likely not be the case for large 3-D problems.

1Communicated directly and indirectly via class and other interaction.
Comparison of Multigroup Solvers

Using GMRES on the full multigroup equations leads to a reduction in sweeps and computational time by roughly an order of magnitude compared to GS+SI(20), with slight improvement for tighter convergence.

All the multigroup preconditioners are successful in reducing the number of sweeps. Only at $\tau = 10^{-4}$ is MG-DSA(LU) more expensive than the unpreconditioned case. MG-DSA(LU) and MG-DSA(ILU) exhibit the same reduction in sweeps for all values of $\tau$, meaning that the preconditioner tolerance of $10^{-4}$ is tight enough to conserve all spectral benefits of the operator. MG-DSA(ILU) is less expensive than MG-DSA(LU) by a factor of 2-3, with diminishing improvement for tighter tolerances.

Compared to DSA, CMDSA leads to roughly 1.5-2 times as many sweeps but with execution times just ender that of MG-DSA(ILU). Because the CMDSA process is based on homogenization over $7 \times 7$ cells, inversion of the corresponding coarse mesh diffusion operator is a trivial expense, even by LU factorization. By adding smoothing
Table 4.2: Sweeps and timings for a C5G7 assembly with $\tau = 10^{-4}$.

( denoted CMDSA+S), the cost of applying the fine mesh diffusion operator (but not its inverse) is incurred, but the reduction in sweeps leads to a slight but increasing reduction in execution time with tighter tolerance.

Finally, adding transport correction to CMDSA+S further reduces the number of sweeps, but at the cost of applying the low order transport operator. Because the low order operator uses 4 angles per octant, and the high order uses 64 per octant, one high order space-angle sweep should be equivalent to 16 low order space-angle sweeps. This is not entirely true in practice, since construction of the scattering source is independent of the quadrature. Even so, for $\tau = 10^{-8}$, TC-CMDSA+S required 588 low order sweeps, equivalent to 36 high order sweeps. Hence, a more representative sweep count is 127 rather than 91. However, its cost at 6.8 s is higher than the cost of CMDSA at about the same sweep count, and this apparent discrepancy can likely be attributed to the angle-independent costs of the low order operator.

4.3 Summary

The results strongly suggest that full multigroup GMRES with a coarse mesh diffusion preconditioner is an ideal approach for solving fixed source multiplying problems.
<table>
<thead>
<tr>
<th>Method</th>
<th>Sweeps</th>
<th>Sweeps/Groups</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GS, SI(20)</td>
<td>2960</td>
<td>422</td>
<td>109.3</td>
</tr>
<tr>
<td>GS, SI(1)</td>
<td>2569</td>
<td>367</td>
<td>95.3</td>
</tr>
<tr>
<td>GS, GMRES</td>
<td>1998</td>
<td>285</td>
<td>73.0</td>
</tr>
<tr>
<td>GS, GMRES+DSA(LU)</td>
<td>1092</td>
<td>156</td>
<td>42.2</td>
</tr>
<tr>
<td>GS, GMRES+DSA(ILU)</td>
<td>1092</td>
<td>156</td>
<td>44.4</td>
</tr>
<tr>
<td>GMRES</td>
<td>315</td>
<td>45</td>
<td>11.7</td>
</tr>
<tr>
<td>GMRES+DSA(LU)</td>
<td>63</td>
<td>9</td>
<td>10.4</td>
</tr>
<tr>
<td>GMRES+DSA(ILU)</td>
<td>63</td>
<td>9</td>
<td>4.8</td>
</tr>
<tr>
<td>GMRES+CMDSA</td>
<td>105</td>
<td>15</td>
<td>4.0</td>
</tr>
<tr>
<td>GMRES+CMDSA+S</td>
<td>91</td>
<td>13</td>
<td>3.6</td>
</tr>
<tr>
<td>GMRES+TC-CMDSA+S</td>
<td>77</td>
<td>11</td>
<td>5.6</td>
</tr>
</tbody>
</table>

Table 4.3: Sweeps and timings for a C5G7 assembly with $\tau = 10^{-6}$.

While several combinations were considered, it is recognized that many possible parameter permutations were omitted. Some of these were eliminated based on past experience, and others were simply deemed outside the intended scope. Of the latter, a more careful study of the coarse mesh parameters (meshing, spectra, and smoothing, etc.) and preconditioners for the fine mesh diffusion solver would be of value.

This chapter along with Chapters 2 and 3 have focused exclusively on solving fixed source multiplying problems, the general class of problems into which response generation falls. Over the next several chapters, the focus will shift from solution of the local transport problems, to defining the boundary conditions needed to specify the local problems, and finally to methods for solving the resulting response matrix equations.
<table>
<thead>
<tr>
<th>Method</th>
<th>Sweeps</th>
<th>Sweeps/Groups</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GS, SI(20)</td>
<td>4345</td>
<td>620</td>
<td>162.2</td>
</tr>
<tr>
<td>GS, SI(1)</td>
<td>3297</td>
<td>471</td>
<td>122.9</td>
</tr>
<tr>
<td>GS, GMRES</td>
<td>3447</td>
<td>492</td>
<td>127.1</td>
</tr>
<tr>
<td>GS, GMRES+DSA(LU)</td>
<td>1776</td>
<td>253</td>
<td>68.7</td>
</tr>
<tr>
<td>GS, GMRES+DSA(ILU)</td>
<td>1811</td>
<td>258</td>
<td>74.0</td>
</tr>
<tr>
<td>GMRES</td>
<td>406</td>
<td>58</td>
<td>15.3</td>
</tr>
<tr>
<td>GMRES+DSA(LU)</td>
<td>77</td>
<td>11</td>
<td>11.1</td>
</tr>
<tr>
<td>GMRES+DSA(ILU)</td>
<td>77</td>
<td>11</td>
<td>5.8</td>
</tr>
<tr>
<td>GMRES+CMDSA</td>
<td>126</td>
<td>18</td>
<td>4.9</td>
</tr>
<tr>
<td>GMRES+CMDSA+S</td>
<td>105</td>
<td>15</td>
<td>4.2</td>
</tr>
<tr>
<td>GMRES+TC-CMDSA+S</td>
<td>91</td>
<td>13</td>
<td>6.8</td>
</tr>
</tbody>
</table>

Table 4.4: Sweeps and timings for a C5G7 assembly with $\tau = 10^{-8}$. 
Chapter 5

Deterministic Response Generation

The basic terminology associated with the response matrix method was developed in Chapter 1, and transport approximations used to compute responses were discussed in Chapter 2. The purpose of this chapter is to complete the numerical development of the response matrix equations. In particular, orthogonal basis sets for each phase space variable are studied, with the ultimate goal of identifying optimal expansion schemes for deterministic responses. Additionally, a simple semi-analytical response model based on transverse integration of the multigroup diffusion equation is presented.

5.1 Orthogonal Expansions

Because the response matrix method depends so fundamentally on expansions of all phase space variables in a basis of orthogonal functions, it is prudent to review in brief several of the most common orthogonal bases applicable to the method and to set notation for the remainder of this chapter.

5.1.1 Two Common Bases

While the transport equation is a continuous equation, the numerical schemes discussed in Chapter 2 used to approximate solutions are inherently discrete. Conse-
quently, two generic classes of orthogonal functions are applicable: those with continuous functions and those with discrete functions. Of the continuous bases used in mathematical physics, the Legendre polynomials (LPs) are one of the most common. In transport theory, the LPs are the standard basis used for expanding angular dependence of the scattering cross section, though this was not treated explicitly in the preceding chapters. The LPs are orthogonal over the range $x \in [-1,1]$, satisfying

$$
\int_{-1}^{1} dx P_i(x)P_j(x) = \frac{2\delta_{ij}}{2i + 1}, \quad (5.1)
$$

where $P_i$ will denote a generic basis member of order $i$; for polynomial bases, the order corresponds to the degree. The first few of these well-known functions are defined

$$
P_0(x) = 1, \quad P_1(x) = x, \quad P_2(x) = \frac{1}{2}(3x^2 - x). \quad (5.2)
$$

The Legendre polynomials have a discrete analog known as the discrete Legendre Polynomials (DLPs) [48]; however, these same polynomials—to within a normalization constant—are equivalent to discrete Chebyshev polynomials [1] and Gram polynomials [15]. Because these polynomials were first introduced to the nuclear engineering community as DLPs, we will use that designation [46]. The DLPs are defined in much the same way as the LPs: the zeroth order function is a discrete vector of ones, while the second term is a vector with entries proportional to the integers $i \in [-N,N]$ where $2N + 1$ is the number of terms. Because scaling is arbitrary, the DLPs are normalized here so that $P_i^jP_i = \delta_{ij}$, thus eliminating a secondary scaling coefficient upon reconstructing and approximation. For a vector having 5 terms, the first few normalized DLPs are

$$
P_0 = \frac{1}{\sqrt{5}} [1,1,1,1,1], \quad P_1 = \frac{1}{\sqrt{10}} [-2,1,0,1,2],
$$

$$
P_2 = \frac{1}{\sqrt{14}} [2,-1,2,-1,2], \quad P_3 = \frac{1}{\sqrt{10}} [-1,2,0,-2,1]. \quad (5.3)
$$
5.1.2 Expansion in All Variables

The phase space of interest includes space, angle, and energy. The response matrix method exploits the linearity of the transport equation by reconstructing solution based on response functions, previously introduced in Chapter 1. In more generic mathematical terms, response functions are generalized Green's function; generalized here refers to the fact that the forcing function is not simply a Dirac-$\delta$ function in the variables, but rather any orthogonal function that is complete over the phase space domain.

In this work (as in most, if not all, previous work on response matrix methods), each phase space variable is expanded individually, and the combined tensor product of one dimensional functions represents the total basis function. For example, consider the 1-D transport case having just (polar) angle and energy dependence. The basis function for the $m$th order angular expansion and $h$th energy expansion can be represented generically as

$$\Gamma_{m,h}(\mu, g) = P_m(\mu)P_h(g),$$

(5.4)

where again $P$ denotes the underlying basis and $g$ denotes the energy group since continuous energy is not considered in this work.

A somewhat natural alternative applicable to angular dependence in 2-D is to use a set of 2-D orthogonal functions such as the spherical harmonics. However, this approach is not considered, as it lacks the flexibility afforded by separate polar and azimuthal expansions.

5.2 Spatial Expansion

This work deals exclusively with Cartesian geometries associated with typical light water reactors. Consequently, expansions in the spatial variable are limited to one side of a 2-D rectangular cell or one face of a 3-D rectangular prism.
5.2.1 Standard Orthogonal Bases

As was mentioned briefly in Chapter 1, the original work on response matrix methods used relatively crude approximations in both space and angle. An early improvement assumed a linear spatial dependence within a diffusion-based response matrix method. Lindahl generalized this to an arbitrary polynomial dependence using the Legendre polynomials.

For the homogeneous nodes used in diffusion analyses, a continuous basis is rather natural. For heterogeneous models based on finite difference discretizations, a continuous basis is not generally guaranteed to remain orthogonal since the underlying dependence of the flux is assumed to be flat across a cell. To integrate over a Cartesian mesh (for S_n or diffusion), a simple midpoint rule is employed for integrating over the surface. While this is a conservative scheme consistent with the discretizations used, it is not an accurate quadrature for the Legendre basis members beyond zeroth order. For MOC, the spatial points along a surface may be nonuniform. Associated with each point is a weight used for the spatial integration. Hence, more accurate spatial integration is available for MOC than for the other methods. In particular, for 2-D problems, the Gaussian track spacing, possibly with subdivisions, can yield highly accurate moments.

As an alternative to a continuous basis, Mosher applied the discrete Legendre polynomials (DLPs) to both the spatial and angular variables for responses based on 2-D discrete ordinates [46]. This approach was later extended to treat heterogeneous diffusion problems based on a finite difference discretization. For methods discretized on a Cartesian mesh, a discrete basis is quite easy to apply. In theory, there is no requirement on the mesh spacing, but physically, the basis is accurate for low order expansions only if the mesh has evenly-spaced cells; otherwise, the basis gives preferential treatment for smaller cells since an even weighting is assumed. Because any basis can be orthogonalized after construction, it is possible to build in spatial dimensions explicitly, which could improve its performance, but that is not investigated here.
5.2.2 A Modified Basis

A different approach, and one echoed below for treatment of energy, is to use a basis that represents a portion of the physics \textit{a priori}. For example, the spatial dependence of the flux in a typical $17 \times 17$ PWR assembly along any surface is not likely well-represented by any simple low order expansion, and certainly not over all energies. Rather, the boundary flux has intensities influenced by boundary pins, and taking this shape effect into account beforehand can potentially minimize the order needed to achieve a desired accuracy.

For the case of pin lattices, an easy approach for incorporating the physics in the basis is to include a shape function that approximates the pin structure as the second member of the basis. After orthogonalizing this member against a flat, zeroth order member, additional members are added based on orthogonalizing first and higher order functions against both the constant function and the shape function. Typically, a good shape function is sinusoidal, matching the periodic nature of the lattice.

To illustrate, consider Figure 5-1, which depicts the spatial dependence of the fast and thermal group partial currents along one edge for a reflected $5 \times 5$ lattice of UO$_2$ pin cells from the C5G7 benchmark (see Appendix C). The partial currents are strongly sinusoidal, and, outside of a constant additive factor and multiplicative constant, can be well-approximated using a two term cosine series expansion of the form

$$J(x) \approx A + B \cos(\omega x).$$

(5.5)

For this particular case, the pin pitch is 1.26 cm, and hence $\omega = \frac{2\pi}{1.26} \approx 4.9867$ works well. While $A$ and $B$ are group-dependent, these factors are largely irrelevant, since the basis constructed consists of a constant and cosine term as its first two members.

Since these curves are based on a uniform lattice of one pin type, they cannot capture all the spatial effects of a realistic assembly having guide tubes and nonuniform fuel loadings. To demonstrate their accuracy in representing more complicated spatial dependence, we consider the $2 \times 2$ assembly minicore represented by the fueled portion of the quarter core 2-D C5G7 benchmark subject to periodic boundary conditions. The
corresponding thermal and fast group fluxes are depicted in Figures 5-2a and 5-2b, respectively. The UO₂ assemblies are easily picked out noting their far larger thermal flux than the neighboring MOX assemblies.

Figure 5-2: Fast and thermal eigenspectra of UO₂-MOX minicore.

Figure 5-3 shows the third order DLP and mDLP basis for the full assembly. The boundary partial current and third order DLP and mDLP approximations for a UO₂ assembly are shown in Figures 5-4 and 5-5 for the thermal and fast groups. Additionally, the partial current is approximated using a third order DLP (dashed lines) and a four term modified DLP basis (dash-dot lines). In this case the linear term gives no extra
information due to symmetry. The modulated shape does quite well at capturing the pin level detail, while the basic polynomial expansion captures just the gross shape.

![DLP vs mDLP](image)

**Figure 5-3:** DLP and mDLP basis sets for 17 × 17 UO₂ assembly.

Tables 5.1 and 5.2 provide an additional comparison, giving the relative $L_1$, $L_2$ and $L_\infty$ norms of the expansion errors, defined as $\| J_{\text{approx}} - J \| / \| J \|$ for each norm. Both bases are identical for zeroth order. The modified basis yields a very low error using just two terms, the constant and cosine, obtained by the second order expansion. Because the third term is linear, it provides no improvement, but near global boundaries, this linear term would likely help maintain accuracy. In general, including the cosine shape function leads to a reduction in the $L_1$ and $L_2$ norms by roughly a factor 2-3 for the fast group and nearly an order of magnitude for the thermal group. The $L_\infty$ is also reduced, but to a smaller extent. Somewhat surprisingly, the second order mDLP expansion yields lower errors than any order DLP expansion, and increasing order leads to comparatively larger improvement for mDLP than for DLP.
Recent studies typically employ a fourth order (five term) expansions in space that, in conjunction with angular expansions, yield to pin power errors on the order of 1%. If a modulated basis were used instead, pin power errors might be reduced by factors similar to those observed here for the partial current.

Using a physics-informed basis requires a relatively smooth solution to yield improvements in the expansion (and ultimately, a response matrix solution). This tends not to be the case for the simple discrete ordinates discretizations used here, and so application of this spatial basis would be best suited to MOC, though this has not yet been investigated.

5.3 Angular Expansion

In this section, our interest is the basis used for expanding the boundary flux in angle, and so we will temporarily suppress any space or energy indices. As above, we denote a member of the orthogonal basis to be $\Gamma_{lm}$, where $l$ and $m$ index the polar and azimuthal
angles. The (incident) angular flux on a surface can be represented as

\[ \psi(\Omega) = \sum_{l=0}^{\infty} r_{lm} \Gamma_{lm}(\Omega), \]

where

\[ r_l = \int_{\hat{n} \cdot \Omega > 1} W(\Omega)\Gamma_l(\Omega)\psi(\Omega)d\Omega, \]

\( \hat{n} \) is the inward surface normal, \( \Omega \) represents angle, and dependence of other variables is suppressed. Furthermore, the basis functions are subject to the orthogonality constraint

\[ \int_{\hat{n} \cdot \Omega < 1} W(\Omega)\Gamma_l(\Omega)\Gamma_m(\Omega)d\Omega = c_l \delta_{lm} \]

for some weighting function \( W \).
Table 5.1: Fast group error norms of DLP and mDLP

<table>
<thead>
<tr>
<th>order</th>
<th>DLP</th>
<th>mDLP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$L_1$</td>
<td>$L_2$</td>
</tr>
<tr>
<td>0</td>
<td>1.351e-02</td>
<td>1.561e-02</td>
</tr>
<tr>
<td>2</td>
<td>1.332e-02</td>
<td>1.522e-02</td>
</tr>
<tr>
<td>4</td>
<td>1.318e-02</td>
<td>1.491e-02</td>
</tr>
<tr>
<td>6</td>
<td>1.317e-02</td>
<td>1.486e-02</td>
</tr>
</tbody>
</table>

Table 5.2: Thermal group error norms of DLP and mDLP

<table>
<thead>
<tr>
<th>order</th>
<th>DLP</th>
<th>mDLP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$L_1$</td>
<td>$L_2$</td>
</tr>
<tr>
<td>0</td>
<td>1.000e-01</td>
<td>1.108e-01</td>
</tr>
<tr>
<td>2</td>
<td>1.001e-01</td>
<td>1.108e-01</td>
</tr>
<tr>
<td>4</td>
<td>9.931e-02</td>
<td>1.092e-01</td>
</tr>
<tr>
<td>6</td>
<td>9.890e-02</td>
<td>1.086e-01</td>
</tr>
</tbody>
</table>

For simplicity, we consider an angular flux incident on the $xy$ plane, so that

$$r_l = \int_{\hat{\Omega} \geq 1} W(\Omega) \Gamma_l(\Omega) \psi(\Omega) d\Omega$$

$$= \int_0^{2\pi} d\phi \int_0^{\pi/2} W(\theta, \phi) \psi(\theta, \phi) d\theta \sin \theta,$$

where $\theta$ is the polar angle with respect to the $z$ axis, and $\phi$ is the azimuth. For a flux incident on any other surface, we can obtain this representation by a change of variables.

One approach for expanding $\psi(\theta, \phi)$ would be to use spherical harmonics scaled over the half range as a basis. An alternative approach, and the one used here, is to separate the polar and azimuthal dependence. In this case, we assume the form

$$\psi(\theta, \phi) = \sum_{m=0}^{\infty} A_m(\phi) \sum_{l=0}^{\infty} P_l(\theta) r_{lm},$$

86
where $A_m(\phi)$ and $P_1(\theta)$ satisfy

$$
\int_0^{2\pi} W_A(\phi) A_i(\phi) A_j(\phi) d\phi = a_i \delta_{ij} \quad (5.11)
$$

and

$$
\int_0^{\pi/2} W_p(\phi) P_i(\theta) P_j(\theta) \sin \theta d\theta = b_i \delta_{ij}. \quad (5.12)
$$

Then

$$
r_{im} = \int_0^{2\pi} \int_0^{\pi/2} W_A(\phi) W_p(\theta) A_i(\phi) P_m(\theta) \psi(\phi, \theta) d\phi \sin \theta d\theta. \quad (5.13)
$$

### 5.3.1 Conservative Expansions

In reactor simulations, low order expansions in angle work well when the flux is relatively isotropic (i.e. close to uniform in angle). Ideally, a zeroth order expansion in angle should yield an isotropic flux. A second desideratum would be an expansion whose zeroth order conserves the particle flow into a surface as measured by the partial current. Formally, this requires

$$
J = r_0 = \int_0^{2\pi} \int_0^{\pi/2} W_A(\phi) W_p(\theta) A_0(\phi) P_0(\theta) \psi(\phi, \theta) d\phi \sin \theta d\theta, \quad (5.14)
$$

where $J$ is the partial current defined

$$
J = \int_0^{2\pi} \int_0^{\pi/2} \hat{n} \cdot \hat{\Omega} \psi(\theta, \phi) d\phi \sin \theta d\theta. \quad (5.15)
$$

We can simplify notation somewhat by introducing $\xi = \cos \theta$. Then we recognize $\hat{n} \cdot \hat{\Omega} = \xi$, and therefore

$$
J = \int_0^{2\pi} \int_0^1 \xi \psi(\xi, \phi) d\phi d\xi. \quad (5.16)
$$

By conserving currents, particle balance is achieved, and reaction rates should, in theory, be better preserved. Aside from their physical soundness, these criteria would also make coupling of higher (e.g. transport) and lower order (e.g. diffusion) methods more straightforward.
These criteria were first defined by Zhang and Rahnema, who reviewed and extended some of the earlier work on response matrix methods [80]. As they pointed out, the work of Mosher [46] used an expansion that did yield an isotropic flux. Mosher used the discrete ordinates method, in which the angular flux is known at \( N \) discrete angles. In one dimension, the azimuthal-dependence vanishes, and the expansion of Eq. 5.13 becomes

\[
\psi(\xi_i) = \sum_{j=0}^{N-1} r_j P_j(\xi_i)
\]  

(5.17)

where

\[
r_j = \sum_i \omega_i W(\xi_i) P_j(\xi_i) \psi(\xi_i),
\]

(5.18)

and \( \omega_i \) is the quadrature weight such that

\[
\int_{-1}^{1} f(\xi) d\xi \approx \sum_{i=0}^{N-1} \omega_i f(\xi_i).
\]

(5.19)

Mosher used the discrete Legendre polynomials (DLPs) as the orthogonal basis in space in angle and a Gauss-Legendre quadrature in 1-D and Abu-Shumay's product quadrature in 2-D. The orthogonal DLPs are subject to the unity-weighted orthogonality constraint

\[
\sum_i P_{l}^{DLP}(i) P_{m}^{DLP}(i) = d_l \delta_{lm}.
\]

(5.20)

While the DLPs do produce an isotropic flux, they fail in general to conserve the current. For example, consider the \( S_8 \) approximation. The incident flux has four components, for which the DLPs of orders zero and one are

\[
P_0^{DLP} = [1, 1, 1, 1]
\]

\[
P_1^{DLP} = \frac{1}{3} [3, 1, -1, -3].
\]

(5.21)
Consider an arbitrary incident flux, $\psi(\mu_i) = [\psi_0, \psi_1, \psi_2, \psi_3]$ with associated abscissa $\xi_i$ and weights $\omega_i$. The actual partial current is

$$J = \sum_i \omega_i \xi_i \psi_i,$$  \hspace{1cm} (5.22)

while the zeroth order expansion yields

$$r_0 = \sum_{i=0}^3 \omega_i \mu_{0,i} \psi_i = \sum_{i=0}^3 \omega_i \psi_i = J.$$  \hspace{1cm} (5.23)

In the later work of Forget [22], continuous Legendre polynomials were used to represent the angular current, $\mathbf{j}(\Omega) = \Omega \psi(\Omega)$. The zeroth order expansion can then be written

$$r_0 = \int_0^{2\pi} \int_0^1 \hat{n} \cdot \mathbf{j}(\xi, \phi) d\phi d\xi = \int_0^{2\pi} \int_0^1 \xi \psi(\xi, \phi) d\phi d\xi,$$  \hspace{1cm} (5.24)

which is indeed the partial current. However, by setting $|\mathbf{j}| = \xi \psi(\xi, \phi) = 1$, one has a distinctly anisotropic flux $\psi = \frac{1}{\xi}$. Because many reactor systems are relatively isotropic, it can take a high expansion order to obtain an accurate flux expansion in this basis.

Zhang and Rahnema define a basis to satisfy the constraints they proposed [80]. They recognized that to produce an isotropic flux required a constant zeroth order basis function $\Gamma_0$ and an expansion in the angular flux (as opposed to the current). Moreover, to preserve the partial current, they specify that the basis should satisfy the orthogonality constraint

$$\int_0^{2\pi} \int_0^1 \xi A_m(\phi) A_{m'}(\phi) P_l(\xi) P_{l'}(\xi) d\phi d\xi = c_{lm} \delta_{ll'} \delta_{mm'},$$  \hspace{1cm} (5.25)

so that the response coefficients are defined

$$r_{lm} = \int_0^{2\pi} \int_0^1 \xi A_l(\phi) P_m(\xi) \psi(\xi, \phi) d\phi d\xi.$$  \hspace{1cm} (5.26)
In other words, they require orthogonality with respect to the weighting function 
\[ W(\xi)W(\phi) = \xi. \]
They suggest use of the Legendre polynomials and the Chebyshev polynomials of the second kind to construct basis functions of the form 
\[ \Gamma_{lm}(\Omega) = U_l(\cos(\theta))P_m(\cos(\phi)). \] (5.27)

The functions \( U_l(x) \) satisfy 
\[ \int_{-1}^{1} U_l(x)U_l'(x)\sqrt{1-x^2}dx = \frac{\pi}{2} \delta_{ll'}. \] (5.28)

However, they employed a somewhat different geometrical picture from the one used here. Instead of a flux incident on the \( xy \) plane, they consider the \( xz \) plane so that the integration bounds change. In this case, the \( \vec{n} \cdot \hat{\Omega} \) becomes \( \sin(\theta)\sin(\phi) \). Then the coefficients are expressed as 
\[ r_{lm} = \int_{0}^{\pi} \int_{0}^{\pi} \sin(\theta)\sin(\phi)U_l(\cos(\theta))P_m(\cos(\phi))\psi(\theta, \phi)d\phi\sin\theta d\theta. \] (5.29)

Let \( \omega = \cos(\phi) \) and \( \frac{d\omega}{\sin(\phi)} = d\phi \). Similarly, let \( \xi = \cos(\theta) \) as above. Then 
\[ r_{lm} = \int_{-1}^{1} \int_{-1}^{1} \sqrt{1-\xi^2}U_l(\xi)P_m(\omega)\psi(\xi, \omega)d\omega d\xi. \] (5.30)

This is indeed an orthogonal basis, since the \( P_l(x) \) are orthogonal over \( x \in [-1, 1] \) with unity weight, and the \( U_l(x) \) are orthogonal over the same range with weight \( \sqrt{1-x^2} \).

As an alternative, we can derive an expansion using the \( xy \) plane as the reference. To satisfy the constraint of Eq. 5.25, we require a polynomial in \( \phi \) to satisfy 
\[ \int_{0}^{2\pi} A_m(\phi)A_{m'}(\phi)d\phi = a_m\delta_{mm'}. \] (5.31)
The scaled Legendre polynomials satisfy this requirement. Additionally, we require a polynomial in \( \xi \) to satisfy

\[
\int_0^1 \mu P_i(\xi)P_j(\xi) d\xi = p_\ell \delta_{\ell\ell}'. \tag{5.32}
\]

Substituting \( \tau = 2\xi - 1 \), we have alternatively

\[
\int_{-1}^1 (\tau + 1) P_i(\tau)P_j(\tau) \tau = 2p_\ell \delta_{\ell\ell}'. \tag{5.33}
\]

The Jacobi polynomials \( P_i^{(a,b)}(x) \) satisfy the orthogonality constraint

\[
\int_{-1}^1 (1-x)^a(1+x)^b P_i^{(a,b)}(x)P_j^{(a,b)}(x) dx = q_\ell \delta_{\ell\ell}'. \tag{5.34}
\]

By inspection, we find the desired basis \( P_i(\tau) \) is the set of \( P_i^{(0,1)}(2\mu - 1) \) with arbitrary normalization. It turns out these functions were previously identified for the case of slab geometry, though they were not identified as a special case of the Jacobi polynomials \[81\]. The utility in this latter basis of Legendre and Jacobi polynomials for the discrete ordinates method is that the weighting function is at most linear, and so any number of quadratures (e.g. level symmetric) can be expected to integrate low orders exactly. Were the Chebyshev basis employed, a Gauss-Chebyshev quadrature in the polar dimension would be required, which complicates its application to horizontal surfaces in three dimensions.

### 5.3.2 Numerical Examples

In this section, we test various expansion schemes in illustrative 1-D and 2-D problems. For all problems, we are interested in the convergence of the eigenvalue as well as the maximum relative error of region fission densities.
<table>
<thead>
<tr>
<th>Material</th>
<th>$\Sigma_{t1}$</th>
<th>$\Sigma_{t2}$</th>
<th>$\nu \Sigma_{f1}$</th>
<th>$\nu \Sigma_{f2}$</th>
<th>$\Sigma_{t1-1}$</th>
<th>$\Sigma_{t2-1}$</th>
<th>$\Sigma_{t2-2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Water</td>
<td>0.1890</td>
<td>1.4633</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.1507</td>
<td>0.0380</td>
<td>1.4536</td>
</tr>
<tr>
<td>Fuel I</td>
<td>0.2263</td>
<td>1.0119</td>
<td>0.0067</td>
<td>0.1241</td>
<td>0.2006</td>
<td>0.0161</td>
<td>0.9355</td>
</tr>
<tr>
<td>Fuel II</td>
<td>0.2252</td>
<td>0.9915</td>
<td>0.0078</td>
<td>0.1542</td>
<td>0.1995</td>
<td>0.0156</td>
<td>0.9014</td>
</tr>
<tr>
<td>Fuel II + Gd</td>
<td>0.2173</td>
<td>1.0606</td>
<td>0.0056</td>
<td>0.0187</td>
<td>0.1902</td>
<td>0.0136</td>
<td>0.5733</td>
</tr>
</tbody>
</table>

Table 5.3: Two-group cross sections (cm$^{-1}$).

**Slab Reactor**

As first test cases, a set of three two-group BWR benchmarks in slab geometry are analyzed \[33\]. The cores represent increasing heterogeneity and are intended to test the accuracy of coarse mesh approximations. They are depicted schematically in Figure 5-6, and the associated cross section information is listed in Table 5.3. A double Gauss-Legendre with 16 quadrature points is used. This quadrature exactly integrates polynomials of up to 15th order in the half space. Each assembly represents a coarse mesh.

For expansions in the polar angle, the DLPs and and Legendre polynomials are used to expand both the angular flux and current. The Jacobi polynomials are also used to expand the angular flux. In all cases, expansions through 7th order were applied, so in all cases we should (and do) obtain the reference solution to within the tolerances employed.

Figure 5-7 shows the convergence of the eigenvalue and fission densities with polar order. The behavior of the error with expansion is qualitatively similar for all three cores. Notably, the Jacobi expansion yields the best zeroth order results of all the expansions. For example, the maximum relative assembly fission density error for core 3 using Jacobi(0) is roughly 4%, whereas all other expansions are close to 10%. Moreover, Jacobi also yields the lowest error for first order expansions, though its improvement diminishes from core 1 through core 3. This makes some sense: the goal in using the Jacobi expansion is to enforce a zeroth order isotropic flux and preserve the partial current. For the increasingly anisotropic cores, this combination is less relevant.
2-D Reflected Assembly

While a one dimensional problem affords us a means to assess all the expansion schemes to full convergence (via exact integration of the basis sets by the quadrature used), in practice, two and three dimensional problems are of most interest. As a test problem, we consider the four by four quarter assembly depicted in Figure 5-8 subject to reflective boundary conditions. The plot values correspond to the materials of Table 5.3. We use Abu-Shumays' compatible quadruple range quadrature with 8 azimuthal and polar angles per quadrant for 256 total angles. This quadrature exactly integrates the $U_i$ and $P_m$ functions through fourth order. In this problem, the pins represent coarse meshes.

Three basis sets are employed. Each set again uses the discrete delta function in energy, equivalent to a full multigroup representation. In space, a DLP basis is used throughout. In angle, we employ (DLP, DLP), (Legendre, Legendre), and (Chebyshev, Legendre) basis sets in the (polar, azimuth) coordinates. In all cases, a maximum angular expansion order is defined that bounds the sum of the azimuthal and polar expansion orders; this means that a full representation of the phase space is never produced, though in practice, such a representation is never the goal. Note, an essentially exact spatial representation is used so that the effect of the angular expansion is isolated.
(a) Core 1: $k$ error.

(b) Core 1: fission density error.

(c) Core 2: $k$ error.

(d) Core 2: fission density error.

(e) Core 3: $k$ error.

(f) Core 3: fission density error.

Figure 5-7: Absolute relative error in eigenvalue and maximum absolute relative error in fission density as a function of polar order for several expansions.
The eigenvalue and fission density errors are shown as a function of the total angular order (through 4th order) in Figure 5-9. The Chebyshev basis performs the best, achieving roughly 1% average (2% max) error in pin fission densities. By first order, it achieves a sub 1% maximum error in pin fission densities, which is a common metric in reactor analysis methods. The DLP expansions require a higher order expansion to achieve this (assuming the second order dip for DLP-J is extraneous), and the DLP-J expansion exhibits rather erratic convergence behavior. The accuracies observed for the DLP-ψ expansion are consistent with those listed by Mosher. The continuous Legendre expansions seem to diverge at second order, which is likely due to incorrect integration by the quadrature used. To summarize, the Chebyshev basis achieves errors that are roughly a factor of two better for the average fission density errors and even more on the maximum fission density errors than the DLP expansion.
Figure 5-9: Absolute relative error in eigenvalue, and average and maximum absolute relative error in fission density as a function of angle order for several expansions.
5.4 Energy Expansions

While the discretization and expansion in space and angle is relatively straightforward in principle (despite rather tedious details in the implementation), the treatment of the energy variable for multigroup problems can be very trivial. Previous work has used what amounts to a discrete Dirac-$\delta$ basis in the energy groups, which leads to full consistency between the local transport solutions and the global response matrix solution with respect to energy. However, it may be valuable to explore other bases for expanding the energy dependence for the purposes of coupling various methods or simply to reduce computational expense.

5.4.1 Mixed Group/Order Expansions

For certain models, the energy fidelity may not need to be uniform. Consequently, it may unnecessary to communicate $G$ degrees of freedom between two assemblies when perhaps just $G/2$ would suffice. One way to implement this is to adapt the mixed-energy technique of Forget and Zhu [23]. In that work, the discrete generalized multigroup (DGM) method was used to allow coupling between fine group regions and coarse group regions. DGM represents the fine group spectrum with a set of coarse energy groups each having a subset of the fine groups. Within each coarse group, the fine group spectrum is expanded in a complete basis so that the number of degrees of freedom is conserved, if desired. However, the fine group information within a coarse group can be neglected when transferring to a region of less importance.

Consider the trivial three region slab illustrated in Figure 5-10. Region 1 requires a full fine group treatment, which in this case consists of four energy groups. Contrarily, Region 3 can be represented with a coarse group approximation, in this case two groups. Consequently, Region 2 must act as a buffer. On its left, a full fine group representation is communicated. On its right, only the two degrees of freedom are communicated.
<table>
<thead>
<tr>
<th>Region 1</th>
<th>Region 2</th>
<th>Region 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>fine group structure needs fine resolution</td>
<td>fine group structure does not need fine resolution</td>
<td>coarse group structure does not need fine resolution</td>
</tr>
</tbody>
</table>

Figure 5-10: Three region, mixed energy slab.

Assuming that the two groups of Region 3 each contain exactly two of the fine groups used within Region 1, a potential basis set for the 1-to-2 coupling is

\[
P^{1\rightarrow 2}_0 = \frac{1}{2} [1,1,0,0]
\]

\[
P^{1\rightarrow 2}_1 = \frac{1}{2} [0,0,1,1]
\]

\[
P^{1\rightarrow 2}_2 = \frac{1}{2} [1,-1,0,0]
\]

\[
P^{1\rightarrow 2}_3 = \frac{1}{2} [0,0,1,-1],
\]

while for the 2-to-3 coupling, a corresponding truncated set is

\[
P^{2\rightarrow 3}_0 = \frac{1}{2} [1,1,0,0]
\]

\[
P^{2\rightarrow 3}_1 = \frac{1}{2} [0,0,1,1].
\]

The two degrees of freedom passing from Region 2 to Region 3 can be represented in the latter region using the simple basis

\[
P^3_0 = [1,0]
\]

\[
P^3_0 = [0,1].
\]

This approach is not implemented, but there is no reason to believe it would not yield results at least as encouraging as the scoping results of Forget and Zhu [23]. Such an approach is open to much further investigation, particularly with respect to
the coarse group structure and the basis used for low-to-high mapping. A very similar study is the subject of ongoing work for improving DGM.

5.4.2 Low Order Energy Representations

Most recent work on response matrix methods has used relatively coarse multigroup libraries, ranging from as few as two groups [80] or three [34] to the seven group data of the C5G7 library [22]. Since the number of response calculations required to represent a region is proportional to the number of groups, moving beyond the relatively few groups studied in previous work is an extremely difficult task with respect to computational time and memory requirements.

Discrete Energy Bases

One possible alternative is to retain a full fine group treatment but limit the energy degrees of freedom at a boundary to a subset. Of course, doing this with a straightforward basis (e.g. the discrete Legendre polynomials or the discrete cosine transform) that has been used in DGM is not likely to yield good results. However, if a basis can be constructed that contains at least some of the physics a priori, then a truncated basis may be useful.

As an example, suppose a global problem consists of nodes representing fuel assemblies. In traditional lattice physics, a single assembly is analyzed subject to reflecting boundary conditions. To obtain few group data from the fine group library, the infinite medium spectrum, possibly computed via a $B_N$ method, is used to weight the fine group over the desired coarse group energy bins. While the fine group spectrum obtained in this matter is not exact, it does often do a remarkably good job, as evidenced by the use of the approach for decades.

We can apply a similar approach within the response matrix method. We take as a given that each node (assembly, subassembly, or even pin) is a fully-specified transport problem. By solving the eigenvalue problem (or better yet, the $B_0$ problem) using the fine group data, we can obtain a representative critical spectrum for each node.
Then, using the idea of spectral mapping suggested by Forget [22], we can transform a standard basis into one in which the zeroth order basis function is the representative spectrum. In that work, higher order members are simply multiplied pointwise by the spectrum; however, this cannot in general yield an orthogonal basis. To correct the basis, the Gram-Schmidt process can be used to orthogonalize successive members against the zeroth order function.

**Discontinuous Energy Bases**

While a single spectrum (or potentially several spectra) can be used for low order energy expansions, a better approach might be to avoid averaging altogether. Consider the case of a UO$_2$ bundle adjacent to a MOX bundle. The thermal flux in MOX is relatively low, owing to large thermal absorption cross sections. Hence, at least at the boundary, the relative importance of a thermal neutron going from UO$_2$ to MOX is generally less than that from MOX to UO$_2$. Consequently, representing the spectrum of neutrons leaving the UO$_2$ in a way that captures the importance of neutrons in MOX, and likewise, representing the spectrum leaving MOX to capture the importance of neutrons entering UO$_2$ may be desirable. A constant number of degrees of freedom can be retained, but they represent something different in each direction. As a result, the current becomes *discontinuous* by construction. Because this approach requires nontrivial extensions to Serment, implementation is subject to future work.

**5.4.3 A Numerical Example**

To illustrate application of a truncated energy basis, consider the simple 1-D “pin” lattice consisting of UO$_2$ and MOX fuel, depicted in Figure 5-11. Fuel pins are 1.08 cm thick and arranged in a 1.28 cm pitch. Cladding was lumped into the fuel during homogenization from a small 2-D pin model used to produce the 44 group cross sections. All transport calculations were performed using a 16 angle DP$_N$ quadrature and a DD discretization. All response matrix calculations were performed using a first order angular expansion in the Jacobi polynomial basis. While this is a very simple model, it
serves to identify how critical selection of an energy basis is for use in low order energy expansions.

The critical spectra for the reflected lattice and the reflected UO$_2$ and MOX pins are shown in Figure 5-12. As expected, the MOX thermal spectrum is significantly smaller than that of UO$_2$, and this difference is likely to have considerable impact near the boundary between fuel types. The lattice-averaged spectrum falls somewhere between the fuel spectra.

To assess expansions in energy, the DLP basis is compared to two modified DLP bases that use the critical mode for the zeroth order member. The first (mDLP-1) is
defined by multiplying higher order modes pointwise with the spectrum, followed by orthogonalization. This approach is closest to the spectral mapping proposed by Forget [22], adding only the orthogonalization of the higher order functions. A slight variant (mDLP-2) skips the pointwise mapping of higher order members and instead orthogonalizes the original DLP members against the modified zeroth order. Each basis is depicted in Figure 5-13 through third order. The modified bases shown use the lattice-averaged spectrum for the zeroth order function. While mDLP-1 and mDLP-2 seem at first to be very close, the resulting bases differ significantly. Even so, they perform very similarly for the problem studied.

Each basis was used for expansions of orders $0 \leq o \leq 43$, and the results were compared to a reference solution based on a complete energy expansion. Figures 5-14 and 5-15 provide the relative absolute error in the eigenvalue and maximum absolute relative error of the pin fission rate as a function of order for each basis. Only the first 42 orders are plotted, since the next moment completes the expansion and recovers the reference values to within the tolerances used.

A striking feature of Figure 5-14 is the poor performance of the DLP basis relative to the modified bases. While the DLP expansion yields a nearly monotonically converging eigenvalue, the associated error is greater than 1% for $o \leq 35$. This is somewhat intuitive: reactor spectra are complicated functions of energy, and approximating with low order polynomials cannot be expected to conserve neutrons very well. For most orders, the mDLP-1 bases outperform the mDLP-2 bases, which implies the point-wise multiplication of the original basis members with the representative spectrum somehow imparts even more “physics” than the spectrum alone. Of the representative spectra used for mDLP-1, the lattice spectrum outperforms the fuel spectra slightly for $o < 20$; for higher orders, no clear advantage is observed. For $o > 5$, mDLP-1 with the lattice spectrum yields eigenvalue errors on the order of 100's of pcm or less. While this is not outstanding performance, it is encouraging that balance can be nearly recovered with so few energy degrees of freedom.

The more important metric is the error in pin fission rates. The basic trends of Figure 5-15 are similar to those of Figure 5-14. The DLP basis performs quite poorly,
achieving sub-1% pin errors at roughly the same order as sub-1% eigenvalues are achieved. The mDLP-1 bases again are observed to have a fairly consistent advantage over the mDLP-2 bases. With either the lattice or UO\textsubscript{2} spectra, the mDLP-2 basis yields pin errors below 2% for \( o > 3 \) except for \( o = 13 \), and sub-1% errors are achieved for \( o > 14 \).

These results are quite encouraging, as they suggest a simple and inexpensive modification of a standard basis may allow a full multigroup transport solution to be approximated using roughly \( 2/3 \) fewer energy degrees of freedom. In practice, this means \( 2/3 \) fewer responses need be generated and nearly \( 8/9 \) less response function storage need be allocated (since the most expensive data is the block-diagonally dense \( \mathbf{R} \), storage of which goes as the square of the degrees of freedom for a node).

There may exist similarly simple basis modifications that would yield greater or more consistent improvement. These might include incorporating spectral modes beyond the fundamental mode used for the zeroth order member. Using the lattice-averaged spectrum, generally unavailable, yielded just slightly better results than using the UO\textsubscript{2} spectrum. Were the first harmonic of the UO\textsubscript{2} problem used—perhaps to generate the second basis member—the resulting basis' lowest order members might provide a highly accurate subspace from which to draw the global solution.

5.5 Summary

In this chapter, bases for all three phase space variable were developed. After comparing several rather pedestrian bases to more complex bases in each variable, the results indicate a rather simple trend: use a basis that somehow captures the physics. For space and energy, this implies using basis functions that look a lot like the result we ultimately expect, while for angle, this implies conserving the physics by enforcing balance. Using bases that capture the physics should lead to models requiring lower orders (and hence fewer responses) to achieve a desired accuracy.
Figure 5-13: Comparison of DLP and modified DLP energy bases. Each basis is orthonormal.
Figure 5-14: Comparison of eigenvalue errors for various energy bases.

Figure 5-15: Comparison of pin fission rate errors for various energy bases.
Chapter 6

Analysis of a Fixed Point Iteration

In this chapter, the fixed point iteration typically used to solve the eigenvalue response matrix equations is presented and analyzed. In particular, the convergence properties of the algorithm are investigated using a simple, analytic diffusion model. Application of the model is facilitated through several numerical examples.

6.1 Fixed Point Iteration

Recall the global eigenvalue problem, expressed mathematically as

\[ T\phi(\bar{\rho}) = \frac{1}{k} F\phi(\bar{\rho}). \]  \hspace{1cm} (1.1)

The \( k \)-eigenvalue can be interpreted physically as the ratio of neutrons produced in one generation to the previous generation. This can alternatively be viewed as the ratio of gains to losses in a given generation. Applying this interpretation to the response matrix formalism, we can update \( k \) via the process

\[ k_{n+1} = \frac{F(k_n)J_-}{A(k_n)J_- + L(k_n)J_-}, \]  \hspace{1cm} (6.1)

where \( F(k)J_- \) is the global fission rate, \( A(k)J_- \) is the global absorption rate, and \( L(k)J_- \) is the total leakage rate from global boundaries.
Equation 6.1 represents a Picard (fixed point) iteration for the $k$-eigenvalue. Our goal is to analyze this iteration. Our previous experience suggests the method converges very quickly with respect to the number of $k$ updates required, and it is useful to examine what influences this convergence.

We begin with a short review of fixed point iteration convergence theory [17]. Consider the fixed point iteration

$$x_{n+1} = f(x_n),$$

with some fixed point $x^*$, i.e. $x^* = f(x^*)$. The sequence defined by Eq. 6.2 converges if and only if

$$\lim_{n \to \infty} x_n - x^* = 0.$$  

(6.3)

Suppose we compute the Taylor expansion of $f(x_n)$ about the fixed point, yielding

$$f(x_n) = f(x^*) + \Delta f'(x^*) + \frac{1}{2} \Delta^2 f''(x^*) + \mathcal{O}(\Delta^3),$$

(6.4)

where $\Delta = x_n - x^*$. If $f'(x^*)$ does not vanish, then

$$f(x_n) - f(x^*) = x_{n+1} - x^* = (x_n - x^*)f'(x^*) + \mathcal{O}(\Delta^2)$$

(6.5)

or

$$\lim_{n \to \infty} \frac{|x_{n+1} - x^*|}{|x_n - x^*|} = |f'(x^*)|. \quad (6.6)$$

Contrarily, if $f'(x^*)$ vanishes, then

$$\lim_{n \to \infty} \frac{|x_{n+1} - x^*|}{|x_n - x^*|^2} = \left| \frac{1}{2} f''(x^*) \right|. \quad (6.7)$$

In general, we can write

$$\lim_{n \to \infty} \frac{|x_{n+1} - x^*|}{|x_n - x^*|^p} = C,$$

(6.8)

where $p$ is the order of convergence and $C$ is the asymptotic error constant. We note that for the iteration to converge requires $C < 1$. The Picard iteration defined by Eq. 6.1
converges linearly (with $p = 1$), and we seek to quantify analytically the asymptotic error constant.

### 6.2 Analytic Model

In this study, we employ the monoenergetic diffusion approximation in a homogeneous slab. In this case, Eq. 1.1 becomes

$$-D \frac{d^2 \phi}{dx^2} + \Sigma_a \phi(x) = \frac{\nu \Sigma_f}{k} \phi(x), \quad (6.9)$$

subject to zero incident current (vacuum) boundary conditions

$$J_-(0) = J_-(L) = 0, \quad (6.10)$$

for a slab of width $L$, and where the notation is standard. The partial current in either direction is defined

$$J_{\text{right/left}}(x) = \frac{\phi(x)}{4} \pm \frac{D d\phi}{2 dx}, \quad (6.11)$$

and so the left boundary condition is such that $J_-(0) = J_{\text{right}}(0) = 0$, and similarly, at the right, $J_-(L) = J_{\text{left}}(L) = 0$.

To simplify the analysis, we nondimensionalize the spatial dimension by substituting $\bar{x} = \frac{x}{\Sigma_i}$, $\bar{\Sigma}_t = 1$, $\bar{\Sigma}_a = \frac{\Sigma_a}{\Sigma_i}$, $\bar{\Sigma}_f = \frac{\Sigma_f}{\Sigma_i}$, and $\bar{D} = D \Sigma_i = \frac{1}{3 \Sigma_i} \Sigma_i = \frac{1}{3}$. However, we drop the tilde notation in subsequent expressions.

In the one group diffusion approximation, the local problem defined in Eq. 1.2 becomes

$$-\frac{1}{3} \frac{d^2 \phi_i}{dx_i^2} + \Sigma_a \phi(x_i) = \frac{\nu \Sigma_f}{k} \phi(x_i), \quad (6.12)$$

in subvolume $\mathcal{V}_i$ of width $\Delta$, subject to the incident current condition on the left boundary at $x_{i_1}$

$$J_{i-}(x_{i_1}) = \frac{\phi(x_{i_1})}{4} - \frac{1}{6} \frac{d\phi}{dx_i} \bigg|_{x_i=x_{i_1}} = 1 \quad (6.13)$$
with vacuum on the right. We rewrite Eq. 6.12 as

\[ \frac{d^2 \phi}{dx_i^2} + B^2 \phi(x_i) = 0, \quad (6.14) \]

where the buckling \( B \) is defined by

\[ B^2 = 3 \left( \frac{\nu \Sigma_f}{k} - \Sigma_u \right). \quad (6.15) \]

Because the eigenvalue \( k \) of a finite slab will remain below the eigenvalue of an infinite slab, i.e. \( k < k_\infty = \frac{\nu \Sigma_f}{\Sigma_u} \), we need only deal with the case of \( B^2 > 0 \), for which the general solution to Eq. 6.12 is

\[ \phi(x_i) = a \cos(Bx_i) + b \sin(Bx_i). \quad (6.16) \]

For the given boundary conditions, one can show that

\[ R = \frac{J_+(x_{i_1})}{J_-(x_{i_1})} = \frac{\sin(B\Delta)(9 + 4B^2)}{12B \cos(B\Delta) + (9 - 4B^2) \sin(B\Delta)} \quad (6.17) \]

and

\[ T = \frac{J_+(x_{i_1} + \Delta)}{J_-(x_{i_1})} = \frac{12B}{12B \cos(B\Delta) + (9 - 4B^2) \sin(B\Delta)}, \quad (6.18) \]

where \( R \) and \( T \) are referred to as the reflection and transmission coefficients, respectively.

Using these expressions, the nodal balance equation can be written

\[ \hat{\vec{j}}_+ = \begin{bmatrix} j_{i1+} \\ j_{i2+} \end{bmatrix} = \begin{bmatrix} R & T \\ T & R \end{bmatrix} \begin{bmatrix} j_{i1-} \\ j_{i2-} \end{bmatrix}, \]

where \( j_{is} \) is the outgoing/incident partial current on surface \( s \) of node \( i \). Here, \( s = 1 \) indicates the left surface, and \( s = 2 \) the right. Note that this formulation assumes symmetry of the node. For a sequence of identical adjacent slabs, the corresponding
operator \( MR \) is given by

\[
MR = \begin{bmatrix}
0 & 0 & R & T \\
0 & T & R & 0 \\
T & R & 0 & 0 \\
& & \ddots & \ddots \\
0 & R & T & 0 \\
T & R & 0 & 0 \\
& & & 0
\end{bmatrix}.
\] (6.19)

For a node, the fission and absorption rate operators are defined as

\[
F = \nu \sum_j \Phi
\] (6.20)

and

\[
A = \sum_\alpha \Phi,
\] (6.21)

where the integrated flux response is defined as

\[
\Phi = \int_{x_{i1}}^{x_{i1}+\Delta} \phi(x_i)dx = \frac{24(B - \cos(B\Delta)) + 36 \sin(B\Delta)}{B\cos(B\Delta)(9 + 4B^2) + 9 - 4B^2}.
\] (6.22)

Noting that each node is identical, the Picard iteration for \( k \) can then be written

\[
k = \frac{F(\sum_{i=1}^N j_{i1} + \sum_{i=1}^N j_{i2})}{A(\sum_{i=1}^N j_{i1} + \sum_{i=1}^N j_{i2}) + T j_{12} + T j_{N1}},
\] (6.23)

where \( N \) is the number of nodes. Since the slab is homogeneous, the corresponding solution is symmetric about its center. Thus, we need only consider the current incident
on one side of each node, here the left, so that

\[ k = \frac{F \sum_{i=1}^{N} j_{i1}}{A \sum_{i=1}^{N} j_{i1} + T j_{N1}}. \]  

(6.24)

To simplify the analysis to come, we note that eigenvectors are defined up to a multiplicative constant, and so we can normalize such that \( j_{N1} = 1 \). Furthermore, we define

\[ J = \sum_{i=1}^{N} j_{i1}, \]  

(6.25)

yielding the concise form

\[ f(k) = \frac{F}{A + \frac{T}{J}}. \]  

(6.26)

Substituting Eqs. 6.18, 6.20, and 6.21 into Eq. 6.26 yields the simplified form

\[ f(k) = \frac{\nu \Sigma_f}{B^2} \frac{1}{\Sigma_a + \frac{3 - 3 \cos(B\Delta) + 2B \sin(B\Delta)}{\Sigma_a + G(B)}} \]  

(6.27)

From Eq. 6.15 we have \( k = \nu \Sigma_f / (\Sigma_a + B^2/3) \). Hence, in the limit \( k \to k^* \), we must have \( G(B) = B^2/3 \) and

\[ J(B) = \frac{3}{3 - 3 \cos(B\Delta) + 2B \sin(B\Delta)}. \]  

(6.28)

## 6.3 Convergence of ERMM

Our analysis is divided into two components. The first aims to probe the asymptotic error constant of the Picard iteration of Eq. 6.1 as a function of the slab width and node width. In Section 6.4, the theoretical results of this section are verified numerically and used to interpret convergence of more complex problems.

The exact solution to Eq. 6.27 requires two components: the \( k \)-eigenvalue and the current eigenvector, \( J \), or, more specifically, the sum of the rightward partial currents,
It can be shown that $k^*$ satisfies the transcendental equation

$$\tan BL = \frac{12B}{4B^2 - 9}, \quad (6.29)$$

which can be solved numerically using Eq. 6.15. The resulting $k^*$ and Eq. 6.28 provide the current sum, $J$.

To compute the asymptotic error constant, we take the derivative of $f(k)$ evaluated at $k^*$,

$$\frac{df}{dk} = \frac{df}{dB} \times \frac{dB}{dk}$$

$$= \left\{ -\frac{\nu \Sigma_f}{(\Sigma_a + G(B))^2} \frac{dG}{dB} \right\} \times \left\{ \frac{3 (\Sigma_a + B^2/3)^2}{2 B \nu \Sigma_f} \right\}$$

$$= \left\{ -\frac{\nu \Sigma_f}{(\Sigma_a + B^2/3)^2} \frac{dG}{dB} \right\} \times \left\{ \frac{3 (\Sigma_a + B^2/3)^2}{2 B \nu \Sigma_f} \right\}$$

$$= \frac{3}{2B} \times \frac{dG}{dB}, \quad (6.30)$$

where

$$\frac{dG}{dB} = \frac{2B}{3} - \frac{B^2(2 \cos(B\Delta)\Delta B + 2 \sin(B\Delta) + 3 \sin(B\Delta)\Delta)}{6 \sin(B\Delta)B + 9 - 9 \cos(B\Delta)}$$

$$- \frac{B^2}{9} \left( 2 \sin(B\Delta)B + 3 - 3 \cos(B\Delta) \right) \frac{dJ}{dB}. \quad (6.31)$$

From Eq. 6.30, we observe that for our simple monoenergetic diffusion approximation, the asymptotic error constant is independent of the cross sections $\nu \Sigma_f$ and $\Sigma_a$.

The remaining unspecified term is the derivative of the current sum, $\frac{dj}{dB}$. While the current vector can be computed directly from the flux solution of Eq. 6.9, the resulting expression does not contain the true dependence of the current vector on the eigenvalue $k$. However, given the current eigenvalue $\lambda$, it is possible to obtain the current vector directly by solving Eq. 1.13. By assuming $j_{1+} = 1$ and a symmetric solution, the resulting system leads to a recursive formula for $j_{1+}$ in terms of $R$, $T$, and $\lambda$.

112
For the simplest case of two nodes, the current vector is simply \((0 1 1 0)\), and because the nonzero elements are fixed, the derivative vanishes, and hence \(\frac{df}{db} = 0\), so

\[
\frac{df}{dk}_{k=k^*} = 1 - \frac{2 \cos(B\Delta)\Delta B^2 + 2 \sin(B\Delta)B + 3 \sin(B\Delta)B\Delta}{2}.
\] (6.32)

Because the expressions quickly become cumbersome with the number of nodes, we seek a different approach to analyze \(\frac{df}{db}\) for arbitrary numbers of nodes. Starting with the original response matrix equation for global balance,

\[
\text{MR}(k)J_\lambda = \lambda J_\lambda,
\] (1.13)

we define an operator \(A\) such that

\[
AJ_\lambda = J_\lambda,
\] (6.33)

where

\[
A = \begin{pmatrix}
I & 0 & 0 & 0 & 0 & \cdots \\
W & 0 & 0 & 0 & \cdots & 0 \\
0 & W & 0 & \cdots & 0 & 0 \\
0 & 0 & \cdots & 0 & 0 & 0 \\
0 & \cdots & 0 & W & 0 & 0 \\
\cdots & 0 & 0 & 0 & W & 0
\end{pmatrix}
\] (6.34)

and

\[
W = \begin{pmatrix}
\frac{T}{\lambda} & \frac{R}{\lambda} & \frac{R^2}{\lambda T} \\
-\frac{R}{\lambda} & \frac{\lambda}{T} & -\frac{R^2}{\lambda T}
\end{pmatrix}.
\] (6.35)

The utility of this form is that it allows the incident boundary current for a cell \(i\) to be related directly to the left global boundary condition via

\[
\begin{pmatrix}
j_{i1-} \\
j_{i2-}
\end{pmatrix} = W^{i-1} \begin{pmatrix}
j_{i1-} \\
j_{i2-}
\end{pmatrix},
\] (6.36)
from which the current sum \( J \) is found to be

\[
J = (1 \quad 1) \sum_{i=1}^{N} \begin{pmatrix} j_{1i} \\ j_{2i} \end{pmatrix} = (1 \quad 1) \sum_{i=1}^{N} W^{i-1} \begin{pmatrix} j_{1i} \\ j_{2i} \end{pmatrix}.
\] (6.37)

Furthermore, every matrix-vector multiplication can be replaced by multiplication with the eigenvalues of \( \lambda_1 \) and \( \lambda_2 \) of \( W \) to obtain

\[
W^i j_{1i} = \alpha_1 W^i \tilde{v}_1 + \alpha_2 W^i \tilde{v}_2 = \alpha_1 \tilde{v}_1 \lambda_1^i + \alpha_2 \tilde{v}_2 \lambda_2^i,
\] (6.38)

where the eigenvectors are defined

\[
\tilde{v}_{1/2} = \begin{pmatrix} 1 \\ \frac{\lambda_1/2 - T}{R} \end{pmatrix}.
\] (6.39)

Assuming a boundary condition of \( j_{1-} = (0 \quad 1)^T \), the multiplicative constants of the eigenvector expansion are found to be

\[
\alpha_1 = \frac{R}{\lambda(\lambda_1 - \lambda_2)} \quad \alpha_2 = -\alpha_1.
\] (6.40)

Now \( J \) can be defined in terms of the eigenvalues of \( W \), as well as \( \lambda, R \) and \( T \):

\[
J(B, \lambda) = \alpha_1 \left( \tilde{v}_1 \sum_{i=1}^{N} \lambda_1^{i-1} - \tilde{v}_2 \sum_{i=1}^{N} \lambda_2^{i-1} \right)
= \frac{R - T}{\lambda(\lambda_1 - \lambda_2)} \left( \sum_{i=1}^{N} \lambda_1^{i-1} - \lambda_2^{i-1} \right) + \left( \sum_{i=1}^{N} \lambda_1^i - \lambda_2^i \right).
\] (6.41)
At the solution \( k = k^* \), global neutron balance is achieved, and \( \lambda = 1 \) so that the eigenvalues of \( W \) simplify to

\[
\lambda_{1/2} = e^{\pm iB\Delta}.
\]  

(6.42)

Substituting Eq. 6.42 into Eq. 6.41 and using the geometric progression formula yields

\[
J(B) = \frac{3 - 3 \cos(BL) + 2B \sin(BL)}{3 - 3 \cos(B\Delta) + 2B \sin(B\Delta)}.
\]  

(6.43)

However, to determine \( \frac{df}{dk} \) requires we evaluate \( \frac{df}{db} \). Since \( J \) from Eq. 6.41 is a function of both \( B \) and \( \lambda \), the chain rule is used to obtain

\[
\frac{dJ}{dB} = \frac{\partial J}{\partial B} \bigg| _{\lambda=1} + \frac{\partial J}{\partial \lambda} \bigg| _{\lambda=1} \frac{d\lambda}{dB} \bigg| _{\lambda=1}.
\]  

(6.44)

The first term is found by differentiating Eq. 6.43 with respect to \( B \). For the second term, \( J \) as defined in Eq. 6.41 must be differentiated with respect to \( \lambda \) before the substitution \( \lambda = 1 \). All that remains is to determine \( \frac{df}{db} \).

To do this, the dot product of Eq. 1.13 with a vector of ones is performed, and solving for \( \lambda \) yields

\[
\lambda = \frac{1^TMR(k)J}{1^TJ} = \frac{1^TMR(k)AJ}{2J(B,\lambda)} = H(B,\lambda).
\]  

(6.45)

Since both \( H \) and \( J \) have implicit dependence on \( B \) and \( \lambda \), the chain rule is again used to obtain

\[
\frac{d\lambda}{dB} = \frac{\partial}{\partial B} \left( \frac{H}{2J} \right) + \frac{\partial}{\partial \lambda} \left( \frac{H}{2J} \right) \frac{d\lambda}{dB},
\]  

(6.46)
which can be shown to yield

\[
\frac{d \lambda}{d B} = \frac{\frac{\partial}{\partial B} \left( \frac{H}{2J} \right)}{1 - \frac{\partial}{\partial \lambda} \left( \frac{H}{2J} \right)}
\]

\[
= \frac{-2 \frac{\partial J}{\partial B} \bigg|_{\lambda=1} + \frac{\partial H}{\partial B} \bigg|_{\lambda=1}}{2J + 2 \frac{\partial J}{\partial \lambda} \bigg|_{\lambda=1} - \frac{\partial H}{\partial \lambda} \bigg|_{\lambda=1}}.
\]  

(6.47)

In the limit of small nodes, this reduces to

\[
\lim_{\Delta \to 0} \frac{d \lambda}{d B} = \frac{-12B \Delta(2BL \cos(BL) + (2 + 3L) \sin(BL))}{2B(4B^2 - 9)L \cos(BL) + (4B^2(3L - 2) - 9(2 + 3L)) \sin(BL)}.
\]  

(6.48)

Substitution of Eq. 6.48 into Eq. 6.44, the asymptotic error constant defined by Eq. 6.30 reduces to

\[
\lim_{\Delta \to 0} \left| \frac{df}{dk} \right|_{k=k^*} = \left| -\frac{B}{2 \Delta} \lim_{\Delta \to 0} \left( \frac{1}{J \frac{dJ}{dB}} \right) \right|
\]

\[
= \left| \frac{4B^2(6 - (9 + 4B^2)L)}{(9 + 4B^2)(-6 + (-9 + 4B^2)L)} \right|.
\]  

(6.49)

By making the approximation \( B \approx \frac{\pi}{L + \frac{3}{4}} \), which is valid for large \( L \) and accurate to within fractions of a percent for a 10 mean free path (mfp) slab, we find

\[
\lim_{\Delta \to 0} \left| \frac{df}{dk} \right|_{k=k^*} \approx \frac{4B^2(27\pi + 12\pi B^2 - 54B - 16B^3)}{(9 + 4B^2)(27\pi + 16B^3 - 18B - 12\pi B^2)}
\]

\[
\approx \frac{4}{9} \frac{B^2}{\pi} - \frac{16B^3}{27\pi} + O(B^4).
\]  

(6.50)

This demonstrates a general result: ERMM attains better convergence for problems with smaller buckling or lower leakage. This is in contrast to fine mesh eigenvalue solvers for which the high dominance ratio typical of low leakage cores leads to slow convergence. Furthermore, the ratio of the asymptotic error constants of two systems with different buckling is relatively constant and well-approximated by Eq. 6.50.
6.4 Numerical Studies

Ultimately, we want to understand what practical implications the observations above might have in numerical modeling of reactor systems. To do so, a systematic convergence study on several test cases is performed.

6.4.1 One Group Homogeneous Model

We begin with one group homogeneous models including 2-D diffusion and 1-D transport approximations. In the one group approximation, it is far easier to relate observed convergence rates with the convergence model described above.

The 1-D slab model is first verified numerically. Figure 6-1 shows the asymptotic error constant as a function of node size in mean free paths (mfps) for several slab widths. Interestingly, the curves show an optimal convergence for nodes of about 1 mfp in size. This corresponds to the function $\frac{df}{dk}$ crossing the axis at that point, and as observed below, this is not a feature common to more complicated problems. A more practical interpretation of the curves is that for all node sizes, the method converges uniformly, and for systems with lower leakage (smaller $B$), convergence is faster (going as $B^2$).

In 2-D, the one group diffusion exhibits similar behavior. Figure 6-2 shows the asymptotic error constant as a function of square node size in mfps for square cores of different extents subject to vacuum on all sides (highlighted by the shape markers). The currents are represented in space using a second order expansion, which proved sufficient to yield $k$-values within about 0.001% for all node sizes. The convergence is almost identical to the 1-D case, which is expected. For a box or cube of length $L$, the buckling satisfies (approximately) the conditions

$$B_{2d}^2 = 2\left(\frac{\pi}{L + \frac{4}{3}}\right)^2$$
and
$$B_{3d}^2 = 3\left(\frac{\pi}{L + \frac{4}{3}}\right)^2.$$  \hspace{1cm} (6.51)
Figure 6-1: Dependence of asymptotic error constants on \( \Delta \) for several slab widths.

Substituting this into Eq. 6.49, we find that

\[
\lim_{\Delta \to 0} \left. \frac{df}{dk} \right|_{k=k^*} \approx \frac{4B^2}{9} - \frac{16B^3}{27\sqrt{d}\pi} + \mathcal{O}(B^4),
\]

where \( d \) is the dimension. Hence, the effect of dimension is minimal, as observed.

Using transport based on the discrete ordinates method with a very high order angular expansion between nodes, a result qualitatively similar to the diffusion case is observed in Figure 6-2, highlighted by dashed lines. The general trend appears to be a quicker descent in the error constant for larger nodes, and a slightly larger asymptotic value. However, the amount by which the asymptotic value is larger is not yet understood. Using the homogeneous \( B_0 \) theory described in Appendix B, a buckling can be found for the given data based on the computed eigenvalue \( k \). Comparing the resulting \( B \) to the analytic value for diffusion yields a very small difference that does not explain the rather significant shift in \( \left| \frac{df}{dk} \right| \) using the model alone. This discrepancy will be subject to additional analysis.
Figure 6-2: Dependence of asymptotic error constants on node width $\Delta$. The solid line represents 1-D diffusion, and the markers represent 2-D diffusion. The dashed line corresponds to 1-D transport.

### 6.4.2 Two Group Homogeneous Model

To apply the one group model to multigroup problems, we need a single group buckling that captures the multigroup problem. The most straightforward approach is to perform a reference calculation for the multigroup problem and subsequently compute homogenized one group data with which to determine the one group buckling.

This approach was applied to a two group slab of varying widths, and the resulting convergence is shown in Figure 6-3 as dashed curves. The associated one group results are included as solid lines. The results indicate the convergence behavior of the two group problem is fundamentally different from the corresponding one group problem. The asymptotic behavior in two groups is apparently insensitive to the slab width, unlike the one group case.

It appears the convergence in multigroup problems is approximately described by the largest of the multigroup buckling coefficients, which are defined in Appendix A.
For the case of two groups with no upscatter, these buckling coefficients are defined by

\[ B^2 = \frac{\nu \Sigma_{f1}/k - \Sigma_{r1}}{2D_1} - \frac{\Sigma_{a2}}{2D_2} \]

\[ \pm \sqrt{\left( \frac{\nu \Sigma_{f1}/k - \Sigma_{r1}}{2D_1} - \frac{\Sigma_{a2}}{2D_2} \right)^2 + \frac{\nu \Sigma_{f2}\Sigma_{12}}{D_1D_2k^2}}. \]  

(6.53)

For the problems studied, the largest of these values is very insensitive to the \( k \) and hence the width of the slab, and for this reason, the convergence for all three slab widths is much more similar and much less favorable in two groups than in one group.

An interesting corollary to this observation is that a one group approximation may yield a simple and effective acceleration technique. Previous work has pointed out using low order space-angle approximations as initial guesses for higher order approximations works well [20]. Such approximations work simply because they require less computation; here, a lower order energy approximation suggests a problem with more favorable convergence properties in addition to a reduction in computation per iteration.

6.4.3 Applications to Heterogeneous Models

Here, we investigate the convergence of two heterogeneous cases. The first is the IAEA two group diffusion benchmark [6]. The second is an unreflected three group \( 32 \times 32 \) \( \text{UO}_2 \) pin assembly model in which the pins have explicit fuel and moderator regions. Again, the spatial variable is expanded using a second order basis, and for the assembly problem, a full transport approximation using second order expansions in the polar and azimuth angles is employed. These expansions yield solutions and convergence rates that are insensitive to the node size.

Figure 6-4 shows the convergence for the two group diffusion benchmark. Additionally, we have included three approximations based on one group homogenization. The first uses assembly-averaged cross sections with a spectrum-averaged reflector cross section, \( i.e. \) each assembly remains unique while the reflector region is reduced to a single material. The two material approximation uses a single average fuel cross sec-
Figure 6-3: Dependence of asymptotic error constants on node width \( \Delta \). The dashed line represents the two group case. The solid line represents the same problem homogenized to one group using the reference two group solution.

The results indicate that heterogeneity can have a very strong impact on convergence, as the one group problem with assembly-level heterogeneity has essentially the same convergence behavior as the original two group problem. Even homogenizing the entire fuel region but maintaining a distinct reflector only changes the magnitude but not the shape of the curve. Only when homogenized to a single material does the resulting convergence behave as the homogeneous model predicts.

Figure 6-5 shows the convergence for three group assembly model. Spectrum-averaging is used to define one group fuel and moderator data for the one group results. The one group model has very similar convergence behavior to the homogeneous model, and the three group model has the rather flat behavior exhibited by the homogeneous two group model. In this case, the assembly is just over 20 mfps
along a side, which puts the observed one group curve correctly between the 12.5 and 25 mfp transport curves of Figure 6-2.

6.5 Summary

In this chapter, we have developed a simple model for assessing the convergence behavior of the ERMM when implemented via fixed point iteration for $k$. The key result is that the method has rapid convergence, and in the worst case, that convergence appears insensitive to the node dimension. This is critical for effective use of the method in parallel computation schemes. For a given problem, the amount of parallelism achievable is governed in part by the smallest reasonable node into which the problem can be decomposed. In fine mesh schemes that connect decomposed domains via boundary fluxes, classical iteration techniques like Jacobi or Gauss-Seidel are employed, and the convergence of these methods degrades as the local problem becomes small. Our analysis demonstrates the ERMM does not have this issue, and that for heterogeneous
multigroup problems, the convergence of the ERMM is relatively insensitive to node size.

Additionally, the convergence of the ERMM was found to be sensitive to energy, and in some multigroup cases, the equivalent one group problem converges much more rapidly. This implies that for such problems, a one group treatment may yield an effective acceleration technique. This could be implemented either by a traditional homogenization procedure similar to what was used to generate the results or by employing a basis in energy such that a zeroth order energy term represents the integrated one group flux.

We have also identified several unanswered questions of a more theoretical nature. For example, it is not clear why in the one group model the convergence behaviors of diffusion- and transport-based responses differ as much as they do. It should be possible to extend the model to transport, at least for the case of isotropic incident fluxes, though it is not clear whether this would explain the behavior for the high order angu-
lar results observed. Additionally, an extension to two groups is also feasible, though initial work suggests the associated analysis becomes significantly more complex.
Chapter 7

Accelerating $\lambda$- and $k$-Convergence

In Chapter 6, we showed the eigenvalue response matrix equations lead naturally to a fixed point iteration similar in spirit to the traditional power method for $k$-eigenvalue problems. In this chapter, we examine efficient methods for solving the inner $\lambda$-eigenvalue problem within a fixed point iteration. Additionally, we seek alternative iteration schemes that lead to fast convergence of $k$ and thus minimize the number of response function evaluations needed. This is critical because for most problems, the computation of responses is by far the most time-intensive component of the process.

7.1 Solving the Inner Eigenvalue Problem

7.1.1 Power Iteration

In the fixed point iteration analyzed in Chapter 6, the current eigenvalue problem

$$ MR(k)J_\lambda = \lambda J_\lambda $$  \hspace{1cm} (1.13)

must be solved for global balance, after which $k$ can be updated. Historically, the method to solve Equation 1.13 amounts to simple power iteration. For a given $k^{(n)}$, the current vector is guessed and normalized, $MR(k^n)$ is applied to the guess, and the
resulting vector points closer to the dominant eigenvector of interest. The result is normalized, and the process repeats until converged.

Unfortunately, the asymptotic convergence rate to the dominant mode is equal to \( \ln(1/\rho) \), where the dominance ratio \( \rho \) is defined

\[
\rho = |\lambda_2|/|\lambda_1|, \tag{7.1}
\]

and the eigenvalues of \( \textbf{MR} \) satisfy \( \lambda > |\lambda_2| \geq |\lambda_i|, \forall i > 2 \). For many problems, \( \rho \) typically falls above 0.99, significantly larger than the dominance ratio associated with \( k \).

Due to the large \( \rho \), 1000's of iterations are required to reduce residual norms \( ||\textbf{MR}_- - \lambda \textbf{J}_-|| \) to within the tolerances used in our later analysis. Chebyshev acceleration was considered for accelerating convergence, but its utility is severely limited due to the nature of the eigenspectrum of \( \textbf{MR} \), a significant portion of which sits away from the real axis. Figure 7-1 shows the current eigenspectrum for the 2-D IAEA problem [6] for converged \( k \) using zeroth- and second-order DLP expansions, and \( 2 \times 2 \) coarse meshes per assembly; see Appendix C for more problem details. Superimposed on the graph is an ellipse bounding the spectrum, centered at the origin with semi-major axis equal to the second largest eigenvalue and semi-minor axis equal to the maximum modulus along the imaginary axis, \( \epsilon \). Were this spectrum completely real instead (so \( \epsilon = 0 \)), it can be shown [25] that the optimal asymptotic speedup via Chebyshev acceleration is close to 20 for this problem. On the contrary, for the true complex spectrum, the optimal speedup is limited to less than 2, and that assumes the dominance ratio and \( \epsilon \) are known a priori which is generally untrue.

Despite these theoretical limitations, recent work claims to have used Chebyshev acceleration successfully for solving the inner problem [80]. However, it seems likely its success would be highly problem-dependent and subject to significant tuning, two features we would ultimately choose to avoid.
7.1.2 Krylov Subspace Methods

As an alternative to the power method, we investigate Krylov subspace methods for the eigenvalue problem, the most well-known of which is the Arnoldi method. The Arnoldi method entails generation of the Krylov subspace defined by Eq. 2.71. The Arnoldi process is used as in GMRES to obtain the upper Hessenberg matrix $H \in \mathbb{R}^{n \times n}$. It turns out that the eigenvalues of $H$, called “Ritz values,” tend to be good estimates of eigenvalues of $A$, and given an eigenpair $(\lambda, y)$ of $H_m$, the Rayleigh-Ritz estimate of the corresponding eigenvector of $A$ is defined $x = V_m y$, and is called a “Ritz vector.”

The eigenpairs of $H$ are found via a dense eigensolver, such as the QR method. While this is done on a small system (since $m \ll n$), it is done repeatedly for increasing $m$ until some criterion is met. If $m$ becomes too large, the dense methods become too expensive. To circumvent this issue, the Arnoldi method must be restarted. In the explicitly restarted Arnoldi method (ERAM), some combination of the existing $m$ Ritz vectors is used to choose a single starting guess, from which a new Arnoldi factorization is generated. There are several ways to do this; in this work, the default SLEPc implementation is used, described in Ref. [28].

An alternative to explicit restart is implicit restart, where the desired portion of the spectrum is retained continuously by contracting from a subspace of dimension $m$ to a smaller space of size $p$ and mapping back to the larger space. This is done...
via implicitly-shifted QR, and leads to the implicitly-restarted Arnoldi method (IRAM) [73]. An alternative to IRAM used in this work is the Krylov-Schur (KS) method [74], which transforms a general Krylov decomposition (of which the Arnoldi decomposition is a special case) into a Krylov-Schur decomposition, where the upper Hessenberg matrix \( H \) above becomes a strictly upper triangle matrix \( T \). Using this decomposition, it is comparatively easier numerically to keep the desired spectral information, and the method is apparently often more efficient than IRAM. As for ERAM, the default SLEPc implementation of KS is used; the method and its implementation in SLEPc are described in Ref. [29].

7.2 Accelerating the Fixed Point Iteration

In the sections to follow, we investigate several techniques for accelerating the outer fixed point iteration for \( k \). All the methods are based on some form of extrapolation with respect to \( k \), and hence no machinery beyond that needed for the fixed point iteration is required for their use.

7.2.1 Regula Falsi and Related Methods

Recall that the current eigenvalue \( \lambda \) approaches unity in the limit \( k \to k^* \). For conservative responses with negligible iteration error, \( \lambda \) tends to exactly unity. Various schemes have been used to capitalize on this relationship. In each case, an initial guess \( k_0 \) is made for which the corresponding \( \lambda_0 \) is found. Subsequently, \( k_1 \) is selected, potentially via balance, and \( \lambda_1 \) is computed. All successive values \( k_n \) are selected so that \( \lambda_n \approx 1 \). Such a scheme is often called regula falsi or the method of false points [42].

Lindahl studied the relationship between \( \lambda \) and \( k \) and found that \( \tilde{k} = 1/k \) varies quite linearly with \( \tilde{\lambda} \propto 1/\lambda \). Lindahl extended the concept by storing three or more pairs for interpolation via higher order polynomials [42].

Anghel and Gheorghu [4] modified the approach of Lindahl by assuming the exponential relation

\[
\lambda \propto ae^{b/k}.
\]  

(7.2)
Because response functions tend to have exponential dependence on $k$, they assumed that would also apply to the current eigenvalue. As the results below will indicate, this form compares favorably with Lindahl’s.

A more recent study by Forget and Rahnema [21] rediscovered the relationship between $k$ and $\lambda$, the latter of which they denoted the “normalization constant.” Moving from a $k$-update via balance, they assumed the relation $k \propto 1/\lambda$ and found good results. In theory, we might expect this to be the case. Expanding the result of Eq. 6.48 for small $B$, we find that

$$\frac{d\lambda}{dB} \propto B,$$

suggesting that

$$\lambda \approx aB^2 + b \approx \frac{a'}{k} + b'. \quad (7.4)$$

Of course, this is based on a one group approximation in the limit of very small nodes. Interestingly, Lindahl found that $k^{-1} \propto \lambda^{-1}$ produced better results, having compared to both $k \propto \lambda^{-1}$ and $k \propto \lambda$ (though without giving numerical results) [42].

All four two-term schemes have been implemented and are assessed in the results to follow. A key limitation of these schemes is that they depend on $\lambda$ getting to unity, or at least close enough so that its departure from unity is within the convergence criteria used. If either the responses or the inner iterations are poorly converged, or the response expansions are not conservative, the schemes can become unstable.

### 7.2.2 Steffensen’s Method

A similar approach to those described above is Steffensen’s method, which, like the others, relies on a sequence of evaluations of the fixed point. Steffensen’s method is most easily motivated via use of Aitken’s $\delta^2$ process. Consider a monotonic sequence

$$\{k\} \equiv \{k_0, k_1, k_2, \ldots, k_n, \ldots\} \quad (7.5)$$

such that

$$\lim_{n \to \infty} k_n = k^*. \quad (7.6)$$
If the sequence is linearly convergent, then
\[
\frac{k_{n-1} - k^*}{k_{n-2} - k^*} \approx \frac{k_n - k^*}{k_{n-1} - k^*}.
\] (7.7)

Consequently,
\[
(k_{n-1} - k^*)^2 \approx (k_n - k^*)(k_{n-2} - k^*)
\] (7.8)
yielding Aitken's \(\delta^2\) approximation
\[
k^* \approx k^*_n = k_{n-2} - \frac{(k_{n-1} - k_{n-2})^2}{k_{n-2} - 2k_{n-1} + k_{n-2}}.
\] (7.9)

Suppose the sequence is generated by a series of Picard iterations as defined in Eq. 6.1. If the improved estimate \(k^*_n\) is used in place of \(k_n\), the resulting iteration is referred to as Steffensen's method. Since this is akin to extrapolation by a second order polynomial fit, we might expect the improved estimate yields an iterative scheme of second order, which is indeed true for Steffensen's method.

Steffensen's method can be written as the one step fixed point iteration
\[
k' = g(k) = k - \frac{(f(k) - k)^2}{f(f(k)) - 2f(k) + k}.
\] (7.10)

Following Section 6.1, we expand \(g(k)\) about the fixed point \(k^*\), yielding
\[
g(k) = g(k^*) + \Delta g'(k^*) + \frac{\Delta^2}{2} g''(k^*) + \mathcal{O}(\Delta^3).
\] (7.11)

To be (at least) second order, \(g'(k^*)\) must vanish. Here,
\[
\lim_{k \to k^*} g'(k) = 1 - \frac{2(k - f(k))(1 - f'(k))}{f(f(k)) - 2f(k) + k}
+ \frac{(k - f(k))^2(f'(f(k))f'(k) - 2f'(k) + 1)}{(f(f(k)) - 2f(k) + k)^2}
\] (7.12)
\[
= 1 - (2) + (1)
\]
\[
= 0,
\]
where the second two terms are reduced via L'Hôpital's rule.

In practice, Steffensen's method is highly sensitive to the accuracy of the sequence estimates. Within the ERMM, it has been observed that Steffensen's method becomes unstable unless very small tolerances ($\approx 10^{-9}$) are used for solving the $\lambda$-eigenvalue problem.

Furthermore, note that once the responses are evaluated for the initial guess, each successive Steffensen iteration requires two response evaluations. In general, the savings gained by second order convergence may or may not outweigh the cost of additional evaluations depending on the problem.

### 7.3 Solving ERME's via Newton's Method

The eigenvalue response matrix problem has been recognized as a nonlinear problem since it was first solved, but it does not appear it has been cast in a form for solution directly by Newton-based methods until quite recently [55].

The eigenvalue response matrix equation, $k$ update equation, and $L_2$ normalization of $J_-$ can be written as the nonlinear residual

$$f(x) = \begin{bmatrix} (MR(k) - \lambda I)J_- \\ F(k)J_- - (kL(k)J_-) \\ \frac{1}{2}J_J^-J_- - \frac{1}{2} \\ J^T_f \end{bmatrix} = 0,$$

(7.13)

and the associated Jacobian is defined

$$f'(x) = \begin{bmatrix} (MR - \lambda I) & MRkJ_- & J_- \\ (F - kL) & (F_k - kL_k - L)J_- & 0 \\ J^T_f & 0 & 0 \end{bmatrix}.$$

(7.14)

For $R(k)$ of size $m \times m$, the Jacobian is of size $(m+2) \times (m+2)$. Moreover, after one evaluation of the response quantities, only the first $m + 1$ rows of the $(m + 1)$th column of $f'$ are not known a priori, and that unknown column requires only one additional eval-
uation of the response quantities to allow for a finite difference approximation of the partial derivatives with respect to \( k \). Hence, like Steffensen's method, Newton's method requires two evaluations of \( k \) per iteration, if the latter approximates the derivative via functions evaluated at \( k \) and \( k + \delta k \). Typically, a value of \( \delta k \approx \sqrt{\varepsilon_{\text{machine}}} \approx 10^{-8} \) is close to optimal. However, at likely reduced performance, the finite difference can make use of previous values of \( k \); in this case, convergence would likely improve every iteration as each \( k \) is closer to the solution.

### 7.3.1 Newton's Method

Newton's method [35] solves a nonlinear system via the sequence

\[
s = -f'(x^{(n)})^{-1}f(x^{(n)})
\]

(7.15)

where \( s \) is the Newton step, and the Newton update is

\[
x^{(n+1)} = x^{(n)} + ls,
\]

(7.16)

with a step length \( l \) defined to guarantee a decrease in \( ||f(x)||_2 \). If a solution \( x^* \) exists, and \( f' \) is Lipschitz continuous near and nonsingular at \( x^* \), then Newton's method is known to exhibit quadratic convergence [35].

For the nonlinear residual of Eq. 7.13 and Jacobian of Eq. 7.14, these constraints appear to be true in practice. For a standard, non-parameterized eigenvalue problem \( Ax = \lambda x \), Peters and Wilkinson [50] have shown the associated Jacobian (akin to Eq. 7.14 without the the \((m + 1)\)th column and row) is nonsingular at the solution if \( \lambda \) is simple (which is true for the dominant mode of interest [43]); however, it does not appear this is always true for the full Jacobian in Eq. 7.14, though in practice the conditions for singularity have not been encountered.
7.3.2 Inexact Newton and JFNK

An inexact Newton method uses an approximate linear solve for the Newton step satisfying
\[
\|f'(x^{(n)})s + f(x^{(n)})\|_2 \leq \eta\|x^{(n)}\|_2,
\] (7.17)
where \(\eta\) is the “forcing term” that may vary at each iteration [35]. The inexact solution of the Newton step necessarily impacts convergence of Newton’s method, but typically convergence remains superlinear. In general, any iterative method for a linear system could be used, but we focus on the specific application of Krylov solvers, yielding Newton-Krylov methods.

Since solving for \(s\) involves only the action of the Jacobian, the Jacobian need not be explicitly formed. Then Newton-Krylov methods become Jacobian-Free Newton-Krylov (JFNK) methods, for which Knoll and Keyes provide an extensive survey [36]. If the action of \(MR\) were performed in a matrix-free manner, the same algorithm could be used in evaluating the action of \(f\) for a fully matrix-free approach.

Often in the context of JFNK, the action of the Jacobian is approximated as a finite difference approximation; however, since the Jacobian in Eq. 7.14 is defined almost entirely \textit{a priori}, only a relatively small portion of the action need be approximated via finite differences. This is critical for on-the-fly response generation, for which evaluation of \(k\)-dependent responses is the dominant cost, since each Krylov vector generally represents a perturbed \(k\).

7.3.3 Preconditioning JFNK

The key to effective use of Krylov-based linear solvers is often adequate preconditioning. For JFNK, a preconditioner \(M\) is in some way “close” to the Jacobian \(f'\) but is easier to construct or apply. Moreover, \(M\) can be applied either to the left, yielding the system
\[
M^{-1}fs = -M^{-1}f,
\]
or to the right by solving
\[
fM^{-1}s = -f\]
and setting \(s = M^{-1}s\).

In the numerical studies of the next chapter, we employ incomplete factorization using an approximate Jacobian matrix as a basis. The easiest choice is to use a full Jacobian computed using the initial guess (which itself has been preconditioned by
a single coarse Picard iteration), possibly omitting the partial derivatives to save on a function evaluation. While this approximate Jacobian requires the explicit construction of $\mathbf{MR}$, this is relatively easy to do given the extreme sparsity of $\mathbf{M}$. 
Chapter 8

Comparison of ERME Solvers

In this chapter, the solvers developed in Chapter 7 are applied to several benchmark problems. The purpose is largely two-fold. While previous chapters have discussed in some depth the accuracy of various approximations, a further comparison based on somewhat larger, more representative problems is valuable. Secondly, and more importantly, we aim to assess which of the various algorithms is best suited for solving response matrix equations.

8.1 Diffusion Benchmarks

Several diffusion benchmarks were first used to investigate the various response matrix algorithms previously described. While such benchmarks do not represent particularly "challenging" problems by modern standards, they are simple to model and provide easy application problems for testing the solvers (and expansions schemes) to a fully converged state.

The problems considered are the 2-D and 3-D IAEA benchmarks, and the 2-D Biblis problem, all two-group problems, and the four-group, 2-D Koeberg benchmark. Each problem is based on homogenized assemblies, and brief specifications are provided in Appendix C.

For all problems, the multigroup dependence is exactly treated while the spatial dependence is expanded in DLPs. Each 2-D assembly is discretized using a $20 \times 20$
spatial mesh, corresponding roughly to 1 cm square cells. For the 3-D IAEA problem, 20 cm cubes are represented by a $10 \times 10 \times 10$ spatial mesh to reduce the size of the reference calculation. While the discretizations used are not fully spatially-converged grids, they are adequate for the present study. The reference solution uses the same mesh spacing for consistency.

When computing responses, no symmetry is considered. For homogeneous problems, exploiting symmetry would reduce the number of responses by a factor of 4 in 2-D or a factor of 6 in 3-D. Such tricks are of course handy for benchmarking, but in reality, reactor assemblies are only symmetric at beginning-of-life (and that is on the order of 1/3 of the core fuel), and even then, only at cold zero power. Once any realistic treatment of temperature feedback is considered, all symmetry is lost, and hence there is essentially no insight to be gained from artificially reducing our problem size.

Unless stated otherwise, convergence is defined by $\|f\|_2 < 10^{-12}$, where $f$ is the nonlinear residual defined by Eq. 7.13. Using this criterion makes comparison of Picard and Newton methods much more straightforward. The reference fine mesh solution uses a tolerance of $10^{-14}$, though this applies to the residual $A\phi - \lambda \phi$ of the fine mesh eigenvalue problem.

### 8.1.1 Tolerances and Errors

As was discovered in early chapters, a convergence criterion based on one quantity may not be an accurate measure of the error in some other quantity. To illustrate, the assembly powers for the 2-D IAEA problem were computed for spatial orders of 0 through 4 subject to a tolerance on the residual norm ranging from $10^{-1}$ down to $10^{-12}$. The relative eigenvalue error for each order as a function of tolerance is shown in Figure 8-1, while the maximum relative assembly error for each order is shown in Figure 8-2. The latter figure indicates that the convergence error in assembly powers for a given order is vanishingly small compared to the truncation error due to the order for tolerances below about $10^{-6}$. A similar trend appears for the eigenvalue error, but
for even looser tolerances. For the subsequent diffusion analyses, a tolerance of $10^{-7}$ was selected to ensure only truncation errors affect the solutions observed.

Figure 8-1: Absolute relative eigenvalue error for the 2-D IAEA problem as a function of residual norm tolerance for several spatial orders.

### 8.1.2 Orders and Accuracy

While some attention was paid to the accuracy of spatial expansions in Chapter 5, it is illustrative to assess convergence of the DLP basis applied to the diffusion benchmarks. Figure 8-4 shows the maximum relative error in the assembly powers as a function of spatial order, while Figure 8-3 does the same for the absolute relative error in the eigenvalue. For the 3-D IAEA problem, two cases are performed. The first uses a full expansion of order $m$ in both spatial variables. That means on a given side, the two-dimensional expansion is equivalent to the form

$$F(x, y) \approx a + bx + cy + dx^2 + exy + fy^2 + \ldots.$$  \hspace{1cm} (8.1)
Figure 8-2: Maximum absolute relative assembly power error for the 2-D IAEA problem as a function of residual norm tolerance for several spatial orders.

The second case uses an order reduction scheme that limits the sum of the $x$ and $y$ orders [20]. In this case,

$$F(x, y) \approx a + bx + cy + dx^2 + fy^2 + \ldots,$$

where the cross term $exy$ has been omitted. Previous experience has demonstrated these cross terms, particularly at high order, have little value, and this is demonstrated quite strongly in the results of Figures 8-3 and 8-4.

For all the problems, a fourth order expansion yields assembly (or nodal, for the IAEA-3D problem) errors below a tenth of a percent and eigenvalue errors on the range of a few pcm. Consequently, a fourth order expansion was selected for use in comparing the solvers in subsequent performance analyses.
8.1.3 Inner Solver Comparison

For use in the Picard solver, several eigenvalue solvers were investigated to solve the inner $\lambda$-eigenvalue problem, including the power (PI), Krylov-Schur (KS), and explicitly-restarted Arnoldi methods (ERAM), each of which is implemented in SLEPC [30]. Because convergence of the outer Picard iteration is somewhat sensitive to the inner convergence, the tolerance $\tau_\lambda$ of the inner problem was set more tightly at $10^{-10}$.

Table 8.1 provides the number of inner and outer iterations and total computational time for each method for each of the diffusion problems. The computational time is further divided into the global (response matrix) time and the response (diffusion solver) time. For all problems, KS outperforms ERAM by a small margin and PI by a factor of two or three. Initial studies demonstrated that IRAM (not included by default with SLEPC) performs at about the same level as KS [58]. Were the tolerance smaller, the improvement of KS over PI would likely diminish, and were the tolerance greater, KS would likely perform increasingly better.
Figure 8-4: Maximum absolute relative assembly power error as a function of spatial order.

For the three 2-D problems, the response time constitutes a significant portion of the total computational time, range from about a third to half depending on the solver. For the 3-D IAEA problem, the global solver becomes the dominant cost. This makes some sense, as the diffusion problems underlying the response generation are quite cheap compared to the much larger global problem.

8.1.4 Outer Solver Comparison

Because KS was the best of the inner solvers investigated, it was selected for studies of the Picard-based outer iteration schemes. For this study, Picard iteration along with the accelerated variant based on the regula falsi method was compared to Steffensen’s method and Newton’s method.
<table>
<thead>
<tr>
<th>solver</th>
<th>time [s]</th>
<th>r. time [s]</th>
<th>inners</th>
<th>outers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>2-D IAEA</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PI</td>
<td>1.87</td>
<td>0.37</td>
<td>3416</td>
<td>6</td>
</tr>
<tr>
<td>KS</td>
<td>0.69</td>
<td>0.41</td>
<td>33</td>
<td>6</td>
</tr>
<tr>
<td>ERAM</td>
<td>0.77</td>
<td>0.40</td>
<td>36</td>
<td>6</td>
</tr>
<tr>
<td><strong>Biblis</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PI</td>
<td>2.16</td>
<td>0.68</td>
<td>3437</td>
<td>5</td>
</tr>
<tr>
<td>KS</td>
<td>0.97</td>
<td>0.68</td>
<td>31</td>
<td>5</td>
</tr>
<tr>
<td>ERAM</td>
<td>1.05</td>
<td>0.69</td>
<td>34</td>
<td>5</td>
</tr>
<tr>
<td><strong>Koeberg</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PI</td>
<td>5.83</td>
<td>1.62</td>
<td>2012</td>
<td>3</td>
</tr>
<tr>
<td>KS</td>
<td>2.38</td>
<td>1.66</td>
<td>21</td>
<td>3</td>
</tr>
<tr>
<td>ERAM</td>
<td>2.63</td>
<td>1.67</td>
<td>21</td>
<td>3</td>
</tr>
<tr>
<td><strong>3-D IAEA</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PI</td>
<td>910.63</td>
<td>18.77</td>
<td>4427</td>
<td>6</td>
</tr>
<tr>
<td>KS</td>
<td>210.91</td>
<td>19.47</td>
<td>75</td>
<td>6</td>
</tr>
<tr>
<td>ERAM</td>
<td>294.48</td>
<td>19.75</td>
<td>70</td>
<td>6</td>
</tr>
</tbody>
</table>

*a Response generation time*

Table 8.1: Picard inner solver comparison for diffusion problems.

**Picard Acceleration**

The four Picard acceleration schemes were applied to the 2-D IAEA and Koeberg problems using a fourth order expansion. Figure 8-5 shows the nonlinear residual as a function of outer iteration for the unaccelerated case along with the four accelerated cases.

As the numerical results of the last section and of Chapter 6 show, the Picard iteration by itself is a quickly converging process. However, the acceleration schemes can offer some improvement. Here, exponential and inverse-inverse extrapolation provide the most robust improvement, though they actually provide no reduction in iterations for the Koeberg problem. Recall that all the schemes depend critically on the limit \( \lambda \to 1 \), and this only occurs if the responses are computed very accurately and the responses are conservative. Here, the diffusion responses are computed essentially by
LU factorization since they are such small problems and are conservative based on the uniform mesh and DLP expansion. Even so, the linear-inverse and linear-linear suffer from their sensitivity to round-off errors, and while care was taken when implementing the coefficients, the convergence tolerances used are apparently not tight enough to ensure stability. Because the exponential scheme yields a slightly smaller final residual norm, it was included for study with the remaining algorithms.

Figure 8-5: Comparison of Picard acceleration schemes for the 2-D IAEA problem (solid lines) and Koeberg problem (dashed lines).

Newton Variants

For Newton's method, both an unpreconditioned and ILU-preconditioned variant were studied. The ILU preconditioner is based on an explicit Jacobian constructed either once, using the initial responses, or every iteration, using the updated responses. In all cases, the underlying linear solves were performed with GMRES(30), and the ILU preconditioner was applied with 0 through 2 levels.

Table 8.2 provides results for the 2-D IAEA and Koeberg problems using a fourth order spatial expansion and $4 \times 4$ nodes per assembly. This yields a somewhat larger
problem that better illustrates differences in the preconditioner; the preconditioners offer no benefit for the single node case. The problems have 184962 and 369922 unknowns, respectively.

For both problems, ILU(0) preconditioning offers the best performance with respect to time despite higher levels yielding lower numbers of inner iterations. Interestingly, not updating the preconditioner has no discernible effect on the iteration count and yields lower computational times than when the preconditioner is updated at every iteration. This can be explained by noting that the majority of the Jacobian is relatively insensitive to small changes in \( k \). Given the initial guess (unity in these cases) is expected to be pretty close to the final answer, the original Jacobian should be pretty close to its final value.

<table>
<thead>
<tr>
<th>preconditioner</th>
<th>time [s]</th>
<th>r. time [s]</th>
<th>inners</th>
<th>outers</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>12.62</td>
<td>0.46</td>
<td>477</td>
<td>4</td>
</tr>
<tr>
<td>no update ILU(0)</td>
<td>10.43</td>
<td>0.46</td>
<td>144</td>
<td>4</td>
</tr>
<tr>
<td>no update ILU(1)</td>
<td>11.68</td>
<td>0.46</td>
<td>118</td>
<td>4</td>
</tr>
<tr>
<td>no update ILU(2)</td>
<td>12.37</td>
<td>0.46</td>
<td>86</td>
<td>4</td>
</tr>
<tr>
<td>ILU(0)</td>
<td>10.69</td>
<td>0.46</td>
<td>144</td>
<td>4</td>
</tr>
<tr>
<td>ILU(1)</td>
<td>14.07</td>
<td>0.45</td>
<td>118</td>
<td>4</td>
</tr>
<tr>
<td>ILU(2)</td>
<td>17.12</td>
<td>0.45</td>
<td>86</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>preconditioner</th>
<th>time [s]</th>
<th>r. time [s]</th>
<th>inners</th>
<th>outers</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>45.82</td>
<td>3.29</td>
<td>403</td>
<td>4</td>
</tr>
<tr>
<td>no update ILU(0)</td>
<td>43.18</td>
<td>3.34</td>
<td>157</td>
<td>4</td>
</tr>
<tr>
<td>no update ILU(1)</td>
<td>52.18</td>
<td>3.35</td>
<td>136</td>
<td>4</td>
</tr>
<tr>
<td>no update ILU(2)</td>
<td>58.26</td>
<td>3.35</td>
<td>91</td>
<td>4</td>
</tr>
<tr>
<td>ILU(0)</td>
<td>45.61</td>
<td>3.35</td>
<td>157</td>
<td>4</td>
</tr>
<tr>
<td>ILU(1)</td>
<td>67.44</td>
<td>3.38</td>
<td>136</td>
<td>4</td>
</tr>
<tr>
<td>ILU(2)</td>
<td>89.90</td>
<td>3.35</td>
<td>91</td>
<td>4</td>
</tr>
</tbody>
</table>

\( ^a \) Response generation time

Table 8.2: Newton solver ILU preconditioner comparison for diffusion problems.
Comparing Picard, Steffensen, and Newton

To compare the methods, several metrics are of interest. Ultimately, the wall clock time is of most importance. However, the number of iterations of each method, both outer and inner, is also indicative of the algorithm performance independent of any particular implementation. These data are provided in Table 8.3 for each of the benchmarks. A fourth order spatial expansion was used throughout, with order reduction being applied to the 3-D IAEA problem. For the 2-D problems, a $4 \times 4$ node-per-assembly model is used, while for the 3-D IAEA problem, a single node is used. This corresponds to 184962 unknowns for the 2-D IAEA and Biblis problems, 369922 for the Koeberg problem, and 988382 for the 3-D IAEA problem.

The solvers tested included Picard (P) with and without exponential extrapolation (exp), Steffensen’s method (S), and Newton’s method (N). For Newton’s method, two schemes were examined. The first is the same as used for testing the preconditioning and is based on a Jacobian with $k$-derivatives computed using a finite difference with a $\Delta k = 10^{-8}$. The second scheme lags the $k$ evaluation, using the last iterate for computing $\Delta k$. This results in a larger $\Delta k$ and hence less accurate finite difference. However, the accuracy of the finite difference has no observable effect on the convergence, and for all four problems, the lagged approach has the same number of outer iterations as the regular approach but reduces the number of $k$ evaluations by nearly half.

Steffensen’s method provides the fastest convergence with respect to outer iterations, but it relies on two $k$-evaluations per outer iteration. Picard with exponential extrapolation yields the lowest computational time and the fewest $k$-evaluations. While Newton’s method with a coarse $\Delta k$ is competitive with respect to $k$-evaluations, the overhead of solving the linear systems is higher than the cost of the $\lambda$-eigenvalue problem in the Picard iteration.

8.1.5 Comments

From the diffusion analyses, it appears that Picard iteration using KS for the inners and the exponential extrapolation for accelerating the outers yields the best performance.
<table>
<thead>
<tr>
<th>solver</th>
<th>time [s]</th>
<th>r. time [s]</th>
<th>inners</th>
<th>outers</th>
<th>k-evaluations</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-D IAEA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P\textsuperscript{a}</td>
<td>10.26</td>
<td>0.32</td>
<td>76</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>P+exp\textsuperscript{b}</td>
<td>9.09</td>
<td>0.27</td>
<td>65</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>S\textsuperscript{c}</td>
<td>11.20</td>
<td>0.39</td>
<td>80</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>N+δk\textsuperscript{d}</td>
<td>10.35</td>
<td>0.44</td>
<td>144</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>N+Δk\textsuperscript{e}</td>
<td>10.22</td>
<td>0.28</td>
<td>146</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Koeberg</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P</td>
<td>9.26</td>
<td>0.56</td>
<td>64</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>P+exp</td>
<td>9.06</td>
<td>0.56</td>
<td>62</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>S</td>
<td>9.36</td>
<td>0.56</td>
<td>64</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>N+δk</td>
<td>10.89</td>
<td>0.92</td>
<td>140</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>N+Δk</td>
<td>10.49</td>
<td>0.57</td>
<td>140</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Biblis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P</td>
<td>25.65</td>
<td>1.61</td>
<td>54</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>P+exp</td>
<td>25.76</td>
<td>1.62</td>
<td>54</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>S</td>
<td>29.61</td>
<td>2.02</td>
<td>60</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>N+δk</td>
<td>43.10</td>
<td>3.26</td>
<td>157</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>N+Δk</td>
<td>42.07</td>
<td>2.05</td>
<td>157</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>3-D IAEA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P</td>
<td>213.51</td>
<td>17.93</td>
<td>75</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>P+exp</td>
<td>167.93</td>
<td>12.79</td>
<td>62</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>S</td>
<td>214.07</td>
<td>17.98</td>
<td>75</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>N+δk</td>
<td>269.36</td>
<td>20.93</td>
<td>129</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>N+Δk</td>
<td>261.27</td>
<td>13.02</td>
<td>129</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

\textsuperscript{a} Picard  
\textsuperscript{b} Picard with exponential extrapolation  
\textsuperscript{c} Steffensen  
\textsuperscript{d} Newton with fine $k$ difference  
\textsuperscript{e} Newton with coarse $k$ difference  
\textsuperscript{f} Response generation time

Table 8.3: Outer solver comparison for diffusion problems.
The Newton methods perform about as well with respect to $k$-evaluations, but the cost of applying the method is higher per iteration than the Picard variants, implying further work on preconditioning the inner solves is warranted.

8.2 2-D C5G7

The 2-D C5G7 transport benchmark is a small quarter core model consisting of two UO$_2$ assemblies and two MOX assemblies, all surrounded by an assembly-width reflector. The problem is specified in Appendix C. Additionally, the transport approximations used are described, and the Detran reference results used in the following analysis are compared to the MCNP reference results from the original specification.

8.2.1 Orders and Accuracy

To assess the accuracy of the response schemes available for transport problems, the C5G7 benchmark was solved using both a DLP and Chebyshev-Legendre basis in angle. A DLP spatial basis and a full energy expansion were used throughout. Because the spatial mesh is not exactly uniform, the DLP spatial expansion is not likely to converge as quickly for the C5G7 problem as for the uniformly-meshed diffusion problems (or the small test problems of Chapter 5). For all cases, the response transport calculations were converged to a relative residual norm of $10^{-8}$, and the outer calculation was converged to a nonlinear residual norm of $10^{-7}$.

Figure 8.4 shows the convergence of the eigenvalue and pin power errors as a function of space-angle order. By third order, the conservative basis appears to approach a very limited improvement. This is most likely due to the DLP spatial basis used; since the mesh is not uniform, a DLP expansion is not conservative, and low order expansions cannot be expected to yield good results. Since both the DLP and Chebyshev bases yield maximum relative pin power errors of just over 2%, it appears the spatial expansion becomes the dominant source of error. These results are in contrast to those of Chapter 5, for which a full order spatial basis helped isolate just the error due to ex-
panding in angle. This indicates that a full implementation and systematic study of the modified spatial basis developed in Section 5.2.2 would potentially be of great value.

\[
\begin{align*}
\text{basis} & \quad \text{order} & \quad e_k^a & \quad \max |e_i|^b & \quad \max |e_i| \frac{p_i}{p_i^{\text{ref}}} & \quad \frac{\sum |e_i|}{N} & \quad \frac{\sqrt{\sum e_i^2}}{N} & \quad \frac{\sum |e_i|p_i^{\text{ref}}}{N} - \bar{p}_i^{\text{ref}} \\
\text{DLP-\(\psi\)} & 0 & 1.00 & 33.35 & 108.87 & 9.49 & 0.36 & 11.11 \\
\text{DLP-\(\psi\)} & 1 & 0.72 & 37.78 & 18.52 & 2.63 & 0.12 & 3.07 \\
\text{DLP-\(\psi\)} & 2 & 0.13 & 4.67 & 7.74 & 0.92 & 0.04 & 1.25 \\
\text{DLP-\(\psi\)} & 3 & 0.01 & 2.35 & 6.24 & 0.35 & 0.02 & 0.37 \\
\text{Chebyshev-\(\psi\)} & 0 & 2.61 & 40.73 & 107.01 & 9.57 & 0.37 & 11.03 \\
\text{Chebyshev-\(\psi\)} & 1 & 0.07 & 19.54 & 11.09 & 1.01 & 0.06 & 1.26 \\
\text{Chebyshev-\(\psi\)} & 2 & 0.04 & 2.95 & 6.70 & 0.39 & 0.02 & 0.37 \\
\text{Chebyshev-\(\psi\)} & 3 & 0.04 & 2.42 & 6.43 & 0.35 & 0.02 & 0.35 \\
\end{align*}
\]

\(a\) \(e_k = |k - k^{\text{ref}}|/k^{\text{ref}}\)

\(b\) \(e_i = p_i - p_i^{\text{ref}},\) for \(i\)th pin

Table 8.4: 2-D C5G7 order convergence. All errors in \%, with reference results from Detran on same mesh.

**Solver Comparison**

The same set of global solvers used in Section 8.1.4 were also applied to the 2-D C5G7 problem. In this case, parallel computation was necessary, and 64 processes were used throughout with a single process for the global balance problem; see Chapter 9 for more details on the parallel implementation of ERMM.

Table 8.5 provides the wall time, as well as the total and response time summed over all processes. Included also are the number of inner iterations, outer iterations, and \(k\) evaluations. Newton's method with the coarse \(k\) derivative yields the best performance. Surprisingly, using the fine \(k\) derivative requires more outer iterations and hence significantly greater time. Steffensen's method, as for the diffusion problems, requires the fewest outer iterations but at the cost of more \(k\) evaluations. Standard Picard iteration performs reasonably well, but the extrapolated Picard iteration fails miserably. However, this is completely expected: since the spatial basis is not conservative, \(\lambda\) does not tend toward unity, and hence extrapolation does not apply. This
highlights a significant value in selecting a conservative basis, since the extrapolated Picard iteration was the best performing method for the diffusion problems.

<table>
<thead>
<tr>
<th>solver</th>
<th>w. time[^f][s]</th>
<th>time[^g][s]</th>
<th>r. time[^h][s]</th>
<th>inner</th>
<th>outer</th>
<th>k-evals.</th>
</tr>
</thead>
<tbody>
<tr>
<td>P[^a]</td>
<td>1.67 \cdot 10^3</td>
<td>1.07 \cdot 10^5</td>
<td>1.07 \cdot 10^5</td>
<td>16</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>P+exp[^b]</td>
<td>6.26 \cdot 10^3</td>
<td>4.00 \cdot 10^5</td>
<td>4.00 \cdot 10^5</td>
<td>36</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>S[^c]</td>
<td>1.73 \cdot 10^3</td>
<td>1.11 \cdot 10^5</td>
<td>1.11 \cdot 10^5</td>
<td>16</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>N+\delta k[^d]</td>
<td>4.60 \cdot 10^3</td>
<td>2.94 \cdot 10^5</td>
<td>2.94 \cdot 10^5</td>
<td>76</td>
<td>8</td>
<td>16</td>
</tr>
<tr>
<td>N+\Delta k[^e]</td>
<td>1.09 \cdot 10^3</td>
<td>6.97 \cdot 10^4</td>
<td>6.96 \cdot 10^4</td>
<td>33</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

\[^a\] Picard
\[^b\] Picard with exponential extrapolation
\[^c\] Steffensen
\[^d\] Newton with fine k difference
\[^e\] Newton with coarse k difference
\[^f\] Wall time
\[^g\] Total time summed over all processes
\[^h\] Total response generation time summed over all processes

Table 8.5: Outer solver comparison for 2-D C5G7 problem with first order expansion. Picard with exponential extrapolation fails due to the nonconservative spatial basis, i.e. \( \lambda \neq 1 \).

### 8.3 3-D Takeda

The 3-D Takeda benchmark is an old and rather simple benchmark, but it allows us to examine in more depth the convergence properties of the basis sets. The model consists of three homogeneous regions using a two group approximation. More details can be found in Appendix C.

#### 8.3.1 Order Convergence

For the response matrix model, each node is a 5 cm cube discretized with uniform 0.25 cm cells in all dimensions. Using order reduction in both space and angle (see Section 8.1.2), the DLP was used for both \( \psi \) and \( J \), and the conservative basis (Chebyshev-Legendre + Jacobi-Legendre) was used for \( \psi \); all bases were used for orders 0 through 4.
Figures 8-6 and 8-7 provide the absolute relative error in the eigenvalue and the maximum absolute relative error in the nodal powers as a function of angular order for several spatial orders. It is readily apparent little is gained with increasing angular order when the spatial order is limited to zero. For higher spatial orders, an increasing angular order yields a monotonically decreasing error for both $k$ and the nodal powers. The conservative basis outperforms the DLP variants, yielding nearly sub-1% nodal errors for a third order angular expansion and spatial orders greater than 1. DLP-$J$ yields slightly better nodal powers than DLP-$\psi$ at higher orders, but yields higher $k$ errors for all orders.

For all the bases, a significant trend is that spatial orders above 2 yield diminishing returns; that is, the most consistent improvement over all angular orders and bases is a shift from first to second order in space. This is reasonable because the nodes are homogeneous and boundary quantities should therefore be relatively smooth functions of space.

![Figure 8-6](image-url)

Figure 8-6: Takeda problem absolute relative eigenvalue error as a function of angular order for several spatial orders. The solid lines indicate the conservative basis, while the dashed and dashed-dot lines indicate the DLP basis used to expand the angular flux $\psi$ and current $J$, respectively.
Figure 8-7: Takeda problem absolute relative nodal power error as a function of angular order for several spatial orders. The solid lines indicate the conservative basis, while the dashed and dashed-dot lines indicate the DLP basis used to expand the angular flux $\psi$ and current $J$, respectively.

### 8.3.2 Solver Comparison

The same set of solvers as used for the diffusion problems and the 2-D C5G7 problem were applied to a the Takeda problem. A second order spatial expansion with a third order angular expansion in the azimuth and polar variables was used. Order reduction was applied to both the spatial and angular terms. The problem was run on 64 processors, with one process for the global problem. Table 8.6 provides the wall time, total time summed over all processors, and the total response function time summed over all processors.

Similar to the diffusion results, the extrapolated Picard iteration proves to be the most efficient of the solvers studied. Newton's method with the coarse $k$ finite difference yielded just as few $k$-evaluations but with a slightly higher overall cost.
<table>
<thead>
<tr>
<th>solver</th>
<th>w. time$^a$[s]</th>
<th>time$^b$[s]</th>
<th>r. time$^c$[s]</th>
<th>inners</th>
<th>outers</th>
<th>k-evals.</th>
</tr>
</thead>
<tbody>
<tr>
<td>P$^a$</td>
<td>$9.76 \cdot 10^2$</td>
<td>$6.25 \cdot 10^4$</td>
<td>$5.91 \cdot 10^4$</td>
<td>56</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>P+exp$^b$</td>
<td>$3.51 \cdot 10^2$</td>
<td>$2.24 \cdot 10^4$</td>
<td>$2.12 \cdot 10^4$</td>
<td>23</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>S$^c$</td>
<td>$1.05 \cdot 10^3$</td>
<td>$6.72 \cdot 10^4$</td>
<td>$6.36 \cdot 10^4$</td>
<td>58</td>
<td>7</td>
<td>15</td>
</tr>
<tr>
<td>N+δk$^d$</td>
<td>$7.30 \cdot 10^2$</td>
<td>$4.67 \cdot 10^4$</td>
<td>$4.26 \cdot 10^4$</td>
<td>57</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>N+Δk$^e$</td>
<td>$3.87 \cdot 10^2$</td>
<td>$2.48 \cdot 10^4$</td>
<td>$2.14 \cdot 10^4$</td>
<td>44</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

$^a$ Picard  
$^b$ Picard with exponential extrapolation  
$^c$ Steffensen  
$^d$ Newton with fine k difference  
$^e$ Newton with coarse k difference  
$^f$ Wall time  
$^g$ Total time summed over all processes  
$^h$ Total response generation time summed over all processes

Table 8.6: Outer solver comparison for 3-D Takeda problem with second order spatial expansion and third order polar and azimuthal angle expansions.

### 8.4 Summary

Based on the results for both the diffusion and transport problems, Picard iteration with the exponential extrapolation appears generally to be the most efficient of the methods, yielding minimum numbers of k-evaluations while providing the lowest global solver overhead. However, the extrapolation is based on $\lambda$ converging to unity, and because this is not always guaranteed, Newton’s method with the coarse finite difference provides a more consistently robust solver, with nearly as few k-evaluations and only relatively small overhead due to the inner linear solves.

It is anticipated further work on preconditioners for Newton’s method would put the corresponding (global) computation time more in line with that of the extrapolated Picard iteration. Even so, the efficacy and simplicity of the Picard iteration suggests that care should be taken to select a conservative basis leading to $\lambda = 1$ upon convergence.
Chapter 9

Parallel Algorithms for ERME Solvers

All chapters to this point have focused primarily on the methods to discretize the transport equation for application to response generation and on algorithms for solving the resulting transport or response matrix equations. In this chapter, the focus turns to solving the response matrix equations on parallel machines. An algorithm is proposed that highlights a natural parallel decomposition of the problem. A simple test study illustrates the potential of the algorithm for larger problems and larger machines.

9.1 A Parallel Decomposition

Response matrix methods in general are naturally parallelizable due to responses being completely independent. Moreover, the sheer number of responses required for real analysis makes parallel computation a necessity moving forward. In support of this work, a parallel response matrix code for Solving Eigenvalue Responses Matrix Equation with Nonlinear Techniques (Serment) was developed. Here, we describe the parallel algorithm implemented in Serment, while technical details of the code (language, libraries, etc.) are provided in Appendix E.

Figure 9-1 provides one possible way to view the various levels of parallelism available for response matrix calculations. The first level decomposes the problem in space, leading in full core models to hundreds of independent nodes. Each node is represented by hundreds or even thousands of responses depending on the underlying transport ap-
proximation and the group structure of the cross section data used. For example, the third order 2-D CSG7 results of Chapter 8 required 1120 responses per node, and that was insufficient to achieve pin power errors below 1%. With these two levels alone, the method offers the potential for parallelism on up to $\mathcal{O}(10^6)$ processes, which represents the upper bound of the largest systems of today. If parallel processing is used also for the local transport solves, even limited to threaded parallelism on $\mathcal{O}(10)$ threads, the potential process use goes beyond present day machines, making the method an interesting candidate for next generation machines.

\[
\begin{align*}
\text{Global eigenvalue problem} & : \quad T\phi(\rho) = \frac{1}{k} F\phi(\rho) \\
\text{Local fixed source problems} & : \quad T_l\phi(\bar{\rho}) = \frac{1}{k} F_l\phi(\bar{\rho})
\end{align*}
\]

Figure 9-1: Schematic of a parallel decomposition for the eigenvalue response matrix method.
9.2 Implementation

The parallel implementation of Serment largely follows the breakdown implied by Figure 9-1. MPI is used throughout, as are the PETSc and SLEPC libraries. The processes used are organized into three MPI communicators: world, global, and local.

All processes belong to the world communicator\(^1\). The world processes are divided into local groups each with its own communicator. The root process of each local communicator belongs to the global communicator.

All response generation takes place within local communicators, and each local communicator is assigned a subset of the unique nodes (and their responses) comprising the global problem. The local root process manages the response generation via a fixed work approach, though load balancing would be a straightforward addition using a simple master-slave approach.

The global balance equations are solved only on the global communicator. Scoping studies suggested the communications overhead incurred by including all processes in the global solve outweigh any benefit of using them.

9.3 Numerical Results

9.3.1 A Response-Dominated Problem

As a preliminary test of Serment for parallel computations, the 2-D C5G7 problem with a zeroth order space and angle expansions was examined. This yields an extremely small global problem of just 252 unknowns (a 7 group current on each of four faces over 9 nodes). However, the remaining transport work required for responses is significant.

The problem was run on up to 14 cluster nodes each having 8 cores. The decomposition used just a single global process, with the rest contributing via response generation. Figure 9-2 shows the parallel speedup as function of the number of processors. In all cases, the response generation was the overwhelming contribution to

\(^{1}\text{equivalent to } \text{MPI\_COMM\_WORLD}\)
the elapsed time, explaining the good, albeit erratic, scaling observed. It is unclear exactly why the performance dips at 80 processes, but the result was repeatable. This implies a hardware-dependency. The cluster used has 18 nodes in total, 8 of which are newer. While having similar CPU speeds to the older nodes, these new nodes have significantly higher memory bandwidth. As soon as an older node is used (the case for 80 processes), the entire process is limited to some extent by the performance of that node.

![Graph](image)

Figure 9-2: Speedup of 2-D C5G7 problem with zeroth order expansions. The single process time was 8091 seconds over 5 $k$-evaluations.

Because the response generation was the dominant cost, the fact that the global problem was limited to one process did not lead to any observable (consistent) performance degradation.
9.3.2 A Balance-Dominated Problem

For cases in which the global problem is not trivial, limiting the global problem to a small group of processes necessarily limits the parallelism of the proposed algorithm. Consequently, one must balance carefully the distribution of local and global processes.

To illustrate, consider the 3-D IAEA problem. From the single process results of Table 8.3, the response generation time represents roughly 9% of the total time. Hence, by parallelizing only the responses, we would expect a small reduction in wall time. However, were the global problem parallelized instead, better but still limited scaling should be attainable.

For the special case of each process representing a local group, meaning each process computes all the responses it needs, the maximum expected speedup is

\[ S_{\text{expected}} = \frac{t_0}{t_0 + \frac{t_{0p}}{n}} \]

where \( t_0 \) is the total time on one process, \( t_{0p} \) represents time spent on parallelizable tasks, \( t_{0s} \) represents time spent on tasks performed in serial, and \( t_0 = t_{0s} + t_{0p} \). Here, response generation represents the majority of the serial work, and at 9% of the total, we expect a maximum speedup of roughly 10.

Figure 9-3 shows the observed scaling for up to 48 processes (limited to identical compute nodes). The observed results generally follow the theoretical maximum, which is much less than perfect scaling. In general, applying more processes to the global problem leads to better scaling, as indicated by the variation in speedup as a function of local group count (denoted “nlg”) in the figure for 48 processes.

9.4 Comments

The results of the previous section indicate that the MPI-enabled, domain- and data-decomposed method implemented works largely as expected. For response-dominated problem, the speedup observed is nearly ideal outside of minor hardware-related effects. Moreover, for balance-dominated problems, the global solver was shown to scale
as expected, though the speedup is limited intrinsically by the local-global structure.

However, this approach was selected knowing that problems of greatest interest—response-dominated transport models—will never have costs dominated by the global solver.

The major limitation of the current implementation is that response generation, while being completely parallelized, does not incorporate automatic load balancing. This should be a relatively straightforward feature to include in future development and would be critical when moving analysis to larger machines.

Additionally, no attempt has yet been made to exploit the threaded-parallelism offered by the Detran solvers responsible for response generation. Appendix D shows that Detran scales well on multicore machines. Hence, it might be the case that using only one transport solver instance with multiple threads may yield an improvement.
over multiple processes each driving its own transport solver, and this is especially likely for problems having large memory foot prints (for which cache would be better used in a threaded approach) and for which the angular quadrature is very high order (allowing many threads to be of use).
Chapter 10

Conclusion and Future Work

10.1 Summary

The thrust of this thesis has aimed at developing response matrix methods for application to full core modeling using deterministic transport methods. Three essential areas were considered in depth: the transport solvers needed for efficient generation of response functions, accurate bases that best represent the physics at boundaries, and efficient algorithms for solving the large, nonlinear eigenvalue problem characteristic of the method.

For solving the transport equation to define response functions, GMRES was shown to be a robust and efficient solver, particularly for application to the multigroup equations with multiplication. By preconditioning the multigroup GMRES solver with a coarse mesh diffusion scheme, even better results were obtained. While an initial scoping study of the Newton-Schulz algorithm for preconditioning did improve upon the basic coarse mesh diffusion preconditioner, some improvement seems possible.

Defining response functions requires expansion in orthogonal bases for each phase space variable, and several bases were investigated. While the standard bases in space and energy were used for much of the numerical analysis of later chapters, the conservative angular bases along with a consistent quadrature set were employed throughout with success. Better space and energy bases were developed, and the preliminary re-
results of Chapter 5 suggest these bases may be the ticket in reducing significantly the number of unknowns required in large models.

Several algorithms for solving the eigenvalue response matrix equations were developed and analyzed. While the standard Picard iteration was found to be rapidly converging, even better convergence was obtained by using acceleration techniques or via use of the second order nonlinear methods Steffensen and Newton. Ultimately, the cost of the problem is largely due to response generation, and an algorithm requiring a minimum number of $k$ evaluations is desirable. For the problems studied, Picard iteration with extrapolation was found to be the most effective method when conservative bases are used. For cases in which the extrapolated Picard iteration failed, Newton’s method provided a robust alternative, especially when the $k$-derivatives are approximated using only previous $k$ values.

Finally, initial studies using the MPI-enabled, domain- and data-decomposed parallel response matrix code Serment showed that the methods implemented scale well on the research cluster available during this work. Moreover, theoretical analysis of the parallel algorithm used suggests it could scale well on much larger systems given the potential process use the method entails.

10.2 Broader Implications

Ultimately, our goal is to use ERMM in production level analyses, and to get there, significant challenges remain. Chief among these remains the shear number of responses required for realistic modeling. The solvers and preconditioners studied go a long way to minimizing the time spent generating these responses, and as was suggested in Chapter 1, any time saved during response generation is applicable to any implementation of ERMM (including pre-generation of responses).

While the bases developed in Chapter 5 are quite promising in their own right, the success so far observed suggests there may even be better bases—“silver bullets” so to speak—that would further reduce the degrees of freedom needed for highly accurate modeling. As an example, an energy basis that would allow 44 group fidelity in as few
as 4 or 5 degrees of freedom would be extremely valuable, with similar value possible in space and probably less so in angle.

Finally, as was discussed in Chapter 9, the potential process use of ERMM for large 3-D reactor models is extremely large. Because so much of the work involved is contained in the independent response calculations, ERMM is nearly "embarrassingly" parallel, which can seldom be said of deterministic methods. While the parallel implementation of Serment has not yet been tuned for anything but a modest research cluster, further refinement of the algorithm and code would make it ideal for much larger systems.

10.3 Future Work

This work has spanned a variety of topics, and in many cases, there was simply insufficient time to explore all the details of many tangential items or those topics which were examined toward the end of this study.

10.3.1 Linear and Nonlinear Coarse Mesh Preconditioners

The coarse mesh diffusion preconditioners developed in Chapter 3 led to significant improvement on the already strong multigroup GMRES algorithm. While a generic development of the coarse mesh scheme based on flux homogenization, no study of weighting schemes beyond the flat flux case was performed due to time considerations. More work should be done to investigate this seemingly low-hanging fruit.

A more detailed study of linear and nonlinear preconditioners, for response generation and for standard reactor transport problems, should be performed. Recent work by Larsen and a student hints that nonlinear operators outperform their linear counterparts, but the analysis appears mostly limited to 1-D problems. Detran is already equipped to perform nonlinear acceleration, and so a study of realistic 2-D and 3-D problems is a near term possibility. In particular, the question to be answered is whether a linear preconditioner, diffusion-based or otherwise, coupled with a Krylov solver like GMRES does better or worse than a nonlinear acceleration technique applied
to a classical algorithm like Gauss-Seidel or even a nonlinear-capable Krylov method like FGMRES.

### 10.3.2 Sparse Approximate Inverse for Preconditioning

In Section 3.3, we developed a preconditioner using Newton’s method to provide a transport correction to a diffusion preconditioner in a way that avoids explicit construction of transport operators. However, for response generation, the amortization of the construction process may make explicit factorizations attractive.

Rather than factorizing the transport operator directly, we can construct an approximate inverse $M^{-1}$ directly. To do so, we seek $M^{-1}$ to minimize

$$f(M^{-1}) = ||R||_F,$$

(10.1)

where

$$R = I - AM^{-1}$$

(10.2)

is the residual matrix, and $|| \cdot ||_F$ denotes the Frobenius norm. Originally due to Benson and Frederickson [9], this approach has been widely used in development of sparse approximate inverse (SPAI) preconditioners for which Benzi and Tuma provide a thorough survey [10].

A particularly appealing SPAI method that assumes no a priori sparsity pattern for $M^{-1}$ was developed by Chow and Saad [13]. Their approach is based on minimizing the objective function of Eq. 10.1 using a descent method. Given an initial preconditioner $M$, they suggest as one alternative the “Global Minimal Residual” descent algorithm, defined by the process

$$M_{\text{new}}^{-1} = M^{-1} + \alpha R,$$

(10.3)

where

$$\alpha = \frac{\text{tr}(R^T AR)}{\text{tr}((AR)^T AR)}.$$

(10.4)
The process is repeated some predefined number of times or until $f$ falls below some threshold. Along the way, the updated $M^{-1}$ can be subjected to numerical dropping to maintain sparsity.

The obvious drawback of this algorithm is that it requires matrix-matrix multiplications involving $A$, a completely impractical operation for most problems. For the case of response generation in parallel, this approach may be practical if the preconditioner for a node is constructed in parallel by and shared among processes involved in computing the responses for that node.

While the method was not completely implemented, initial scoping studies suggest that even a few iterations yield very good preconditioners. However, since each multiplication $A$ with another matrix requires $m$ space-angle-energy sweeps, a balance must be found between sweeps for the preconditioner and the sweeps ultimately eliminated in the response solutions. Analysis of this issue is highly recommended for future work.

10.3.3 Advanced Space and Energy Bases

Part of the focus of Chapter 5 was to study space and energy bases. For the spatial case, a very simple periodic function was shown to reduce errors in the partial current by a significant margin compared to a standard DLP expansion. Incorporating this in Serment and testing on full problems should be done. In our best estimate, such bases should be capable of reducing the required spatial order to third order, which includes the flat and shape function along with a linear and quadratic term.

For the energy treatment, a modified DLP basis was produced using a pin cell spectrum as its zeroth order member. While the results were promising, the convergence was quite slow in energy order, and only at order 15 (for 44 groups) were the errors reduced to satisfactory levels. The chief goal in a new energy basis would be to reduce errors to acceptable levels using roughly a 10th of the original degrees of freedom. For the 44 groups used in the example, this means that a 4th or 5th order expansion is the desired target. One could incorporate more physics into the basis by using additional spectral information from the pin cell calculation (i.e. using higher order modes). An
alternative would be to optimize the basis more heuristically so that on average the coefficients tend toward an exponential decay (meaning higher order terms become increasingly less important with order).

### 10.3.4 MOC Responses

An original goal of this work was to use the method of characteristics as the transport discretization. Substantial progress toward this were made in Detran, but because the main focus of this work is essentially independent of the discretization used, and because the $S_n$ method was already completed, little MOC analysis was performed. Scoping studies investigated a space-angle quadrature that would allow exact, full-order expansions of MOC boundary fluxes [57]. However, the analysis of Chapter 5 showed that good results can be obtained with proper selection of an angular basis. Coupled with a high order representation in space, MOC should be a natural fit for future response work.

### 10.3.5 Large Scale Parallel Computing

While Serment is completely parallelized, only small-scale scoping studies have been performed. Moving to larger machines will help identify weakness in the algorithm or its implementation, and more importantly, allow much larger problems to be solved. Of particular interest would be to compare the efficiency of Serment + Detran to that of a parallel fine mesh code such as PARTISN or Denovo.

### 10.3.6 Time-Dependent Responses

Recent work by Pounders and Rahnema investigated use of time-dependent responses in the context of the incident flux expansion method [52]. They expanded boundary fluxes in the Legendre polynomials over a predetermined time interval. Probably unknown to those authors was the much earlier work of Sicilian and Pryor [67]. Rather than expand in a continuous basis, these authors defined responses at discrete points in time, between which a linear shape was assumed. In essence, Sicilian and Pryor
applied a linear spline expansion, whereas Pounders and Rahnema used a continuous basis. In theory, both can be viewed as instances of a more general approach that uses a finite expansion within finite intervals.

The chief difficulty noted in both works is the added requirement of computing volume responses. Whereas for eigenvalue problems, the entire balance equation can be cast in terms of boundary unknowns, time-dependent problems of interest to reactor physics necessarily include delayed neutron precursors. While Pounders and Rahnema did not account for precursors explicitly, they did incorporate volume responses that could be used for this purpose. Sicilian and Pryor did account for precursors, and introduced approximations that eliminated the need for volume-to-volume responses. Future work should examine the method of Sicilian and Pryor and test it in a modern code solving realistic problems.
Appendix A

Multigroup Buckling

The buckling $B$ was defined in Eq. 6.15 for a one group problem and can be related to the curvature of the flux. In multigroup problems, we can define more generalized buckling terms [27]. Consider the multigroup diffusion equations in operator form

\[-D \nabla \vec{\phi} + \Sigma_r \vec{\phi} = \Sigma_s \vec{\phi} + \frac{1}{k} \vec{\gamma}^T \nu \Sigma_f \vec{\phi}. \tag{A.1}\]

We can write this in Helmholtz form

\[\nabla \vec{\phi} + H \vec{\phi}(\vec{r}) = 0. \tag{A.2}\]

Defining $B^2$ and $V$ such that

\[H = VB^2V^T, \tag{A.3}\]

we can represent the flux as

\[\vec{\phi} = V\vec{\psi}. \tag{A.4}\]

The functions $\psi_g$ are the so-called modal fluxes that satisfy

\[\nabla \vec{\psi} + B^2 \vec{\psi}(\vec{r}) = 0. \tag{A.5}\]
The square root of the eigenvalues $B_g^2$ are recognized as the buckling coefficients. In the case of one group, the development presented here can be seen to simplify to the form used in the original one group analysis.
Appendix B

Homogeneous $B_0$ Theory

Consider the one group, one dimensional transport equation for isotropic scattering and fission

$$\mu \frac{d\psi}{dx} + \Sigma_t \psi(x, \mu) = \frac{1}{2} \left( \frac{\nu \Sigma_f}{k} + \Sigma_s \right) \int_{-1}^{1} \psi(x, \mu) d\mu. \tag{B.1}$$

In the homogeneous $B_0$ approximation [51], the angular flux is factorized as $\psi(x, \mu) = \psi(\mu) e^{iBx}$, and substituting this into the transport equation yields

$$(iB\mu + \Sigma_t)\psi(\mu) = \frac{1}{2} \left( \frac{\nu \Sigma_f}{k} + \Sigma_s \right) \int_{-1}^{1} \psi(\mu) d\mu. \tag{B.2}$$

Dividing both sides by $iB\mu + \Sigma_t$ and integrating over $\mu$ yields the identity

$$1 = \left( \frac{\nu \Sigma_f}{k} + \Sigma_s \right) \int_{-1}^{1} \frac{d\mu}{iB\mu + \Sigma_t}. \tag{B.3}$$

Performing the integral, we find the transcendental equation for the buckling

$$\frac{1}{B} \tan^{-1} \left( \frac{B}{\Sigma_t} \right) = \left( \frac{\nu \Sigma_f}{k} + \Sigma_s \right)^{-1}. \tag{B.4}$$
Appendix C

Benchmark Problem Descriptions

This Appendix provides descriptions of the various example and benchmark problems used throughout this work. Most problems come from the literature, although a few were developed to highlight features of the methods presented.

C.1 Diffusion Benchmarks

In the following sections, several diffusion benchmarks are described. All the models were originally specified as quarter cores subject to reflection. In all models analyzed here, the full core was modeled to provide a somewhat larger problem.

C.1.1 IAEA Two Group Benchmarks

One of the most widely used diffusion benchmarks is the IAEA two group diffusion benchmark [6]. A 2-D and 3-D version are defined. While not difficult by today’s standards, in its time, its goal was to tax the best solvers of the day by presenting a core with significant (albeit unrealistic) thermal flux gradients near the reflector. The model consists of four (2-D) or five (3-D) unique material regions.
C.1.2 Biblis Benchmark

The Biblis benchmark (described in Ref. [47]; original documentation unavailable) provides a more realistic 2-D core model. The core consists of 7 unique materials, and fuel is arranged in a typical checker-board pattern.

C.1.3 Koeberg Benchmark

The Koeberg benchmark is a four-group model that, like the Biblis benchmark, is a fairly accurate model of an operating PWR [47]. In this case, it is a model of the Koeberg power station in South Africa [47].

C.2 Simplified BWR Bundle

To provide a relatively small transport problem for examining the transport operator in detail, a small BWR bundle was modeled, illustrated in Figure C-1. The model contains just two materials, a smeared fuel-clad mixture and a moderator, and consists of 1.15 cm square pins with a 1.4 cm pitch. The cross sections are provided in Tables C.1 and C.2.

<table>
<thead>
<tr>
<th></th>
<th>$g = 1$</th>
<th>$g = 2$</th>
<th>$g = 3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Sigma_{t,g}$</td>
<td>0.2822058997</td>
<td>0.4997685502</td>
<td>0.4323754911</td>
</tr>
<tr>
<td>$\Sigma_{f,g}$</td>
<td>0.0028231045</td>
<td>0.0096261203</td>
<td>0.1123513981</td>
</tr>
<tr>
<td>$\nu_g$</td>
<td>2.7202775245</td>
<td>2.4338148428</td>
<td>2.4338000000</td>
</tr>
<tr>
<td>$\chi_g$</td>
<td>0.9996892490</td>
<td>0.0003391680</td>
<td>0.0000000000</td>
</tr>
<tr>
<td>$\Sigma_{s1-g}$</td>
<td>0.2760152893</td>
<td>0.0000000000</td>
<td>0.0000000000</td>
</tr>
<tr>
<td>$\Sigma_{s2-g}$</td>
<td>0.0011230014</td>
<td>0.4533430274</td>
<td>0.0000378305</td>
</tr>
<tr>
<td>$\Sigma_{s3-g}$</td>
<td>0.0000000000</td>
<td>0.0014582502</td>
<td>0.2823864370</td>
</tr>
</tbody>
</table>

Table C.1: Three group BWR fuel cross sections.
### 2-D C5G7 Benchmark

The C5G7 benchmark was originally proposed to assess transport methods for use in (mostly) explicit representation of pin cell assemblies containing MOX fuel [41]. The fuel pin and cladding are smeared, but the moderator is represented explicitly. The benchmark has been widely used to benchmark discretizations in space and angle. The C5G7 problem uses 7-group data for 7 material regions: one UO$_2$ fuel pin, three MOX pins, and one each for a moderator, guide tube, and instrumentation. The cross sections are provided in the original documentation. The core schematic is provided in Figure C-2, where the indices (starting at zero) correspond to the 7 materials.

<table>
<thead>
<tr>
<th></th>
<th>$g = 1$</th>
<th>$g = 2$</th>
<th>$g = 3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Sigma_{t,g}$</td>
<td>0.3333694980</td>
<td>0.5887110656</td>
<td>1.6512246291</td>
</tr>
<tr>
<td>$\Sigma_{s,1-g}$</td>
<td>0.2432946408</td>
<td>0.0000000000</td>
<td>0.0000000000</td>
</tr>
<tr>
<td>$\Sigma_{s,2-g}$</td>
<td>0.0898364840</td>
<td>0.4413753398</td>
<td>0.0000203109</td>
</tr>
<tr>
<td>$\Sigma_{s,3-g}$</td>
<td>0.0000387911</td>
<td>0.1465683257</td>
<td>1.6300848232</td>
</tr>
</tbody>
</table>
C.3.1 Spatial Discretization

Because the implementation of the discrete ordinates method in Detran is limited to a cartesian mesh, modeling approximations of the fuel pin are needed. Two basic approaches are available. In the first, an arbitrary discretization is used, and the material data assigned to each cell is a volume-average based on the true geometry. This is not ideal, since volume-averaging generally does not conserve neutrons. The preferred approach is to use a relatively accurate discretization of the pin that preserves the volume of each material region exactly but generally not its shape. For the C5G7 problem, we employ a $8 \times 8$ volume-preserving discretization illustrated in Figure C-3.

C.3.2 Reference Values

For assessing the accuracy of response matrix results, there are two reasonable references. The first is the “true” solution based on a reference MCNP calculation. The second represents the full order solution based on the underlying transport method used in the response matrix method. Using the spatial discretization described above
and a $6 \times 6$ QR-TG quadrature, a reference Detran solution was computed, the results of which are compared to the MCNP solution in Tables C.3 and C.4. For reference, the total time was 1738 seconds using PI with the flattened transport operator (i.e. no scattering iterations) and a tolerance on the successive flux norm of $10^{-8}$.

<table>
<thead>
<tr>
<th></th>
<th>$k$</th>
<th>error [%]</th>
<th>$\max p_i$</th>
<th>error [%]</th>
<th>$\min p_i$</th>
<th>error [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCNP</td>
<td>1.18655</td>
<td>0.00800</td>
<td>2.49770</td>
<td>0.16000</td>
<td>0.23150</td>
<td>0.58000</td>
</tr>
<tr>
<td>Detran</td>
<td>1.18642</td>
<td>-0.01104</td>
<td>2.50391</td>
<td>0.24873</td>
<td>0.23167</td>
<td>0.07249</td>
</tr>
</tbody>
</table>

Table C.3: Detran calculation results for eigenvalue and pin power. Here, $p_i$ represents the power of the $i$th pin. The MCNP errors are the stochastic uncertainties, while the Detran errors are relative to the MCNP values.

The Detran results are surprisingly accurate given the relative crude spatial discretization. In particular, the Table C.4 shows the maximum pin power error is about 0.01, and the maximum relative error is just under 1%. Several additional measures are included, following the format of previous CSG7 benchmarking results.
Table C.4: Detran calculation results for pin power distribution. Here, \( p_i \) is the power of the \( i \)th pin, \( e_i \) is the error \( p_i - p_i^{\text{ref}} \), and \( \bar{p} \) is a region average.

For the response matrix method, it is likely more useful from a methods development view to compare the approximations to the full order solution. For analysis, of course, it makes sense to compare to the true reference.

### C.4 3-D Takeda Benchmark

In 1991, four 3-D neutron transport benchmarks for reactor analysis were proposed [76]. Here, we consider the first benchmark, which is a model of the Kyoto University Critical Assembly. The model is based on three homogeneous regions and two group cross section. Two cases were presented, one with and one without the control rod inserted; here we consider only the case with the rod inserted.

Table C.5 provides the reference Monte Carlo values for the eigenvalue and the region-averaged fluxes along with the Detran results. The latter were generating using a 0.25 cm uniform mesh and a 6 × 6 product QR-TG quadrature.

| region      | \( \max |e_i| \) [%] | \( \frac{\max |e_i|}{\bar{p}^{\text{ref}}} \) [%] | \( \frac{\sum |e_i|}{N} \) [%] | \( \sqrt{\frac{\sum e_i^2}{N}} \) [%] | \( \frac{\sum |e_i|p_i^{\text{ref}}}{N} - \bar{p}^{\text{ref}} \) [%] |
|-------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| inner UO₂   | 0.91231         | 0.41991         | 0.29450         | 0.02164         | 1.10572         |
| MOX         | 0.51372         | 0.94501         | 0.15319         | 0.01139         | 0.09674         |
| outer UO₂   | 0.29422         | 0.76321         | 0.09205         | 0.00693         | 0.02663         |
| core        | 0.91231         | 0.94501         | 0.17323         | 0.00696         | 0.22097         |

Table C.5: Detran calculation results for eigenvalue and pin power. Here, \( p_i \) represents the power of the \( i \)th pin. The MCNP errors are the stochastic uncertainties, while the Detran errors are relative to the MCNP values.
Appendix D

A DETermistic TRANsport Code

A DETerministic TRANsport code (Detran) was developed in support of this thesis. Many of the approximations and methods implemented in Detran were discussed in Chapters 2 and 3. This appendix provides a brief description of its implementation and features. The code is free and open source under the MIT license. The source code and documentation is available at github.com/robertsjo/libdetran. A user's guide is currently under development.

D.1 Language and Design

Detran is an object-oriented deterministic transport code written in C++. Originally intended as the transport kernel of Serment, Detran has become a relatively flexible transport code for a variety of problems. The initial design of Detran was heavily influenced by previous experience with the Denovo code of ORNL. In fact, Detran retains many coding conventions inherited from Denovo. Detran was further influenced by the design of Jaguar, a deterministic transport code developed by KAPL and Bettis.

The primary design goal of Detran is to be a useful, completely self-contained tool. Currently, only a relatively recent C++ compiler (and CMake) is needed to build the Detran executable, for which a limited text-based input is available.

However, scientific codes are quickly moving toward input/output schemes that are user friendly. Gone (almost, anyway) are the days of ASCII inputs and outputs coupled
with a hodge podge of Perl (or another scripting language) parsing routines. A more flexible approach is to build the scripting language right into the code. Detran does this by providing a flexible SWIG-generated Python front end that gives users much more flexibility in defining problems and manipulating output. This interface requires the user have Python 2.7 or greater and an up-to-date Numpy distribution. For plotting, Matplotlib is highly recommended but not required.

D.2 Build System and External Libraries

The CMake build system is used for Detran. Using CMake, Detran and its Python front end have been successfully built on a variety of Linux machines, OS X, and Windows 7. A particularly nice feature of CMake is that it produces a Makefile compatible with parallel builds on at least Linux and OS X machines.

Detran can optionally be built with PETSc and SLEPc, giving access to several dozen linear solvers and eigensolvers in addition to those implemented. Additionally, a small set of routines written in Fortran are optionally available to extend some of the built-in routines.

Detran can optionally be built with several Boost libraries. In support of Serment, Detran contains several orthogonal basis sets. These can be extended by compiling with Boost's math library. When used with Serment, Detran must be compiled with the Boost Archive library, which allows generic datatypes to be written and read in a binary format.

Detran can produce simple flux (and other) plots using a simple built in PPM plotter. For more advanced plots, Detran can be compiled with Silo to produce files for viewing in Visit or Paraview. For general input and output, Detran can be compiled with HDF5.

D.3 Discretizations and Solvers

Detran incorporates the following discretizations
• Mesh-centered diffusion

• Step difference, diamond difference, step characteristic discrete ordinates with about a dozen angular quadratures

• Method of characteristics using a step characteristic (for use with any of the product quadratures)

All transport approximations are currently limited to isotropic scattering.

Detran includes a limited but self-contained sparse linear algebra package (internally called callow\(^1\)) that implements a variety of linear solvers, including:

1. Richardson
2. Jacobi
3. Gauss-Seidel
4. SOR
5. GMRES

Additionally, the following eigensolvers are implemented:

1. power iteration
2. QR/QZ (via optionally-compiled Fortran EISPACK code)
3. generalized Davidson (which requires QZ)

Essentially any solver implemented by callow or PETSc/SLEPc can be used for transport solves. Additionally, specialized implementations of source iteration, Gauss-Seidel, and power iteration are provided that eliminate some steps characteristic of more general implementations. For example, all the callow solvers test convergence using the norm of the residual, either \( \|Ax - b\| \) for linear problems or \( \|Ax - \lambda x\| \) for eigenvalue problems. Forming the residual requires an additional application of

\(^1\)Callow, as in the callow youth who has little business writing linear algebra routines. Usually it is best to live by OMILAT: only masochists implement linear algebra themselves.
A, which is undesirable in practice. In the transport implementations, convergence is based on the norm of the difference of successive iterates, $||\phi^{(n)} - \phi^{(n-1)}||$. While this was shown to underestimate the residual norm in Chapter 4, as long as a user knows how to select and interpret a reasonable tolerance, there is no problem in practice. Hence, the transport implementation of source iteration and power iteration can be expected to outperform the generalized implementation by a small margin. Recall that in transport problems, Gauss-Seidel really means block Gauss-Seidel in energy, for which there is no generic solver in callow or otherwise.

For use with generic solvers, Detran has several diffusion-based preconditioners. These include a standard fine mesh DSA-like preconditioner along with several coarse mesh variants, all of which are described in more detail in 3.

**D.4 Time Dependent Transport**

**D.4.1 Time Dependent Features**

One of the original goals of this thesis was to study ways to apply the response matrix method to time-dependent problems. While the direction of this work ultimately excluded this analysis, Detran was extended to include a rather extensive capability for time-dependent transport. Detran currently has the following time-dependent features:

- time stepping based on any of the spatial discretization schemes
- time integration via BDF 1-6 or the implicit midpoint rule
- full treatment of delayed neutron precursors
- time-dependent materials via linear interpolation of user-defined discrete time materials
- time-dependent external source via linear interpolation of user-defined discrete time sources or user-defined Gaussian pulse
• synthetic materials formed as part of the time step that may be approximated by the isotropic approximation for transport problems (thus eliminating the need to carry the full discrete flux; future work will allow stepping with arbitrary order spherical harmonic moments)

• lightweight coupling for multiphysics based on fixed point iteration

D.4.2 Application to the 2D LRA Problem

As part of a course project, Detran was used to solve the 2D LRA dynamics problem with the discrete ordinates method. The problem models the ejection of a control rod, including a simple adiabatic feedback model. A clear problem specification can be found in Ref. [69]. However, in that reference, as in all others found, the model given is apparently inconsistent with the results presented; see the footnote in Ref. [75] for the correction. While further details of the problem are outside of the present scope, a few results are worth including here for posterity.

A fully consistent temporal discretization of the discrete ordinates equations results in time stepping schemes that must propagate the fine mesh discrete angular flux in time. For large problems, particularly in three dimensions, storing these fluxes is extraordinarily expensive. Contrarily, for steady state calculations, the angular flux is never needed, as the discrete contribution to flux moments can be added on-the-fly, as discussed in Chapter 2.

We are interested in the magnitude of the error in assembly powers due to propagating only a low order flux approximation, here the scalar flux. For the DD discretization with 15 cm mesh, Table D.1 summarizes several key parameters. Surprisingly, the time to peak is identical for both cases, and the other values are very close.

Figure D-1 shows the core power for the full angular, scalar flux, and diffusion cases as a function of time. Figure D-2 shows the relative error for the scalar flux case with respect to the full angular flux case, while Figure D-3 shows the assembly peaking factors and relative errors for the scalar flux case at the peak time of 1.496 seconds. The maximum error is roughly 21%, while the error in the core power density...
### Table D.1: Summary of transient results.

<table>
<thead>
<tr>
<th></th>
<th>Diffusion</th>
<th>DD 15 (cm) approx.</th>
</tr>
</thead>
<tbody>
<tr>
<td>peak core power [W/cc]</td>
<td>5973</td>
<td>5428</td>
</tr>
<tr>
<td>time to peak [sec]</td>
<td>1.301</td>
<td>1.496</td>
</tr>
<tr>
<td>final core average temp. [K]</td>
<td>1354</td>
<td>1038</td>
</tr>
<tr>
<td>final maximum temp. [K]</td>
<td>4438</td>
<td>2640</td>
</tr>
</tbody>
</table>

is roughly 4%. While these errors are likely too large to support use of the scalar flux only in $S_N$-based transient analysis, it does seem likely that including one or two higher order moments could yield both high accuracy and a smaller memory footprint.

![Graph showing core power for DD and diffusion cases with 15 cm mesh.](image)

**Figure D-1:** Core power for DD and diffusion cases with 15 cm mesh.

### D.5 Parallelism

The transport sweeps in Detran are parallelized on angles within an octant using OpenMP. The implementation is not optimal, being used only for initial testing. Even
Figure D-2: Relative core power error due to scalar flux approximation.

Figure D-3: Relative assembly error in DD using only scalar flux at 1.496 sec.
so, the implementation can achieve roughly 80% efficiency. To illustrate, Figures D-4 and D-5 show the speedup and parallel efficiency of Detran on a one group problem with a $200 \times 200$ spatial mesh and 64 angles per octant. The machine used is an Intel Core i7 with four cores at 2.7 GHz. The speedup is defined as

$$S_n = \frac{T_1}{T_n},$$  \hspace{1cm} (D.1)

where $n$ is the number of threads and $T_n$ is the wall time using $n$ threads. The parallel efficiency is defined

$$E_{np} = \frac{S_n}{p} = \frac{T_1}{pT_n},$$  \hspace{1cm} (D.2)

where $p$ is the number of physical cores. For $n > 4$, a value of $p = 4$ is assumed, and hence the ideal speedup is taken to be $S_n = \min(n, 4)$. The speedup tapers significantly after four threads, which may be due to the machine having only 4 physical cores. However, both the speedup and efficiency reach optimal values at 8 threads. It is believed Intel’s hyperthreading may contribute to the better performance beyond the physical core count.

Detran was never intended to be a massively parallel code, but rather was to be an optimized serial (or threaded) kernel for use in Serment. However, with minimal development, it is believed Detran could become a cluster-level tool, which would be especially useful for time-dependent analysis of large problems. As a first step, the shared memory parallelism can be further integrated. Currently, threading is done solely within the space-angle sweep routine, and the overhead of starting and stopping threads is high. Moving the threaded block up to the solver level would minimize this overhead and open up other routines (like source construction) to threading. Additionally, multigroup Krylov solvers are easily parallelized in energy, for which distributed memory parallelism via MPI is a natural solution. For even relatively few groups (such as the 7 groups of the C5G7 problem), parallelism across 8 cores of one node in angle and 7 nodes in energy represents the size of machine fairly typical of academic research groups.

182
Figure D-4: Detran speedup on four core machine.

Figure D-5: Detran parallel efficiency on four core machine.
D.6 Future Work

While Detran has a relatively complete set of features, time constraints prevented implementation of the following desiderata:

- Flexible multi-unknown discretization schemes, similar to the Equation templating of Denovo. This would allow standard discretizations like discontinuous Galerkin. It would also facilitate use of more complicated schemes, such as an embedded response matrix approach for handling heterogeneous cells.

- Anisotropic scattering

- More complete threaded parallelism

- Distributed memory parallelism in energy

- Rigorous performance testing

- Rigorous benchmarking
Appendix E

Solving Eigenvalue Response Matrix Equations with Nonlinear Techniques

The parallel response matrix code Serment was developed in support of this thesis. Chapters 5-7 detailed the theoretical nature of this work, while Chapters 8 and 9 provided numerical studies of the algorithms developed.

All the results presented were generated using Serment coupled with Detran for the response generation. Serment is planned to be open sourced in the near future.

E.1 Language, Design, and External Packages

The design and implementation of Serment largely follows that of Detran. Written in C++, Serment relies on MPI for parallel communications, and the PETSc and SLEPc packages for numerical algorithms. Additionally, Serment relies on much of the utilities built into Detran, such as smart pointers and other constructs. The Boost library is used for provide an archive functionality, which allows just about any data to be passed via MPI. This helps significantly when passing generic containers. However, standard MPI calls are used for all numerically-intensive work not performed by PETSc or SLEPc.
E.2 Future Work

While the current implementation of Serment is limited to response generation by Detran, the structure of the code is intended to handle any possible response source. One valuable extension would be to include the OpenMC Monte Carlo code as a response source.

Additionally, only Cartesian nodes are currently implemented, but nodes can be completely arbitrary outside of requiring (in 3-D) a designated “top” and “bottom.” While this limitation is not strictly needed, its purpose is to provide a definitive polar axis for defining polar expansions. Because Detran has a nearly complete MOC module, extending Serment to hexagonal cells in 2-D and 3-D is warranted.

Finally, the representation of nodal boundary conditions is completely general. Each surface may be assigned its own basis in space, angle, and energy. This makes modeling with varying fidelity or different transport approximations very straightforward. For example, it is currently possible to model a reactor core having some assemblies that are represented with a full transport approximation while others are represented with diffusion. Recent work has attempted to develop a coupled diffusion-transport framework using a response-based approach, but only a limited scoping code and not a full implementation was performed. With Serment and Detran, such an approach is built into the method by construction. Consequently, there exists a significant parameter space for modeling alone that was left completely untouched by the more methods-oriented work of this thesis.
Appendix F

Angular Quadratures and Orthogonal Bases

A significant component of Chapter 5 was development of basis sets for use in expanding the angular dependence of boundary fluxes or currents. In particular, it was shown that basis sets that have an isotropic flux as their zeroth order member and also conserve the partial current tend to outperform other basis sets, especially for the low orders of interest in practical calculations. Sections F.1

F.1 Exact Integration

The conservative bases are all continuous polynomial bases, and for their use to be effective, appropriate quadratures must be used to integrate the expansion coefficients. Here, we compare several common quadratures for 1-D, 2-D, and 3-D integration and identify those which are exact for a given basis and order.

In 1-D slab geometry and for the polar component of horizontal surfaces in 3-D problems, the Jacobi polynomials, $P_l^{(0,1)}$, are the natural basis. The Jacobi moments for an isotropic flux are defined

$$J_{l} = \int_{0}^{1} \mu P_l^{(0,1)}(2\mu - 1) \cdot 1d\mu = \frac{1}{2}\delta_{0l},$$

(F.1)
and the partial current is $J = J_0$.

For 2-D and 3-D problems, we require a polar integration having the form

$$J_{p,l} = \int_{\xi_0}^{1} \sqrt{1 - \xi^2} U_l(\xi) \cdot 1 d\xi = \frac{\pi}{2} \delta_{l0}, \quad (E2)$$

where $\xi_0 = -1$ for 3-D and $\xi_0 = 0$ for 2-D, and an azimuthal integration of the form

$$J_{a,m} = \int_{-1}^{1} P_m(\omega) \cdot 1 d\omega = 2\delta_{m0}, \quad (E3)$$

where an isotropic flux is again assumed. Here, the partial current is $J = J_{p,0} J_{a,0}$. Note, for 2-D, only even order $U_l$ functions are used due to symmetry.

In all cases, we seek exact integration for at least the zeroth order component, since we otherwise cannot conserve the partial current, $J$. However, it turns out even this is a rather difficult task given standard (product) quadratures. While integrating the azimuthal Legendre moments is particularly straightforward, requiring exact integration for the polar weighting functions $x$ and $\sqrt{1 - x^2}$ characteristic of the Jacobi and Chebyshev bases makes a standard Gaussian quadrature impossible to define. However, notice that $x$ and $\sqrt{1 - x^2}$ essentially represent $\cos \theta$ and $\sin \theta$. Recent work has developed Gaussian-like quadratures for integrating functions that are polynomials in $\cos \theta$ and $\sin \theta$ over partial periods [14]. The abscissa and weights are given in Table E1 for several quadrature orders.

<table>
<thead>
<tr>
<th>$\mu$</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>$w$</td>
<td></td>
<td>$w$</td>
<td>$w$</td>
</tr>
<tr>
<td>0.98372041</td>
<td>0.07933872</td>
<td>0.99371729</td>
<td>0.99710142</td>
</tr>
<tr>
<td>0.70710678</td>
<td>0.48635618</td>
<td>0.86659691</td>
<td>0.93298500</td>
</tr>
<tr>
<td>0.17970575</td>
<td>0.43430509</td>
<td>0.49900882</td>
<td>0.70710678</td>
</tr>
<tr>
<td></td>
<td>0.11191942</td>
<td>0.27729632</td>
<td>0.35991526</td>
</tr>
<tr>
<td></td>
<td>0.07608393</td>
<td>0.19076617</td>
<td></td>
</tr>
</tbody>
</table>

Table E1: 3-, 4-, and 5-point trigonometric Gaussian quadrature rules for polar integration over the range $\mu \in [0, 1]$. 

188
F.2 Numerical Results

Because the azimuthal Legendre moments are easily integrated, we limit our tests to polar integration of the Jacobi and Chebyshev moments. Several polar quadratures are examined, including Gauss-Legendre (GL), double Gauss-Legendre (DGL), Gauss-Chebyshev (GC), double Gauss-Chebyshev (DGC), Abu-Shumays' double range (ASDR; see Ref. [2]), and the trigonometric Gaussian (TG) quadratures. Each is applied using 3, 4, 5, and 6 angles per half space to integrate $J_l$ for $l = 0, 1, \text{and} 2$. Tables F.2-F.4 provide the results.

For integrating the Jacobi moments, the double Gauss-Legendre quadrature appears to be ideal, yielding the exact answer for all quadrature orders. In fact, DGL needs just one angle, $\mu = 0.5$, which grabs the average of the linear integrand. TG also performs well, capturing exact values with four angles. The remaining quadratures are unable to compute the partial current for an isotropic flux exactly for any finite number of angles. While GL is optimal for the range $\mu \in [-1, 1]$ in that an $N$-point GL quadrature exactly integrates polynomials of degree $2N - 1$ or less, recall we are integrating over the half space. For the isotropic flux, using GL yields an integration over the equivalent flux $\psi(\mu) = H(1 - \mu)$, where $H$ is the Heaviside step function. Since $H$ has no finite polynomial representation, GL cannot yield an exact integration. An 2N-point DGL quadrature, on the other hand, can exactly represent polynomials in the half space of degree $2N - 1$ or less; hence, even the 1-point DGL quadrature captures the first order integrand $\mu$ exactly.

However, DGL fails when integrating the Chebyshev moment, since the additional weighting term $\sqrt{1 - x^2}$ is not polynomial. For both the 2-D and 3-D cases, GC captures the exact values. This makes some sense, since GC is designed exactly for integrating functions subject to the weight $\sqrt{1 - x^2}$, yielding a polynomial integrand for polynomial functions. ASDR also computes exact moments, as it too is a Gaussian quadrature, in this case with a weight proportional to $\frac{1}{\sqrt{1-x^2}}$, thus canceling the Chebyshev moment weighting. TG yields exact moments when 5 angles are used.


\section*{F.3 Discussion}

For 1-D problems, DGL appears to be the optimal choice for integrating the polar Jacobi moments. For 2-D problems, either GC or ASDR are optimal. To select one over the other would require analysis of the cross moments $J_{in}$, which has not been performed. For 3-D problems, however, only the TG polar quadrature yields exact conservation and hence is optimal when the conservative bases are used.

While these quadratures are being called “optimal,” this is only in the sense that they are the only ones investigated here that yield exact moments for the polar bases used. For application to more general reactor problems, using a DLP polar quadrature and uniform azimuthal quadrature appears to be the best general approach, particularly when anisotropic scattering is present [62].

\section*{F.4 Detran Quadratures}

For reference, the quadratures currently available in Detran are provided in the following tables. All quadratures in Detran are octant-symmetric, meaning that values for the first octant (i.e., $\mu, \eta, \xi > 0$) are defined, and signs are applied when applicable.

Table F.5 provides a list of quadratures implemented for use as polar quadratures. These are applicable to 1-D, 2-D, and 3-D problems. Table F.6 shows the quadratures implemented for use as azimuthal quadrature in product quadratures. There are 9 polar quadratures and 5 azimuthal quadratures, yielding 45 possible product quadratures. Table F.7 gives the quadratures that are not product quadratures.
<table>
<thead>
<tr>
<th>#</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$l = 0$</td>
<td>GL</td>
</tr>
<tr>
<td>3</td>
<td>9.95e-03</td>
<td>2.81e-02</td>
</tr>
<tr>
<td>4</td>
<td>5.76e-03</td>
<td>1.63e-02</td>
</tr>
<tr>
<td>5</td>
<td>3.76e-03</td>
<td>1.06e-02</td>
</tr>
<tr>
<td>6</td>
<td>2.65e-03</td>
<td>7.49e-03</td>
</tr>
<tr>
<td></td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
</tbody>
</table>

|    | GC | DGC |
| 3  | 2.36e-02 | 2.36e-02 |
| 4  | 1.31e-02 | 1.31e-02 |
| 5  | 8.32e-03 | 8.32e-03 |
| 6  | 5.76e-03 | 5.76e-03 |

|    | 0.00e+00 | 0.00e+00 | 0.00e+00 |

|    | 5.76e-03 | 4.00e-03 |
| 3  | 2.00e-02 | 5.67e-02 |
| 4  | 1.31e-02 | 2.20e-02 |
| 5  | 8.32e-03 | 1.31e-02 |
| 6  | 5.76e-03 | 8.77e-03 |

|    | 0.00e+00 | 0.00e+00 | 0.00e+00 |

|    | 1.63e-02 | 1.63e-02 |
| 3  | 4.62e-02 | 4.62e-02 |
| 4  | 2.77e-02 | 2.77e-02 |
| 5  | 1.86e-02 | 1.86e-02 |
| 6  | 1.33e-02 | 1.33e-02 |

|    | 0.00e+00 | 0.00e+00 | 0.00e+00 |

|    | 2.74e-03 | 1.34e-02 |
| 3  | 0.00e+00 | 0.00e+00 |
| 4  | 0.00e+00 | 0.00e+00 |
| 5  | 0.00e+00 | 0.00e+00 |
| 6  | 0.00e+00 | 0.00e+00 |

Table E2: Absolute errors in 1-D Jacobi moments
<table>
<thead>
<tr>
<th>#</th>
<th>l = 0</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3.07e-03</td>
<td>1.02e-02</td>
<td>2.14e-02</td>
</tr>
<tr>
<td>4</td>
<td>1.36e-03</td>
<td>4.33e-03</td>
<td>8.20e-03</td>
</tr>
<tr>
<td>5</td>
<td>7.18e-04</td>
<td>2.24e-03</td>
<td>4.05e-03</td>
</tr>
<tr>
<td>6</td>
<td>4.24e-04</td>
<td>1.31e-03</td>
<td>2.31e-03</td>
</tr>
<tr>
<td></td>
<td>DGL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>7.24e-03</td>
<td>2.67e-02</td>
<td>7.61e-02</td>
</tr>
<tr>
<td>4</td>
<td>3.32e-03</td>
<td>1.12e-02</td>
<td>2.41e-02</td>
</tr>
<tr>
<td>5</td>
<td>1.79e-03</td>
<td>5.80e-03</td>
<td>1.13e-02</td>
</tr>
<tr>
<td>6</td>
<td>1.08e-03</td>
<td>3.41e-03</td>
<td>6.35e-03</td>
</tr>
<tr>
<td></td>
<td>GC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td>4</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td>5</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td>6</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td></td>
<td>DGC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4.69e-02</td>
<td>4.54e-02</td>
<td>9.22e-02</td>
</tr>
<tr>
<td>4</td>
<td>2.61e-02</td>
<td>2.59e-02</td>
<td>2.54e-02</td>
</tr>
<tr>
<td>5</td>
<td>1.66e-02</td>
<td>1.66e-02</td>
<td>1.64e-02</td>
</tr>
<tr>
<td>6</td>
<td>1.15e-02</td>
<td>1.15e-02</td>
<td>1.14e-02</td>
</tr>
<tr>
<td></td>
<td>ASDR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td>4</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td>5</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td>6</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td></td>
<td>TG</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.00e+00</td>
<td>2.56e-02</td>
<td>2.56e-02</td>
</tr>
<tr>
<td>4</td>
<td>0.00e+00</td>
<td>5.44e-04</td>
<td>5.44e-04</td>
</tr>
<tr>
<td>5</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td>6</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
</tbody>
</table>

Table F3: Absolute errors in 2-D Chebyshev moments
<table>
<thead>
<tr>
<th></th>
<th>l = 0</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>GL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>6.14e-03</td>
<td>0.00e+00</td>
<td>2.05e-02</td>
</tr>
<tr>
<td>4</td>
<td>2.72e-03</td>
<td>0.00e+00</td>
<td>8.66e-03</td>
</tr>
<tr>
<td>5</td>
<td>1.44e-03</td>
<td>0.00e+00</td>
<td>4.48e-03</td>
</tr>
<tr>
<td>6</td>
<td>8.49e-04</td>
<td>0.00e+00</td>
<td>2.62e-03</td>
</tr>
<tr>
<td>DGL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1.45e-02</td>
<td>0.00e+00</td>
<td>5.34e-02</td>
</tr>
<tr>
<td>4</td>
<td>6.64e-03</td>
<td>0.00e+00</td>
<td>2.24e-02</td>
</tr>
<tr>
<td>5</td>
<td>3.59e-03</td>
<td>0.00e+00</td>
<td>1.16e-02</td>
</tr>
<tr>
<td>6</td>
<td>2.16e-03</td>
<td>0.00e+00</td>
<td>6.83e-03</td>
</tr>
<tr>
<td>GC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td>4</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td>5</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td>6</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td>DGC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>9.39e-02</td>
<td>0.00e+00</td>
<td>9.08e-02</td>
</tr>
<tr>
<td>4</td>
<td>5.23e-02</td>
<td>0.00e+00</td>
<td>5.17e-02</td>
</tr>
<tr>
<td>5</td>
<td>3.33e-02</td>
<td>0.00e+00</td>
<td>3.31e-02</td>
</tr>
<tr>
<td>6</td>
<td>2.30e-02</td>
<td>0.00e+00</td>
<td>2.30e-02</td>
</tr>
<tr>
<td>ASDR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td>4</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td>5</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td>6</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td>TG</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>5.12e-02</td>
</tr>
<tr>
<td>4</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>1.09e-03</td>
</tr>
<tr>
<td>5</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
<tr>
<td>6</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
<td>0.00e+00</td>
</tr>
</tbody>
</table>

Table F4: Absolute errors in 3-D Chebyshev moments
<table>
<thead>
<tr>
<th>Name</th>
<th>Acronym</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gauss-Legendre</td>
<td>GL</td>
<td>Gaussian with unity weight</td>
</tr>
<tr>
<td>Double Gauss-Legendre</td>
<td>DGL</td>
<td>GL quadrature defined in the half space</td>
</tr>
<tr>
<td>Gauss-Chebyshev</td>
<td>GC</td>
<td>Gaussian with weight ((1 - x^2)^{-2})</td>
</tr>
<tr>
<td>Double Gauss-Chebyshev</td>
<td>DGC</td>
<td>GC over the half space</td>
</tr>
<tr>
<td>Uniform</td>
<td>U</td>
<td>Uniform over (\theta)</td>
</tr>
<tr>
<td>Uniform-Cosine</td>
<td>UC</td>
<td>Uniform over (\mu = \cos \theta)</td>
</tr>
<tr>
<td>Tabuchi-Yamamoto</td>
<td>TY</td>
<td>Optimized quadrature based on integrating Bickley function of order 3; particularly successful for 2-D lattice calculations (See Ref. [79])</td>
</tr>
<tr>
<td>Abu-Shumays Double Range</td>
<td>DR</td>
<td>Gaussian with weight (2\sqrt{1 - x^2})^{-1}) over the half space (see Ref. [2])</td>
</tr>
<tr>
<td>Triangular-Gaussian</td>
<td>TY</td>
<td>Gaussian-like quadrature for integrating polynomials in both (\cos \theta) and (\sin \theta)</td>
</tr>
</tbody>
</table>

Table E5: Detran polar quadratures.

<table>
<thead>
<tr>
<th>Name</th>
<th>Acronym</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gauss-Legendre</td>
<td>GL</td>
<td>Gaussian with unity weight</td>
</tr>
<tr>
<td>Double Gauss-Legendre</td>
<td>DGL</td>
<td>GL quadrature defined in the half space</td>
</tr>
<tr>
<td>Uniform</td>
<td>U</td>
<td>Uniform over (\phi)</td>
</tr>
<tr>
<td>Simpson</td>
<td>S</td>
<td>Simpson's rule over (\phi)</td>
</tr>
<tr>
<td>Abu-Shumays Quadruple Range</td>
<td>QR</td>
<td>Integrates certain set of polynomials in (\sin \phi) (see Ref. [2])</td>
</tr>
</tbody>
</table>

Table E6: Detran azimuthal quadratures.

<table>
<thead>
<tr>
<th>Name</th>
<th>Acronym</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level-Symmetric</td>
<td>LS</td>
<td>Exact integration of spherical harmonics.</td>
</tr>
<tr>
<td>Uniform-Equal</td>
<td>UE</td>
<td>Uniform over azimuth and polar cosine (see Ref. [12])</td>
</tr>
</tbody>
</table>

Table E7: Other Detran quadratures.
Bibliography


