HIGHER-ORDER DGFEM TRANSPORT CALCULATIONS ON POLYTOPE MESHES FOR MASSIVELY-PARALLEL ARCHITECTURES

A Dissertation

by

MICHAEL WAYNE HACKEMACK

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Jean Ragusa
Committee Members,	Marvin Adams
	Jim Morel
	Nancy Amato
	Troy Becker
Head of Department,	Yassin Hassan

August 2016

Major Subject: Nuclear Engineering

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

1. INTRODUCTION

1.1 Motivation and Purpose of the Dissertation

1. Polytope mesh cells are now being employed in other physics communities - most notably computational fluid dynamics (CFD) [1];

2. They are believed to reduce the number of unknowns to solve with equivalent accuracy;

3. They can reduce cell/face counts which can reduce algorithm wallclock times depending on the solution method;

4. They can allow for transition elements between different portions of the domain (e.g., tetrahedral elements bordering hexahedral elements at the border of the boundary layer);

5. They can easily be split along cut planes - allowing the mesh to be partitioned into regular or irregular divisions as well as be generated by simplical meshing techniques across processor sets in parallel;

6. Hanging nodes from non-conforming meshes, like those that naturally arise from locally refined/adapted meshes, are no longer necessary.

1.2 Current State of the Problem

1.2.1 Background on the Multigroup DGFEM S_N Transport Equation

Multigroup goes here...

Angular discretization goes here...

DGFEM Sn goes here...

1.2.2 Diffusion Synthetic Acceleration

1.2.3 Polytope Grids Formed from Voronoi Mesh Generation

Since this dissertation work is on the solution of the transport equation on polytope meshes, we next describe how these grids can be generated. Traditionally, FEM calculations have been performed on simplical (triangles and tetrahedra) and tensor based meshes (quadrilaterals and hexahedra). In fact, it is still a standard practice to refer to any type of mesh as a *triangulation* in some communities [2]. Many different mesh generation software has been developed to build these simple grids [3, 4, 5, 6]. However, multiple fields including *computational fluid dynamics* (CFD) and *solid mechanics* are now finding benefits to utilizing polygonal and polyhedral meshes for their calculations.

However, polytope mesh generation is still in its infancy.

1.2.4 Adaptive Mesh Refinement for the DGFEM S_N Transport Equation

1.3 Organization of the Dissertation

In this introductory chapter, we have presented a summary of work performed. We also gave our motivation for choosing this work as well as a brief discussion of previous work that has directly influenced this dissertation. We conclude this intoduction by briefly describing the remaining chapters of this dissertation.

In Chapter 2, we present the DGFEM formulation for the multigroup, S_N transport equation. We then describe the transport equation's discretization in energy, angle, and space. We have left the FEM spatial interpolation function as arbitrary at this point to be defined in detail in Chapter 3. For the spatial variable, we provide the theoretical convergence properties of the DGFEM form. We also detail the elementary assembly procedures to form the full set of spatial equations. We con-

clude by providing the methodology to be used to solve the full phase-space of the transport problem.

In Chapter 3, we present all the finite element basis functions that we will use in this work. In two dimensions, we present four different linearly-complete polygonal coordinate systems that we will use to generate our finite element basis functions. We then present the methodology that converts each of these linear coordinate systems into quadratically-complete coordinates for use as higher-order basis functions. We also present the single linearly-complete polyhedral coordinate system that we will use for the 3D transport problems.

In Chapter 4, we present the methodologies to be used for DSA preconditioning of the DGFEM transport equation for optically thick problems. We give a discontinuous form of the diffusion equation which can be used on 2D and 3D polytope grids. The theoretical limits of the DSA scheme are analyzed and we conclude with a real-world problem of accelerating the thermal neutron upscattering of a large multigroup, heterogeneous transport problem. In doing so, we demonstrate that our methodology will work on massively-parallel computer architectures.

We then finalize this dissertation work by drawing conclusions and discussing open topics of research stemming from this dissertation in Chapter 5. We note that our detailed literature reviews, numerical results, and conclusions pertaining to each topic are presented in their corresponding chapter.

Additional material that is not included in the main body of the dissertation for the sake of brevity is appended for completeness. The appendices are organized in a simple manner:

- Appendix **??**: addendum to Section 2, corresponding to additional material relating to the multigroup S_N equations.

- Appendix **??**: addendum to Section 3, corresponding to additional material relating to the various polytope coordinate systems to be utilized as finite element basis functions.

- Appendix **??**: addendum to Section 4, corresponding to additional material relating to DSA preconditioning on polytope grids.

2. THE DGFEM FORMULATION OF THE MULTIGROUP S_N EQUATIONS

2.1 Fundamental Aspects of the Transport Equation

The movement of bulk materials and particles through some medium can be described by the statistical behavior of a non-equilibrium system. Boltzmann first devised these probabilistic field equations to characterize fluid flow via driving temperature gradients [7]. His work was later extended to model general fluid flow, heat conduction, hamiltonian mechanics, quantum theory, general relativity, and radiation transport, among others. The Boltzmann Equation can be written in the general form:

$$\frac{\partial u}{\partial t} = \left(\frac{\partial u}{\partial t}\right)_{force} + \left(\frac{\partial u}{\partial t}\right)_{advec} + \left(\frac{\partial u}{\partial t}\right)_{coll} \tag{2.1}$$

where $u(\vec{r}, \vec{p}, t)$ is the transport distribution function parameterized in terms of position, $\vec{r} = (x, y, z)$, momentum, $\vec{p} = (p_x, p_y, p_z)$, and time, t. In simplified terms, Eq. (2.1) can be interpreted that the time rate of the change of the distribution function, $\frac{\partial u}{\partial t}$, is equal to the sum of the change rates due to external forces, $\left(\frac{\partial u}{\partial t}\right)_{force}$, advection of the particles, $\left(\frac{\partial u}{\partial t}\right)_{advec}$, and particle-to-particle and particle-to-matter collisions, $\left(\frac{\partial u}{\partial t}\right)_{coll}$ [8].

For neutral particle transport, the following assumptions [9] about the behavior of the radiation particles can be utilized:

1. Particles may be considered as points;

2. Particles do not interact with other particles;

3. Particles interact with material target atoms in a binary manner;

4. Collisions between particles and material target atoms are instantaneous;

5. Particles do not experience any external force fields (*e.g.,* gravity).

These assumptions lead to the first-order form of the Boltzmann Transport Equation, which we simply call the transport equation for brevity. The remainder of the chapter is outlined as follows. Section 2.2 provides the general form of the neutron transport equation with some variants. Section 2.3 describes how we discretize the transport equation in energy with the multigroup methodology and Section 2.4 presents the angular discretization via collocation. Section 2.5 details which boundary conditions will be employed for our work. Section 2.6 will conclude our discretization procedures in the spatial domain. Section 2.7 will present the iterative procedures used to obtain a numerical solution. We then present concluding remarks for the chapter in Section 2.8.

2.2 The Neutron Transport Equation

The time-dependent neutron angular flux, $\Psi(\vec{r}, E, \vec{\Omega}, t)$, at spatial position $\vec{r}$, with energy E moving in direction $\vec{\Omega}$ and at time t, is defined within an open, convex spatial domain $\mathcal{D}$, with boundary, $\partial\mathcal{D}$, by the general neutron transport equation:

$$\begin{aligned}\frac{1}{v(E)}\frac{\partial\Psi}{\partial t} + \vec{\Omega}\cdot\vec{\nabla}\Psi(\vec{r}, E, \vec{\Omega}, t) + \sigma_t(\vec{r}, E, t)\Psi(\vec{r}, E, \vec{\Omega}, t) &= Q_{ext}(\vec{r}, E, \vec{\Omega}, t) \\ &+ \frac{\chi(\vec{r}, E, t)}{4\pi}\int dE'\nu\sigma_f(\vec{r}, E', t)\int d\Omega'\Psi(\vec{r}, E', \vec{\Omega}', t) \\ &+ \int dE'\int d\Omega'\sigma_s(E'\to E, \Omega'\to\Omega, t)\Psi(\vec{r}, E', \vec{\Omega}', t)\end{aligned} \tag{2.2}$$

with the following, general boundary condition:

$$\Psi(\vec{r},E,\vec{\Omega},t)=\Psi^{inc}(\vec{r},E,\vec{\Omega},t)+\int dE'\int d\Omega'\gamma(\vec{r},E'\rightarrow E,\vec{\Omega}'\rightarrow\vec{\Omega},t)\Psi(\vec{r},E',\vec{\Omega}',t)$$
$$\text{for } \vec{r}\in\partial\mathcal{D}^{-}\left\{\partial\mathcal{D},\vec{\Omega}'\cdot\vec{n}<0\right\}. \tag{2.3}$$

In Eqs. (2.2) and (2.3), the physical properties of the system are defined as the following: $\sigma_t(\vec{r},E,t)$ is the total neutron cross section, $\chi(\vec{r},E,t)$ is the neutron fission spectrum, $\sigma_f(\vec{r},E',t)$ is the fission cross section, $\nu(\vec{r},E',t)$ is the average number of neutrons emitted per fission, $\sigma_s(E'\rightarrow E,\Omega'\rightarrow\Omega,t)$ is the differential scattering cross section, $Q_{ext}(\vec{r},E,\vec{\Omega},t)$ is a distributed external source, $\Psi^{inc}(\vec{r},E,\vec{\Omega},t)$ is the incident boundary source, and $\gamma(\vec{r},E'\rightarrow E,\vec{\Omega}'\rightarrow\vec{\Omega},t)$ is the boundary albedo.

We define the operator notation of Eq. (2.2):

$$\frac{1}{\mathbf{v}}\frac{\partial\Psi}{\partial t}+\mathbf{L}\Psi=\mathbf{F}\Psi+\mathbf{S}\Psi+\mathbf{Q}, \tag{2.4}$$

by dropping the dependent variable parameters and using the following operators:

$$\begin{aligned}
\mathbf{L}\Psi&=\vec{\Omega}\cdot\vec{\nabla}\Psi(\vec{r},E,\vec{\Omega},t)+\sigma_t(\vec{r},E,t)\Psi(\vec{r},E,\vec{\Omega},t),\\
\mathbf{F}\Psi&=\frac{\chi(\vec{r},E,t)}{4\pi}\int dE'\nu\sigma_f(\vec{r},E',t)\int d\Omega'\Psi(\vec{r},E',\vec{\Omega}',t),\\
\mathbf{S}\Psi&=\int dE'\int d\Omega'\sigma_s(E'\rightarrow E,\Omega'\rightarrow\Omega,t)\Psi(\vec{r},E',\vec{\Omega}',t),\\
\mathbf{Q}&=Q_{ext}(\vec{r},E,\vec{\Omega},t),
\end{aligned} \tag{2.5}$$

where $\mathbf{L}$ is the loss operator which includes total reaction and streaming, $\mathbf{F}$ is the fission operator, and $\mathbf{S}$ is the scattering operator. If we wish to analyze a transport problem at steady-state conditions, we simply omit the temporal derivative to form

$$\mathbf{L}\Psi=\mathbf{F}\Psi+\mathbf{S}\Psi+\mathbf{Q}, \tag{2.6}$$

and note that the operators of Eq. (2.5) no longer depend on time, t.

There is a special subset of transport problems that is routinely analyzed to determine the neutron behavior of a fissile system called the *k-eigenvalue problem*. In Eq. (2.2), $\nu(\vec{r}, E)$ acts as a multiplicative factor on the number of neutrons emitted per fission event. We replace this multiplicative factor in the following manner:

$$\nu(\vec{r}, E) \rightarrow \frac{\tilde{\nu}(\vec{r}, E)}{k}, \tag{2.7}$$

where we have introduced the eigenvalue, k. By also dropping the external source term, the steady-state neutron transport equation in Eq. (2.6) can be rewritten into

$$(\mathbf{L} - \mathbf{S})\,\tilde{\Psi} = \frac{1}{k}\mathbf{F}\tilde{\Psi}, \tag{2.8}$$

where $(k, \tilde{\Psi})$ forms an appropriate eigenvalue-eigenvector pair. Of most interest is the eigenpair corresponding to the eigenvalue of largest magnitude.

We can then gain knowledge of the behavior of the neutron population in the problem by taking the full phase-space integrals of the loss operator $\int\int\int \mathbf{L}\tilde{\Psi}\, dE\, d\Omega\, d^3r$, the fission operator $\int\int\int \mathbf{F}\tilde{\Psi}\, dE\, d\Omega\, d^3r$, and the scattering operator $\int\int\int \mathbf{S}\tilde{\Psi}\, dE\, d\Omega\, d^3r$. With the appropriate eigenvector solution, $\tilde{\Psi}$, the k eigenvalue then has the meaning as the multiplicative value which balances Eq. (2.8) in an integral sense. This means that k also has a physical meaning as well. A value $k < 1$ is called subcritical and corresponds to a system whose neutron population decreases in time; a value $k = 1$ is called critical and corresponds to a system whose neutron population remains constant in time; and a value $k > 1$ is called supercritical and corresponds to a system whose neutron population increases in time [10].

2.3 Energy Discretization

We begin our discretization procedures by focusing on the angular flux's energy variable. An ubiquitous energy discretization procedure in the transport community is the multigroup method [11, 12]. The multigroup method is defined by splitting the angular flux solution into G number of distinct, contiguous, and non-overlapping energy intervals called groups. We begin by restricting the full energy domain, $[0, \infty)$, into a finite domain, $E \in [E_G, E_0]$. E_0 corresponds to some maximum energy value and E_G corresponds to some minimum energy value (typically 0). We have done this by defining $G+1$ discrete energy values that are in a monotonically continuous reverse order: $E_G < E_{G-1} < \ldots < E_1 < E_0$.

From this distribution of energy values, we then say that a particular energy group, g, corresponds to the following energy interval:

$$\Delta E_g \in [E_g, E_{g-1}]. \tag{2.9}$$

Figure 2.1 provides a visual representation between the $G+1$ discrete energy values and the G energy groups. While the order that we have prescribed may seem illogical (high-to-low) to those outside of the radiation physics community, it has been historically applied this way because radiation transport problems are iteratively solved from high energy to low energy.

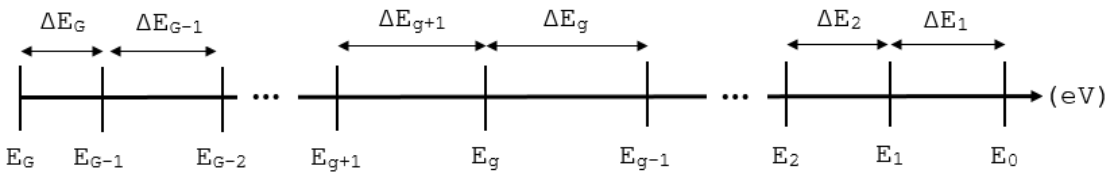


Figure 2.1: Interval structure of the multigroup methodology.

For the remainder of this energy discretization procedure, we will utilize the steady-state form of the transport equation in Eq. (2.6). The time-dependent and eigenvalue forms are analagous and would be derived identically. Taking the energy interval for group g as defined in Eq. (2.9), the energy-integrated angular flux of group g is

$$\Psi_g(\vec{r},\vec{\Omega}) = \int_{E_g}^{E_{g-1}} \Psi(\vec{r},E,\vec{\Omega})\,dE. \tag{2.10}$$

We can then use the energy-integrated angular flux to form the following coupled, $(g = 1,...,G)$, discrete equations (we have dropped the spatial parameter and some of the angular parameters for further clarity):

$$\left(\vec{\Omega}\cdot\vec{\nabla}+\sigma_{t,g}\right)\Psi_g = \sum_{g'=1}^{G}\left[\frac{\chi_g}{4\pi}\nu\sigma_{f,g'}\int_{4\pi}\Psi_{g'}(\vec{\Omega}')\,d\Omega' + \int_{4\pi}\sigma_s^{g'\to g}(\vec{\Omega}',\vec{\Omega})\Psi_{g'}(\vec{\Omega}')\,d\Omega'\right]+Q_g \tag{2.11}$$

where

$$\begin{aligned}
\sigma_{t,g}(\vec{r}) &\equiv \frac{\int_{E_g}^{E_{g-1}}\sigma_t(\vec{r},\vec{\Omega},E)\int_{4\pi}\Psi(\vec{r},\vec{\Omega},E)\,dEd\Omega}{\int_{E_g}^{E_{g-1}}\int_{4\pi}\Psi(\vec{r},\vec{\Omega},E)\,dEd\Omega}\\
\nu\sigma_{f,g}(\vec{r}) &\equiv \frac{\int_{E_g}^{E_{g-1}}\nu\sigma_f(\vec{r},E)\int_{4\pi}\Psi(\vec{r},\vec{\Omega},E)\,dEd\Omega}{\int_{E_g}^{E_{g-1}}\int_{4\pi}\Psi(\vec{r},\vec{\Omega},E)\,dEd\Omega}\\
\chi_g &\equiv \int_{E_g}^{E_{g-1}}\chi(\vec{r},E)\,dE\\
\sigma_s^{g'\to g}(\vec{r},\vec{\Omega}',\vec{\Omega}) &\equiv \frac{\int_{E_{g'}}^{E_{g'-1}}\left[\int_{E_g}^{E_{g-1}}\sigma_s(\vec{r},E'\to E,\vec{\Omega}',\vec{\Omega})\,dE\right]\Psi(\vec{r},\vec{\Omega}',E')\,dE'}{\int_{E_g}^{E_{g-1}}\Psi(\vec{r},\vec{\Omega},E)\,dE}\\
Q_g(\vec{r},\vec{\Omega}) &\equiv \int_{E_g}^{E_{g-1}}Q(\vec{r},\vec{\Omega},E)\,dE
\end{aligned} \tag{2.12}$$

The above equations are mathematically exact to those presented in Eqs. (2.2 - 2.6) and we have made no approximations at this time. However, this requires full knowledge of the energy distribution of the angular flux solution at all positions in our problem domain since we weight the multigroup cross sections with this solution. This is obviously impossible since the energy distibution is part of the solution space we are trying to solve for. Instead, we now define the process to make the multigroup discretization an effective approximation method.

We first define an approximate angular flux distribution for a region s:

$$\Psi(\vec{r},\vec{\Omega},E) = \hat{\Psi}(\vec{r},\vec{\Omega})f_s(E), \tag{2.13}$$

which is a factorization of the angular flux solution into a region-dependent energy function, $f_s(E)$, and a spatially/angularly dependent function, $\hat{\Psi}(\vec{r},\vec{\Omega})$. With this approximation, we can redefine the energy-collapsed cross sections of Eq. (2.12):

$$\begin{aligned}
\sigma_{t,g}(\vec{r}) &\equiv \frac{\int_{E_g}^{E_{g-1}} \sigma_t(\vec{r},E)f_s(E)\,dE}{\int_{E_g}^{E_{g-1}} f_s(E)\,dE}, \\
\nu\sigma_{f,g}(\vec{r}) &\equiv \frac{\int_{E_g}^{E_{g-1}} \nu\sigma_f(\vec{r},E)f_s(E)\,dE}{\int_{E_g}^{E_{g-1}} f_s(E)\,dE}, \\
\sigma_s^{g'\to g}(\vec{r},\vec{\Omega}',\vec{\Omega}) &\equiv \frac{\int_{E_{g'}}^{E_{g'-1}} \left[\int_{E_g}^{E_{g-1}} \sigma_s(\vec{r},E'\to E,\vec{\Omega}',\vec{\Omega})\,dE\right] f_s(E')\,dE'}{\int_{E_g}^{E_{g-1}} f_s(E)\,dE}.
\end{aligned} \tag{2.14}$$

It is noted that we do not need to redefine the fission spectrum or the distributed external sources since they are not weighted with the angular flux solution. With this energy factorization, we would expect, in general, that the approximation error will tend to zero as the number of discrete energy groups increases (thereby making the energy bins thinner). This is especially true if the group structure is chosen with

many more bins in energy regions with large variations in the energy solution. For certain problems, the region-dependent energy function is well understood (*i.e.,* almost exactly known). This means, that for these problems, we can achieve reasonable solution accuracy with only a few groups where the energy bins of the multigroup discretization are well chosen.

2.4 Angular Discretization

Now that we have provided the discretization of the energy variable, we next focus on the discretization of the transport problem in angle. We will do this in two stages: 1) expand the scattering source and the distributed external source in spherical harmonics and 2) collocate the angular flux at the interpolation points of the angular trial space. We will perform these discretization procedures by taking the steady-state equation presented in Eq. (2.6), dropping spatial parameterization, combining the fission and external sources into a single term, and using only 1 energy group:

$$\vec{\Omega} \cdot \vec{\nabla}\Psi(\vec{\Omega}) + \sigma_t \Psi(\vec{\Omega}) = \int_{4\pi} d\Omega' \, \sigma_s(\vec{\Omega}' \rightarrow \vec{\Omega})\Psi(\vec{\Omega}') + Q(\vec{\Omega}). \tag{2.15}$$

We first develop an approximation for the scattering term in Eq. (2.15) by expanding the angular flux and the scattering cross section in spherical harmonics functions and Legendre polynomials, respectively. We begin by first assuming that the material is isotropic in relation to the radiation's initial direction. From this assumption, the parameterization of the scattering cross section can be written in terms of only the scattering angle, μ_0,

$$\sigma_s(\vec{\Omega}' \rightarrow \vec{\Omega}) = \frac{1}{2\pi}\sigma_s(\vec{\Omega}' \cdot \vec{\Omega}) = \frac{1}{2\pi}\sigma_s(\mu_0), \tag{2.16}$$

where $\mu_0 \equiv \vec{\Omega}' \cdot \vec{\Omega}$. With this simplification, the scattering cross section can now be expanded in an infinite series in terms of the Legendre polynomials,

$$\sigma_s(\vec{\Omega}' \rightarrow \vec{\Omega}) = \sum_{p=0}^{\infty} \frac{2p+1}{4\pi} \sigma_{s,p} P_p(\mu_0), \tag{2.17}$$

where $\sigma_{s,p}$ is the p angular moment of the scattering cross section. These angular moments of the scattering cross section have the form:

$$\sigma_{s,p} \equiv \int_{-1}^{1} d\mu_0 \, \sigma_s(\mu_0) P_p(\mu_0). \tag{2.18}$$

With the scattering cross section redefined, we can now expand the angular flux in terms of an infinite series of the spherical harmonics functions, Y,

$$\Psi(\vec{\Omega}) = \frac{1}{4\pi} \sum_{k=0}^{\infty} \sum_{n=-k}^{k} \Phi_{k,n} Y_{k,n}(\vec{\Omega}) \tag{2.19}$$

where the angular moments of the angular flux, $\Phi_{k,n}$, have the form:

$$\Phi_{k,n} \equiv \int_{4\pi} d\Omega \, \Psi(\vec{\Omega}) \, Y_{k,n}(\vec{\Omega}). \tag{2.20}$$

We note that the p and k orders of the scattering cross section and angular flux expansions, respectively, are not corresponding. We then take the scattering cross section expansion of Eq. (2.17) and the angular flux expansion of Eq. (2.19), and insert them into the original scattering term of the right-hand-side of Eq. (2.15). After significant algebra and manipulations, which we will not include here for brevity, the scattering term can be greatly simplified (the full details of this are located in Appendix **??**). Eq. (2.15) can now be written again with this alternate and simplified scattering term that is composed of the cross section and angular flux moments:

$$\vec{\Omega} \cdot \vec{\nabla} \Psi(\vec{\Omega}) + \sigma_t \Psi(\vec{\Omega}) = \sum_{p=0}^{\infty} \frac{2p+1}{4\pi} \sigma_{s,p} \sum_{n=-p}^{p} \Phi_{p,n} Y_{p,n}(\vec{\Omega}) + Q(\vec{\Omega}). \tag{2.21}$$

From the initial assumption of material isotropy (which may or may not be an approximation), the scattering term of Eq. (2.21) has introduced no approximation. Unfortunately, this form requires an infinite series expansion which we cannot use with only finite computational resources. Instead, we truncate the series at some maximum expansion order, N_p, which, in general, introduces an approximate form for the scattering. However, we note that if the problem scattering can be exactly captured with moments through order N_p, then we have introduced no approximation with this truncation. With this order of truncation, we again write the angularly continuous Eq. (2.21), but also fold the source term into the spherical harmonics expansion,

$$\vec{\Omega} \cdot \vec{\nabla} \Psi(\vec{\Omega}) + \sigma_t \Psi(\vec{\Omega}) = \sum_{p=0}^{N_p} \frac{2p+1}{4\pi} \sum_{n=-p}^{p} Y_{p,n}(\vec{\Omega}) \left[\sigma_{s,p} \Phi_{p,n} + Q_{p,n} \right]. \tag{2.22}$$

At this point, one may wonder why we have altered the scattering operator so that it is terms of moments of the scattering cross sections and the angular flux. The reason is two-fold which will also be discussed in further detail later in this chapter. First, it greatly reduces the phase space of the scattering cross sections. With proper preprocessing, the scattering cross sections can be simplified into just their Legendre moments, instead of having to store angle-to-angle quantities ($\vec{\Omega}' \rightarrow \vec{\Omega}$). For every group-to-group combination in energy ($g' \rightarrow g$), there are only the N_p moments of the scattering cross section. Secondly, the contribution of the angular flux into the

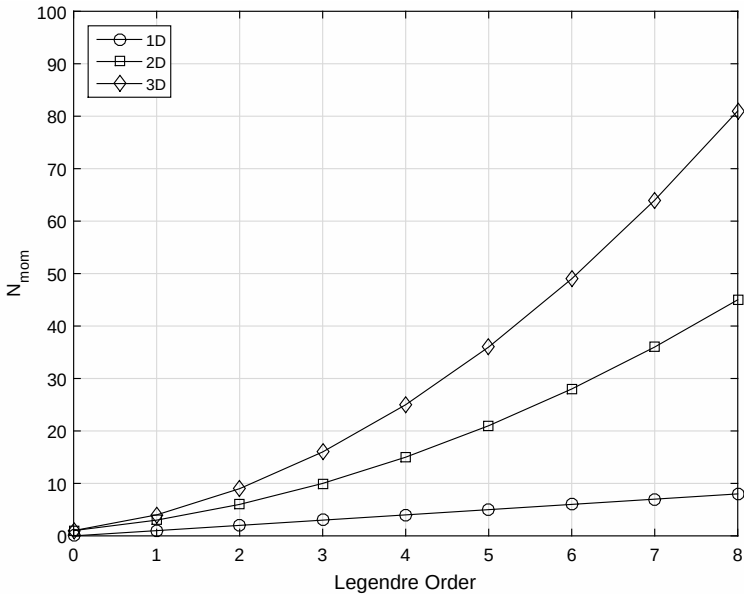


Figure 2.2: Number of spherical harmonics moments, N_{mom}, in 1D, 2D, and 3D as a function of the expansion order, p.

scattering source with its moments can also greatly reduce the dimensional space that needs to be stored in computer memory. This will be discussed later in further detail in Section 2.7.1, but it simply means that we only have to store N_{mom} angular flux moments for use in the scattering source. In 1 dimension, N_{mom} is equal to $(N_p + 1)$. In 2 dimensions, N_{mom} is equal to $\frac{(N_p+1)(N_p+2)}{2}$. In 3 dimensions, N_{mom} is equal to $(N_p + 1)^2$. For comparative purposes, we have plotted N_{mom} for 1, 2, and 3 dimensions up to order 8 in Figure 2.2.

Up to this point, we have only presented the methodology to express our source terms with expansions of the spherical harmonics functions. Next, we describe the second portion of our angular discretization by deriving the standard S_N equations using a collocation technique. We begin by choosing a set of M distinct points and weights to form a quadrature set in angular space: $\{\vec{\Omega}_m, w_m\}_{m=1}^{M}$. We will give further details about the required characteristics of this quadrature set as well as a

couple of common options a little later. Using this quadrature set, we can further define a trial space for the angular flux,

$$\Psi(\vec{\Omega}) = \sum_{m=1}^{M} B_m(\vec{\Omega})\Psi_m, \tag{2.23}$$

where the angular bases, B_m, satisfy the *Kronecker* property,

$$B_j(\vec{\Omega}_m) = \delta_{j,m}, \tag{2.24}$$

as well as the *Lagrange* property,

$$\sum_{m=1}^{M} B_m(\vec{\Omega}_m) = 1, \tag{2.25}$$

and the singular value of the angular flux along a given direction has the following notation:

$$\Psi_m = \Psi(\vec{\Omega}_m). \tag{2.26}$$

Next, we substitute Eq. (2.23) into Eq. (2.22), drop the external source for clarity, and collocate at the $(k = 1, ..., M)$ interpolation (quadrature) points,

$$\begin{aligned} &\vec{\Omega} \cdot \vec{\nabla} \left(\sum_{m=1}^{M} B_m(\vec{\Omega}_k)\Psi_m \right) + \sigma_t \left(\sum_{m=1}^{M} B_m(\vec{\Omega}_k)\Psi_m \right) \\ &= \sum_{p=0}^{N_p} \frac{2p+1}{4\pi} \sigma_{s,p} \sum_{n=-p}^{p} Y_{p,n}(\vec{\Omega}) \left(\sum_{m=1}^{M} \Psi_m \int_{4\pi} d\Omega \, B_m(\vec{\Omega}_k) \, Y_{p,n}(\vec{\Omega}) \right), \\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad k = 1, ..., M \end{aligned} \tag{2.27}$$

where we inserted Eq. (2.23) into Eq. (2.20) to form a slightly modified form for the

angular flux moments:

$$\Phi_{p,n} = \sum_{m=1}^{M} \Psi_m \int_{4\pi} d\Omega \, B_m(\vec{\Omega}) \, Y_{p,n}(\vec{\Omega}). \tag{2.28}$$

The *Kronecker* property of Eq. (2.24), is then used at the collocation points so that Eq. (2.27) can be simplified into the following form (where we have reintroduced the distributed source term in terms of its contribution for angle m):

$$\vec{\Omega}_m \cdot \vec{\nabla}\Psi_m + \sigma_t \Psi_m = \sum_{p=0}^{N_P} \frac{2p+1}{4\pi} \sigma_{s,p} \sum_{n=-p}^{p} Y_{p,n}(\vec{\Omega}_m)\Phi_{p,n} + Q_m \quad . \qquad m = 1, ..., M \tag{2.29}$$

Equation (2.29) represents the transport equation that has been discretized into M separate equations in angle (for 1 energy group and no spatial discretization). Up to this point, we have simply stated that there is some angular quadrature set composed of M directions and weights that will satisfy some conditions of the solution, but we have not explicitly stated these conditions. For this work we will require our angular quadrature set to maintain the following properties:

1. The weights can sum to 1 by some normalization procedure: $\sum_m w_m = 1$.
2. The odd angular moments sum to $\vec{0}$: $\sum_m w_m \left(\vec{\Omega}_m\right)^n = \vec{0}$ $(n = 1, 3, 5, ...)$.
3. $\sum_m w_m \vec{\Omega}_m \vec{\Omega}_m = \frac{1}{3}\mathbb{I}$, where $\mathbb{I}$ is the identity tensor.
4. The points and weights are symmetric about the primary axes in angular space.
5. The points and weights also need to have symmetry about the problem domain boundary (this is not an issue if the domain is a rectangle in 2D or an orthogonal parallelepiped in 3D). This point is important for reflecting boundary conditions and is described in greater detail in Section 2.5.

Point 4 requires some additional explanation. In 1 dimension, this corresponds to symmetry about the point 0 on the interval $[-1, 1]$. In 2 dimensions, this corresponds to quadrant-to-quadrant symmetry about the x-y primary axes of the unit circle. In 3 dimensions, this corresponds to octant-to-octant symmetry about the x-y-z primary axes of the unit sphere.

From these properties, especially property 4, our 2D and 3D quadrature sets can be constructed in a simple and consistent manner (1D quadrature sets have different construction and our work does not include them). For both 2D and 3D problems, we can generate a subset of the quadrature points and weights on a single octant of the unit sphere, where each quadrature point in this subset has the form: $\vec{\Omega} = [\mu, \eta, \xi]$. If we are solving a 2D problem, we would then project the quadrature points onto the $(0 < \theta < \frac{\pi}{2})$ portion of the unit circle so that they have the form: $\vec{\Omega} = [\mu, \eta]$ (we can then view the primary octant as the primary quadrant). Once we have defined the quadrature points and weights for the primary quadrant or octant, we can then directly calculate the remainder of the quadrature set by mapping to the other quadrants or octants. Table 2.1 presents the mapping from the primary quadrant to the other 3 quadrants for 2D problems. Table 2.2 presents the mapping from the primary octant to the other 7 octants for 3D problems. In these tables, the '1' subscript corresponds to those angles generated in the primary quadrant or octant.

Table 2.1: 2D angle mapping from the first quadrant into the other 3 quadrants.

Quadrant	μ	η
1	$\mu_1 = \mu_1$	$\eta_1 = \eta_1$
2	$\mu_2 = -\mu_1$	$\eta_2 = \eta_1$
3	$\mu_3 = -\mu_1$	$\eta_3 = -\eta_1$
4	$\mu_4 = \mu_1$	$\eta_4 = -\eta_1$

Table 2.2: 3D angle mapping from the first octant into the other 7 octants.

Octant	μ	η	ξ
1	$\mu_1 = \mu_1$	$\eta_1 = \eta_1$	$\xi_1 = \xi_1$
2	$\mu_2 = -\mu_1$	$\eta_2 = \eta_1$	$\xi_2 = \xi_1$
3	$\mu_3 = -\mu_1$	$\eta_3 = -\eta_1$	$\xi_3 = \xi_1$
4	$\mu_4 = \mu_1$	$\eta_4 = -\eta_1$	$\xi_4 = \xi_1$
5	$\mu_5 = \mu_1$	$\eta_5 = \eta_1$	$\xi_5 = -\xi_1$
6	$\mu_6 = -\mu_1$	$\eta_6 = \eta_1$	$\xi_6 = -\xi_1$
7	$\mu_7 = -\mu_1$	$\eta_7 = -\eta_1$	$\xi_7 = -\xi_1$
8	$\mu_8 = \mu_1$	$\eta_8 = -\eta_1$	$\xi_8 = -\xi_1$

We conclude our discussion of angular discretizations by presenting two common angular quadrature sets that will be employed in this dissertation work. Section 2.4.1 presents the Level Symmetric (LS) quadrature set and Section 2.4.2 presents the Product Gauss-Legendre-Chebyshev (PGLC) quadrature set. Both of these quadrature sets can be formed from the procedure outlined before: form the primary octant and then map appropriately.

2.4.1 Level Symmetric Quadrature Set

The first quadrature set we present is the common Level Symmetric set that has had extensive use in the radiation transport community [12, 13]. Its defining characteristic is the restriction that it is rotationally symmetric (invariant) about all three axes of the primary octant. This leads to a two-fold additional set of restrictions: 1) once the location of the first ordinate is selected, then all other ordinates are known; and 2) the weights can become negative. This negativity of the weights can be problematic and lead to unphysical solutions if the angular flux is not sufficiently smooth.

We begin our description of the LS quadrature by analyzing the 3D angular coordinate system for a particular direction, $\vec{\Omega}$, as depicted in Figure 2.3. The

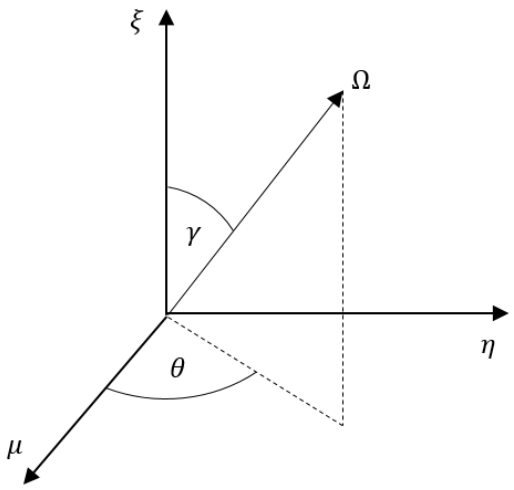


Figure 2.3: Angular coordinate system for the direction $\vec{\Omega}$.

angular direction, $\vec{\Omega} = [\vec{\Omega}_x, \vec{\Omega}_y, \vec{\Omega}_z]$, is typically described with its directional cosines: μ, η, and ξ. These are described by the angles θ and γ of the coordinate system, which are the azimuthal and polar angles, respectively, and allow us to give a functional form for each direction component:

$$\begin{aligned} \vec{\Omega}_x =& \mu = \cos(\theta)\sin(\gamma) = \cos(\theta)\sqrt{1-\xi^2} \\ \vec{\Omega}_y =& \eta = \sin(\theta)\sin(\gamma) = \sin(\theta)\sqrt{1-\xi^2}\ . \\ \vec{\Omega}_z =& \xi = \cos(\gamma) \end{aligned} \tag{2.30}$$

The direction cosines are related and necessarily must have a Euclidean norm of 1:

$$\mu^2 + \eta^2 + \xi^2 = 1. \tag{2.31}$$

We next specify the order of the quadrature set, N, which we restrict to only positive even integers. Each direction cosine (μ, η, and ξ) then contains exactly $N/2$ positive values with respect to each of the three axes. This leads to exactly $\frac{N(N+2)}{8}$ total angular directions in the primary octant. Because of the rotational invariance

of the quadrature set, no ordinate axis receives preferential clustering of the nodes. This means that the index value of each ordinate is identical:

$$\mu_i = \eta_i = \xi_i, \qquad i \in (1, N/2) \tag{2.32}$$

As previously stated, once the location of the first ordinate, μ_1, is selected, then the remaining are directly known. However, to maintain the relation of Eq. (2.31), this first ordinate has restrictions placed on it. It must maintain a positive value: $\mu_1^2 \in (0, 1/3]$. Also, for the $S2$ set (N=2), there is exactly one direction cosine with no degrees of freedom. This requires that $\mu_1^2 = 1/3$ for the $S2$ case.

With μ_1 now selected, we can consider an ordinate set $[\mu_i, \eta_j, \xi_k]$, where $i+j+k = N/2+2$. To maintain the appropriate Euclidean norm, a recursion relation can be derived (which we will not do for brevity):

$$\mu_i^2 = \mu_{i-1}^2 + \Delta \tag{2.33}$$

where the spacing constant, Δ, has the form:

$$\Delta = \frac{2(1-3\mu_1^2)}{N-2}. \tag{2.34}$$

Based on this recursion form, we can see that if μ_1^2 is close to 0, then the ordinates will be clustered around the poles of the primary octant. Likewise, if μ_1^2 is close to 1/3, then the ordinates will be clustered away from the poles. For this work, we choose to select values of μ_1 in conformance with the LQ_N quadrature set because they can exactly integrate the polynomials of the Legendre expansion of the scattering cross sections [14]. We finally note that the weights of the LQ_N set become negative for $N \geq 20$.

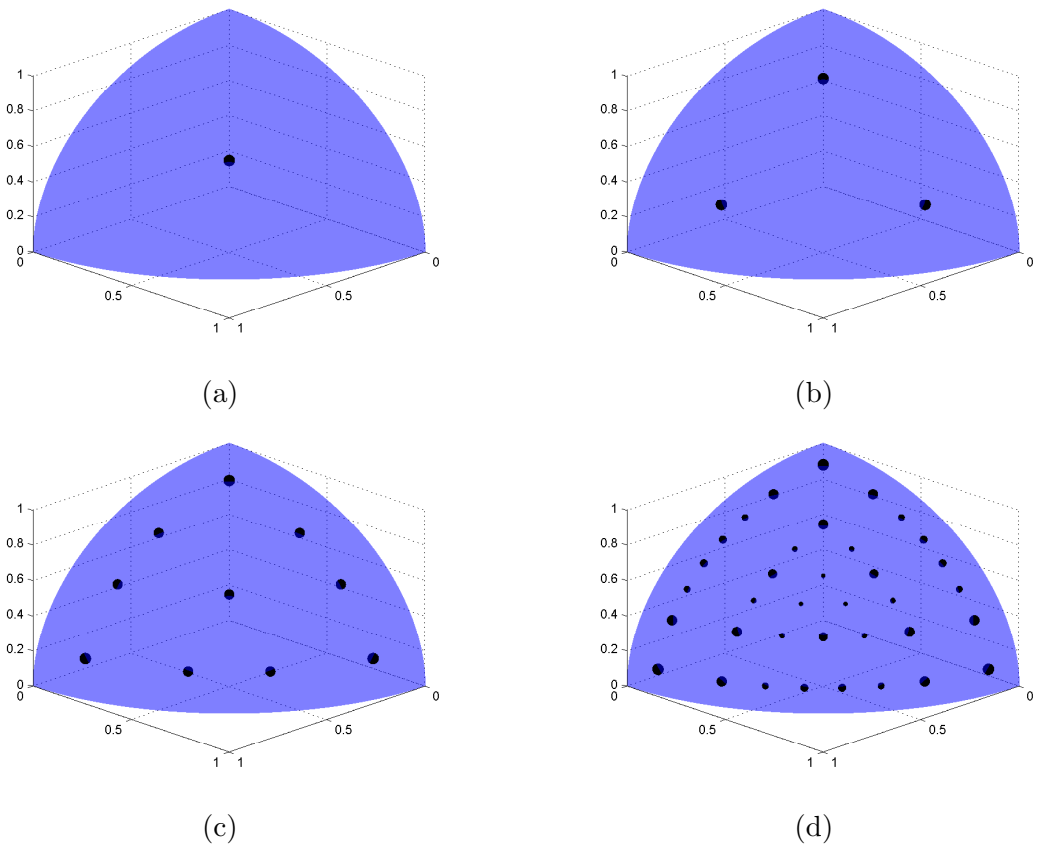


Figure 2.4: Level-Symmetric angular quadrature sets of order (a) 2, (b) 4, (c) 8, and (d) 16.

We conclude this discussion of the LS quadrature set with some examples. Figure 2.4 provides a visual depiction of the LS nodes and weights in the primary octant for varying orders. The magnitude of the weights is characterized by the relative size of the nodes. Figure 2.5 then provides the projection of the 3D LS quadrature set onto the unit circle for various orders for use in 2D problems. We have included the full quadrature set in this image including the quadrant-to-quadrant mapping.

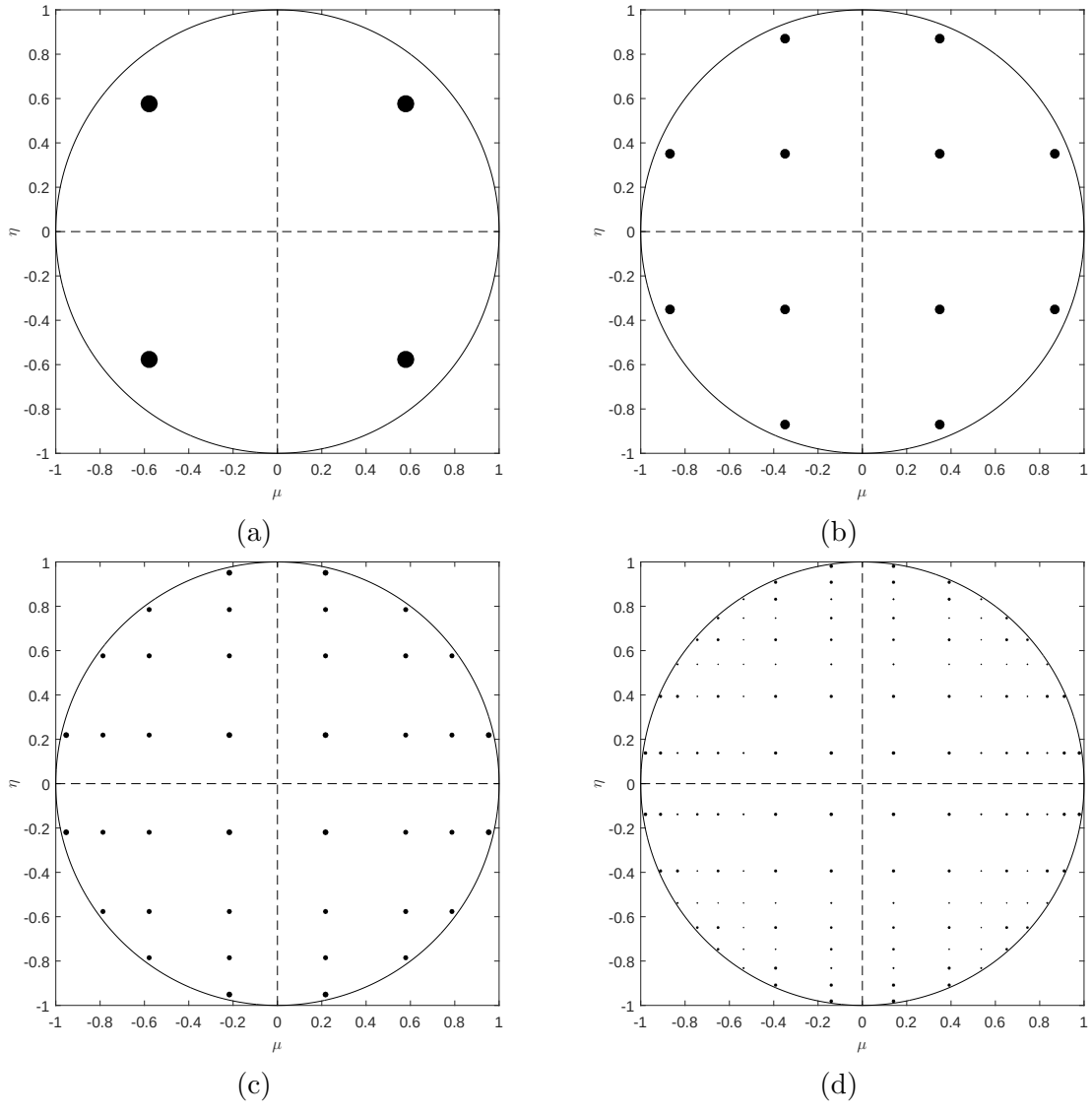


Figure 2.5: Projection of the 3D Level-Symmetric angular quadrature set with orders (a) 2, (b) 4, (c) 8, and (d) 16 onto the x-y space on the unit circle.

2.4.2 Product Gauss-Legendre-Chebyshev Quadrature Set

The second angular quadrature set we will present is a Product Gauss-Legendre-Chebyshev (PGLC) set [15]. It is formed by the product-wise multiplication of a Gauss-Chebyshev quadrature in the azimuthal direction and a Gauss-Legendre quadrature in the polar direction. It has the following key differences from the Level Symmetric set:

- Does not have 90^o rotational invariance within the primary octant; however, we still maintain octant-to-octant symmetry via mapping;
- Has more control over the placement of the anglular directions within the primary octant;
- Quadrature weights are aligned with the polar level;
- Has strictly positive weights for all polar and azimuthal combinations.

From the listed differences, we can already discern some clear advantages and disadvantages from a fully-symmetric quadrature set like LS. If a high number of angles are required for a problem, then negative weights do not arise. This is beneficial for transport problems with significant discontinuities. Also, the quadrature directions can be preferentially distributed in the primary octant if required for a particular problem. For example, if the transport solution is smoothly varying in the polar direction and not in the azimuthal direction, then we can specify a larger number of quadrature points in the azimuthal direction, with much fewer points in the polar direction. However, this also highlights the fact that the quadrature weights are aligned with the polar level, which can lead to less accurate moment integrations for certain transport problems.

Because the PGLC quadrature set is formed by product-wise multiplication, we simply need to specify the component nodes and weights in both the azimuthal and polar directions to fully define all ordinates in the primary octant. The azimuthal direction, θ, uses the positive range of the Gauss-Chebyshev quadrature set [16]. With the azimuthal direction restricted to its positive values in the primary octant, this corresponds to the upper-right portion of the unit circle: $\theta \in [0, \pi/2]$. If we specify A azimuthal directions for our quadrature set in the primary octant, then the azimuthal nodes and weights can be directly stated as

$$\theta_m = \frac{2m-1}{4A}\pi \qquad \text{and} \qquad w_m = \frac{\pi}{2A}, \tag{2.35}$$

respectively.

For the polar direction, a Gauss-Legendre quadrature set is used [17]. Similar to the azimuthal direction, we restrict the integration of the polar direction to its positive values: $\xi \in (0, 1)$. If we specify P polar directions, then the cosine of the polar nodes, ξ, of our quadrature set are the positive roots of the $2P$-order Legendre polynomials taken over the inveral $[-1, 1]$. In this case, we simply discard the negative roots. The corresponding Legendre weights are given by the following formula,

$$w_n = \frac{2}{(1-\xi_n^2)(L'_{2P}(\xi_n))^2}, \tag{2.36}$$

where L'_{2P} is the derivative of the $2P$-order Legendre polynomial.

With the azimuthal directions specified by Eq. (2.35) and the polar cosines specified by the Legendre polynomial roots, any ordinate can now be determined by the definition of the angular directions in Eq. (2.30). The ordinate weights can be specified in a similar manner. From Eqs. (2.35) and (2.36), any ordinate weight,

$w_{m,n}$, can be calculated by the pairwise products of the azimuthal and polar weights: $w_{m,n} = w_m w_n$. This means that we can specify the integral, F, of some function $f(\theta, \gamma)$ over the primary octant of the unit sphere,

$$F = \sum_{m=1}^{A} \sum_{n=1}^{P} w_m w_n f(\theta_m, \gamma_n). \tag{2.37}$$

For this dissertation, we will use the following notation to define the product nature of the PGLC quadrature points: S_A^P. Here, A and P correspond to the number of azimuthal and polar directions in the primary octant, respectively. We demonstrate this definition in Figure 2.6 for the primary octant with several combinations of azimuthal and polar directions. Figure 2.7 then presents the projections of these quadrature sets onto the unit circle for use in 2D transport problems. Again, the size of the direction marker corresponds to the relative weight of the quadrature point. One can clearly see that the weights vary on the polar levels, and all azimuthal weights on a given polar level are constant.

2.5 Boundary Conditions

Using the energy and angular discretizations presented in Sections 2.3 and 2.4, respectively, we write the standard, steady-state, multigroup S_N transport equation for one angular direction, m, and one energy group, g:

$$\left(\vec{\Omega}_m \cdot \vec{\nabla} + \sigma_{t,g}\right) \Psi_{m,g} = \sum_{g'=1}^{G} \sum_{p=0}^{N_P} \frac{2p+1}{4\pi} \sigma_{s,p}^{g' \to g} \sum_{n=-p}^{p} \Phi_{p,n,g'} Y_{p,n}(\vec{\Omega}_m) + \frac{\chi_g}{4\pi} \sum_{g'=1}^{G} \nu \sigma_{f,g'} \Phi_{g'} + Q_{m,g}, \tag{2.38}$$

where we have dropped the spatial parameter for clarity and is beholden to the

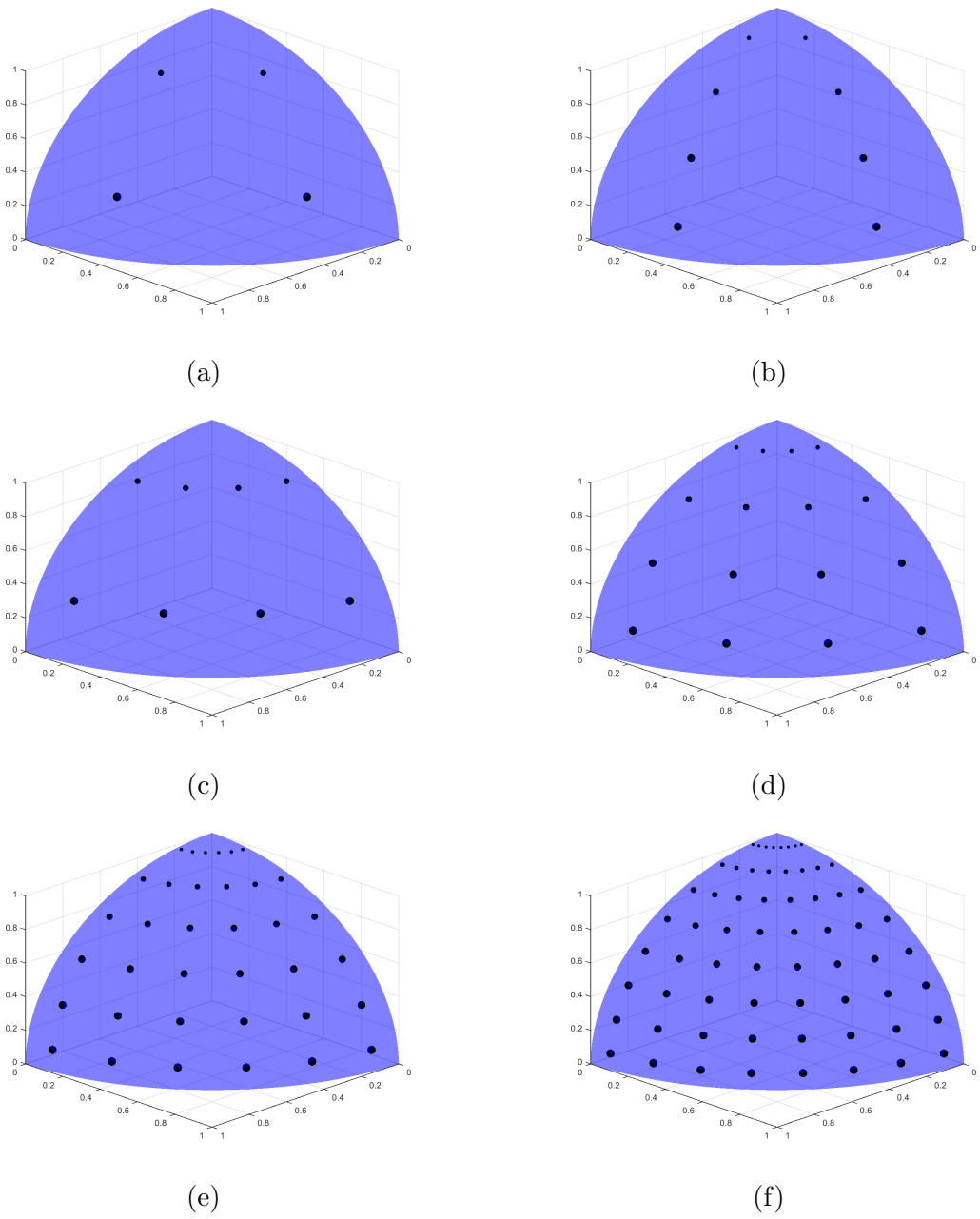


Figure 2.6: Product Gauss-Legendre-Chebyshev angular quadrature set with orders: (a) S_2^2, (b) S_2^4, (c) S_4^2, (d) S_4^4, (e) S_6^6, and (f) S_8^8.

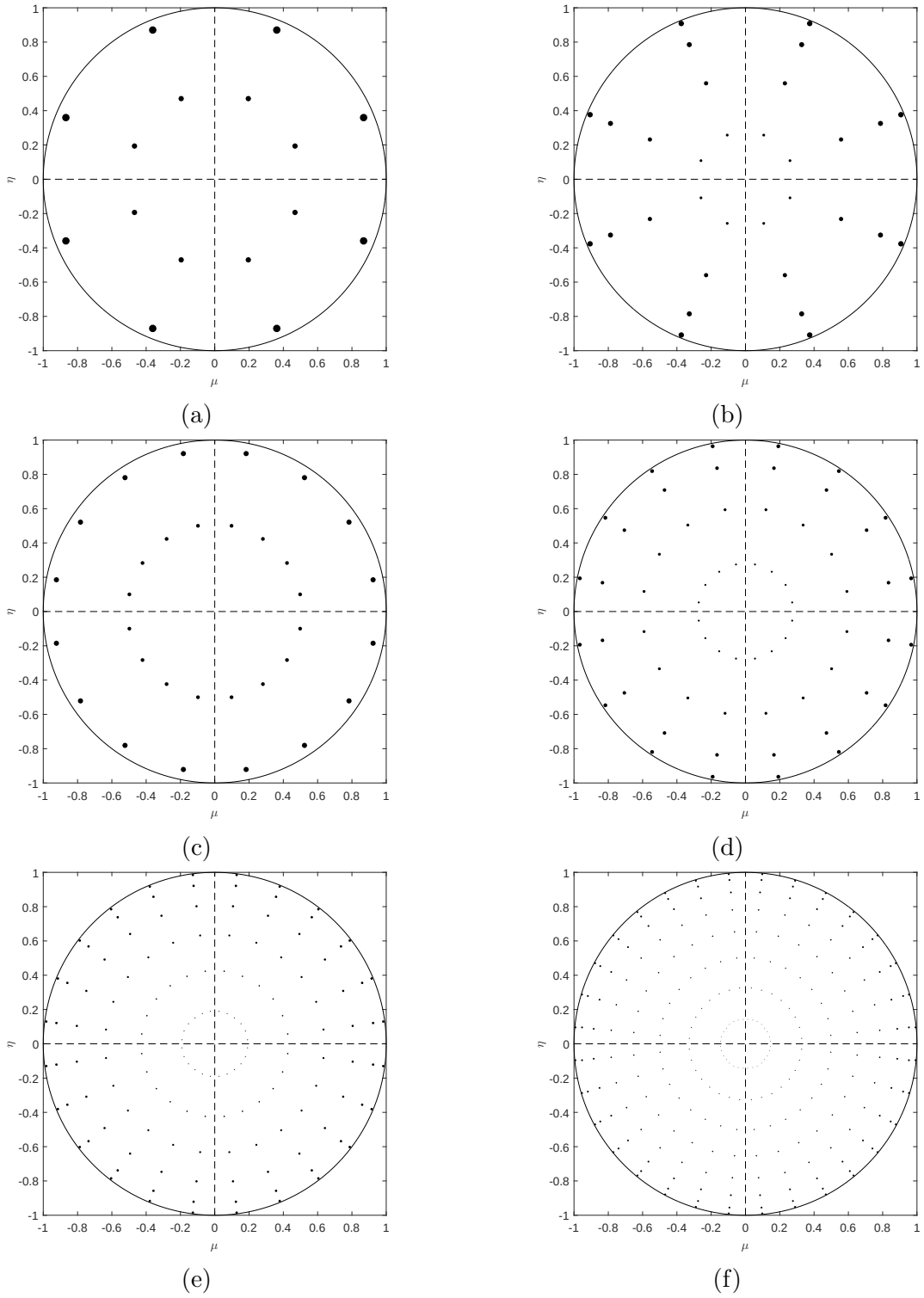


Figure 2.7: Projection of the 3D Product Gauss-Legendre-Chebyshev angular quadrature set with orders: (a) S_2^2, (b) S_2^4, (c) S_4^2, (d) S_4^4, (e) S_6^6, and (f) S_8^8 onto the x-y space on the unit circle.

following general, discretized boundary condition:

$$\Psi_{m,g}(\vec{r}) = \Psi^{inc}_{m,g}(\vec{r}) + \sum_{g'=1}^{G} \sum_{\vec{\Omega}_{m'}\cdot\vec{n}>0} \gamma^{m'\to m}_{g'\to g}(\vec{r})\Psi_{m',g'}(\vec{r}). \tag{2.39}$$

These (MxG) number of discrete, tightly-coupled equations are currently defined as continuous in space.

For this dissertation work, we will consider only one type of boundary conditions: Dirichlet-type boundaries (also called *first-type boundary condition* in some physics and mathematical communities). In particular, we will only utilize incoming-incident and reflecting boundary conditions which correspond to $\vec{r} \in \partial\mathcal{D}^d$ and $\vec{r} \in \partial\mathcal{D}^r$, respectively. The full domain boundary is then the union: $\partial\mathcal{D} = \partial\mathcal{D}^d \cup \partial\mathcal{D}^r$. This leads to the boundary condition being succinctly written for one angular direction, m, and one energy group, g as

$$\Psi_{m,g}(\vec{r}) = \begin{cases} \Psi^{inc}_{m,g}(\vec{r}), & \vec{r} \in \partial\mathcal{D}^d \\ \Psi_{m',g}(\vec{r}), & \vec{r} \in \partial\mathcal{D}^r \end{cases} \tag{2.40}$$

where the reflecting angle is $\vec{\Omega}_{m'} = \vec{\Omega}_m - 2\left(\vec{\Omega}_m \cdot \vec{n}\right)\vec{n}$ and $\vec{n}$ is oriented outward from the domain. To properly utilize the reflecting boundary condition that we have proposed, the angular quadrature set defined in Section 2.4 needs the following properties:

1. The reflected directions, $\vec{\Omega}_{m'}$, are also in the quadrature set for all $\vec{r} \in \partial\mathcal{D}^r$.

2. The weights of the incident, w_m, and reflected, $w_{m'}$, angles must be equal.

For problems where the reflecting boundaries align with the x, y, z axes, this will not be an issue with standard quadrature sets (*e.g.,* level-symmetric or Gauss-Legendre-

Chebyshev). However, if the reflecting boundaries do not align in this manner, then additional care must be employed in calculating appropriate angular quadrature sets.

2.6 Spatial Discretization

For the spatial discretization of the problem domain, we simplify Eq. (2.38) into a single energy group and drop the fission term (it can be lumped into the 0th order external source term and will act similarly to the total interaction term)

$$\vec{\Omega}_m \cdot \vec{\nabla}\Psi_m + \sigma_t \Psi_m = \sum_{p=0}^{N_P} \frac{2p+1}{4\pi} \sum_{n=-p}^{p} Y_{p,n}(\vec{\Omega}_m) \left[\sigma_{s,p}\Phi_{p,n,} + Q_{p,n}\right] \tag{2.41}$$

to form M ($m = 1, ..., M$) angularly discrete equations. We then lay down an unstructured mesh, $\mathbb{T}_h \in \mathbb{R}^d$, over the spatial domain, where d is the dimensionality of the problem ($d = 1, 2, 3$). This mesh consists of non-overlapping spatial elements to form a complete union over the entire spatial domain: $\mathcal{D} = \bigcup_{K \in \mathbb{T}_h} K$. To form the DGFEM set of equations [2, 18], we consider a spatial cell $K \in \mathbb{R}^d$ which has N_V^K vertices and N_f^K faces. Each face of cell K resides in dimension $\mathbb{R}^{d-1}$ and is formed by a connection of a subset of vertices. For a 1D problem, each face is a single point. For a 2D problem, each face is a line segment connecting two distinct points. For a 3D problem, each face is a $\mathbb{R}^2$ closed polygon (not necessarily coplanar) which may or may not be convex. An example of this interconnection between elements is given for a $\mathbb{R}^2$ problem in Figure 2.8 between our cell of interest, K, and another cell, K', separated by the face f. We have chosen the normal direction of the face to have orientation from cell K to cell K' while we form the DGFEM equations for cell K. This means that if we were instead analyzing cell K', then the face f normal, $\vec{n}'$, would be opposite (*i.e.*, $\vec{n}' = -\vec{n}$).

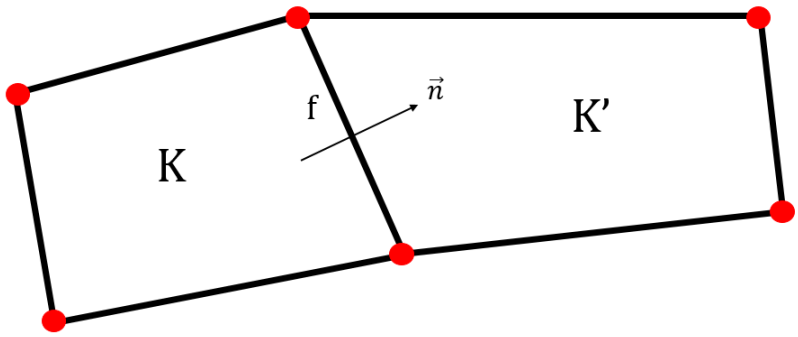


Figure 2.8: Two cells of the spatial discretization with the connecting face, f, with normal direction, $\vec{n}$, oriented from cell K to cell K'.

Next, we left-multiply Eq. (2.41) by an appropriate test function b_m, integrate over cell K, and apply Gauss' Divergence Theorem to the streaming term to obtain the Galerkin weighted-residual for cell K for an angular direction $\vec{\Omega}_m$:

$$\begin{aligned} &-\left(\vec{\Omega}_m \cdot \vec{\nabla} b_m, \Psi_m\right)_K + \sum_{f=1}^{N_f^K} \left\langle \left(\vec{\Omega}_m \cdot \vec{n}\right) b_m, \tilde{\Psi}_m \right\rangle_f + \left(\sigma_t b_m, \Psi_m\right)_K \\ &= \sum_{p=0}^{N_P} \sum_{n=-p}^{p} \frac{2p+1}{4\pi} Y_{p,n}(\vec{\Omega}_m) \left[\left(\sigma_{s,p}\, b_m, \Phi_{p,n,}\right)_K + \left(b_m, Q_{p,n}\right)_K \right] \end{aligned} . \tag{2.42}$$

The cell boundary fluxes, $\tilde{\Psi}_m$, will depend on the cell boundary type and will be defined shortly. The cell boundary $\partial\mathcal{D}_K = \bigcup_{f \in N_f^K} f$ is the closed set of the N_f^K faces of the geometric cell. The inner products:

$$\left(u, v\right)_K \equiv \int_K u\, v\, dr \tag{2.43}$$

and

$$\left\langle u, v \right\rangle_f \equiv \int_f u\, v\, ds \tag{2.44}$$

correspond to integrations over the cell and faces, respectively, where $dr \in \mathbb{R}^d$ is within the cell and $ds \in \mathbb{R}^{d-1}$ is along the cell boundary. We note that we will use this notation of the inner product for the remainder of the dissertation unless otherwise stated. We then separate the summation of the cell K boundary integration terms into two different types: outflow boundaries ($\partial K^+ = \{\vec{r} \in \partial K : \vec{n}(\vec{r}) \cdot \vec{\Omega}_m > 0\}$) and inflow boundaries ($\partial K^- = \{\vec{r} \in \partial K : \vec{n}(\vec{r}) \cdot \vec{\Omega}_m < 0\}$). The inflow boundaries can further be separated into inflow from another cell: $\partial K^- \backslash \partial \mathcal{D}$; inflow from incident flux on the domain boundary: $\partial K^- \cap \partial \mathcal{D}^d$; and reflecting domain boundaries: $\partial K^- \cap \partial \mathcal{D}^r$. At this point, we note that the derivation can comprise an additional step by using Gauss' Divergence Theorem again on the streaming term. This is sometimes performed for radiation transport work so that mass matrix lumping can be performed, but we will not do so here at this time. Therefore, with the cell boundary terminology as proposed, Eq. (2.42) can be written into the following form:

$$\begin{aligned} &- \left(\vec{\Omega}_m \cdot \vec{\nabla} b_m, \Psi_m\right)_K + \left(\sigma_t b_m, \Psi_m\right)_K \\ &+ \left\langle \left(\vec{\Omega}_m \cdot \vec{n}\right) b_m, \tilde{\Psi}_m \right\rangle_{\partial K^+} + \left\langle \left(\vec{\Omega}_m \cdot \vec{n}\right) b_m, \tilde{\Psi}_m \right\rangle_{\partial K^- \backslash \partial \mathcal{D}} \\ &+ \left\langle \left(\vec{\Omega}_m \cdot \vec{n}\right) b_m, \tilde{\Psi}_m \right\rangle_{\partial K^- \cap \partial \mathcal{D}^d} + \left\langle \left(\vec{\Omega}_m \cdot \vec{n}\right) b_m, \tilde{\Psi}_m \right\rangle_{\partial K^- \cap \partial \mathcal{D}^r} \\ &= \sum_{p=0}^{N_P} \sum_{n=-p}^{p} \frac{2p+1}{4\pi} Y_{p,n}(\vec{\Omega}_m) \left[\left(\sigma_{s,p}\, b_m, \Phi_{p,n,}\right)_K + \left(b_m, Q_{p,n}\right)_K \right] \end{aligned} \tag{2.45}$$

We can now deal with the boundary fluxes, $\tilde{\Psi}_m$, by enforcing the ubiquitously-used *upwind scheme*. In simple nomenclature, the upwind scheme corresponds to using the angular flux values within the cell for outflow boundaries and angular flux values outside the cell for inflow boundaries. Mathematically, the upwind scheme can succinctly be written as the following for all boundary types,

$$\tilde{\Psi}_m(\vec{r}) = \begin{cases} \Psi_m^-, & \partial K^+ \\ \Psi_m^+, & \partial K^- \backslash \partial \mathcal{D} \\ \Psi_m^{inc}, & \partial K^- \cap \partial \mathcal{D}^d \\ \Psi_{m'}^-, & \partial K^- \cap \partial \mathcal{D}^r \end{cases}, \tag{2.46}$$

when the following trace is applied to the angular fluxes :

$$\Psi_m^{\pm}(\vec{r}) \equiv \lim_{s \to 0^{\pm}} \Psi_m\Big(\vec{r} + s(\vec{\Omega}_m \cdot \vec{n})\vec{n}\Big). \tag{2.47}$$

This trace has the notation, with $\vec{n}$ pointing outwards from cell K, of Ψ_m^- corresponding to fluxes within the cell and Ψ_m^+ corresponding to fluxes out of the cell. Now, using the upwind scheme as previously defined, we can write our complete set of DGFEM equations for cell K as

$$\begin{aligned} &-\Big(\vec{\Omega}_m \cdot \vec{\nabla} b_m, \Psi_m\Big)_K + \Big(\sigma_t b_m, \Psi_m\Big)_K + \Big\langle (\vec{\Omega}_m \cdot \vec{n})\, b_m, \Psi_m^- \Big\rangle_{\partial K^+} \\ &\quad + \Big\langle (\vec{\Omega}_m \cdot \vec{n})\, b_m, \Psi_m^+ \Big\rangle_{\partial K^- \backslash \partial \mathcal{D}} + \Big\langle (\vec{\Omega}_m \cdot \vec{n})\, b_m, \Psi_{m'}^- \Big\rangle_{\partial K^- \cap \partial \mathcal{D}^r} \\ &\quad = \sum_{p=0}^{N_P} \sum_{n=-p}^{p} \frac{2p+1}{4\pi} Y_{p,n}(\vec{\Omega}_m) \left[\Big(\sigma_{s,p}\, b_m, \Phi_{p,n,}\Big)_K + \big(b_m, Q_{p,n}\big)_K \right] \cdot \\ &\quad - \Big\langle (\vec{\Omega}_m \cdot \vec{n})\, b_m, \Psi_m^{inc} \Big\rangle_{\partial K^- \cap \partial \mathcal{D}^d} \end{aligned} \tag{2.48}$$

We note that fluxes without the trace superscript are all within the cell. By completely defining our mathematical formulation for an arbitrary spatial cell, it is easy to see that the full set of equations to define our discretized solution space for a single angle and energy group comprises of a simple double integration loop. The full set of equations can be formed by looping over all spatial cells, $\mathcal{D} = \bigcup_{K \in \mathbb{T}_h} K$,

while further looping over all faces within each cell, $\partial\mathcal{D}_K = \bigcup_{f\in N_f^K} f$.

2.6.1 Convergence Rates of the DGFEM S_N Equation

Because we seek to investigate the use of high-order spatial basis functions for the transport equation, we need to form an estimate of the spatial error based on some measure of the mesh. We do this by taking Eq. (2.48), performing another integration-by-parts on the streaming term, multiplying by the angular weight, w_m, and summing over all elements and all angular directions. We also change the notation of the test function from b_m to Ψ_m^* to ease notation at a later step. This leads to the variational form for the 1-group S_N equation:

$$\begin{aligned}
&\sum_{m=1}^{M} w_m \Big[\Big(\Psi_m^*, \vec{\Omega}_m\cdot\vec{\nabla}\Psi_m\Big)_{\mathcal{D}} + \Big(\sigma_t\Psi_m^*, \Psi_m\Big)_{\mathcal{D}}\Big] \\
&+\sum_{m=1}^{M} w_m \Big\langle (\vec{\Omega}_m\cdot\vec{n})\,\Psi_m^{*+}, [\![\Psi_m]\!]\Big\rangle_{E_h^i} \\
&+\sum_{m=1}^{M} w_m \Big[\Big\langle (\vec{\Omega}_m\cdot\vec{n})\,\Psi_m^*, \Psi_{m'}\Big\rangle_{\partial\mathcal{D}_m^{r-}} - \Big\langle (\vec{\Omega}_m\cdot\vec{n})\,\Psi_m^*, \Psi_m\Big\rangle_{\partial\mathcal{D}_m^-}\Big] \\
&=\sum_{m=1}^{M} w_m \sum_{p=0}^{N_P}\sum_{n=-p}^{p} \frac{2p+1}{4\pi} Y_{p,n}(\vec{\Omega}_m)\Big(\Psi_m^*, \sigma_{s,p}\,\Phi_{p,n,} + Q_{p,n}\Big)_{\mathcal{D}} \\
&-\sum_{m=1}^{M} w_m \Big\langle (\vec{\Omega}_m\cdot\vec{n})\,\Psi_m^*, \Psi_m^{inc}\Big\rangle_{\partial\mathcal{D}_m^-}
\end{aligned} \tag{2.49}$$

where the inner products over the whole domain and over all interior faces are

$$\Big(u, v\Big)_{\mathcal{D}} \equiv \sum_{K\in\mathbb{T}_h} \Big(u, v\Big)_K, \tag{2.50}$$

and

$$\Big\langle u, v\Big\rangle_{E_h^i} \equiv \sum_{f\in E_h^i} \Big\langle u, v\Big\rangle_f, \tag{2.51}$$

respectively. The interior faces are designated as the non-repeating set: $E_h^i \in \cup_{K\in\mathbb{T}_h}\partial K\backslash\partial\mathcal{D}$. In Eq. (2.49), the interior jump term, $[\![\Psi_m]\!]$, is defined as,

$$[\![\Psi_m]\!] = \Psi_m^+ - \Psi_m^-, \tag{2.52}$$

and along with the inflow basis function, b_m^+, is beholden to the following trace condition:

$$\begin{aligned} \Psi_m^\pm \equiv \lim_{s\to 0^\pm} \Psi_m(\vec{r} - u(\vec{r})s\,\vec{\Omega}_m) \\ u(\vec{r}) = \text{sgn}\left(\vec{n}(\vec{r})\cdot\vec{\Omega}_m\right) \end{aligned}. \tag{2.53}$$

We can give compact notation to the variational form of the DGFEM transport equation in Eq. (2.49) by defining the bilinear form in Eq. (2.54). We have changed the sequence of the summation over directions and over the elements for the boundary terms.

$$\begin{aligned} a(\Psi^*,\Psi) = \sum_{m=1}^{M} w_m\Big[\Big(\Psi_m^*, \vec{\Omega}_m\cdot\vec{\nabla}\Psi_m\Big)_{\mathcal{D}} + \Big(\sigma_t\Psi_m^*, \Psi_m\Big)_{\mathcal{D}}\Big] \\ + \sum_{m=1}^{M} w_m\Big\langle(\vec{\Omega}_m\cdot\vec{n})\,\Psi_m^{*+}, [\![\Psi_m]\!]\Big\rangle_{E_h^i} - \sum_{f\in\partial\mathcal{D}}\sum_{\vec{\Omega}_m\cdot\vec{n}}\Big\langle(\vec{\Omega}_m\cdot\vec{n})\,\Psi_m^*, \Psi_m\Big\rangle_f \\ + \sum_{f\in\partial\mathcal{D}^r}\sum_{\vec{\Omega}_m\cdot\vec{n}}\Big\langle(\vec{\Omega}_m\cdot\vec{n})\,\Psi_m^*, \Psi_{m'}\Big\rangle_f - \sum_{p=0}^{N_P}\sum_{n=-p}^{p}\frac{2p+1}{4\pi}\Big(\sigma_{s,p}\,\Phi_{p,n,}^*, \Phi_{p,n,}\Big)_{\mathcal{D}} \end{aligned} \tag{2.54}$$

This bilinear form is positive definite ($a(\Psi,\Psi) > 0$) but obviously not symmetric. It also holds Galerking orthogonality, which means that the jumps across elements for the exact solution are zero,

$$a(\Psi^*, \Psi - \Psi_{exact}) = 0, \qquad m = 1, ..., M. \tag{2.55}$$

Because of the positive definiteness and Galerkin orthogonality of the bilinear form, we can use it to define a DG-norm,

$$||u||_{DG} = a(u,u). \tag{2.56}$$

We can also give a more simplified definition of the DG norm,

$$||u||^2_{DG} = \sum_{K\in\mathbb{T}_h}\left[||u||^2_{L^2(K)} + \frac{1}{2}||u^+ - u^-||_{L^2(\partial K)}\right], \tag{2.57}$$

which contains the standard L_2 norm in the element interiors as well as additional L_2 norms on the element boundaries corresponding to the jump terms across the mesh cells.

The convergence of DGFEM methods for the hyperbolic systems has been extensively studied [19, 20, 21, 22]. With the discretized flux solutions, $\Psi_h \in W^h_{\mathcal{D}}$ and $\Phi_h \in W^h_{\mathcal{D}}$, corresponding to our unstructured, $\mathbb{T}_h$, we can define an error for our DGFEM transport solution with the DG norm for the angular fluxes,

$$||\Psi - \Psi_{exact}||_{DG} \leq C\frac{h^q}{(p+1)^q}, \tag{2.58}$$

and flux moments,

$$||\Phi - \Phi_{exact}||_{DG} \leq C\frac{h^q}{(p+1)^q}, \tag{2.59}$$

respectively. In Eqs. (2.58) and (2.59), $q = \min(p+1/2, r-1/2)$, h is the maximum diameter of all the mesh elements, p is the polynomial completeness of the finite element function space (this will be explained further in Chapter 3), r is the regularity index of the transport solution, and C is a constant that is independent of the mesh employed. We can also give an estimate for the transport solution error in the

standard L_2 norm,

$$||\Phi - \Phi_{exact}||_{L_2} \leq C\frac{h^{q+1/2}}{(p+1)^q}, \tag{2.60}$$

where q has the same definition as before. We can see that for the L_2 norm the convergence rate is simply 1/2 integer more than the DG norm for both the polynomial order and the regularity index.

Investigating the definition of the q term in Eqs. (2.58), (2.59), and (2.60) further, we can see that our DGFEM transport convergence rates are all limited by the regularity, r, of the solution space. The spaces to which the transport equation can reside have been investigated by others in previous works [22, 23, 24]. If we have sufficiently smooth data, then the exact transport solution belongs, at most, in the $H^{3/2-\epsilon}(\mathcal{D})$ Hilbert space. This yields a solution regularity of $r = \frac{3}{2} - \epsilon$, where ϵ is positive and extremely small. This means that for most practical occurences, the observed converge regularity is simply $\frac{3}{2}$. For this case it is sufficient to have piecewise polynomial cross section data and incident boundary fluxes that align with our mesh. However, in the case of a pure absorber or void, the exact transport solution lives in the $H^{1/2-\epsilon}(\mathcal{D})$ Hilbert space. Again, the practical irregularity becomes $\frac{1}{2}$. From these spaces, one would think that the transport solution regularity would impede the use of higher-order finite element spaces. We note, however, that these regularity indices only apply to the asymptotic convergence range, which usually only applies to very fine meshes that are much smaller than typically employed meshes. Therefore, we expect to capture up to order $(p+1)$ convergence in preasymptotic ranges that would be employed for a wide variety of transport problems. Results capturing this irregularity behavior are presented in Chapter 3.

Finally, we also seek to define the transport solution convergence rates in terms

other than the maximum element diameter, h. For polytope meshes, this metric may not be the easiest to compute or report if there is large variability in the polygonal or polyhedral elements of the mesh. We seek to re-express the convergence rates in terms of the total number of degrees of freedom in the problem, N_{dof}, instead of the maximum element diameter, h. First, we relate the total number of elements, N_{ele}, to the maximum element diameter,

$$N_{ele} \propto h^{-d}, \tag{2.61}$$

where we assumed that the polytope elements are convex and reasonably regular. We then assume that the finite element functional space on each element is the Serendipity space (we go into further detail in Chapter 3) [25, 26]. This means that the number of degrees of freedom per element is proportional to the polynomial order: $N_{dof} \propto ph^{-d}$ or $h \propto \left(\frac{N_{dof}}{p}\right)^{-1/d}$. We substitute this result into Eq. (2.60) to yield an L_2 convergence rate in terms of the problem's total number of degrees of freedom,

$$||\Phi - \Phi_{exact}||_{L_2} \leq C(p) N_{dof}^{-\frac{q+1/2}{d}}, \tag{2.62}$$

where the proportionality constant, $C(p)$, now has the form,

$$C(p) = \frac{p^{\frac{q+1/2}{d}}}{(p+1)^q}. \tag{2.63}$$

For a transport problem that is not bound by the solution regularity, we can express the simplified convergence rate,

$$||\Phi - \Phi_{exact}||_{L_2} \leq C(p)\, N_{dof}^{-\frac{p+1}{d}}, \qquad C(p) = \frac{p^{\frac{p+1}{d}}}{(p+1)^{p+1/2}}. \tag{2.64}$$

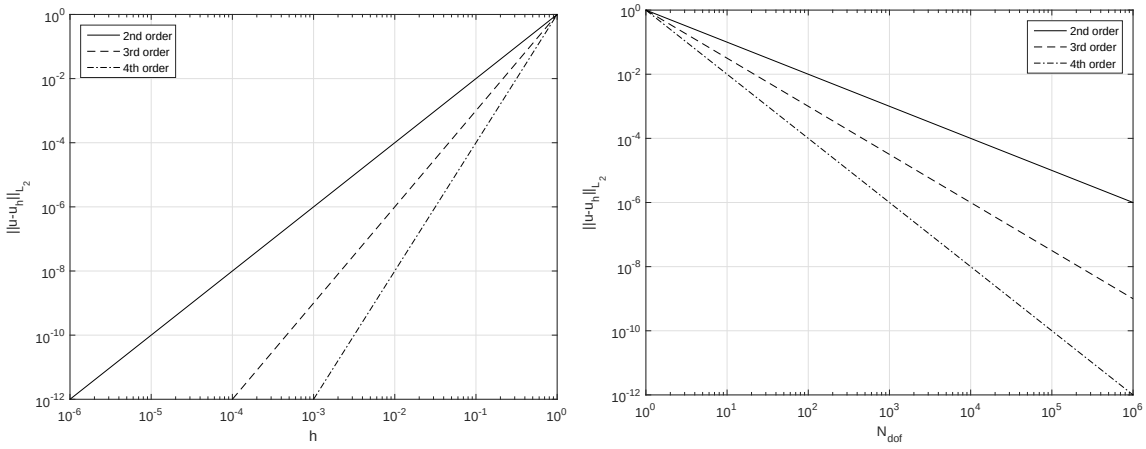


Figure 2.9: Example of the theoretical convergence rates for a DGFEM transport problem that is not bound by solution regularity in terms of the maximum element diameter (left) and number of degrees of freedom (right).

The result of Eq. (2.64) states that we expect convergence rates of N_{dof}^{-1} and $N_{dof}^{-2/3}$ for linear ($p = 1$) functional spaces in 2D and 3D, respectively. Conversely, quadratic ($p = 2$) functional spaces will yield convergence rates of $N_{dof}^{-3/2}$ and N_{dof}^{-1} in 2D and 3D, respectively. The proportionality constant, $C(p)$, in Eq. (2.64) is a monotonically decreasing function for $p \geq 1$ in 2D and 3D (this does not hold for 1D problems). Finally, we show a comparison between the convergence rate estimates of Eqs. (2.60) and (2.62) for a 2D problem not bounded by the solution regularity ($q+1/2 = p+1$) in Figure 2.9. We also simply set $C(p) = 1$ for all the curves to better show a comparison between the rates. We can get a good idea of the power of using higher-order shape functions with curves based on N_{dof}. If our transport problem takes about the same amount of calculation time to solve for the same number of degrees of freedom, we can see that we would yield a more accurate answer for little additional cost.

2.6.2 Elementary Matrices on an Arbitrary Spatial Cell

In Eq. (2.48), we presented the full set of spatially-discretized equations needed to solve the angular flux solution for cell K for a single angular direction. In the equation, several terms of various types arise including interaction, $\left(\sigma b_m, \Psi_m\right)_K$, streaming, $\left(\vec{\Omega}_m \cdot \vec{\nabla} b_m, \Psi_m\right)_K$, and surface, $\left\langle \left(\vec{\Omega}_m \cdot \vec{n}\right) b_m, \Psi_m \right\rangle_{\partial K}$. Each of these correspond to a different elementary matrix type. We now present how to compute the mass, streaming, and surface matrices in Sections 2.6.2.1, 2.6.2.2, and 2.6.2.3, respectively.

2.6.2.1 Elementary Mass Matrices

In the spatially discretized equations presented in Section 2.6, there are several reaction terms that appear with the form: $\left(\sigma b_m, \Psi_m\right)_K$ for a given angular direction, m, and for a spatial cell, K. In FEM analysis these reaction terms are ubiquitously referred to as the mass matrix terms [27, 28]. For cell K, we define the elementary mass matrix, $\mathbf{M}$, as

$$\mathbf{M}_K = \int_K \mathbf{b}_K \, \mathbf{b}_K^T \, dr, \tag{2.65}$$

where $\mathbf{b}_K$ corresponds to the set of N_K basis functions that have non-zero measure in cell K. Depending on the FEM basis functions utilized, the integrals in Eq. (2.65) can be directly integrated analytically. However, if in general, the basis functions cannot be analytically integrated on an arbitrary set of cell shapes, then a numerical integration scheme becomes necessary. If we define a quadrature set, $\left\{\vec{x}_q, w_q^K\right\}_{q=1}^{N_q}$, for cell K, consisting of N_q points, $\vec{x}_q$, and weights, w_q^K, then we can numerically calculate the mass matrix by the following

$$\mathbf{M}_K = \sum_{q=1}^{N_q} w_q^K \mathbf{b}_K(\vec{x}_q)\, \mathbf{b}_K^T(\vec{x}_q). \tag{2.66}$$

In this case, it is necessary that the sum of the weights of this quadrature set exactly equal the geometric measure of cell K. This means that $\sum_{q=1}^{N_q} w_q^K$ is equal to the cell width in 1 dimension, the cell area in 2 dimensions, and the cell volume in 3 dimensions.

Since $\mathbf{b}_K$ consists of a column vector for the basis functions and $\mathbf{b}_K^T$ consists of a row vector, then their multiplication will obviously yield a full $(N_K \,\mathrm{x}\, N_K)$ matrix. This matrix is written for completeness of this discussion on the mass matrix:

$$\mathbf{M}_K = \begin{bmatrix} \int_K b_1\, b_1 & \dots & \int_K b_1\, b_j & \dots & \int_K b_1\, b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_K b_i\, b_1 & \dots & \int_K b_i\, b_j & \dots & \int_K b_i\, b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_K b_{N_K}\, b_1 & \dots & \int_K b_{N_K}\, b_j & \dots & \int_K b_{N_K}\, b_{N_K} \end{bmatrix}, \tag{2.67}$$

where an individual matrix entry is of the form:

$$M_{i,j,K} = \int_K b_i\, b_j. \tag{2.68}$$

2.6.2.2 Elementary Streaming Matrices

Next, we will consider the streaming term that has the form: $\left(\vec{\Omega}_m \cdot \vec{\nabla} b_m, \Psi_m\right)_K$ for a given angular direction, m, and for a spatial cell, K. $\vec{\nabla}$ is the gradient operator in physical space. It has the form of $\vec{\nabla} = \left[\frac{d}{dx}\right]$ in 1 dimension, the form of $\vec{\nabla} = \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right]$ in 2 dimensions, and the form of $\vec{\nabla} = \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right]$ in 3 dimensions. Since for every cell, the streaming term is applied for all M angles in the angular discretization, we

define the analytical elementary streaming matrix:

$$\vec{\mathbf{G}}_K = \int_K \vec{\nabla}\mathbf{b}_K \, \mathbf{b}_K^T \, dr, \tag{2.69}$$

which has dimensionality (N_KxN_Kxd). We choose to store the elementary streaming matrix in this form and not store M separate (N_KxN_K) local matrices corresponding to the application of the dot product ($\vec{\Omega}_m \cdot \int_K \vec{\nabla}\mathbf{b}_K \, \mathbf{b}_K^T \, dr$). Instead, we simply evaluate the dot product with the appropriate angular direction whenever necessary. This has great benefit when trying to run large transport problems when memory becomes a premium and processor operations are not our limiting bottleneck.

Just like the elementary mass matrix, we can use the same spatial quadrature set, $\left\{\vec{x}_q, w_q^K\right\}_{q=1}^{N_q}$, for cell K to numerically calculate the streaming matrix:

$$\vec{\mathbf{G}}_K = \sum_{q=1}^{N_q} w_q^K \vec{\nabla}\mathbf{b}_K(\vec{x}_q) \, \mathbf{b}_K^T(\vec{x}_q). \tag{2.70}$$

In this case, this local cell-wise streaming matrix has the full matrix form:

$$\vec{\mathbf{G}}_K = \begin{bmatrix} \int_K \vec{\nabla} b_1 \, b_1 & \dots & \int_K \vec{\nabla} b_1 \, b_j & \dots & \int_K \vec{\nabla} b_1 \, b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_K \vec{\nabla} b_i \, b_1 & \dots & \int_K \vec{\nabla} b_i \, b_j & \dots & \int_K \vec{\nabla} b_i \, b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_K \vec{\nabla} b_{N_K} \, b_1 & \dots & \int_K \vec{\nabla} b_{N_K} \, b_j & \dots & \int_K \vec{\nabla} b_{N_K} \, b_{N_K} \end{bmatrix}, \tag{2.71}$$

where an individual matrix entry is of the form:

$$\vec{G}_{i,j,K} = \int_K \vec{\nabla} b_i \, b_j. \tag{2.72}$$

2.6.2.3 Elementary Surface Matrices

Finally, the last terms to consider of the discretized transport equation are those found on the faces of the cell boundary: $\vec{\Omega}_m \cdot \left\langle \vec{n}\, b_m, \Psi_m \right\rangle_{\partial K}$. These terms are analagous to the cell mass matrix but are computed on the cell boundary with dimensionality $(d-1)$. Analyzing a single face, f, in cell K, the analytical surface matrix is of the form,

$$\vec{\mathbf{F}}_{f,K} = \int_f \vec{n}(\vec{r})\, \mathbf{b}_K\, \mathbf{b}_K^T\, ds, \tag{2.73}$$

where we allow the outward surface normal, $\vec{n}$, to vary along the cell face. For 1D problems as well as 2D problems with colinear cell faces (no curvature), the outward normals would be constant along the entire face. However, there are many cases where 3D mesh cells would not have coplanar vertices along a face. Then, the outward normal would not be constant along the face and would need to be taken into account during integration procedures. A simple example of non-coplanar face vertices would be an orthogonal hexahedral cell that has its vertices undergo a randomized displacement.

With the analytical form of the surface matrices defined in Eq. (2.73), we can see that they have dimensionality, (N_KxN_Kxd). This is the same dimensionality as the cell streaming term. However, it is possible to reduce the dimensionality of the surface matrices if it is desired to reduce the memory footprint. There are some basis sets where all but $N_b^{f,K}$ basis functions are zero along face f. If we also restrict the mesh cell faces of our transport problems to have colinear (in 2D) or coplanar (in 3D) vertices so that the outward normal is constant along a face f, then we can define the surface matrix as $\int_f \mathbf{b}_K\, \mathbf{b}_K^T\, ds$. For these basis sets with $N_b^{f,K}$ non-zero face values on colinear/coplanar face f, the surface matrix has reduced dimensionality of

$(N_b^{f,K} \mathrm{x} N_b^{f,K})$.

Just like the cell mass and streaming matrices, it is possible that the basis functions cannot be integrated analytically. Analogous to the cell-wise quadrature, we can define a quadrature set for face f: $\left\{\vec{x}_q, w_q^f\right\}_{q=1}^{N_q^f}$. This quadrature set is not specific for just one of the cells that face f separates. If the quadrature set can exactly integrate the basis functions of both cells K and K' (as defined by Figure 2.8), then only 1 quadrature set needs to be defined for both cells. Using this quadrature set, we can numerically calculate the surface matrix for face f along cell K:

$$\vec{\mathbf{F}}_{f,K} = \sum_{q=1}^{N_q^f} w_q^f \vec{n}(\vec{x}_q)\, \mathbf{b}_K(\vec{x}_q)\, \mathbf{b}_K^T(\vec{x}_q). \tag{2.74}$$

Similar to the cell-wise spatial quadrature sets, the sum of the weights of these face-wise quadrature sets needs to exactly equal the geometric measure of face f. This means that $\sum_{q=1}^{N_q^f} w_q^f$ is equal to 1.0 in 1 dimension, the length of the face edge in 2 dimensions and the face area in 3 dimensions.

Using the same notation as the cell-wise mass and streaming matrices, the local face-wise surface matrix for face f has the full matrix form,

$$\vec{\mathbf{F}}_{f,K} = \begin{bmatrix} \int_f \vec{n}\, b_1\, b_1 & \dots & \int_f \vec{n}\, b_1\, b_j & \dots & \int_f \vec{n}\, b_1\, b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_f \vec{n}\, b_i\, b_1 & \dots & \int_f \vec{n}\, b_i\, b_j & \dots & \int_f \vec{n}\, b_i\, b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_f \vec{n}\, b_{N_K}\, b_1 & \dots & \int_f \vec{n}\, b_{N_K}\, b_j & \dots & \int_f \vec{n}\, b_{N_K}\, b_{N_K} \end{bmatrix}, \tag{2.75}$$

where an individual matrix entry is of the form:

$$\vec{F}_{i,j,f,K} = \int_f \vec{n}\, b_i\, b_j. \tag{2.76}$$

2.7 Solution Procedures

To this point, we have properly described the procedures to discretize the transport problem in energy, angle, and space. Combining the results of Sections 2.3, 2.4, and 2.6, we write the fully-discretized DGFEM multigroup S_N equations for an element K, where the test function $b_{m,g}$ for a single direction and energy group is now used:

$$\begin{aligned}
&-\Big(\vec{\Omega}_m\cdot\vec{\nabla}b_{m,g},\Psi_{m,g}\Big)_K+\Big(\sigma_{t,g}b_{m,g},\Psi_{m,g}\Big)_K+\Big\langle(\vec{\Omega}_m\cdot\vec{n})\,b_{m,g},\Psi^-_{m,g}\Big\rangle_{\partial K^+}\\
&\quad+\Big\langle(\vec{\Omega}_m\cdot\vec{n})\,b_{m,g},\Psi^+_{m,g}\Big\rangle_{\partial K^-\setminus\partial\mathcal{D}}+\Big\langle(\vec{\Omega}_m\cdot\vec{n})\,b_{m,g},\Psi^-_{m',g}\Big\rangle_{\partial K^-\cap\partial\mathcal{D}^r}\\
&\quad=\sum_{g'=1}^{G}\sum_{p=0}^{N_P}\frac{2p+1}{4\pi}\sum_{n=-p}^{p}Y_{p,n}(\vec{\Omega}_m)\Big(\sigma_{s,p}^{g'\to g}\,b_{m,g},\Phi_{p,n,g'}\Big)_K\\
&\quad+\big(b_{m,g},Q_{m,g}\big)_K-\Big\langle(\vec{\Omega}_m\cdot\vec{n})\,b_{m,g},\Psi^{inc}_{m,g}\Big\rangle_{\partial K^-\cap\partial\mathcal{D}^d}
\end{aligned}\quad . \tag{2.77}$$

All of the notations used in Eq. (2.77) remain unchanged from Section 2.6.

We now spend the remainder of this chapter discussing various methodologies to efficiently solve the tightly-coupled system of equations composing our transport problem. Section 2.7.1 details the iterative procedures used to solve the transport problem in energy and angle, and Section 2.7.2 then describes how we solve the spatial portion of the problem for a single energy/angle iteration.

2.7.1 Angle and Energy Iteration Procedures

The fully discretized transport equation has an angular flux solution, $\mathbf{\Psi}$, with dimensionality of (GxMxN_{dof}). The angular flux moments, $\mathbf{\Phi}$, have dimensionality of (GxN_{mom}xN_{dof}). Depending on the necessary fidelity of the problem, the full phase-space of the solution can become extremely large to solve. We can have *billions*

of total unknowns to solve for if we simply have $N_{dof} \approx O(10^6)$, $M \approx O(10^2)$, and $G \approx O(10^1)$. These orders of number of unknowns in space, angle, and energy are of reasonable size for 3D transport problems.

In theory, if we had the computer memory, we could construct a left-hand-side matrix of dimensionality (GxMxN_{dof})x(GxMxN_{dof}) with a corresponding right-hand-side vector of dimensionality (GxMxN_{dof})x1, we could then directly solve for the full phase-space angular flux solution at once. However, because the dimensional space of the unknowns can rapidly grow and become too large for hardware memory, transport problems have traditionally been solved iteratively. We now detail the procedures that we will employ to iteratively obtain the phase-space solution in energy and angle.

Because the only coupling that arises between the set of multigroup S_N equations is between the energy groups in the scattering source, our iterative procedures principally lie in the energy domain. Figure 2.10 presents a pair of scattering matrices that show typical coupling between energy groups for neutronics problems. Depending on how the group boundaries are established along with a possible need for higher fidelity in energy, thermal upscattering may or may or may not be present in the problem. We can see from Figure 2.10 that the purely-downscattering matrix only has 1-way coupling, from high-to-low in the group energies. This means that if we progress through the energy groups from $g = 1, ..., G$ (in what is called an *outer iteration*), then we only have to do so once and we are done. However, there are many transport problems where we wish to have several energy groups in the thermal region. This leads to thermal neutron upscattering and causes the lower-right portion of the scattering matrix to be full as seen in the bottom of Figure 2.10. Then, depending on how the groups are structured, multiple outer iterations may be necessary to fully converge the scattering source.

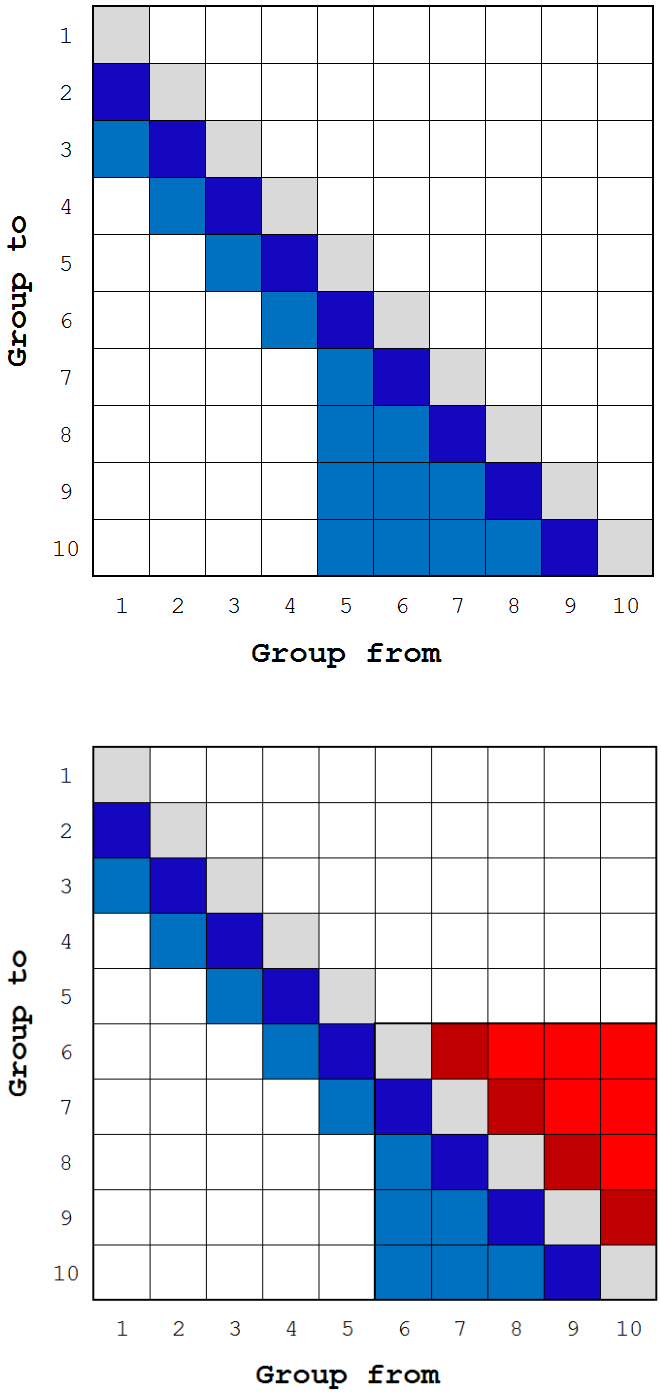


Figure 2.10: Scattering matrices (top) without and (bottom) with upscattering. The gray corresponds to within-group scattering; the blue corresponds to down-scattering in energy; and the red corresponds to up-scattering in energy.

We now describe the full iterative procedures required to converge the scattering source in detail. We begin by defining a new concept called a *group set*, which is simply a collection of contiguous groups. The group sets are ordered from high-to-low energy just like the groups and are non-overlapping with the adjoining sets. We demonstrate this concept with a simple example. If we have a problem with 10 groups ($G = 10$), we then choose to aggregate them into 3 group sets. Group set 1 contains $g \in [1, 3]$, group set 2 contains $g \in [4, 7]$, and group set 3 contains $g \in [8, 10]$. With these group sets defined we then choose to employ a double iteration loop to converge the scattering matrix. The outer iterations in this procedure perform residual calculations by looping through the group sets. Then, for each group set in every outer iteration, *inner iterations* are performed. These inner iterations perform residual calculations to solve the scattering matrix within that group set to some specified tolerance. We will sometimes refer to these inner iterations as *within-group-set* (WGS) *iterations*. Because each outer iteration performs residual calculations for every group set, we will sometimes refer to these outer iteration as *across-group-set* (AGS) *iterations*. We continue performing AGS iterations until the solution residual is below some specified tolerance. At each outer and inner iteration, we possibly perform some acceleration (or preconditioning) step, where we leave the details for this until Chapter 4. This complete iterative procedure involving the outer and inner iteration loops is given in Algorithm 1. In this algorithm, we solve N_{gs} number of group sets by using a maximum number of AGS iterations, I_{ags}, as well as a maximum number of WGS iterations for each group set, I_{gs}.

As previously stated, the residual iterations for each group set only converge the scattering source for that group set. This means that the scattering sources from the other group sets are not modified during these iterations. We show this behavior by decomposing the transport equation into N_{gs} separate equations where we have

decomposed the scattering source into group set and across-group set components. The equations for the angular flux and flux moment solutions for each group set are given by

$$\mathbf{L}_{gs}\boldsymbol{\Psi}_{gs} = \mathbf{M}_{gs}\boldsymbol{\Sigma}_{gs}\boldsymbol{\Phi}_{gs} + \mathbf{M}_{gs}\sum_{ggs\neq gs}\boldsymbol{\Sigma}_{ggs}\boldsymbol{\Phi}_{ggs} + \mathbf{Q}_{gs} \quad , \quad gs = 1, ..., N_{gs} \tag{2.78}$$

and

$$\boldsymbol{\Phi}_{gs} = \mathbf{D}_{gs}\boldsymbol{\Psi}_{gs}, \tag{2.79}$$

respectively, where $\mathbf{L}$ is the fully-discretized loss operator which consists of total interaction and streaming terms, $\mathbf{M}$ is the moment-to-discrete operator of the angular discretization, $\mathbf{D}$ is the discrete-to-moment operator of the angular discretization, $\boldsymbol{\Sigma}$ is the scattering operator of the multigroup and angular discretizations, and $\mathbf{Q}$ is the full phase-space distributed source. In this case, the source contains contributions from boundary, domain sources, and fission sources. We note that $\mathbf{M}$ and $\mathbf{D}$ are also group set dependent since we do not enforce the exact same angular discretization on all group sets. This can be useful if certain ranges of energy groups require higher angular fidelity due to increased anisotropic scattering.

We can further express Eq. (2.78) in terms of only the flux moments. Through algebra and linear algebra techniques, we state the flux-moment-only equation as

$$\left(\mathbf{I} - \mathbf{T}_{gs}\right)\boldsymbol{\Phi}_{gs} = \mathbf{D}_{gs}\mathbf{L}_{gs}^{-1}\mathbf{M}_{gs}\sum_{ggs\neq gs}\boldsymbol{\Sigma}_{ggs}\boldsymbol{\Phi}_{ggs} + \mathbf{D}_{gs}\mathbf{L}_{gs}^{-1}\mathbf{Q}_{gs}, \tag{2.80}$$

where we define,

$$\mathbf{T}_{gs} \equiv \mathbf{D}_{gs}\mathbf{L}_{gs}^{-1}\mathbf{M}_{gs}\mathbf{\Sigma}_{gs}, \tag{2.81}$$

for further brevity. In Eqs. (2.80) and (2.81), the term $\mathbf{L}_{gs}^{-1}$ accounts for the inversion of the loss operator. For this work, we will utilize the full-domain transport sweep, and we leave its details until Section 2.7.2. We now provide the details for the procedures that will be used to solve Eq. (2.80) in Sections 2.7.1.1 and 2.7.1.2.

Algorithm 1 Iterative Solver in Energy for the Multigroup Transport Problem

```
Initialize: Φ_{g,p,n} = 0,    g = 1, ..., G; p = 0, ..., N_p; n = −p, ..., p
for a = 1, ..., I_ags do
    Loop: through group sets
    for gs = 1, ..., N_gs do
        for k = 1, ..., I_gs do
            Perform: residual iteration
            Apply: Acceleration for group set gs
            Check: convergence of group set gs
        end for
    end for
    Perform: residual iteration
    Apply: Acceleration for across-group-set
    Check: across-group-set convergence
end for
```

2.7.1.1 Source Iteration

One simple method to invert the $(\mathbf{I} - \mathbf{T})$ operator of Eq. (2.80) is the *source iteration* technique (SI), also known as *Richardson iteration*. If we isolate a single group set, gs, then the iterative procedure for the transport equation is

$$\mathbf{L}\mathbf{\Psi}^{(k+1)} = \mathbf{M}\mathbf{\Sigma}\mathbf{\Phi}^{(k)} + \mathbf{Q}, \tag{2.82}$$

where we removed the gs subscripts for brevity and note that the driving source $\mathbf{Q}$ contains scattering sources from all other group sets into the current group set of interest. From Eq. (2.82), we can see that the scattering source at iteration $(k+1)$ is calculated from a previous guess for the flux moments at iteration (k). Then, after the application of one transport sweep, we obtain a new guess for the angular flux moments:

$$\begin{aligned} \mathbf{\Psi}^{(k+1)} &= \mathbf{L}^{-1}\left(\mathbf{M\Sigma\Phi}^{(k)} + \mathbf{Q}\right) \\ \mathbf{\Phi}^{(k+1)} &= \mathbf{D\Psi}^{(k+1)} \end{aligned} \tag{2.83}$$

We continue to perform these Richardson iterations until the difference between two iterate solutions is less than some specified tolerance in a given norm. This convergence criterion for SI can be succinctly written as

$$\frac{||\Phi^{(k+1)} - \Phi^{(k)}||}{||\Phi^{(k+1)}||} \leq \text{tol}, \tag{2.84}$$

where the estimate for the error is normalized by the norm of the most recent iteration solution. Steps are taken to ensure that a divide-by-zero does not occur. In general, SI is guaranteed to converge for a large range of transport problems (further precautions need to be taken for problems with high anisotropic scattering). We can estimate how quickly SI will converge by analyzing the spectral radius (SR) of the transport problem [29]. This spectral radius, ρ, can be estimated in the asymptotic convergence region by the ratio of two successive solution differences:

$$\rho \approx \frac{||\Phi^{(k+1)} - \Phi^{(k)}||}{||\Phi^{(k)} - \Phi^{(k-1)}||} \tag{2.85}$$

We can gather a lot of information about how our transport solution will converge from Eq. (2.85). If $\rho > 1$ in the asymptotic region, then the error in our residual

Table 2.3: Average number of iterations required to reduce SI error by 1 order of magnitude for different SR values.

ρ	Iterations
0.1	1.0
0.25	1.6
0.5	3.3
0.75	8.0
0.9	22
0.99	230
0.999	2301
0.9999	23024

will grow without end, and our solution will diverge. However, in the preasymptotic region, Eq. (2.85) may be a bad estimate for the spectral radius, and values greater than 1 may be observed for several iterations. This does not mean that the solution will diverge in the end. If we are in the asymptotic region, then we would like $\rho \ll 1$ because it means that our transport solution will quickly converge. However, if $\rho \approx 1$ (but still strictly less than 1), then our transport problem will slowly converge to our final solution. We demonstrate this behavior in Table 2.3 by giving the average number of SI iterations required to reduce our solution error by 1 order of magnitude. We can see that as ρ approaches 1, the iteration numbers become prohibitively large. For those problems with large spectral radii, we can accelerate our solution convergence by using synthetic acceleration on Preconditioned Richardson Iteration [29]. We leave the details for this in Chapter 4.

2.7.1.2 Krylov Subspace Methods

Source iteration is not the only iterative technique that can be employed to invert $(\mathbf{I} - \mathbf{T})$ for a given group set. In the last 20 years, Krylov subspace methods have

been applied to the discretized transport equation [30, 31, 32]. Because $(\mathbf{I} - \mathbf{T})$ is not symmetric, we want to only use Krylov methods that can solve non-symmetric matrices. The two Krylov subspace methods that we would naturally want to employ are GMRES and BiCGSTAB [33, 34]. We will not describe the implementations of these methods here for brevity. However, we will state that most of the computational machinery required to perform Richardson iterations can also be used for these Krylov methods. The only modifications/extensions that are needed are summarized in the following list.

1. Construction of the right-hand-side: $\mathbf{b} = \mathbf{D}\mathbf{L}^{-1}\mathbf{Q}$. From this equation, it is obvious that we need just one initial transport sweep to properly build this right-hand-side.

2. The operation of the matrix $(\mathbf{I} - \mathbf{T})$ on a Krylov vector, ν. This can easily be accomplished with the same machinery as Richardson iteration by simply subtracting the original flux moment vector by the updated moments after one transport sweep.

3. Modify the calculation of the convergence criterion so that the norm of the iteration residual, normalized by the right-hand-side, is smaller than some prescribed tolerance: $\frac{||\mathbf{b}-(\mathbf{I}-\mathbf{T})\mathbf{x}||_2}{||\mathbf{b}||_2} < \text{tol}$.

Combined with the appropriate linear algebra operations, these three small alterations are all that are required to properly utilize the Krylov solver for the transport equation. It is not immediately obvious how one would precondition the Krylov iterations in the context of transport sweeping. However, just like the Richardson iteration scheme, we will provide the implementation of DSA preconditioning for Krylov in Chapter 4.

2.7.2 Spatial Solution Procedures

Section 2.7.1 presented the methodology that we will employ to iteratively converge our transport solutions in energy and angle (flux moments). Both Richardson iteration and the Krylov methods were presented as methods that can invert the $(\mathbf{I}-\mathbf{T})$ operator. In both of these iterative methods, the common operation of interest is the inversion of the loss operator ($\mathbf{L}$). There are different techniques that could be used to perform this operation, including several matrix-dependent and matrix-free methodologies. For this work, the loss operator inversion on some unstructured mesh, $\mathbb{T}_h$, will be performed by use of the full-domain transport sweep as outlined next in Section 2.7.2.1. We then conclude our discussion on spatial solution procedures by briefly describing how we can utilize adaptive mesh refinement (AMR) to generate irregular polytope meshes (without hanging nodes in this work) in Section 2.7.2.2.

2.7.2.1 Transport Sweeping

Recall from earlier that the continuous definition of the loss operator for an angular direction m and energy group g is,

$$L_{m,g} = \vec{\Omega}_m \cdot \vec{\nabla} + \sigma_{t,g}\,, \tag{2.86}$$

where we suppress the spatial parameter for brevity. From the application of the test function, integration by parts of the streaming term, and the use of the upwind scheme as outlined in Section 2.6, we described the coupling between different mesh cells. The upwind scheme only couples the adjoining cells upwind of any given angular direction. This means that for a given mesh element, if the upwind cells have already had their angular fluxes updated for a given iteration, we then have

all the information needed to solve for the new angular fluxes on the element. This means that if the task dependence graphs are well chosen for all directions, we can then invert the streaming operator by solving Eq. (2.77) one cell at a time for a given group of angular directions.

This cell-by-cell inversion of the streaming operator is known as a *transport sweep*. If we perform this inversion for all angular directions across the entire spatial domain without lagging, then it is called a full-domain transport sweep. The full-domain transport sweep is a beneficial matrix-free scheme because of the following:

- The iterative solver does not need to build any matrices explicitly but only requires the action of $\mathbf{L}^{-1}$.
- Does not require the formation of M separate matrices for each of the angular directions (for 1 energy group). This is both memory and computationaly intensive for any problems of appreciable size.
- The matrix-vector operations on a single element within a transport sweep can be efficiently performed depending on the group set structure and the angle aggregation as outlined in Section 2.7.1.
- The matrix-free transport sweep favors higher-order DGFEM schemes since they will yield more processor work per element with less memory caching (which is the current bottleneck with massively-parallel calculations).
- The number of sweep iterations does not grow with increasing problem size or processor counts. This is in contrast with partial-domain sweeping like *parallel block jacobi* (PBJ) [35].

We can gain knowledge of how our transport sweeps will perform by analyzing the streaming operator, $\mathbf{L}$, in space and angle. If $\mathbf{L}$ is strictly block-lower triangular

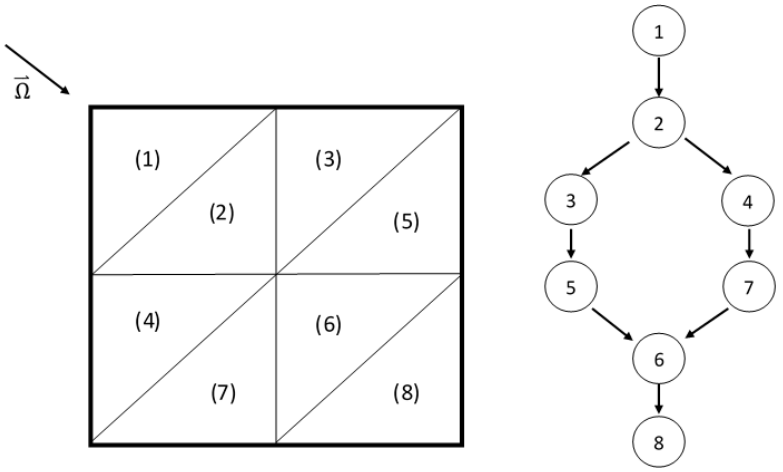


Figure 2.11: Task graph for the transport sweep for a given direction without cycles.

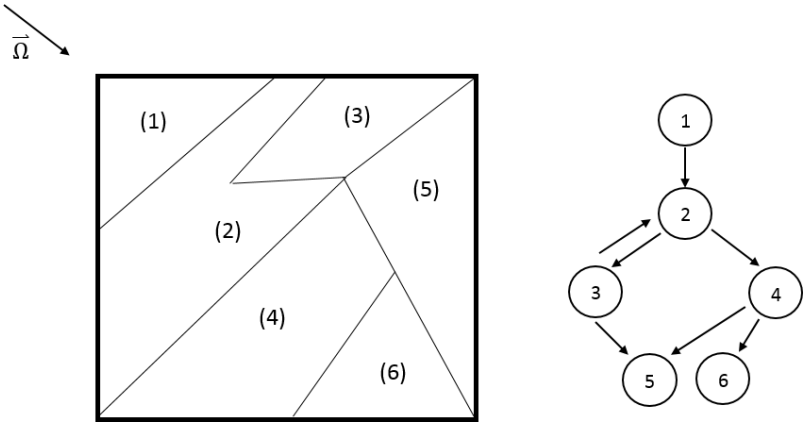


Figure 2.12: Task graph for the transport sweep for a given direction with cycles present.

for a given angular direction, then it is guaranteed that an ordering of the mesh cells can be represented by a Directed Acyclic Graph (DAG) [36]. However, situations can arise where a cycle forms in the graph and a complete sweep for those directions becomes impossible. These situations include concave polytope elements that cause re-entrance, opposing reflecting boundary conditions, and extreme distortion of an $\mathbb{R}^3$ domain (regular mesh of hexahedral cells with 1 dimensional end twisted). Figure 2.11 shows a DAG without any cycles for the given direction. Figure 2.12 then shows a graph with a cycle caused by a re-entrant quadrilateral mesh cell. There are methods to break these cycles and still perform transport sweeping, but we will not consider them for this work.

2.7.2.2 Spatial Adaptive Mesh Refinement

Adaptive Mesh Refinement (AMR) techniques have become more commonplace across many scientific and engineering fields over the last couple of decades [37, 38, 39, 40, 41, 42]. However, while this has become more common practice in other disciplines, the use of AMR for the transport equation is still in its infancy [43, 44, 45, 46, 47, 48]. For this work, we will only utilize *h*-type refinement strategies for 2D transport problems on initial meshes with only quadrilateral cells.

Figure 2.13 provides the logical flow chart used to perform the mesh adaptation procedures for our problem of interest. The AMR procedure begins with an initial and typically coarse mesh. The transport solution is calculated on this mesh with appropriate PDE solvers. Once this solution is determined, *a posteriori* error estimates are used to determine which problem regions contain the largest spatial discretization error [49, 50]. Then, based on some refinement criterion, some subset of mesh elements are flagged for local refinement. From these flagged elements, the mesh can be refined, and a new and more accurate solution can further be obtained.

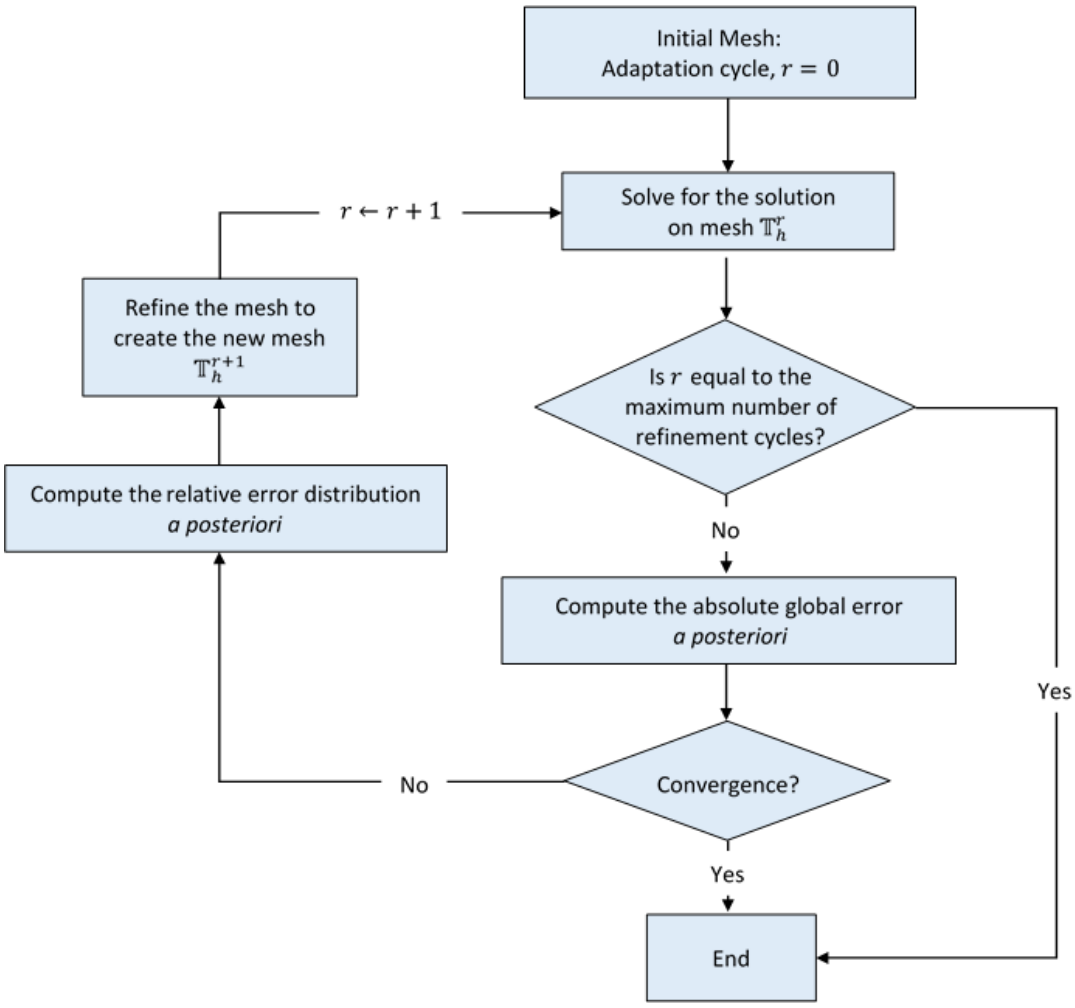


Figure 2.13: Flow chart for mesh adaptation.

This process is repeated until some global convergence criterion is satisfied or some number of mesh adaptation cycles are performed.

At the heart of the mesh adaptation procedures is the reliance on an accurate estimation of the numerical solution error. The state-of-the-art error estimators rely on either adjoint-based methods [43, 44, 46] or projection-based methods [51]. However, these methods require an additional calculation of the solution at each refinement level in a richer (and more difficult to solve) functional space. Therefore,

we will employ a simpler error estimator that does not require an additional solution calculation. For this work, we will utilize the jump-based error estimator since it is the most straightforward to define and execute. The estimate of the error for mesh cell K of the mesh at refinement level r, $\mathbb{T}_h^r$, is given by the following,

$$\eta_K^r = \int_{\partial K} [\![\Phi^r]\!]^2 d\,s = \int_{\partial K} \left(\sum_m w_m [\![\Psi_m^r]\!] \right)^2 , \tag{2.87}$$

where $[\![\cdot]\!]$ is the jump operator along a face defined as,

$$[\![\Phi(\vec{r})]\!] = \Phi^+(\vec{r}) - \Phi^-(\vec{r}), \tag{2.88}$$

and the terms, $\Phi^+(\vec{r})$ and $\Phi^-(\vec{r})$, are subject to the trace:

$$\Phi^\pm(\vec{r}) = \lim_{s \to 0^\pm} \Phi(\vec{r} + s\vec{n}). \tag{2.89}$$

In this case, the outward normal, $\vec{n}$, is determined with respect to the element K along its boundary, ∂K. With this trace, $\Phi^-(\vec{r})$ always corresponds to the solution within cell K. Investigating face f of cell K, the across-face solution, $\Phi^+(\vec{r})$, is dependent on the boundary type of face f. The across-face solutions can be succinctly written:

$$\Phi^+(\vec{r}) = \begin{cases} \lim_{s \to 0^+} \Phi(\vec{r} + s\vec{n}) & \vec{r} \notin \partial\mathcal{D} \\ \sum_{\vec{\Omega}_m \cdot \vec{n} > 0} \Psi_m^-(\vec{r}) + \sum_{\vec{\Omega}_m \cdot \vec{n} < 0} \Psi_m^{inc}(\vec{r}) & \vec{r} \in \partial\mathcal{D}^d \\ \Phi^-(\vec{r}) & \vec{r} \in \partial\mathcal{D}^r \end{cases} . \tag{2.90}$$

From Eq. (2.90), the across-face solutions for interior faces, incident boundaries and reflecting boundaries have different meanings. For an interior face f ($\vec{r} \notin \partial\mathcal{D}$), the across-face solution comes from the cell K' (as defined by Figure 2.8). For

incident boundaries ($\vec{r} \in \partial\mathcal{D}^d$), the across-face solution is a combination of integrals of the outgoing (Ψ_m^-) and incident boundary fluxes (Ψ_m^{inc}). Finally, for reflecting boundaries ($\vec{r} \in \partial\mathcal{D}^r$), the across-face solutions are simply the within-cell solutions. Therefore, the solution jump is exactly zero for all reflecting boundaries and yields no contribution to the error estimate.

With the error estimates defined for all cells $K \in \mathbb{T}_h^r$ for refinement level r, a criterion is needed to determine which cells should be refined. For this work, we choose to employ a refinement criterion of the following form,

$$\eta_K^r \geq \alpha \max_{K' \in \mathbb{T}_h^r} \left(\eta_{K'}^r\right), \tag{2.91}$$

where α is a user-defined value $(0, 1)$. This refinement criterion has a simple meaning. If, for example, $\alpha = 0.2$, then a cell will be refined if its error estimate is greater than 20% of the cell with the largest error estimate. This does not necessarily mean that 80% of the mesh cells will be refined at level r. Instead, the criterion simply states that any cell above a particular threshold will be refined. This means that it is theoretically possible for the extreme cases of 1 or all cells being refined at a particular refinement cycle. The first extreme case could arise with a single spatial cell having a strong solution discontinuity stemming from a strong localized source. The second extreme case could arise when the intercell solution jumps are about equivalent stemming from an incredibly smooth discretized solution at that particular refinement cycle. We could also enforce a uniform refinement of all the spatial cells by setting $\alpha = 0$.

Once we have determined which elements on mesh level r to refine, we then form our new adapted mesh on level $r + 1$ by a combination of logical refinement rules, hierarchical elements, and refinement trees. Each quadrilateral mesh cell with vertices

flagged for refinement (which we denote as the parent element) is further subdivided into four smaller quadrilateral cells (which we denote as daughter elements I, II, III, and IV) that maintain the overall shape of the orginal mesh cell. If the parent element is denoted by the vertices (1), (2), (3), and (4), then the refinement rules, including the new local ordering of each daughter element's vertices are given in Figure 2.14. These refinement rules are detailed as follows:

1. Daughter element I is placed near original vertex (1).
2. Daughter element II is placed near original vertex (2).
3. Daughter element III is placed near original vertex (3).
4. Daughter element IV is placed near original vertex (4).
5. The local numbering of the daughter element vertices is consistent with the parent element.
6. The vertices of the daughter elements common with the original element inherit the same local numbering.

These refinement rules guarantee that we will maintain logical consistency with our counter-clockwise ordering of the element vertices. This consistency is important for the computation of the polytope basis functions that are presented in Chapter 3.

Once a mesh element from level r has been refined, we need to incorporate its daughter elements into the adapted mesh, $\mathbb{T}_h^{r+1}$. We can see from the nesting refinement process, that these AMR procedures lead to a hierarchical representation of the mesh. All of the mesh cells are obtained by subdivisions of cells from the initial mesh, $\mathbb{T}_h^0$. This hierarchical nature of the mesh can succinctly be described in a tree structure. We give an example of this tree structure in Figure 2.15 for the simple case

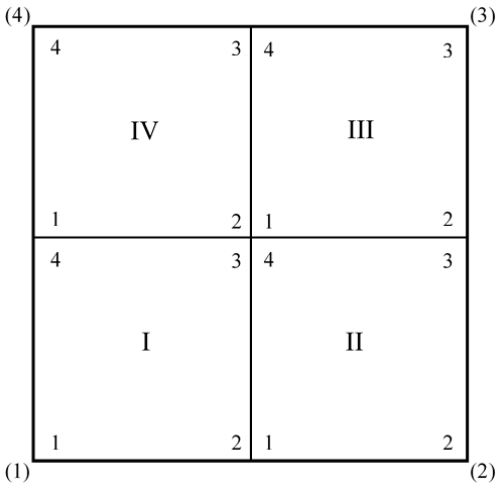
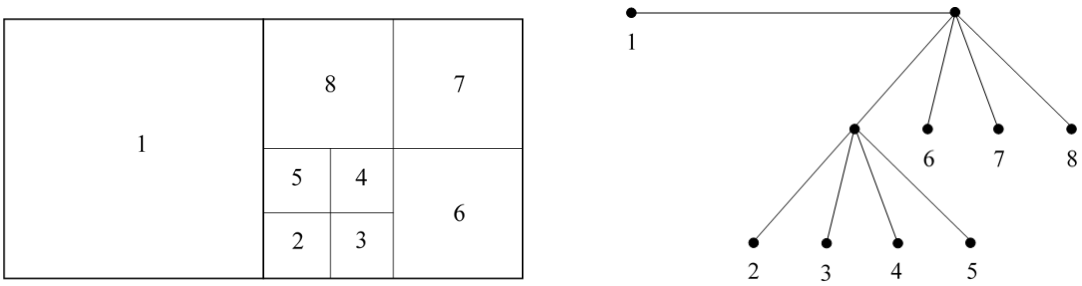


Figure 2.14: Refinement rules for a quadrilateral mesh cell.

of a 2-cell domain that undergoes two refinement cycles with 1 cell refined per cycle. From the tree structure in Figure 2.15b, we introduce the concept of an individual element's *refinement level*, $\ell(K)$, for mesh cell K. The refinement level $\ell(K)$ denotes how many times a cell from the original mesh, $\mathbb{T}_h^0$, has been refined to form element K. By convention, the refinement level for all elements in the original mesh are 0. We note that an element's refinement level is not necessarily the number of refinement cycles that have been performed. In Figure 2.15b, we can see three distinct refinement levels that are represented by the three different vertical levels of nodes. Element 1 has a level of 0 since it was never refined. Elements 2, 3, 4, and 5 have a level of 2. Finally, elements 6, 7, and 8 have a level of 1. We also use Figure 2.15 to further define the concept of an *irregularity index*. This index is the difference in refinement levels between two adjoining mesh elements. The irregularity index for the face separating elements 6 and 7 is 0 since $\ell(6) = \ell(7)$. The irregularity index for the face separating elements 4 and 8 is 1 since $\ell(8) = 1$ and $\ell(4) = 2$. Finally, the irregularity index for the face separating elements 1 and 2 is 2 since $\ell(1) = 0$ and

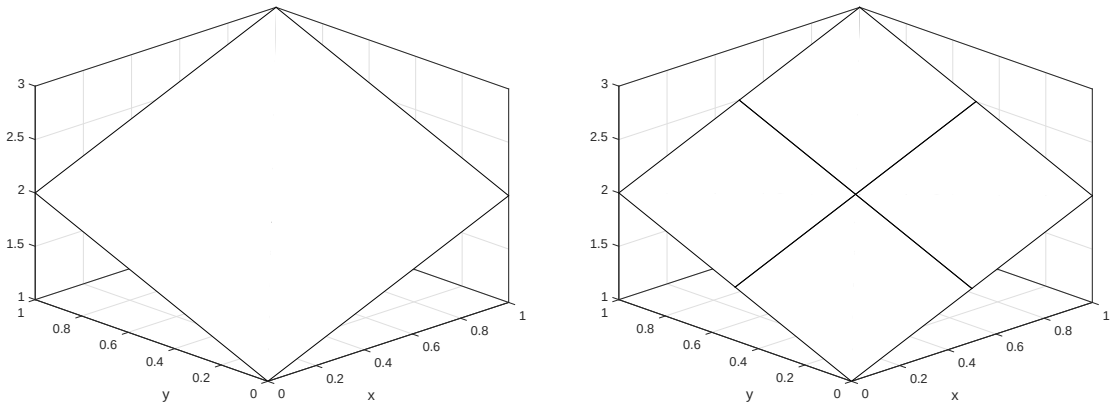


(a) Twice-refined domain (b) Refinement tree structure

Figure 2.15: Hierarchical refinement tree for a simple domain with quadrilateral cells.

$\ell(2) = 2$. For the AMR problems in this work, we restrict the maximum allowable irregularity index for all of the elements for each of the mesh refinement levels.

Following a refinement cycle, we have a new adapted mesh, $\mathbb{T}_h^{r+1}$, but our current solution lives on $\mathbb{T}_h^r$. This means that the solution lives on a lower-dimensional space than the adapted mesh. To solve for the transport solution on $\mathbb{T}_h^{r+1}$, we could simply reinitialize our solution vectors to zero and follow the procedures in Algorithm 1. However, this means that the solution from mesh $\mathbb{T}_h^r$ would simply be discarded. This would discount all the work performed to compute the solution on level r. Furthermore, if there is little difference in the functional space between the two refinement cycles, then the calculated solution at level r would be a good initial guess for the solution to be calculated for level $r + 1$. For this reason, we define the concept of *bootstrapping*, which is the projection of the solution from $\mathbb{T}_h^r$ onto $\mathbb{T}_h^{r+1}$. Figure 2.16 provides an example of bootstrapping by refining a single unit square element onto the four identical daughter elements with $\frac{1}{4}$ the area as the parent element. Figure 2.16a shows a linear planar solution on a single unit square parent element. Then, after this element is refined, its solution is projected onto its four identical daughter elements. Figure 2.16b shows this bootstrapped solution onto the refined mesh, where we can clearly see that the projected solution lives in



(a) Before refinement (b) After refinement

Figure 2.16: Bootstrapping the solution from a single unit square element onto the four daughter elements after refinement.

the dimensional space of the daughter elements but still lives in the functional space of the parent element.

2.8 Conclusions

In this chapter, we have presented the tightly-coupled system of equations that comprise the DGFEM S_N transport equation. We began with the fully-continuous transport equation presented in Section 2.2 and then discretized it in energy, angle and space in Sections 2.3, 2.4, and 2.6, respectively. Appropriate boundary conditions were presented in Section 2.5. For this work, we will only utilize incoming-incident and reflecting boundary conditions and not use any further albedo terms. We finished this chapter in Section 2.7 by describing the procedures that will be utilized to solve our system of equations in energy, angle, and space. We also provided the methodology that we will employ to perform AMR calculations in this work, which yields another mechanism to produce polytope meshes.

3. FEM BASIS FUNCTIONS FOR UNSTRUCTURED POLYTOPES

In Section 2.6, we detailed the spatial discretization of the transport equation. We then proceeded to give the functional forms for the various elementary matrices needed to form the full set of spatially-discretized PDEs. These included the mass, streaming, and surface matrices where the integrations on the element's domain and boundary require combinations of the basis functions' values and gradients.

The remainder of the this chapter is organized as follows. In Section 3.1, we present the 2D, linearly-complete, barycentric, polygonal basis functions that we will analyze in this dissertation. We then present in Section 3.2 the methodology to convert the barycentric polygonal basis functions presented in Section 3.1 into a serendipity space of basis functions with quadratic-completeness. Section 3.4 then presentes the 3D, linearly-complete, polyhedral basis functions that will be exclusively used in Chapter 4 for 3D DSA calculations. We then present numerical results pertaining to our linear and quadratic 2D basis functions in Section 3.5. Section 3.6 concludes with some closing remarks.

3.1 Linear Basis Functions on 2D Polygons

Figure 3.1, gives an image of a reference polygon along with the geometric notations we will use to define the different linear polygonal coordinates. An element, $K \in \mathbb{R}^2$, is defined by a closed set of N_K points (vertices) in $\mathbb{R}^2$. The vertices are ordered $(1, ..., N_K)$ in a counter-clockwise manner without restriction on their convexity. Face j on the polygon, e_j, is defined as the line segment between vertices j and $j+1$. The vertex $j+1$ is determined in general as $j+1 = \mod(j, N_K) + 1$, which gives a wrap-around definition of vertex $N_K + 1 = 1$.

We complete our geometric description for the polygonal coordinate system by

analyzing a point $\vec{x}$ inside the polygon's domain, as also seen in Figure 3.1. α_j is the angle between the points $(\vec{x}_j, \vec{x}, \vec{x}_{j+1})$. Since element K is defined by a closed set of $\mathbb{R}^2$ points, α_j is strongly bounded: $([0, \pi])$. We conclude by defining $|\vec{u}|$ as the Euclidean distance of the vector $\vec{u}$. This means that $|\vec{x} - \vec{x}_j|$ is the distance between the points $\vec{x}$ and $\vec{x}_j$ and $|e_j|$ is the length of face j between points $\vec{x}_j$ and $\vec{x}_{j+1}$.

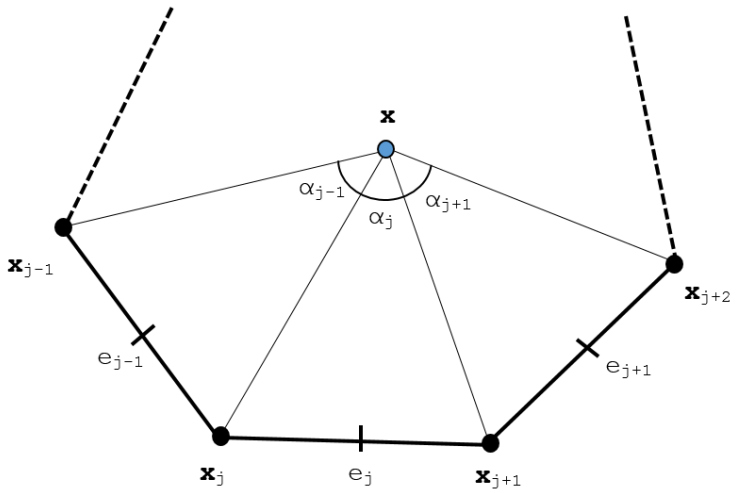


Figure 3.1: Arbitrary polygon with geometric properties used for 2D basis function generation.

In this dissertation, all 1st-order, two-dimensional basis functions for an element K will obey the properties for barycentric coordinates. They will form a *partition of unity*,

$$\sum_{i=1}^{N_K} b_i(\vec{x}) = 1; \tag{3.1}$$

coordinate interpolation will result from an *affine combination* of the vertices,

$$\sum_{i=1}^{N_K} b_i(\vec{x})\vec{x}_i = \vec{x}; \tag{3.2}$$

and they will satisfy the *Lagrange property*,

$$b_i(\vec{x}_j) = \delta_{ij}. \tag{3.3}$$

N_K is again the number of spatial degrees with measure in element K. Using the *partition of unity* of Eq. (3.1), we can rewrite Eqs. (3.1-3.2) into a separate, compact, vectorized form for completeness

$$\sum_{i=1}^{N_K} b_i(\vec{x})\vec{c}_{i,1}(\vec{x}) = \vec{q}_1, \tag{3.4}$$

where $\vec{c}_{i,1}(\vec{x})$ and $\vec{q}_1$ are the lineary-complete constraint and equivalence terms, respectively. These terms are simply:

$$\vec{c}_{i,1}(\vec{x}) = \begin{bmatrix} 1 \\ x_i - x \\ y_i - y \end{bmatrix} \qquad \text{and} \qquad \vec{q}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \tag{3.5}$$

respectively. Equation (3.4) states that our interpolation functions (the basis functions) can exactly reproduce polynomial functions up to order 1. This is why we state that our basis functions are linearly-complete. However, we will not restrict our N_K basis functions to be polynomials. In fact, of the basis functions that we will use, only the PWL coordinates are formed by combinations of polynomial functions.

3.1.1 Wachspress Rational Basis Functions

The first linearly-complete polygonal coordinates that we will consider are the Wachspress rational functions [52]. These rational functions were the first derived for

2D polygons and possess all the properties of the barycentric coordinates previously detailed.

$$b_j^W(\vec{x}) = \frac{w_j(\vec{x})}{\sum_i w_i(\vec{x})} \tag{3.6}$$

where the Wachspress weight function for vertex j, w_j, has the following definition:

$$w_j(\vec{x}) = \frac{A(\vec{x}_{j-1}, \vec{x}_j, \vec{x}_{j+1})}{A(\vec{x}, \vec{x}_{j-1}, \vec{x}_j)\, A(\vec{x}, \vec{x}_j, \vec{x}_{j+1})} \tag{3.7}$$

3.1.2 Piecewise Linear (PWL) Basis Functions

The second linearly-complete 2D polygonal coordinates that we will analyze are the Piecewise Linear (PWL) coordinates proposed by Stone and Adams [53, 54].

$$b_j^{PWL}(x, y) = t_j(x, y) + \alpha_j^K t_c(x, y) \tag{3.8}$$

t_j is the standard 2D linear function with unity at vertex j that linearly decreases to zero to the cell center and each adjoining vertex. t_c is the 2D cell "tent" function which is unity at the cell center and linearly decreases to zero to each cell vertex. α_j^K is the weight parameter for vertex j in cell K.

$$\begin{aligned} b_1(r, s) &= 1 - r - s \\ b_2(r, s) &= r \\ b_3(r, s) &= s \end{aligned} \tag{3.9}$$

3.1.3 Mean Value Basis Functions

At this point, we now introduce the first new polygonal basis set for use with the transport equation: the *mean value coordinates* (MV) developed by Floater [55, 56].

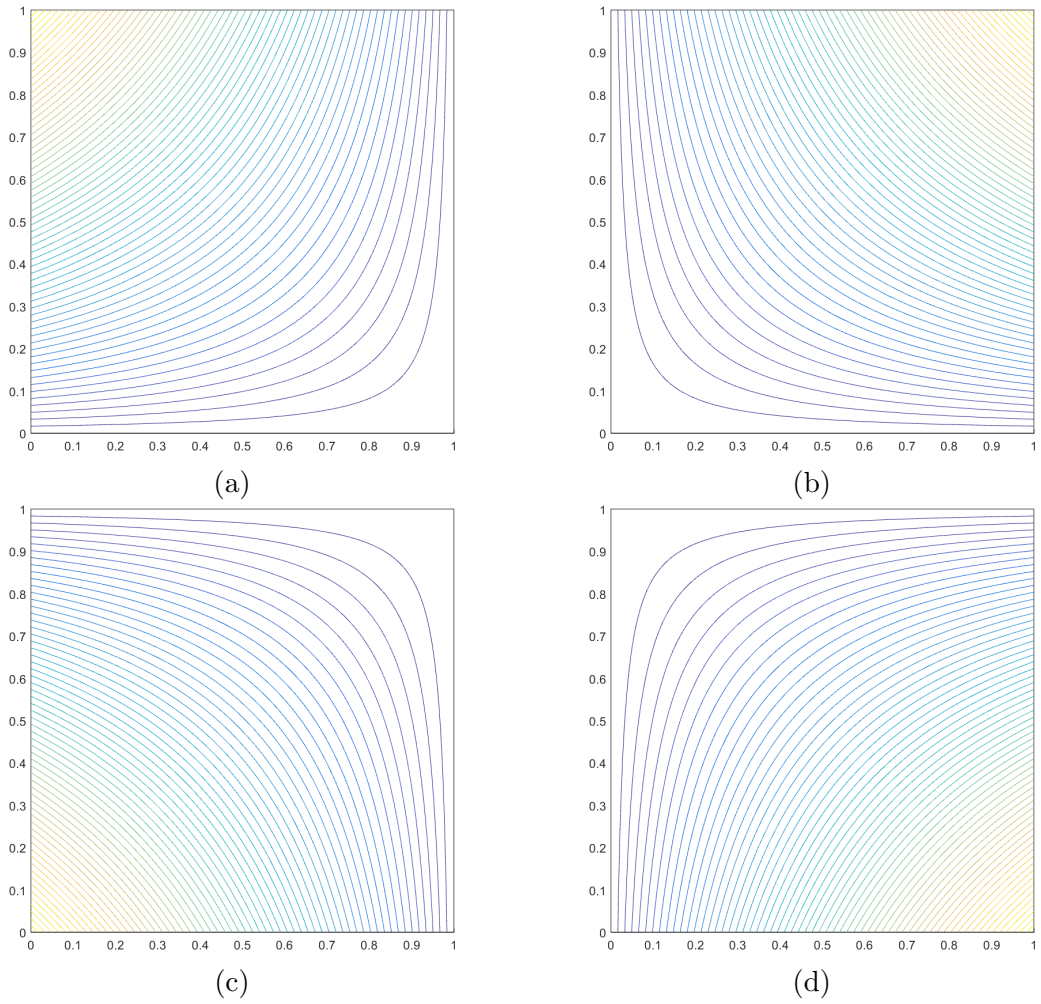


Figure 3.2: Contour plots of the linear Wachspress basis functions on the unit square for the vertices located at: (a) (0,1), (b) (1,1), (c) (0,0), and (d) (1,0).

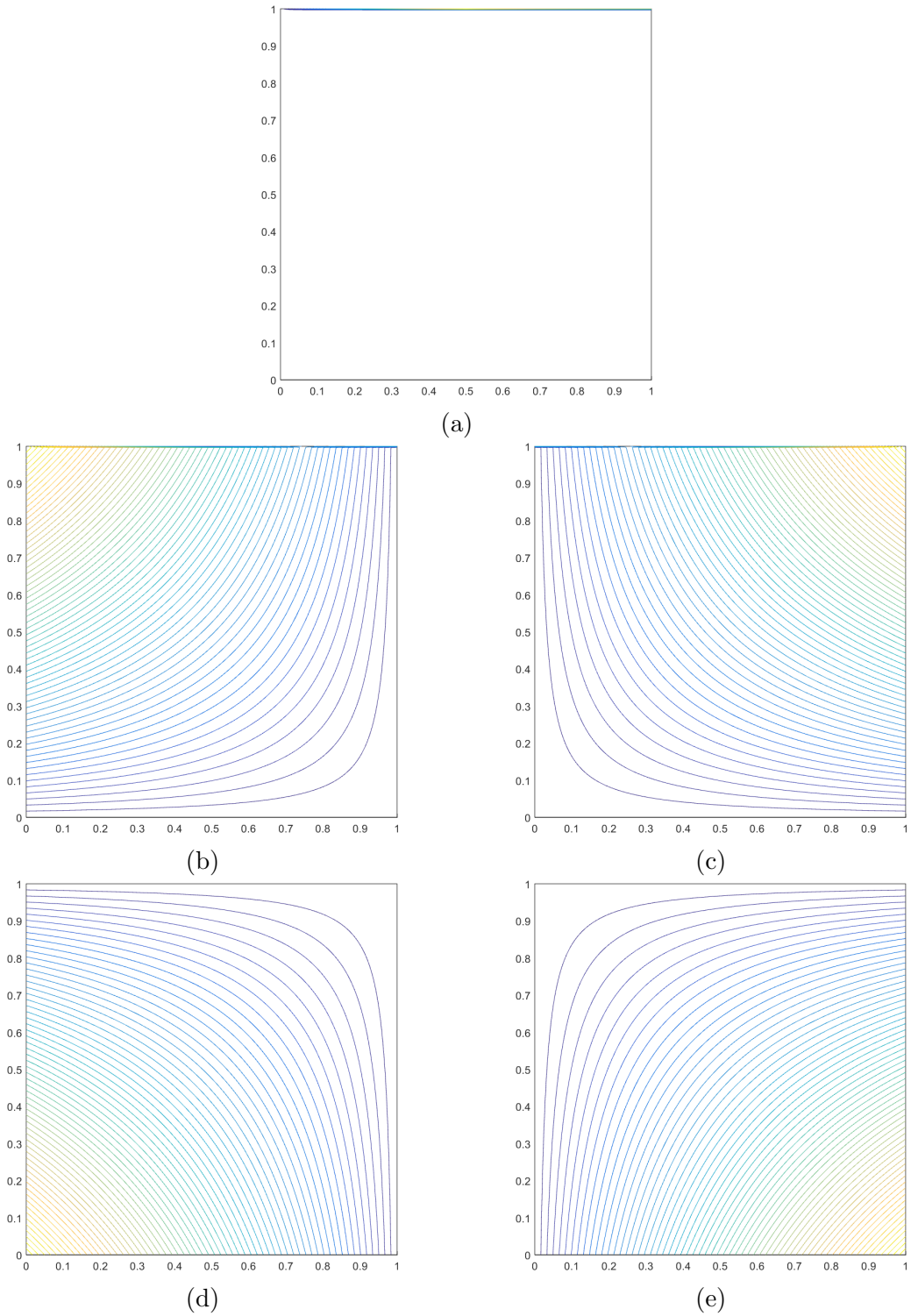


Figure 3.3: Contour plots of the linear Wachspress basis functions on the degnerate pentagon for the vertices located at: (a) (1/2,1), (b) (0,1), (c) (1,1), (d) (0,0), and (e) (1,0).

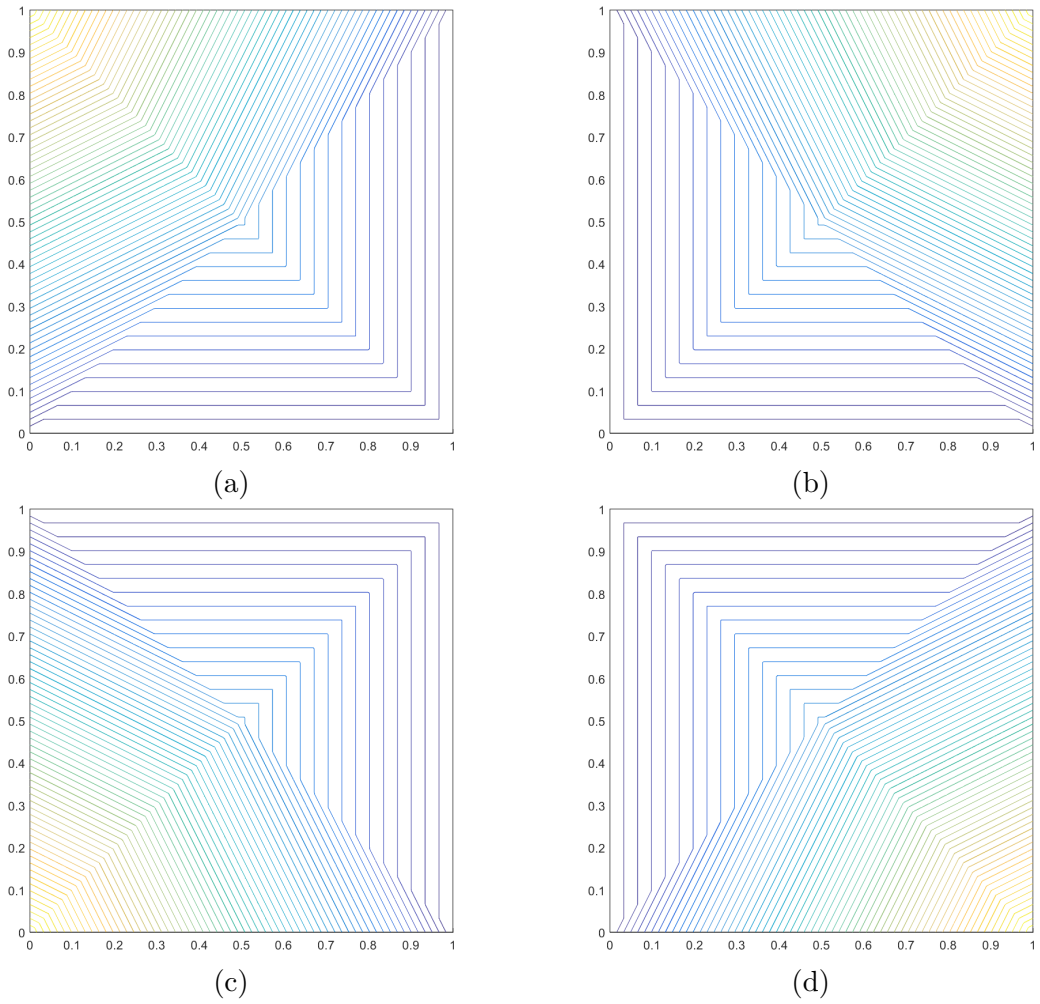


Figure 3.4: Contour plots of the linear PWL basis functions on the unit square for the vertices located at: (a) (0,1), (b) (1,1), (c) (0,0), and (d) (1,0).

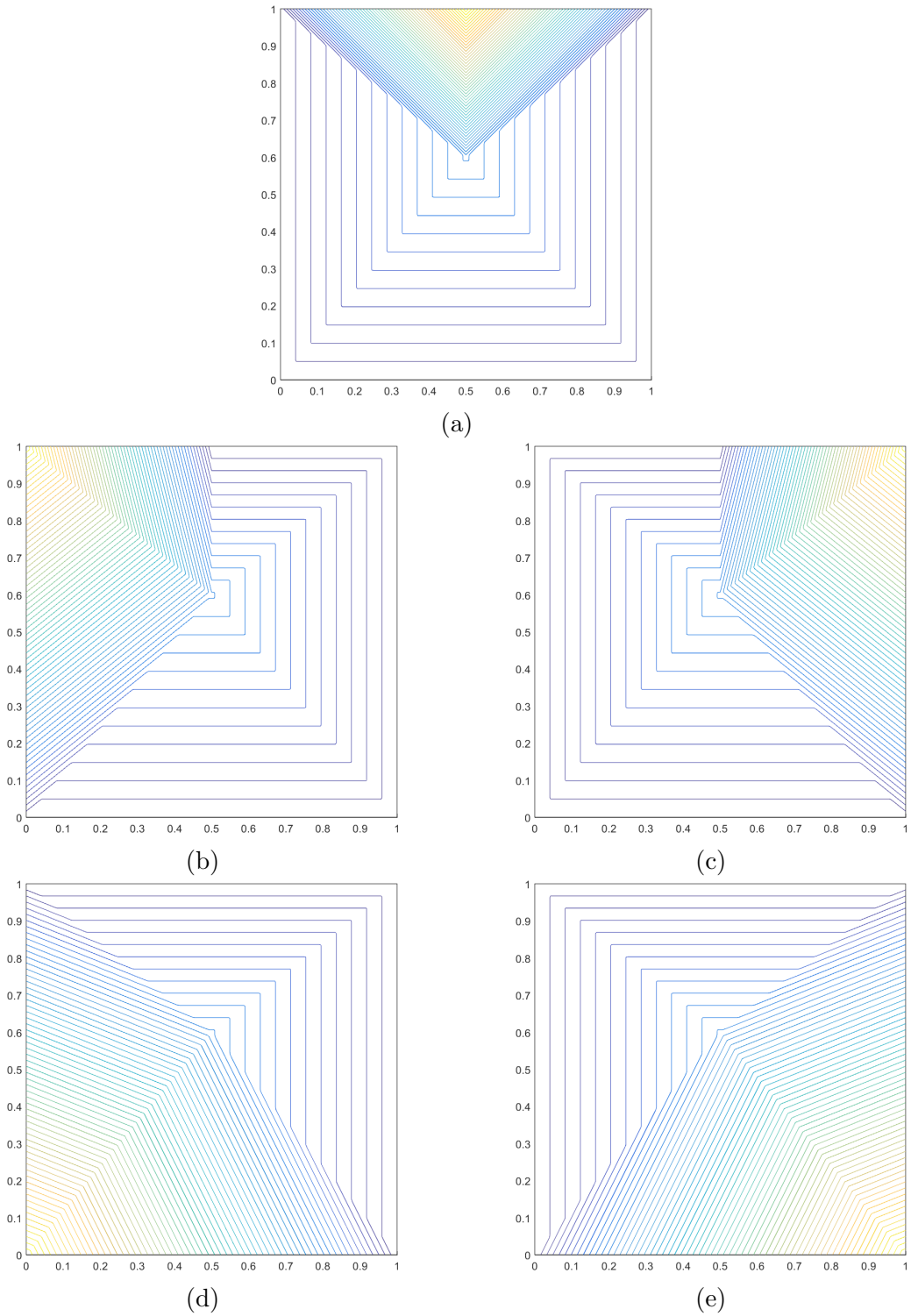


Figure 3.5: Contour plots of the linear PWL basis functions on the degenerate pentagon for the vertices located at: (a) (1/2,1), (b) (0,1), (c) (1,1), (d) (0,0), and (e) (1,0).

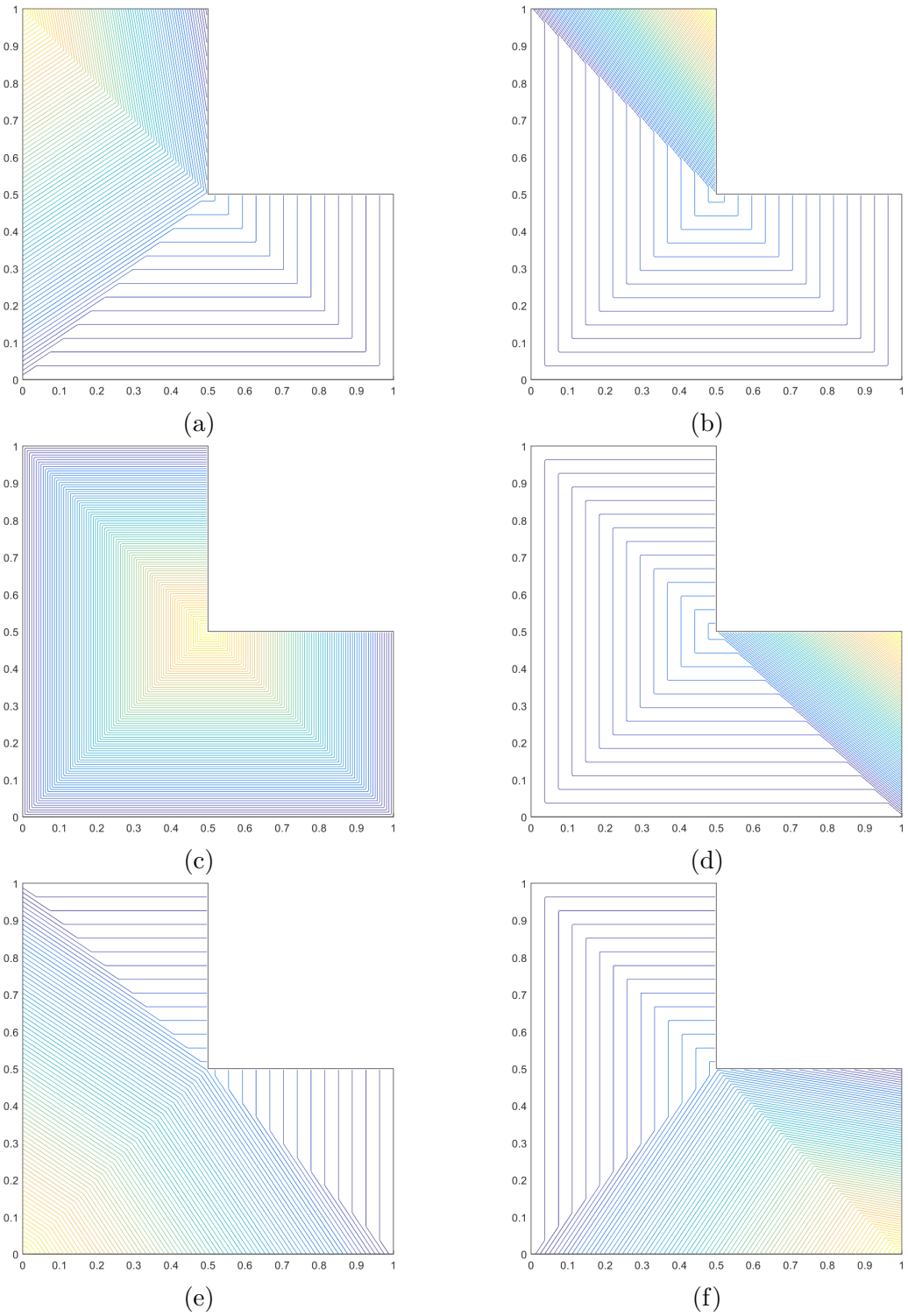


Figure 3.6: Contour plots of the linear PWL basis functions on the L-shaped domain for the vertices located at: (a) (0,1), (b) (1/2,1), (c) (1/2,1/2), (d) (1,1/2), (e) (0,0), and (f) (1,0).

The original motivation behind the MV coordinates was to approximate harmonic maps on a polygon by a set of piecewise linear maps over a triangulation of the polygon for use in computer aided graphic design.

$$\nabla^2 u = 0, \tag{3.10}$$

with $u(\vec{r}) = u_0$ constituting a piecewise linear function

$$b_j^{MV}(\vec{x}) = \frac{w_j(\vec{x})}{\sum_i w_i(\vec{x})} \tag{3.11}$$

where the mean value weight function for vertex j, w_j, has the following definition:

$$w_j(\vec{x}) = \frac{\tan(\alpha_{j-1}/2) + \tan(\alpha_j/2)}{|\vec{x}_j - \vec{x}|} \tag{3.12}$$

3.1.4 Maximum Entropy Basis Functions

The final linearly-complete 2D basis functions that we will analyze in this work are generated by use of the *maximum entropy coordinates* (ME) [57, 58, 59].

$$b_j^{ME}(\vec{x}) = \frac{w_j(\vec{x})}{\sum_i w_i(\vec{x})}. \tag{3.13}$$

where the maximum entropy weight function for vertex j, w_j, has the following definition,

$$w_j(\vec{x}) = m_j(\vec{x}) \exp(-\kappa \cdot (\vec{x}_j - \vec{x})), \tag{3.14}$$

where κ is a vector value of dimension d that will be explained shortly. In Eq. (3.14), m_j is called the prior distribution and is a key component of Bayesian inference [60, 61]. In the context of Eq. (3.14), the prior distribution m_j can be viewed as

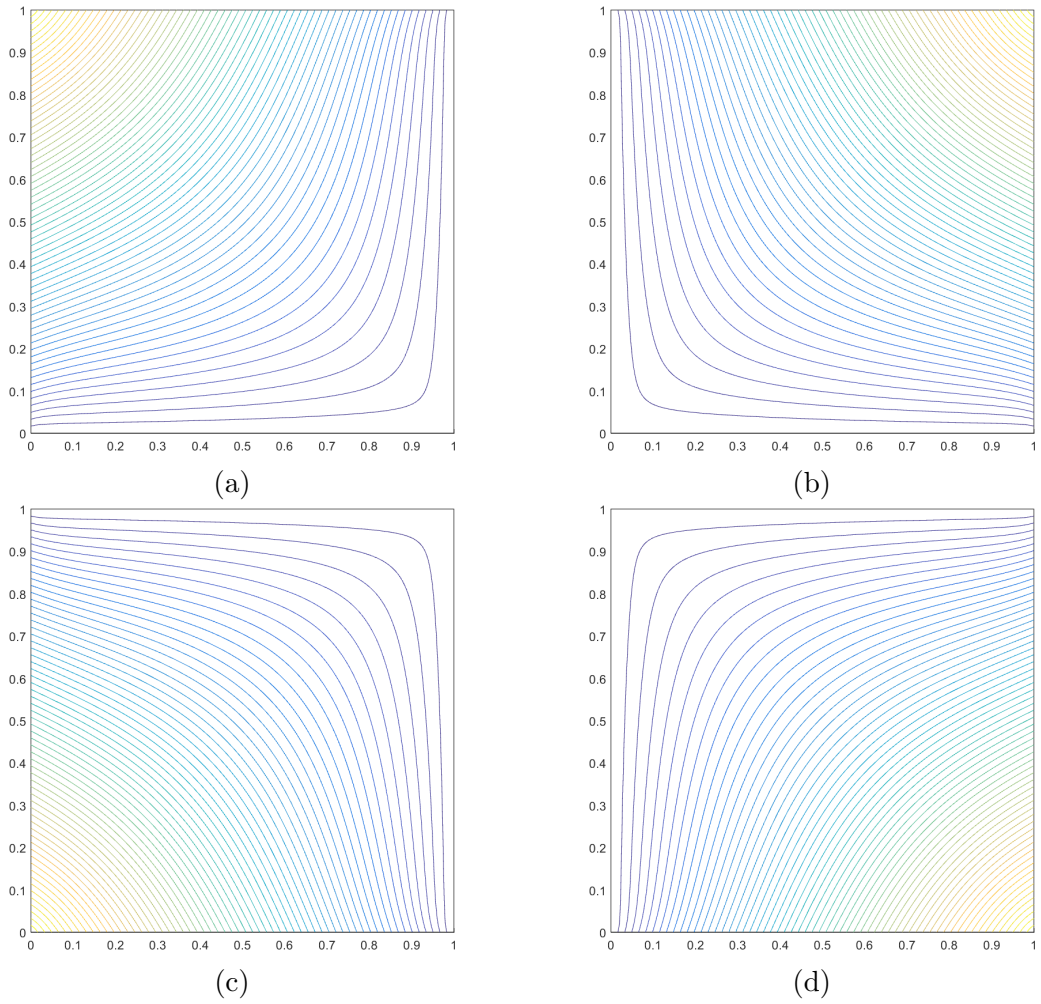


Figure 3.7: Contour plots of the linear mean value basis functions on the unit square for the vertices located at: (a) (0,1), (b) (1,1), (c) (0,0), and (d) (1,0).

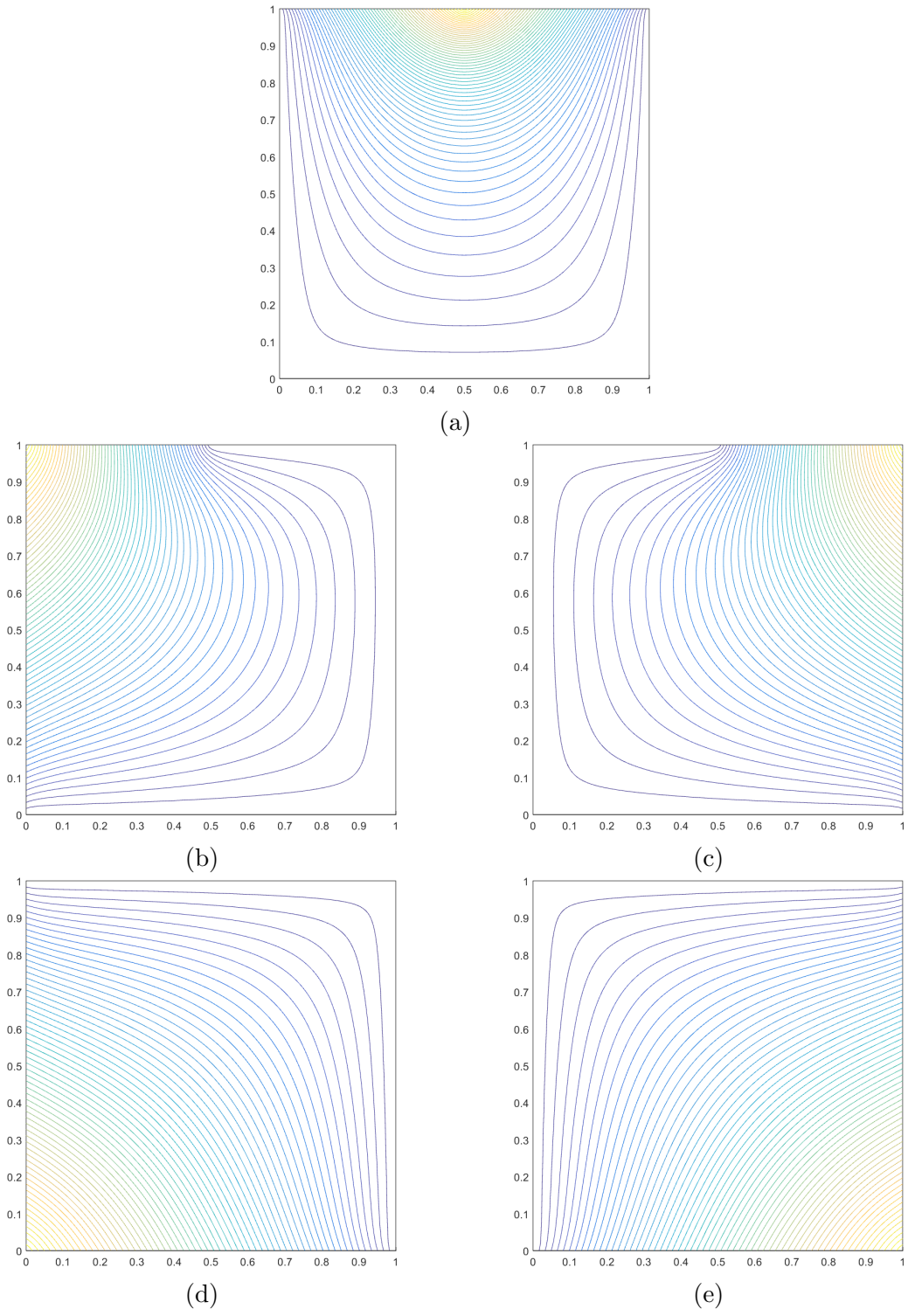


Figure 3.8: Contour plots of the linear mean value basis functions on the degnerate pentagon for the vertices located at: (a) (1/2,1), (b) (0,1), (c) (1,1), (d) (0,0), and (e) (1,0).

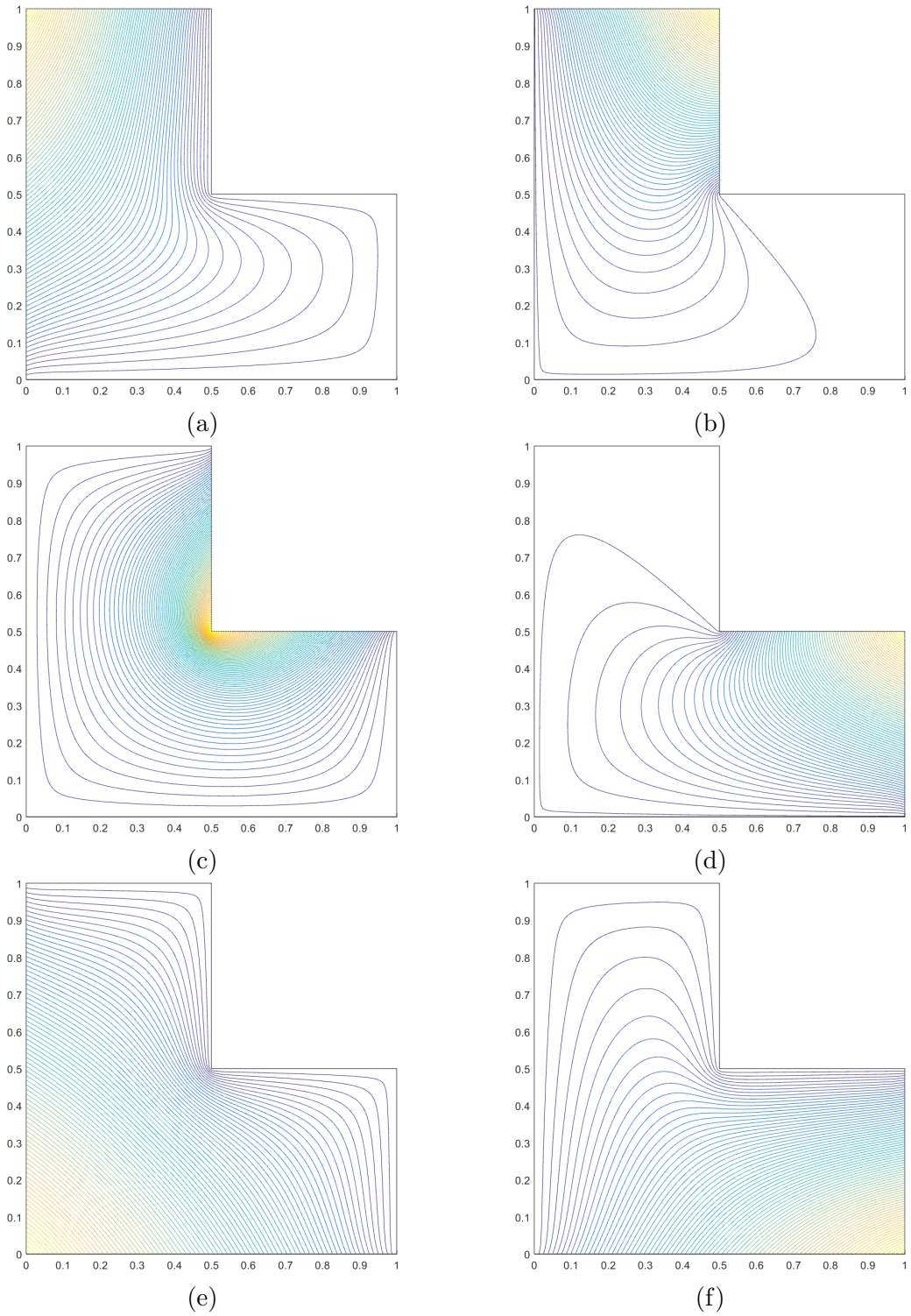


Figure 3.9: Contour plots of the linear mean value basis functions on the L-shaped domain for the vertices located at: (a) (0,1), (b) (1/2,1), (c) (1/2,1/2), (d) (1,1/2), (e) (0,0), and (f) (1,0).

Table 3.1: Summary of the 2D coordinate systems used on polygons.

Basis Function	Dimension	Polytope Types	Integration	Direct/Iterative
Wachspress	2D/3D	Convex	Numerical	Direct
PWL	1D/2D/3D	Convex/Concave	Analytical	Direct
Mean Value	2D	Convex/Concave	Numerical	Direct
Max Entropy	1D/2D/3D	Convex/Concave	Numerical	Iterative

a weight function associated with vertex j. This means that there is variability that one can employ for these weight functions. These weight functions can then be tailored depending on the application and the numerical scheme employed.

$$m_j(\vec{x}) = \frac{\pi_j(\vec{x})}{\sum_k \pi_k(\vec{x})} \tag{3.15}$$

where

$$\pi_j(\vec{x}) = \prod_{i \neq j-1,j} \rho_j(\vec{x}) \tag{3.16}$$

where

$$\rho_j(\vec{x}) = ||\vec{x} - \vec{x}_j|| + ||\vec{x} - \vec{x}_{j+1}|| - ||\vec{x}_{j+1} - \vec{x}_j|| \tag{3.17}$$

3.1.5 Summary of 2D Linear Basis Functions on Polygons

3.2 Converting the Linear Polygonal Basis Functions to the Quadratic Serendipity Space

Now that we have given complete details on the linearly-complete generalized barycentric coordinates that we will investigate for this work, we

for the constant constraint,

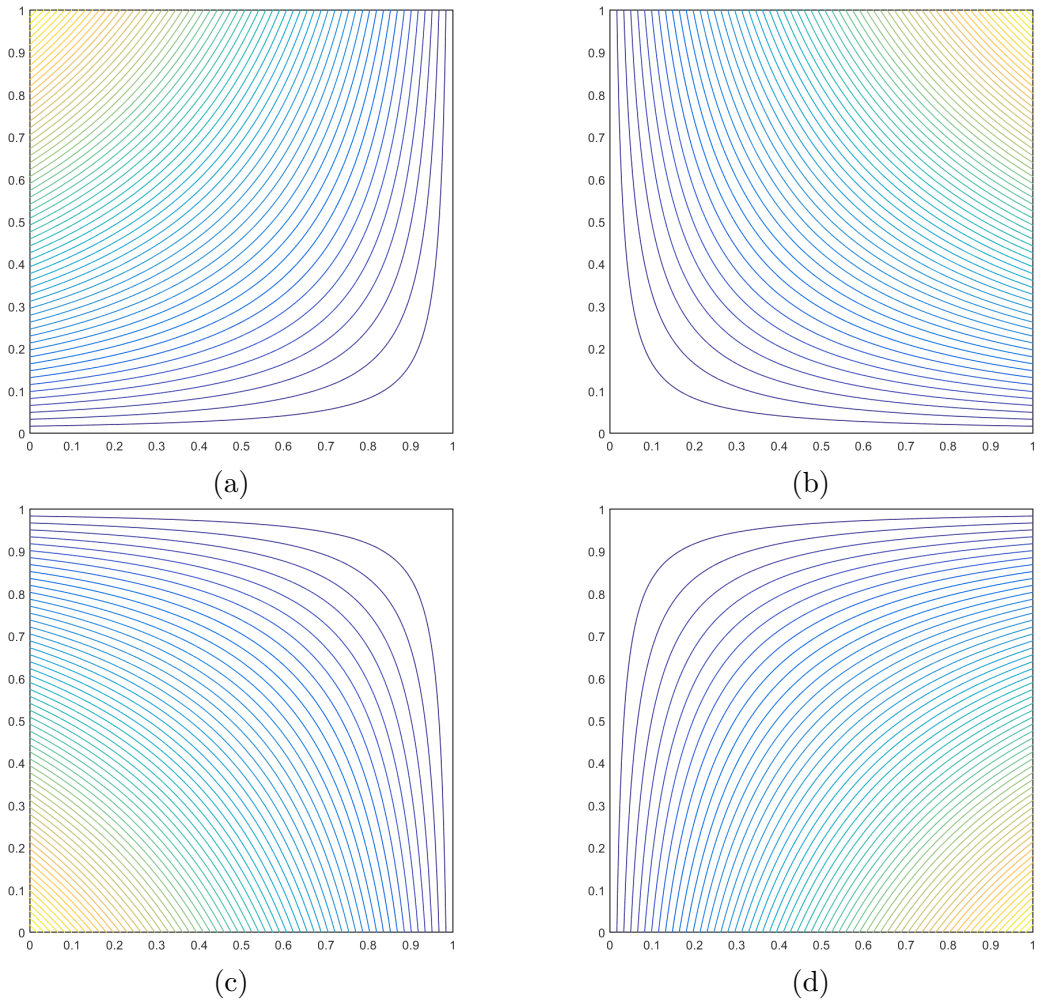


Figure 3.10: Contour plots of the linear maximum entropy basis functions on the unit square for the vertices located at: (a) (0,1), (b) (1,1), (c) (0,0), and (d) (1,0).

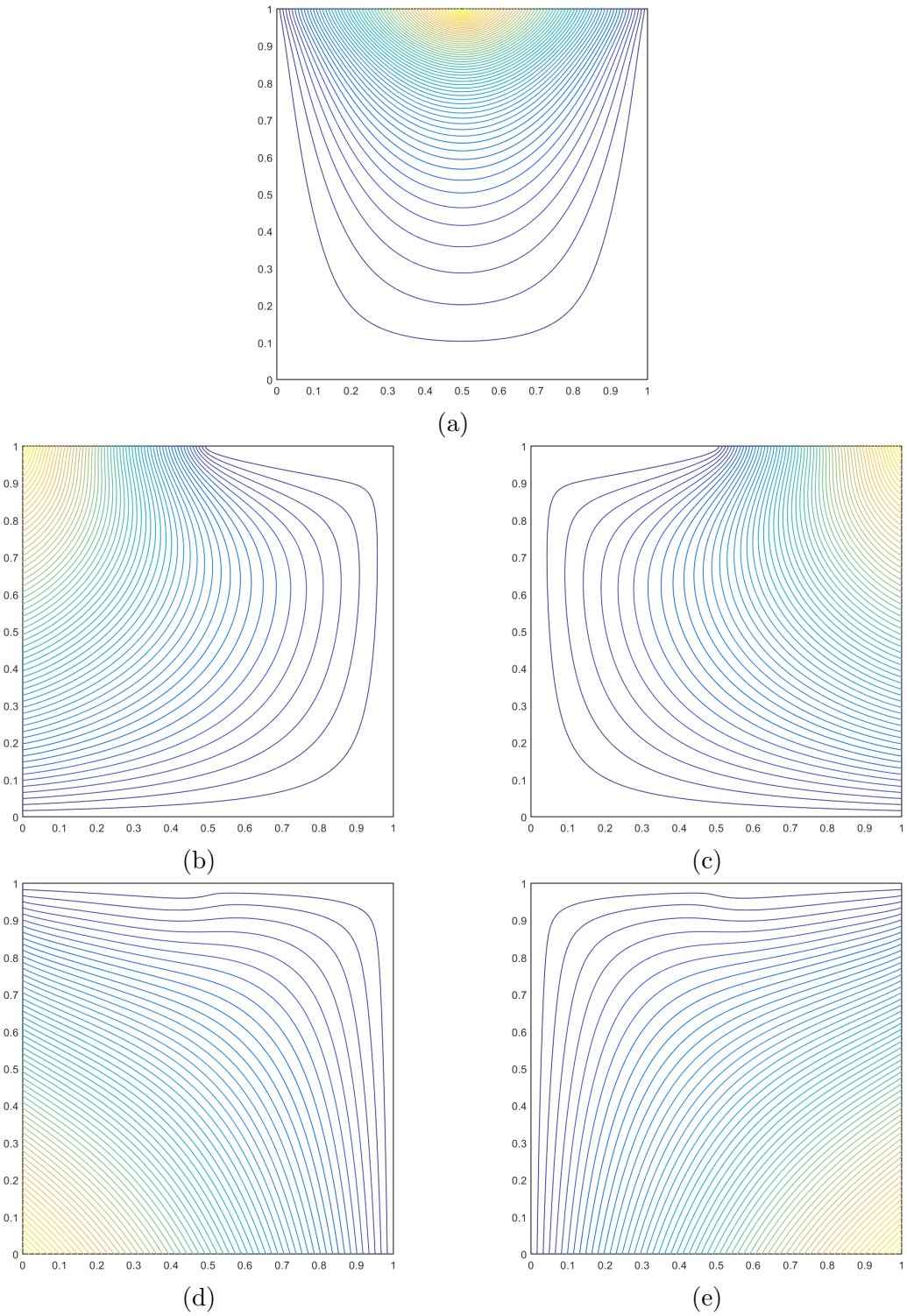


Figure 3.11: Contour plots of the linear maximum entropy basis functions on the degenerate pentagon for the vertices located at: (a) (1/2,1), (b) (0,1), (c) (1,1), (d) (0,0), and (e) (1,0).

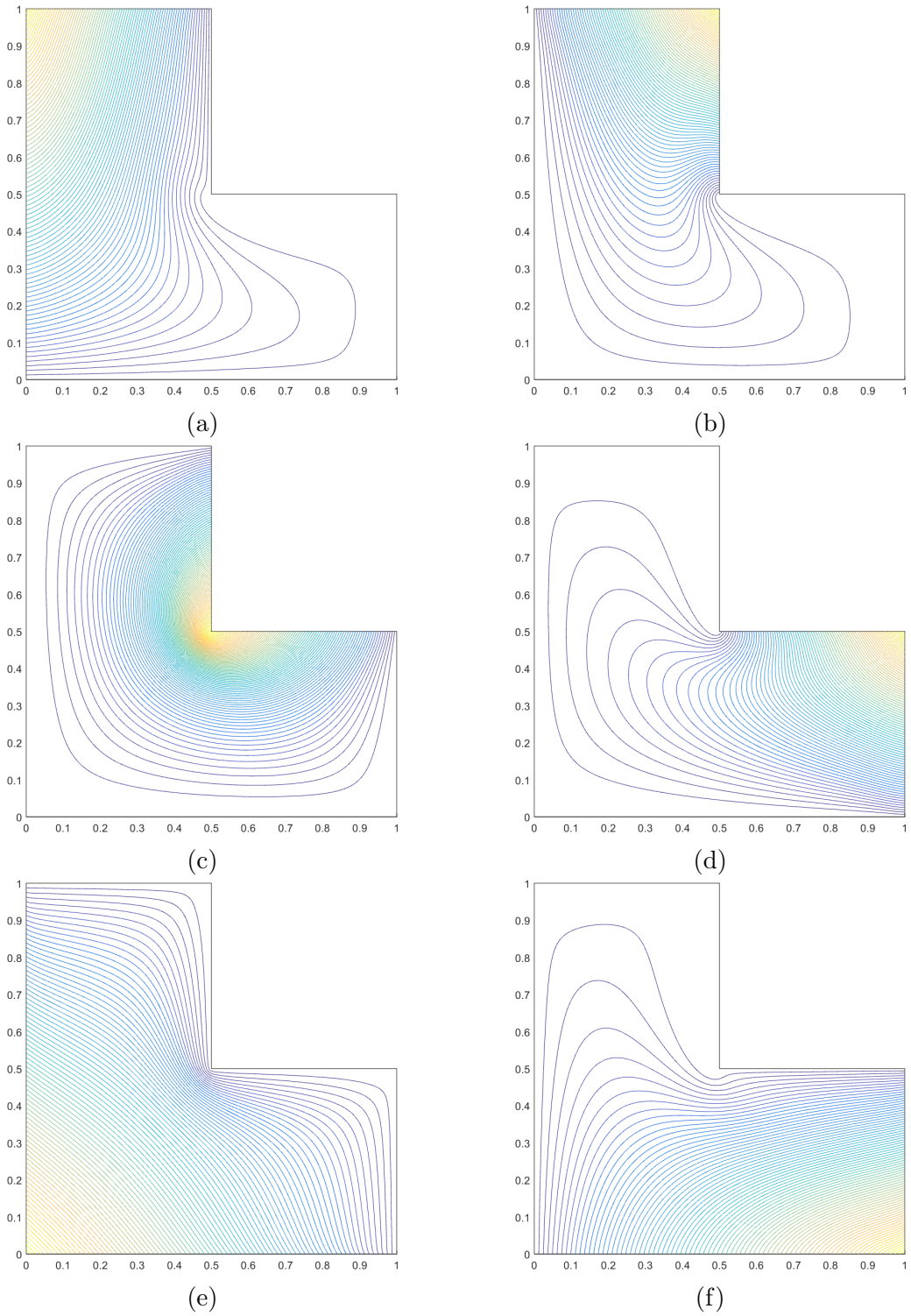
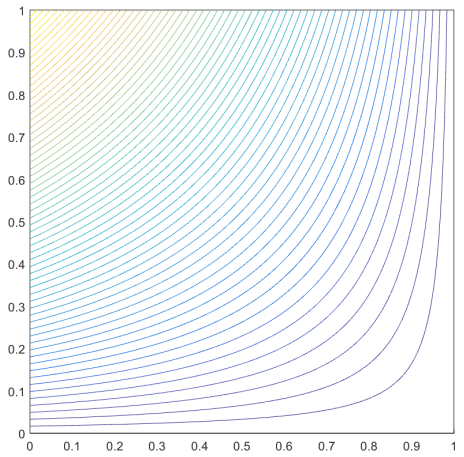
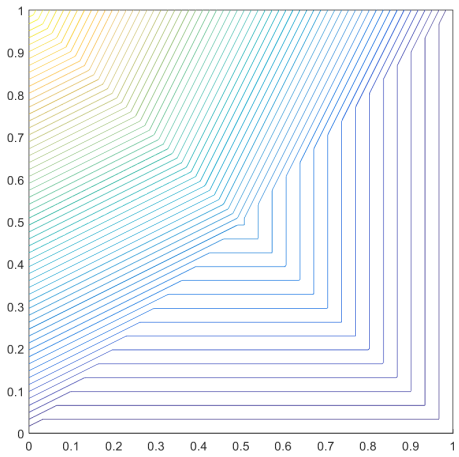


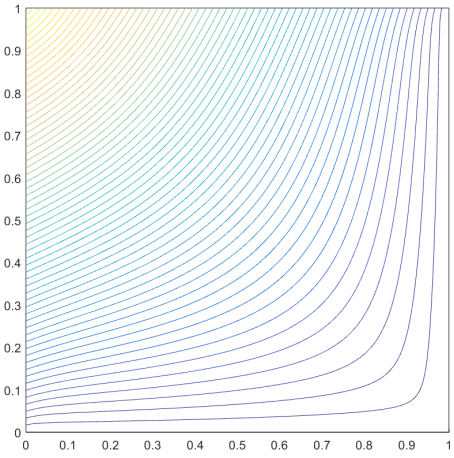
Figure 3.12: Contour plots of the linear maximum entropy basis functions on the L-shaped domain for the vertices located at: (a) (0,1), (b) (1/2,1), (c) (1/2,1/2), (d) (1,1/2), (e) (0,0), and (f) (1,0).



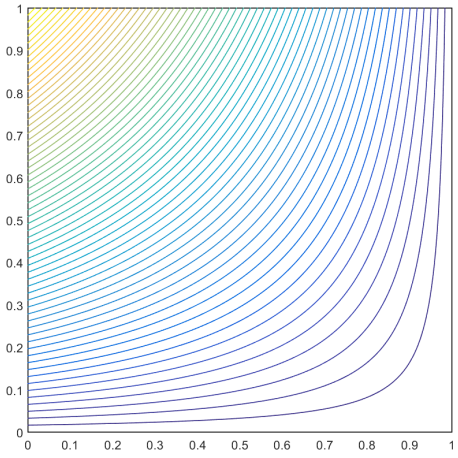
(a) Wachspress



(b) PWL

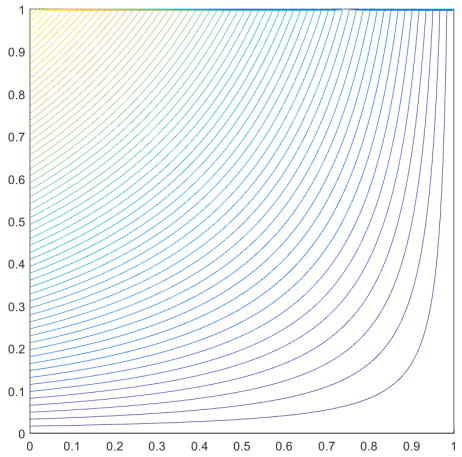


(c) mean value

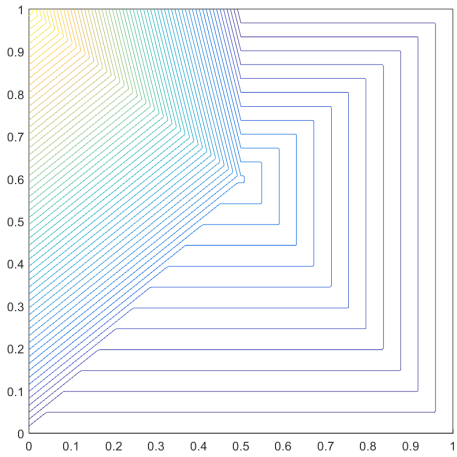


(d) maximum entropy

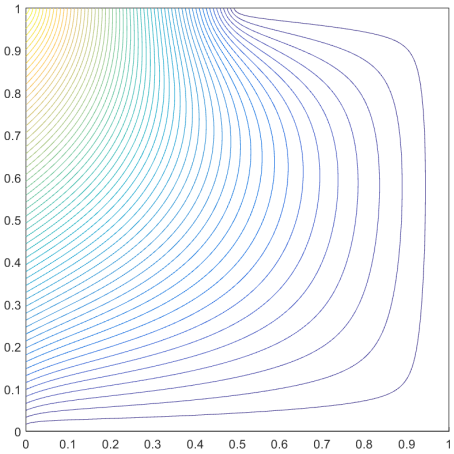
Figure 3.13: Contour plots of the different linear basis function on the unit square located at vertex (0,1).



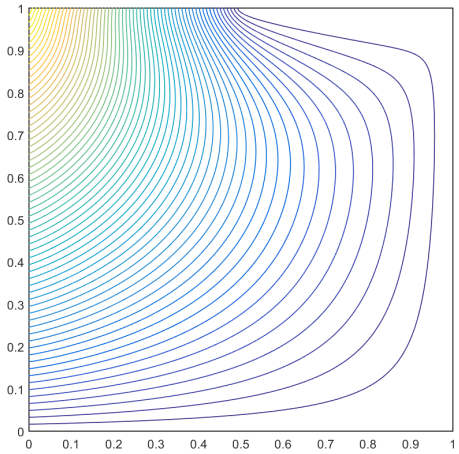
(a) Wachspress



(b) PWL



(c) Mean Value



(d) Maximum Entropy

Figure 3.14: Contour plots of the different linear basis function on the degenerate pentagon located at vertex (0,1). It is clear that the Wachspress coordinates fail for the weakly convex case.

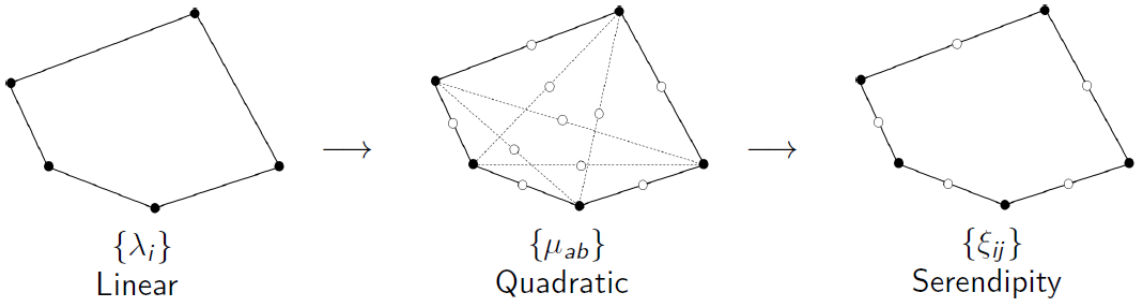


Figure 3.15: Overview of the process to construct the quadratic serendipity basis functions on polygons. The filled dots correspond to basis functions that maintain the Lagrange property while empty dots do not.

$$\sum_{a=1}^{N_K}\sum_{b=1}^{N_K}\mu_{ab}(\vec{x}) = 1, \tag{3.18}$$

for the linear constraint,

$$\sum_{a=1}^{N_K}\sum_{b=1}^{N_K}\mu_{ab}(\vec{x})\,\vec{x}_a = \vec{x}, \tag{3.19}$$

and for the quadratic constraint,

$$\sum_{a=1}^{N_K}\sum_{b=1}^{N_K}\mu_{ab}(\vec{x})\ (\vec{x}_a \otimes \vec{x}_b) = \vec{x} \otimes \vec{x}. \tag{3.20}$$

$$\vec{x}_{ab} = \frac{\vec{x}_a + \vec{x}_b}{2} \tag{3.21}$$

With this definition

for the constant constraint,

$$\sum_{aa\in V}\mu_{aa}(\vec{x}) + \sum_{ab\in E\cup D} 2\mu_{ab}(\vec{x}) = 1, \tag{3.22}$$

for the linear constraint,

$$\sum_{aa \in V} \mu_{aa}(\vec{x})\, \vec{x}_{aa} + \sum_{ab \in E \cup D} 2\mu_{ab}(\vec{x})\, \vec{x}_{ab} = \vec{x}, \tag{3.23}$$

and for the quadratic constraint,

$$\sum_{aa \in V} \mu_{aa}(\vec{x})\ (\vec{x}_a \otimes \vec{x}_a) + \sum_{ab \in E \cup D} \mu_{ab}(\vec{x})\ (\vec{x}_a \otimes \vec{x}_b + \vec{x}_b \otimes \vec{x}_a) = \vec{x} \otimes \vec{x}. \tag{3.24}$$

for the constant constraint,

$$\sum_{ii \in V} \xi_{ii}(\vec{x}) + \sum_{i(i+1) \in E} 2\xi_{i(i+1)}(\vec{x}) = 1, \tag{3.25}$$

for the linear constraint,

$$\sum_{ii \in V} \xi_{ii}(\vec{x})\, \vec{x}_{ii} + \sum_{i(i+1) \in E} 2\xi_{i(i+1)}(\vec{x})\, \vec{x}_{i(i+1)} = \vec{x}, \tag{3.26}$$

and for the quadratic constraint,

$$\sum_{ii \in V} \xi_{ii}(\vec{x})\ (\vec{x}_i \otimes \vec{x}_i) + \sum_{i(i+1) \in E} \xi_{i(i+1)}(\vec{x})\ (\vec{x}_i \otimes \vec{x}_{i+1} + \vec{x}_{i+1} \otimes \vec{x}_i) = \vec{x} \otimes \vec{x}. \tag{3.27}$$

$$\{\xi\} = \mathbb{A}\,\{\mu\} \tag{3.28}$$

$$\mathbb{A} = \begin{bmatrix} c_{11}^{11} & \dots & c_{ab}^{11} & \dots & c_{(n-2)n}^{11} \\ \dots & \ddots & \vdots & \ddots & \vdots \\ c_{11}^{ij} & \dots & c_{ab}^{ij} & \dots & c_{(n-2)n}^{ij} \\ \dots & \ddots & \vdots & \ddots & \vdots \\ c_{11}^{n(n+1)} & \dots & c_{ab}^{n(n+1)} & \dots & c_{(n-2)n}^{n(n+1)} \end{bmatrix} \tag{3.29}$$

$$\mathbb{A} = \left[\mathbb{I} \,|\, \mathbb{A}'\right] \tag{3.30}$$

where $\mathbb{I}$ is the $(2N_K \text{x} 2N_K)$ identity matrix, and $\mathbb{A}'$ is a full $(2N_K \text{x} N_Q)$ matrix. This means that the vertex and face midpoint serendipity functions, ξ_{ij}, are formed by taking their corresponding quadratic function, μij, and adding some linear combination of the interior functions. Therefore, we only need to determine

$$B^* = B^T(BB^T)^{-1} \tag{3.31}$$

3.3 Integrating the Arbitrary 2D Polygonal Elements

Sections 3.1 and 3.2 detail how the basis functions and their gradients can be computed at different points on a 2D polygonal element. These basis functions and gradients can then be used to calculate the integrals of the elementary matrices for a given element K as described in Section 2.6. Because the elementary matrix integrals using the Wachspress, mean value, and maximum entropy coordinates cannot be integrated analytically, we need to define a numerical quadrature scheme. The spatial quadrature sets need to be amenable to arbitrary polygons and also integrate polynomials exactly (the different polynomial orders of the basis functions). Efficient quadrature schemes exist for both triangles and quadrilaterals [62, 63, 64, 65, 66]. These include symmetric rules on triangles and cubature and tensor-product rules on

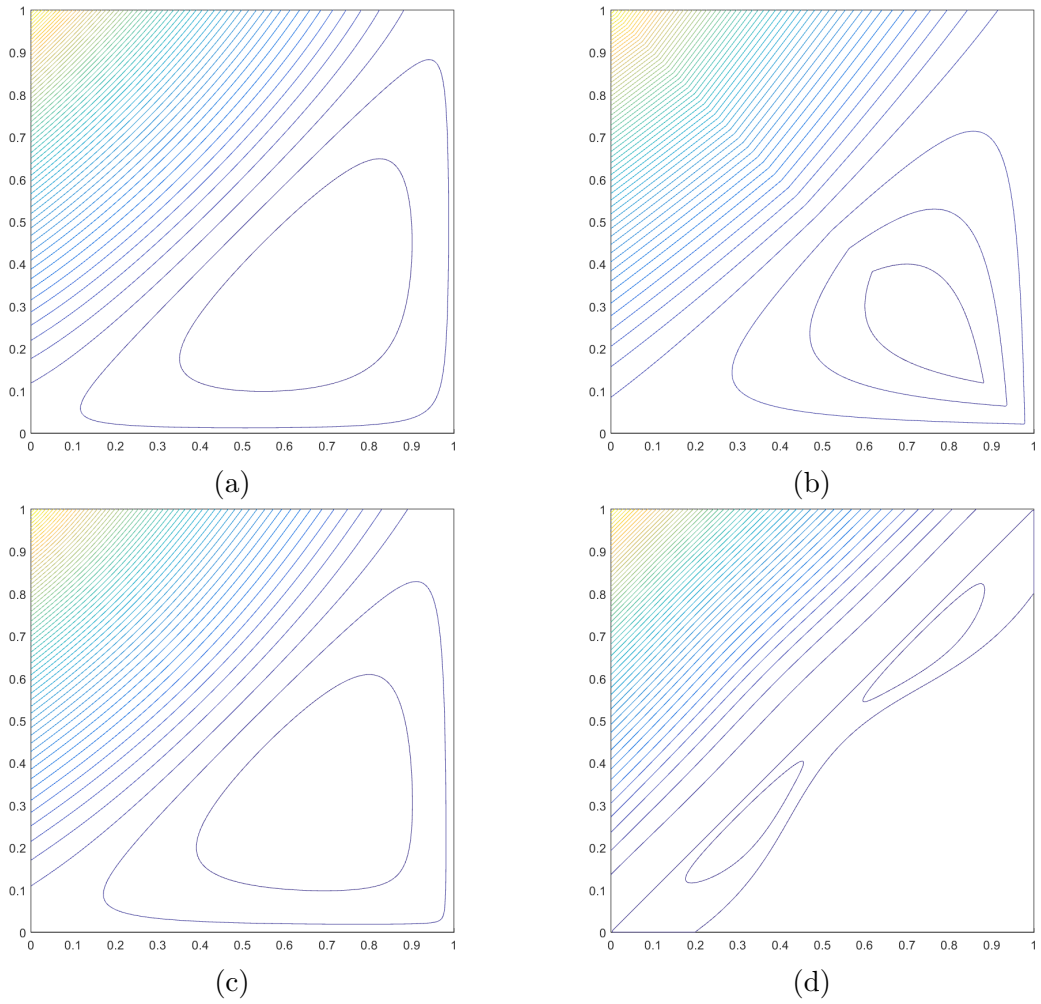


Figure 3.16: Contour plots of the different quadratic serendipity basis function on the unit square located at vertex (0,1): (a) Wachspress, (b) PWL, (c) mean value, and (d) maximum entropy.

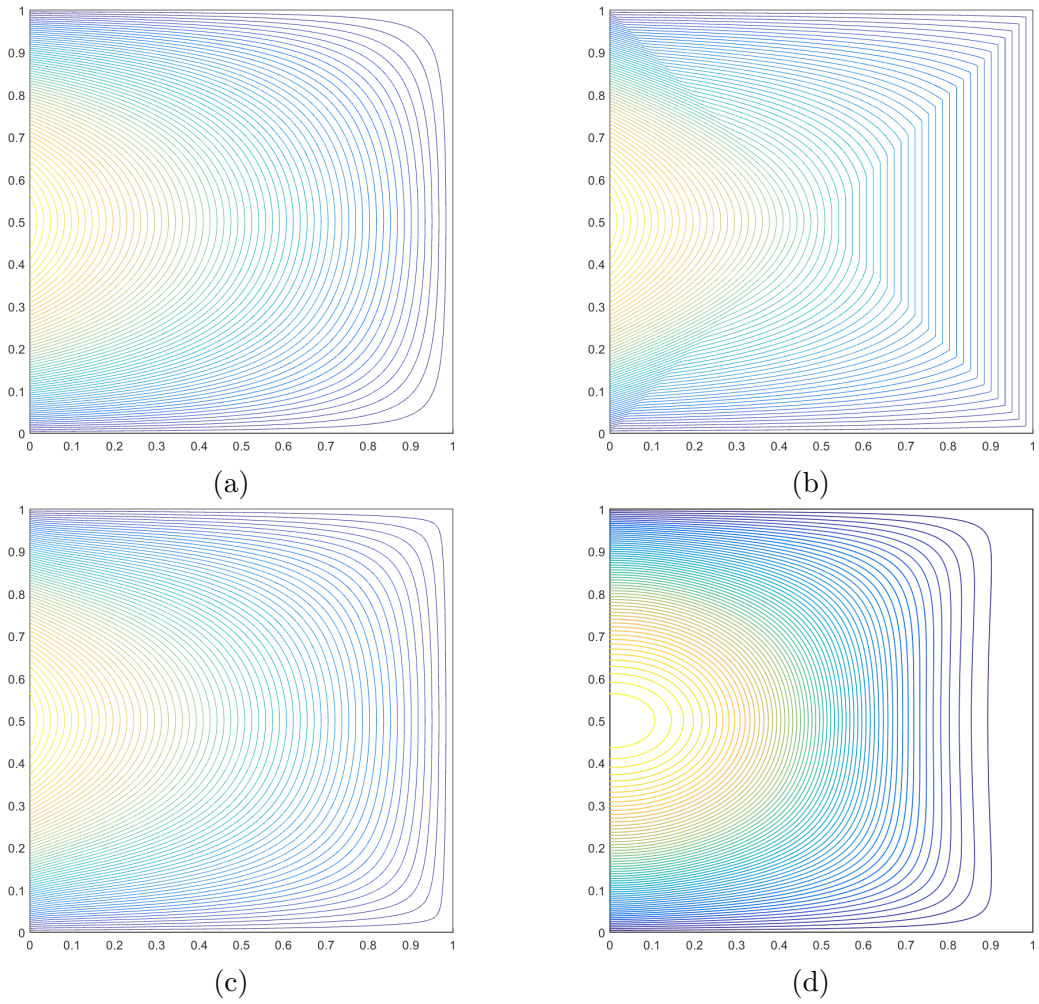


Figure 3.17: Contour plots of the different quadratic serendipity basis function on the unit square at a mid-face node located at (0,1/2): (a) Wachspress, (b) PWL, (c) mean value, and (d) maximum entropy.

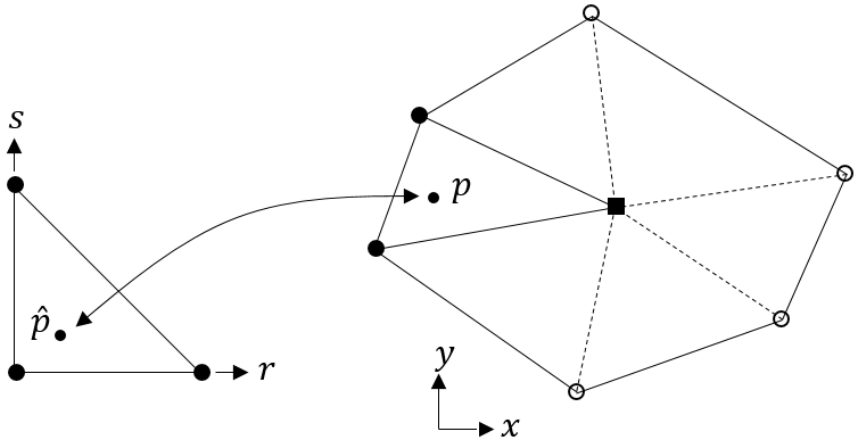


Figure 3.18: blah.

triangles and quadrilaterals, respectively. However, polygons have an infinite number of topological shapes and explicit quadrature rules cannot be defined. Because of this, the development of efficient quadrature rules for arbitrary polytopes is an ongoing field of research [67, 68, 69].

At this time, we were only interested in the accuracy and not the efficiency of the integration of the elementary matrices. Therefore, we simply choose to use a simple triangulation-based scheme. The global polygonal element K with N_K vertices is sub-divided into N_K separate triangles. Each of these triangles is formed from two adjacent vertices and the polygon's centroid, $\vec{c}$. For convex and degenerate (not concave) polygons the centroid can be the average of the vertex coordinates, which is simply given by,

$$\vec{c} = \frac{1}{N_K} \sum_{i=1}^{N_K} \vec{x}_i. \tag{3.32}$$

Then for each sub-triangle, a quadrature rule with N_q nodes is employed (we do not vary the number of nodes between sub-triangles). This quadrature rule is specified

in the reference space of $\{r, s\}$ on the unit triangle with vertices of (0,0), (1,0), and (0,1). We have chosen a symmetric reference quadrature set that is well documented in the literature [63]. We denote this reference quadrature rule by $\left\{\hat{x}_q, \hat{w}_q\right\}_{q=1}^{N_q}$, where the symbol $\hat{}$ denotes any quantity that lives in the reference space. We note that the sum of the reference weights equals the reference triangle area of 1/2. This reference quadrature is mapped into the global space of the sub-triangle by an affine transformation. The mapping of a point from the reference space, $\hat{\mathbf{p}}$, to its corresponding point in global space, $\mathbf{p}$, is done with,

$$\mathbf{p} = \mathbf{x}_0 + J\hat{\mathbf{p}}, \tag{3.33}$$

where $\mathbf{x}_0$ is the global position of one of the sub-triangle vertices and J is the Jacobian matrix of the transformation. This mapping is presented graphically in Figure 3.18. If the global positions of the sub-triangle vertices are given by $\vec{x}_0$, $\vec{x}_1$, and $\vec{x}_2$, then the Jacobian is given by the following,

$$J = \begin{bmatrix} x_1 - x_0 & y_1 - y_0 \\ x_2 - x_0 & y_2 - y_0 \end{bmatrix}. \tag{3.34}$$

The Jacobian matrix can also be used to map the gradients between the reference and global spaces. The gradient of the reference space can be computed in terms of the global space by,

$$\nabla_{\hat{x}} = J\nabla_x, \tag{3.35}$$

and the gradient of the global space can be computed in terms of the reference space by,

$$\nabla_x = J^{-1}\nabla_{\hat{x}} \tag{3.36}$$

With the positions of the nodes mapped to the global space, that just leaves the weights. The value of the global weight q on sub-triangle i within polygon K (given by $w_{i,q}^K$) is mapped from the corresponding reference weight, $\hat{w}_q$, by

$$w_{i,q}^K = \hat{w}_q |J_i|. \tag{3.37}$$

In Eq. (3.37), $|J_i|$ is the determinant of the Jacobian matrix corresponding the transformation to sub-triangle i, and it is equal to 2 times the area of the sub-triangle i. This means that the determinant acts to normalize the weights so that their sum is equal to the sub-triangle's area. Therefore, summing all the weights of all of the sub-triangles will equal the total area of the polygon K.

To this point, we have provided the means to generate the quadrature nodes and weights within the global space of a polygon K. Next, the values and gradients of the basis functions at these quadrature nodes can be calculated by the procedures outlined in Sections 3.1 and 3.2. Then, the function f can be integrated over the polygon K by the double sum,

$$\int_K f = \sum_{i=1}^{N_K} \sum_{q=1}^{N_q} w_{i,q}^K f(\vec{x}_{i,q}), \tag{3.38}$$

where $w_{i,q}^K$ and $\vec{x}_{i,q}$ correspond to the quadrature weights and global positions for node q within sub-triangle i, respectively. In this case, the function f can either be some scalar quantity or an elementary matrix. Thus, the elementary matrices needed for the DGFEM formulation of the transport equation can be computed in a logical manner for any arbitrary polygon. In a similar manner, the integral of f over the

entire mesh, $\mathbb{T}_h$, is simply the sum of integrals over all elements. This integration of f over the entire domain is simply given by

$$\int_{\mathbb{T}_h} f = \sum_{K \in \mathbb{T}_h} \left(\sum_{i=1}^{N_K} \sum_{q=1}^{N_q} w_{i,q}^K f(\vec{x}_{i,q}) \right), \tag{3.39}$$

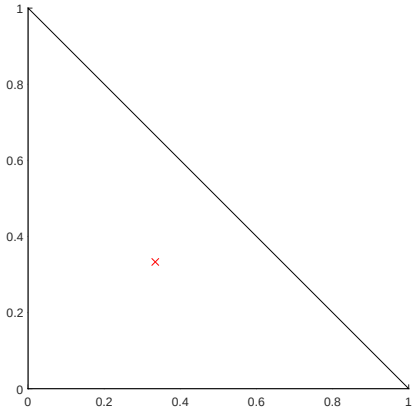
which is clearly just an element-wise sum of Eq. (3.38).

We conclude this section by providing some visual examples of quadrature sets for polygonal elements. Figure 3.19 gives quadrature sets on the reference triangle for orders 1-6. We can see that our reference quadrature is symmetric about any of the three vertices, though we note that true isoparametric symmetry is obtained with equilateral triangles. Then Figures 3.20 and 3.21 provide examples of the mapping of the reference quadrature into the global space for a regular pentagon and hexagon, respectively.

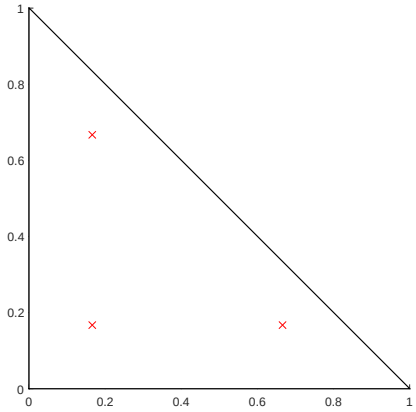
3.4 Linear Basis Functions on 3D Polyhedra

We have defined linearly-complete and quadratically-complete 2D polygonal basis functions. Next, we present an efficient coordinate system for arbitrary 3D polyhedra that is linearly-complete. However, at the time of this work, no analagous methodology to convert 3D linear coordinates to their serendipity basis exists. Therefore, we will utilize only a single linearly-complete 3D coordinate system for some of the analysis to be performed in Chapter 4.

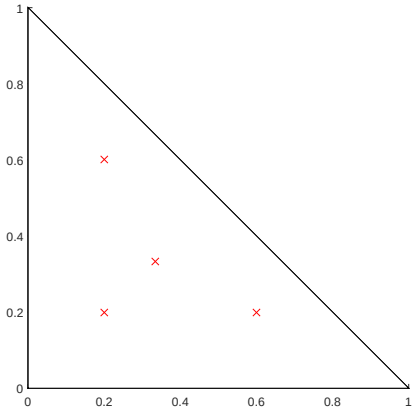
For this work, we will utilize the 3D version of the Piecewise Linear (PWL) coordinates. From Table 3.1, we can see some of the properties of the different 2D polygonal coordinates. The PWL functions are the only coordinates with a 3D analogue that allow for direct, analytical integration of the elementary matrices. They also allow for extremely-distorted concave polyhedra, though we will not analyze



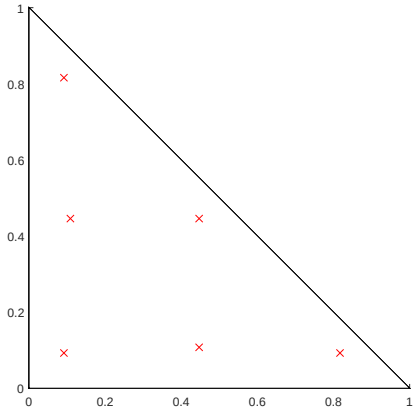
(a) Order 1



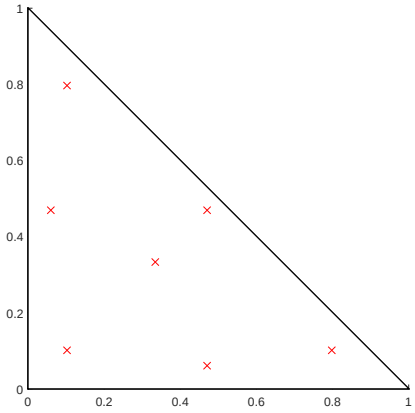
(b) Order 2



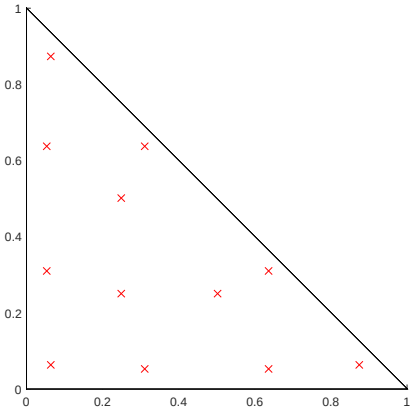
(c) Order 3



(d) Order 4

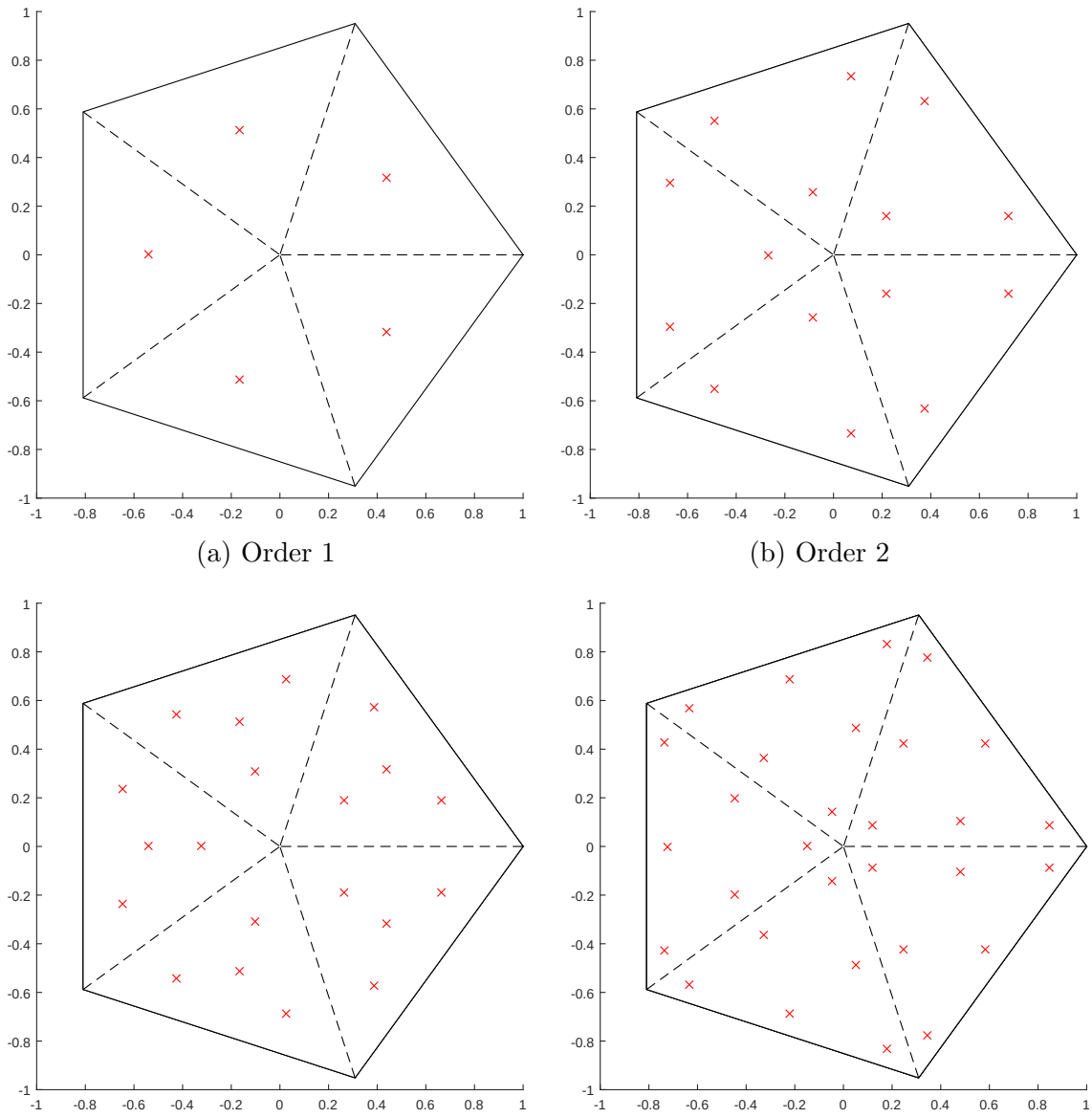


(e) Order 5



(f) Order 6

Figure 3.19: Quadrature sets on the reference triangle of varying order.



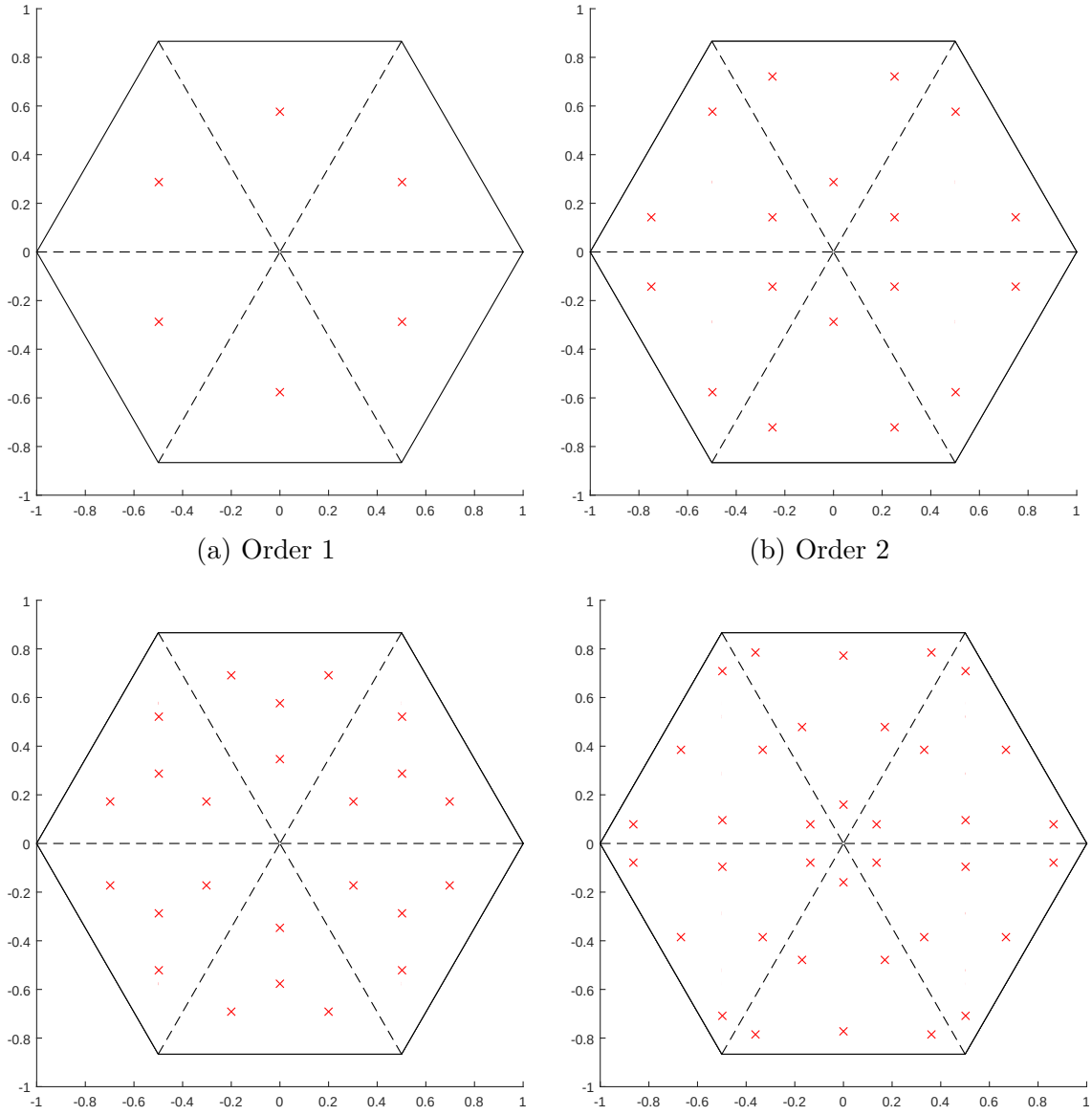
(a) Order 1

(b) Order 2

(c) Order 3

(d) Order 4

Figure 3.20: blah



(a) Order 1

(b) Order 2

(c) Order 3

(d) Order 4

Figure 3.21: blah

those in this work.

$$b_j(x,y,z) = t_j(x,y,z) + \sum_{f=1}^{F_j} \beta_f^j t_f(x,y,z) + \alpha_j^K t_c(x,y,z) \tag{3.40}$$

t_j is the standard 3D linear function with unity at vertex j that linearly decreases to zero to the cell center, the face center for each face that includes vertex j, and each vertex that shares an edge with vertex j. t_c is the 3D cell "tent" function which is unity at the cell center and linearly decreases to zero to each cell vertex and face center. t_f is the face "tent" function which is unity at the face center and linearly decreases to zero at each vertex on that face and the cell center. $\beta_{f,j}$ is the weight parameter for face f touching cell vertex j, and F_j is the number of faces touching vertex j. Like the previous work defining the PWLD method [70], we also choose to assume the cell and face weighting parameters are

$$\alpha_{K,j} = \frac{1}{N_K} \qquad \text{and} \qquad \beta_{f,j} = \frac{1}{N_f}, \tag{3.41}$$

respectively, where N_K is the number of vertices in cell K and N_f is the number of vertices on face f, which leads to constant values of α and β for each cell and face, respectively. This assumption of the cell weight function holds for both 2D and 3D.

$$\begin{aligned} b_1(r,s,t) &= 1 - r - s - t \\ b_2(r,s,t) &= r \\ b_3(r,s,t) &= s \\ b_4(r,s,t) &= t \end{aligned} \tag{3.42}$$

3.5 Numerical Results

Now that we have presented several linear polygonal finite element basis sets along with the methodology to convert them to quadratic serendipity-like basis, we present several numerical problems to demonstrate our methodology. First, we demonstrate that the presented basis sets can capture an exactly-linear transport solution in Section 3.5.1. Next, we present some convergence properties of the basis sets using the method of manufactured solutions (MMS) in Section 3.5.3. We then present a searchlight problem and observe how the basis sets react with adaptive mesh refinement (AMR) to mitigate numerical dispersion through a vacuum in Section 3.5.5.

3.5.1 Two-Dimensional Exactly-Linear Transport Solutions

We present our first numerical example by demonstrating that the linear and quadratic polygonal finite element basis functions capture an exactly-linear solution space. We will show this by the method of exact solutions (MES). Since the coordinate interpolation of the basis functions for the linear basis functions requires exact linear interpolation (Eq. (3.2)), then an exactly-linear solution space can be captured, even on highly distorted polygonal meshes. This also applies to the quadratic serendipity space since it is formed by the product-wise pairings of the linear basis functions. We build our exact solution by investigating the 2D, 1 energy group transport problem with no scattering and an angle-dependent distributed source,

$$\mu\frac{\partial\Psi}{\partial x}+\eta\frac{\partial\Psi}{\partial y}+\sigma_t\Psi=Q(x,y,\mu,\eta), \tag{3.43}$$

where the streaming term was separated into the corresponding two-dimensional terms. We chose to drop the scattering term for this example so that the error

arising from iteratively converging our solution would have no impact.

We then define an angular flux solution that is linear in both space and angle along with the corresponding 0th moment scalar flux ($\Phi_{0,0} \rightarrow \Phi$) solution:

$$\begin{aligned} \Psi(x,y,\mu,\eta) &= ax + by + c\mu + d\eta + e \\ \Phi(x,y) &= 2\pi\left(ax + by + e\right) \end{aligned} . \tag{3.44}$$

One can immediately notice that our 0th moment solution is not dependent on angle. We arrive at this solution by enforcing our 2D angular quadrature set to have the following properties:

$$\sum_q w_q = 2\pi \qquad \text{and} \qquad \sum_q w_q \begin{bmatrix} \mu_q \\ \eta_q \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{3.45}$$

The sum of the quadrature weights is handled by simply renormalizing those that are generated in Section 2.4 to 2π.

Our boundary conditions for all inflow boundaries are then uniquely determined by the angular flux solution of Eq. (3.44). Inserting the angular flux solution of Eq. (3.44) into Eq. (3.43), we obtain the distributed source that will produce our exactly-linear solution space:

$$Q(x,y,\mu,\eta) = a\mu + b\eta + \sigma_t\left(c\mu + d\eta\right) + \sigma_t\left(ax + by + e\right). \tag{3.46}$$

It is noted that the angular dependence of the source can be removed (which can ease the code development burden) if one sets

$$\begin{aligned} a &= -c\,\sigma_t, \\ b &= -d\,\sigma_t. \end{aligned} \tag{3.47}$$

For this example, we test the various 2D polygonal finite element basis functions on six different mesh types. These mesh types include triangular, quadrilateral, and polygonal meshes:

1. Orthogonal cartesian mesh formed by the intersection of 11 equally-spaced vertices in both the x and y dimensions. This forms a 10x10 array of quadrilateral mesh cells.

2. Ordered-triangular mesh formed by the bisection of the previous orthogonal cartesian mesh (forming 200 triangles all of the same size/shape).

3. Quadrilateral shestakov grid formed by the randomization of vertices based on a skewness parameter [71, 72]. With a certain range of this skewness parameter, highly distorted meshes can be generated.

4. Sinusoidal polygonal grid that is generated by the transformation of a uniform orthogonal grid based on a sinusoid functional. The transformed vertices are then converted into a polygonal grid by computing a bounded Voronoi diagram.

5. Kershaw's quadrilateral z-mesh [73]. This mesh is formed by taking an orthogonal quadrilateral grid and displacing certain interior vertices only in the y dimension.

6. A polygonal variant of the quadrilateral z-mesh. The polygonal grid is formed in a similar manner to the sinusoidal polygonal mesh with a Voronoi diagram.

We also wish that both the angular flux solution as well as the 0th moment solution are strictly positive everywhere. Therefore, we set the function parameters in Eq. (3.44) to $\sigma_t = a = c = d = e = 1.0$ and $b = 1.5$. We gave the solution the 40% tilt in space ($a \neq b$) so that it would not align with the triangular mesh. Using an

S8 LS quadrature set, we ran all combinations of the polygonal basis functions and the mesh types. The linear solutions for the Wachspress, PWL, mean value, linear maximum entropy, and quadratic serendipity maximum entropy basis functions are presented in Figures 3.22, **??**, **??**, **??**, and **??**, respectively. We can see that for all the polygonal basis functions, an exact linear solution is captured as shown by the unbroken nature of the contour lines. This even holds on the highly distorted quadrilateral shestakov mesh.

3.5.2 Two-Dimensional Exactly-Quadratic Transport Solutions

$$\begin{aligned} \Psi(x,y,\mu,\eta) &= a + bx + cy + dxy + ex^2 + fy^2 \\ \Phi(x,y) &= 2\pi\left(a + bx + cy + dxy + ex^2 + fy^2\right) \end{aligned} . \tag{3.48}$$

3.5.3 Convergence Rate Analysis by the Method of Manufactured Solutions

The next numerical example we investigate involves calculating the convergence rate of the solution error via the method of manufactured solutions (MMS). Like MES, MMS enforces a given solution by use of a derived functional form for the driving source of the problem (Q_{ext}). However, unlike MES, we enforce a spatial solution that cannot be captured by the interpolation of the finite element space.

For this example, we choose the following solution and problem parameters and characteristics:

1. Constant total cross section so that parameterized material properties are not necessary;
2. No scattering to avoid solution discontinuities from the S_N discretization;
3. No solution dependence in angle to avoid introducing angular discretization error;

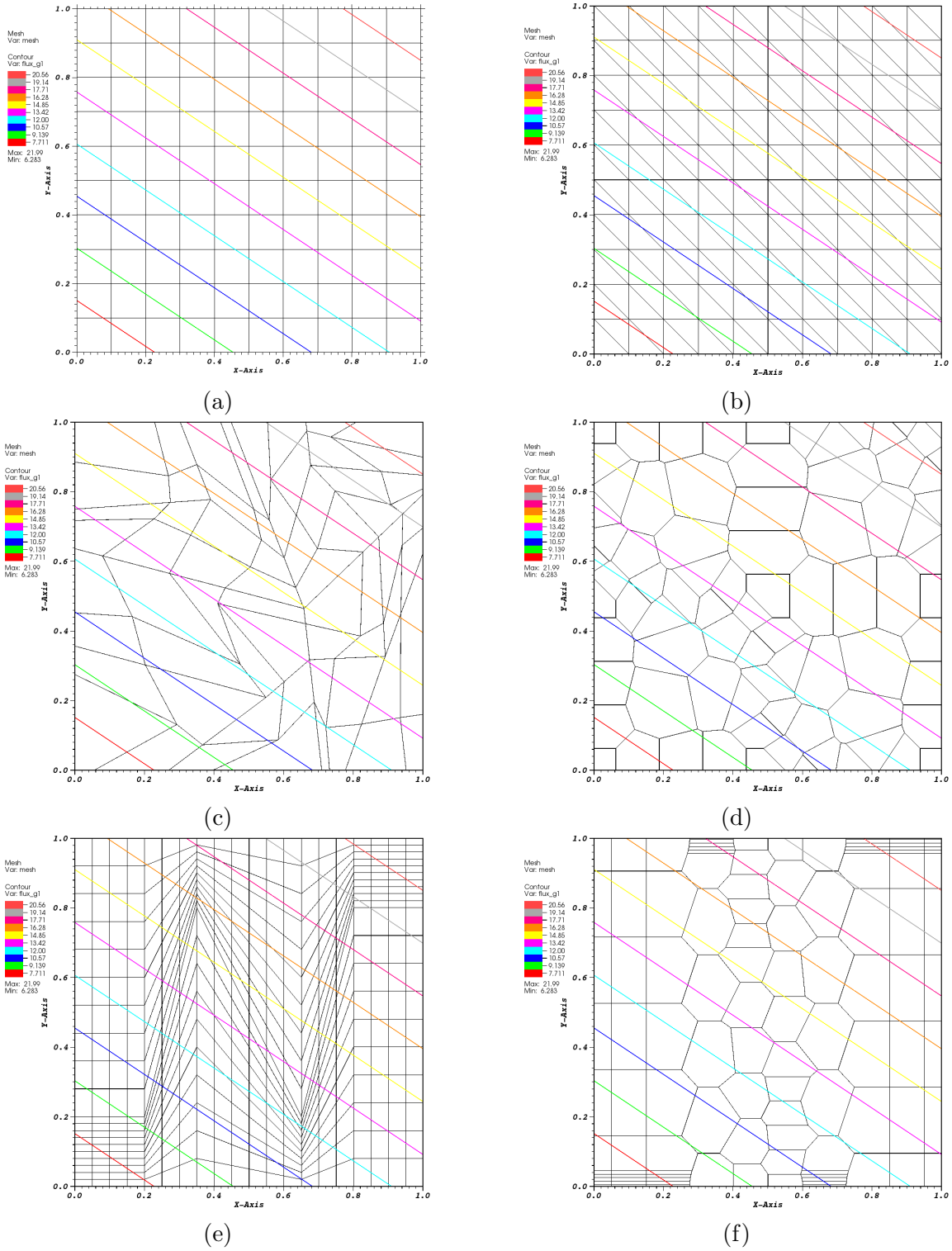
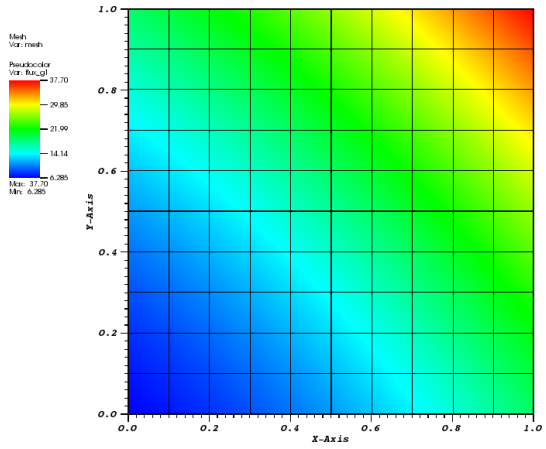
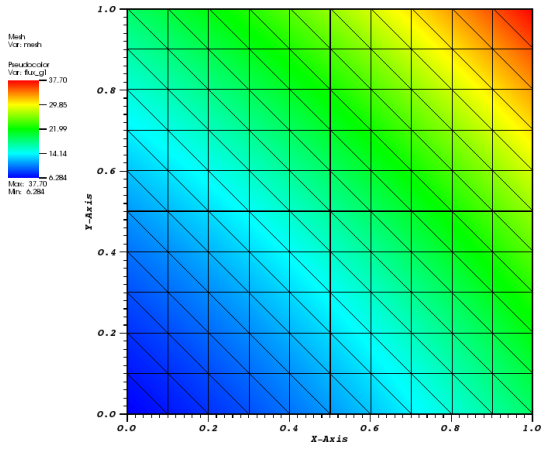


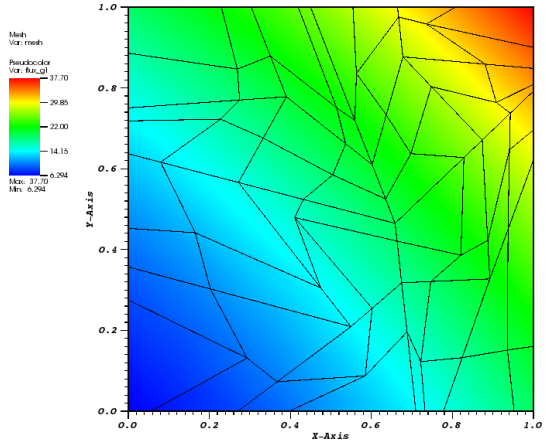
Figure 3.22: Contour plots of the exactly-linear solution with the Wachspress basis functions on (a) cartesian mesh, (b) ordered-triangular mesh, (c) quadrilateral shestakov mesh, (d) sinusoidal polygonal mesh, (e) quadrilateral z-mesh, and (f) polygonal z-mesh.



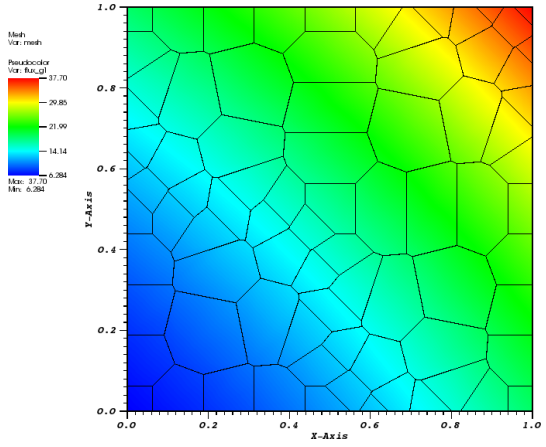
(a) Cartesian



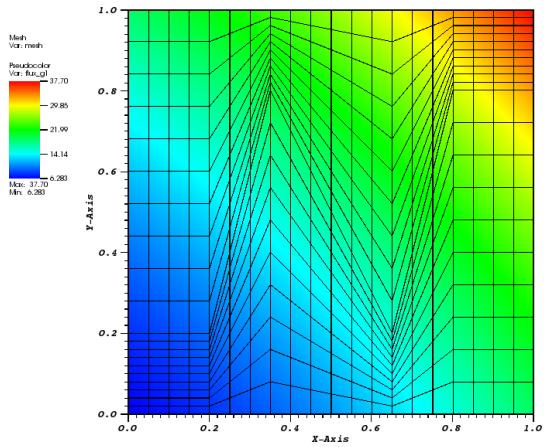
(b) Triangular



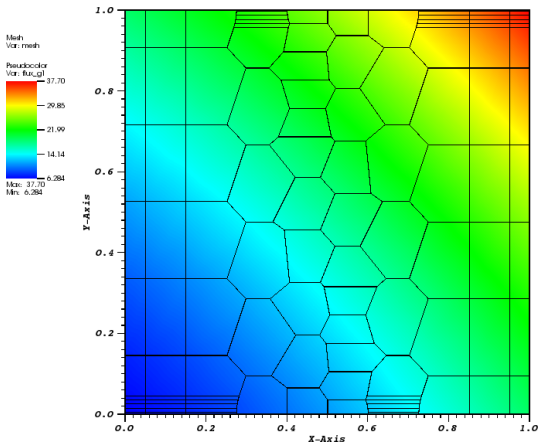
(c) Shestakov Quadrilaterals



(d) Sinusoid Polygons

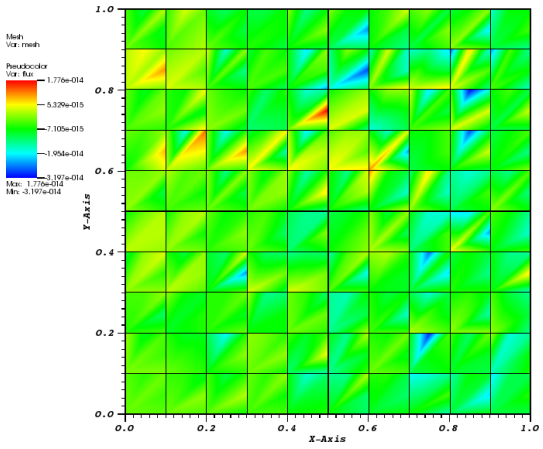


(e) Z-Quadrilaterals

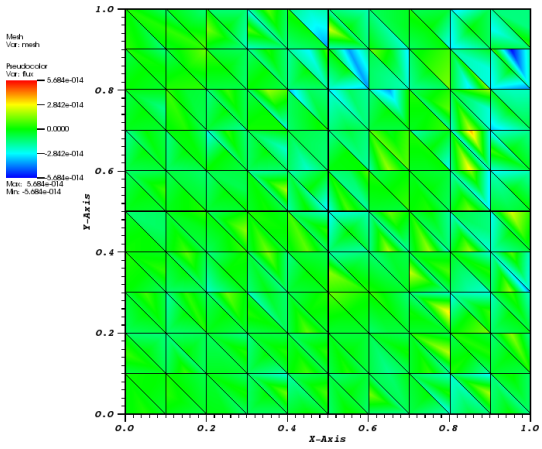


(f) Z-Polygons

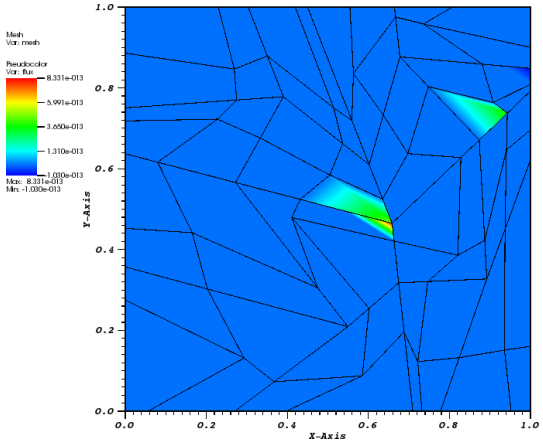
Figure 3.23: Plots of the exactly-quadratic solution with the quadratic serendipity maximum entropy basis functions.



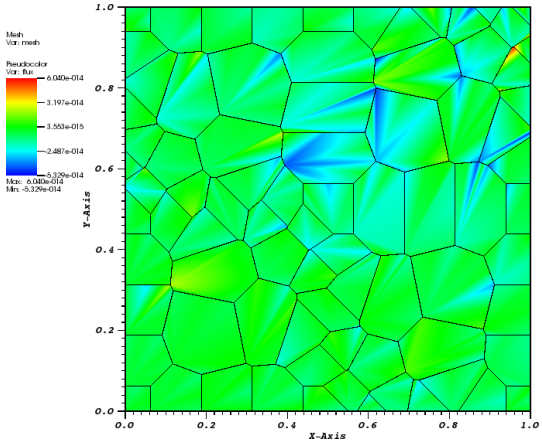
(a) Cartesian



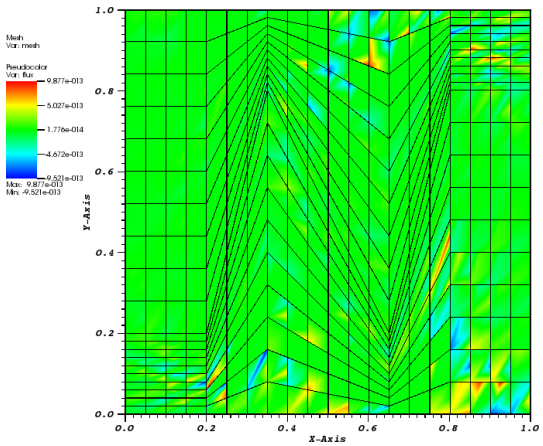
(b) Triangular



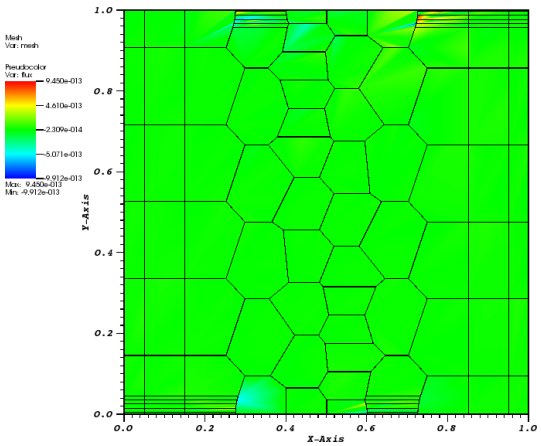
(c) Shestakov Quadrilaterals



(d) Sinusoid Polygons

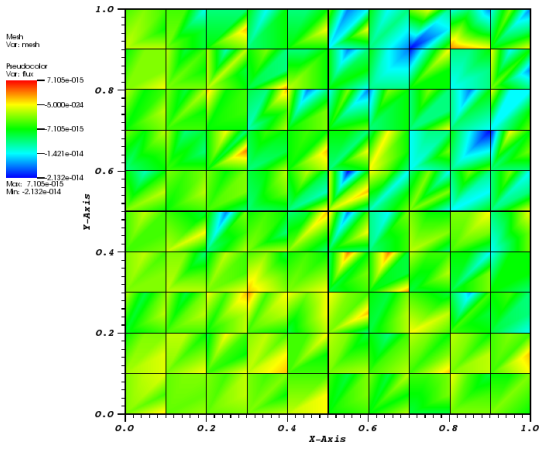


(e) Z-Quadrilaterals

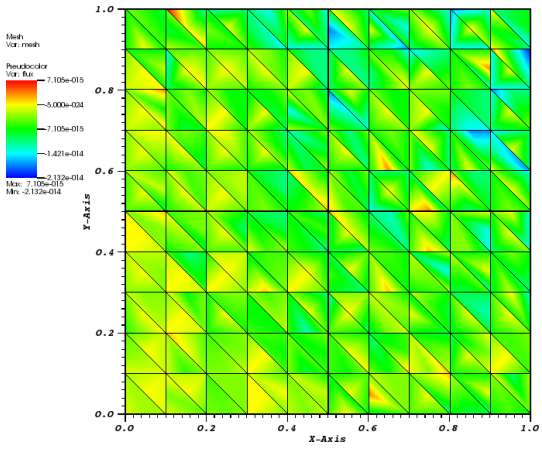


(f) Z-Polygons

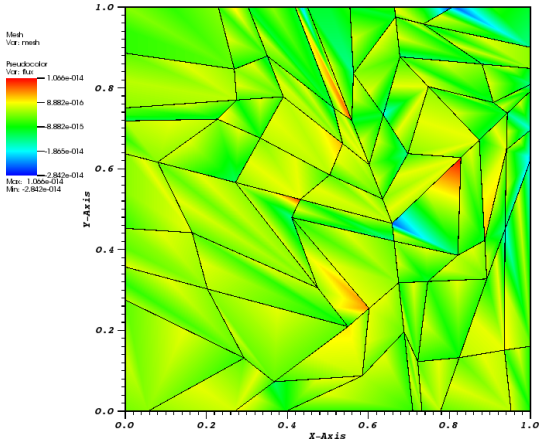
Figure 3.24: Plots of the error of the exactly-quadratic solution with the quadratic serendipity Wachspress basis functions.



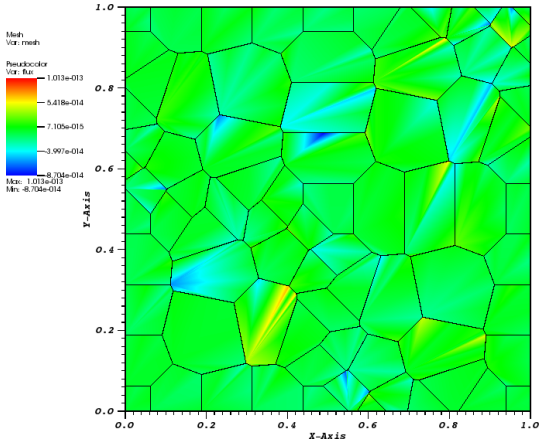
(a) Cartesian



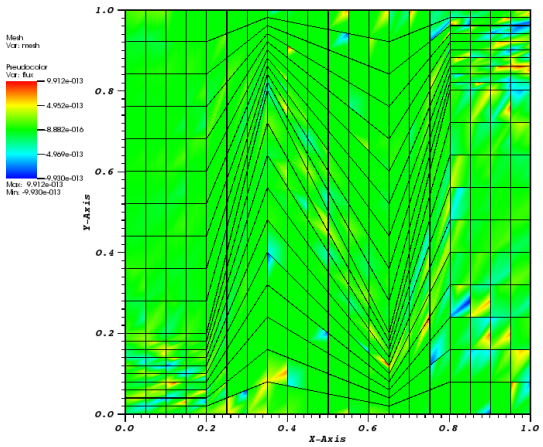
(b) Triangular



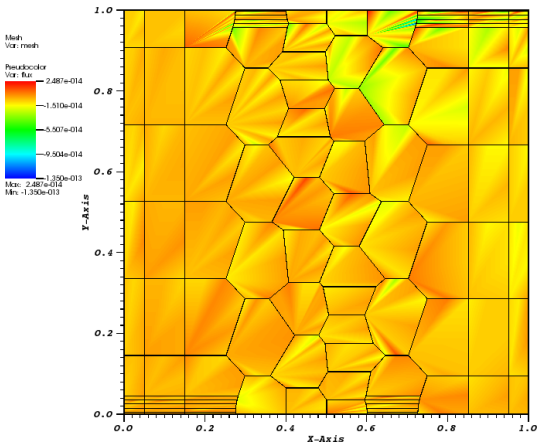
(c) Shestakov Quadrilaterals



(d) Sinusoid Polygons

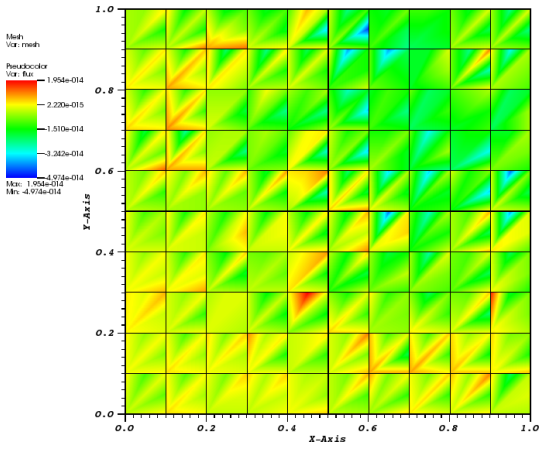


(e) Z-Quadrilaterals

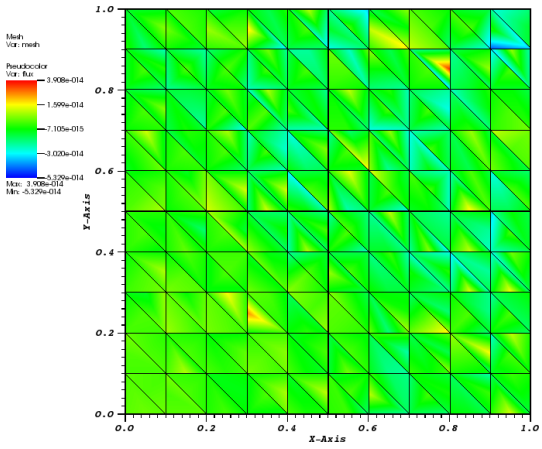


(f) Z-Polygons

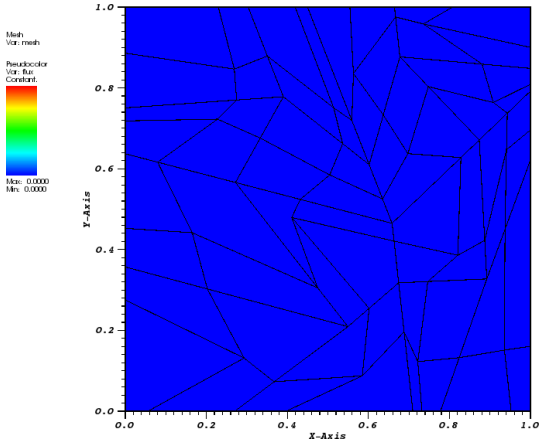
Figure 3.25: Plots of the error of the exactly-quadratic solution with the quadratic serendipity PWL basis functions.



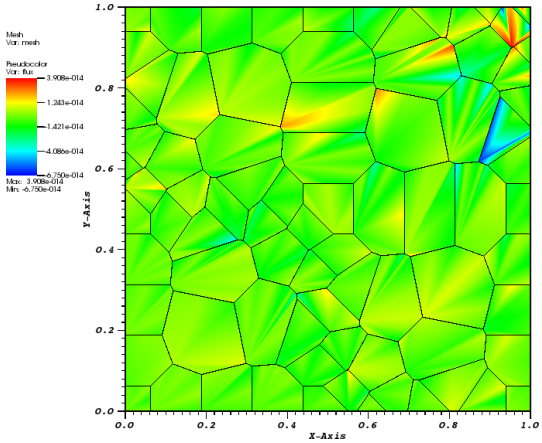
(a) Cartesian



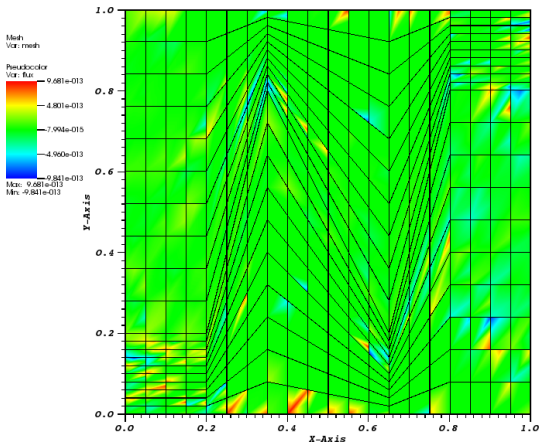
(b) Triangular



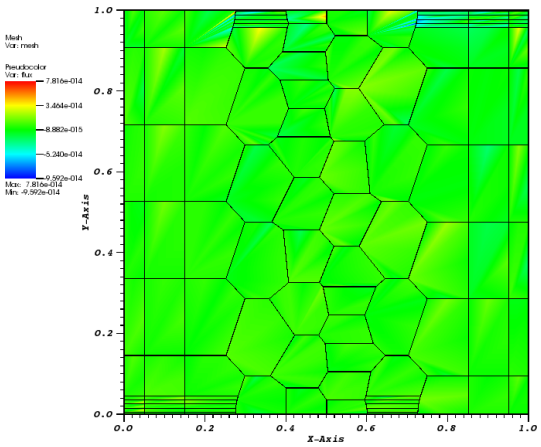
(c) Shestakov Quadrilaterals



(d) Sinusoid Polygons

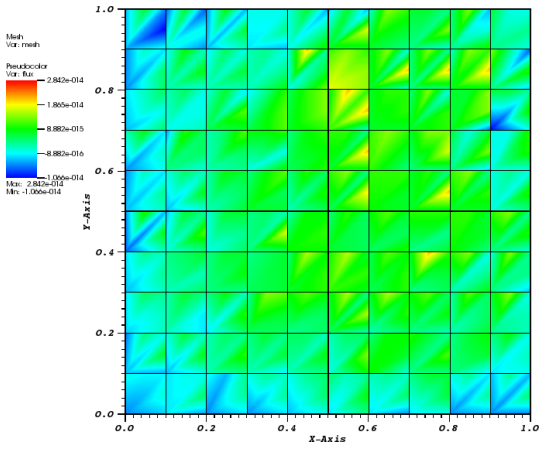


(e) Z-Quadrilaterals

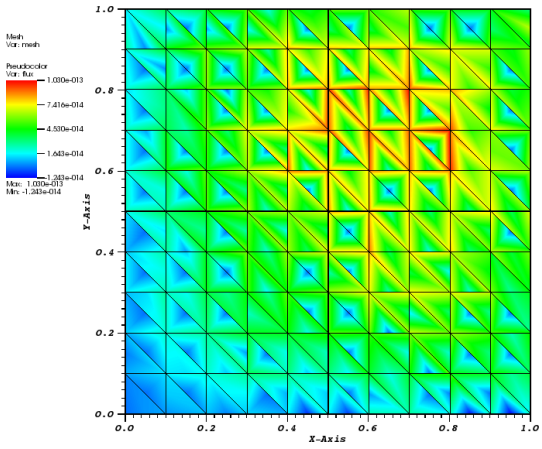


(f) Z-Polygons

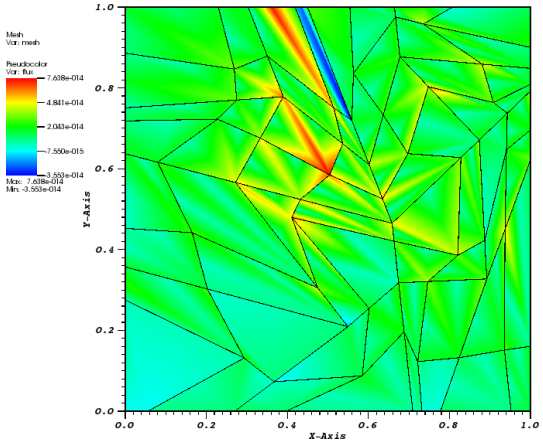
Figure 3.26: Plots of the error of the exactly-quadratic solution with the quadratic serendipity mean value basis functions.



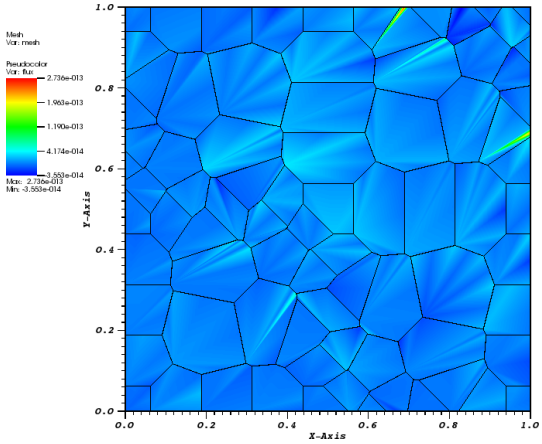
(a) Cartesian



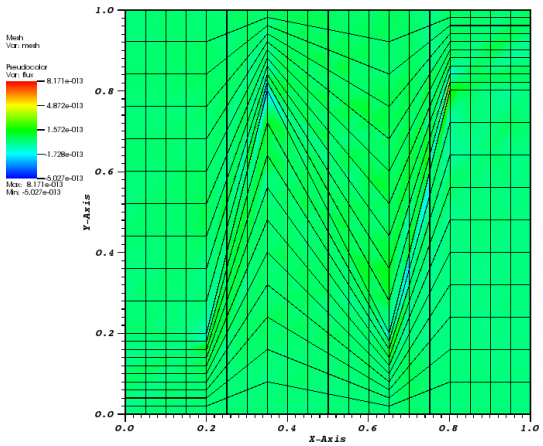
(b) Triangular



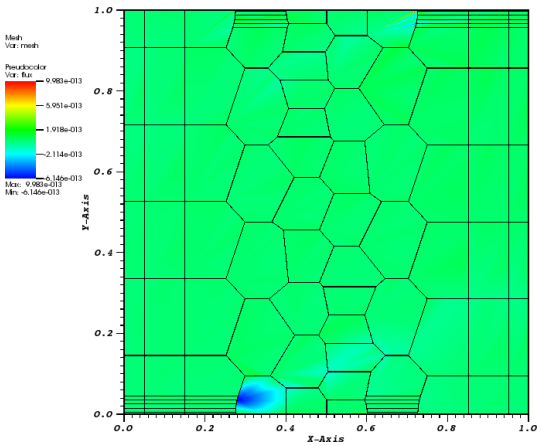
(c) Shestakov Quadrilaterals



(d) Sinusoid Polygons



(e) Z-Quadrilaterals



(f) Z-Polygons

Figure 3.27: Plots of the error of the exactly-quadratic solution with the quadratic serendipity maximum entropy basis functions.

4. Analytical solutions that are C^∞ continuous in space for both the angular flux and 0th order flux moment;;

5. The angular flux solution is zero on the boundary for all incident directions - this is identical to vacuum boundaries which can ease code development.

To satisfy these characteristics, we choose to analyze two different solution spaces. The first is a smoothly varying sinusoid solution with no extreme local maxima. The second solution is a product of a quadratic function and a gaussian which yields a significant local maximum.

The sinusoid flux solutions, $\{\Psi^s, \Phi^s\}$, have the following parameterized form,

$$\begin{aligned} \Psi^s(x,y) =& \sin(\nu\frac{\pi x}{L_x})\sin(\nu\frac{\pi y}{L_y}), \\ \Phi^s(x,y) =& 2\pi\sin(\nu\frac{\pi x}{L_x})\sin(\nu\frac{\pi y}{L_y}), \end{aligned} \tag{3.49}$$

where ν is a frequency parameter. We restrict this paremeter to positive integers ($\nu = 1, 2, 3, ...$) to maintain characteristic 5 of the solution and problem space. The gaussian solution space, $\{\Psi^g, \Phi^g\}$, that has its local maximum centered at (x_0, y_0) has the parameterized form,

$$\begin{aligned} \Psi^g(x,y) =& C_M x(L_x - x)y(L_y - y)\exp(-\frac{(x-x_0)^2 + (y-y_0)^2}{\gamma}), \\ \Phi^g(x,y) =& 2\pi C_M x(L_x - x)y(L_y - y)\exp(-\frac{(x-x_0)^2 + (y-y_0)^2}{\gamma}), \end{aligned} \tag{3.50}$$

where the constants in the equations are:

$$C_M = \frac{100}{L_x^2 L_y^2} \qquad \gamma = \frac{L_x L_y}{100}. \tag{3.51}$$

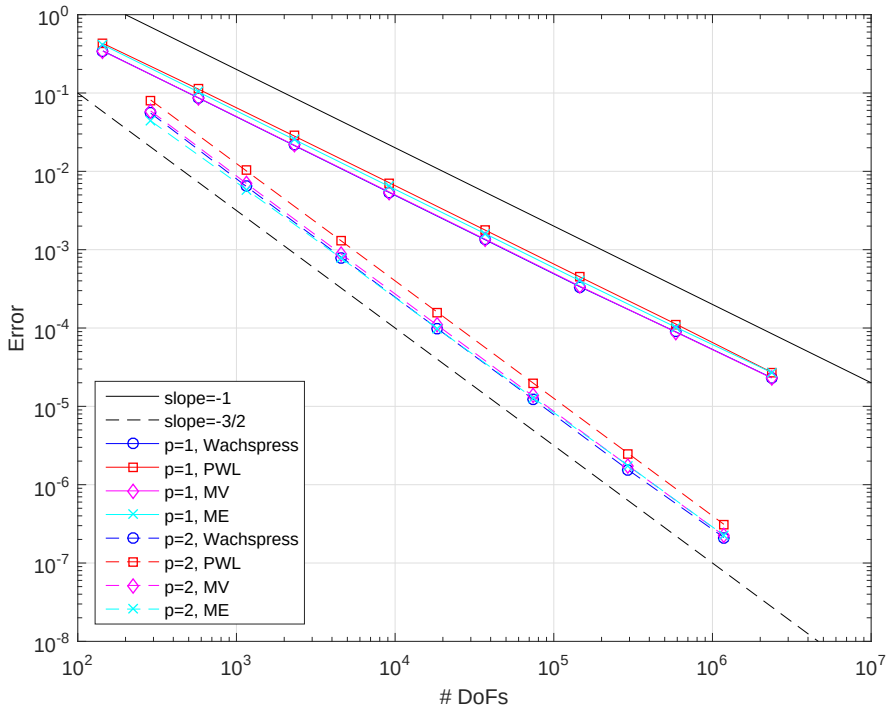


Figure 3.28: Convergence rates for the sinusoid MMS problem on a Cartesian mesh.

For this example, we choose the dimensionality of our problem to be $[0, 1]^2$ which makes $L_x = L_y = 1$ for both the sinusoid and gaussian solutions. For the sinusoid solution, we select the frequency parameter, ν, to be 3 and for the gaussian solution we set the local maximum: $x_0 = y_0 = 0.75$. With these parameters, the sinusoid solution will have local minima and maxima of -2π and 2π, respectively, and the gaussian solution will have a global maximum of $\frac{225}{32}\pi \approx 22.1$.

3.5.4 Convergence Rate Analysis in a Purely-Absorbing Medium

3.5.5 Searchlight Problem

The next example models a beam or searchlight. Similar problems were investigated in Dedner and Vollmöller [45] and Wang and Ragusa [48]. In this problem, an incident beam of neutrons is shined onto a small portion of a boundary, propagates

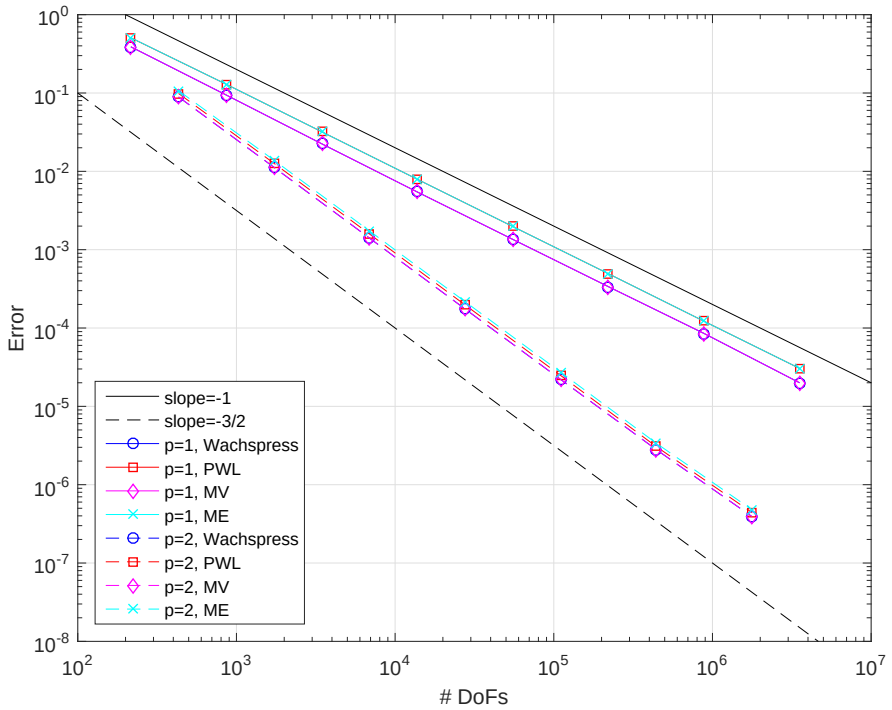


Figure 3.29: Convergence rates for the sinusoid MMS problem on an ordered triangular mesh.

through a vacuum, and then exits through a small portion of a different boundary. As the beam propagates through the vacuum, the spatial discretization causes radiation outflow through all downwind cell faces. This leads to numerical dispersion and will cause to beam to artificially broaden.

In this problem, we investigate an $\mathbb{R}^2$ domain of size $[0,1]^2$ cm. The radiation enters the left boundary between $0.2 \leq y \leq 0.4$ with an un-normalized angular direction of $[1, 0.4]$. For this chosen direction, the radiation beam would analytically leave the right boundary between $0.6 \leq y \leq 0.8$. This means that any radiation leaving the right boundary for all other y values is due to the numerical dispersion of the beam.

We investigated this problem using several of the 2D polygonal basis functions

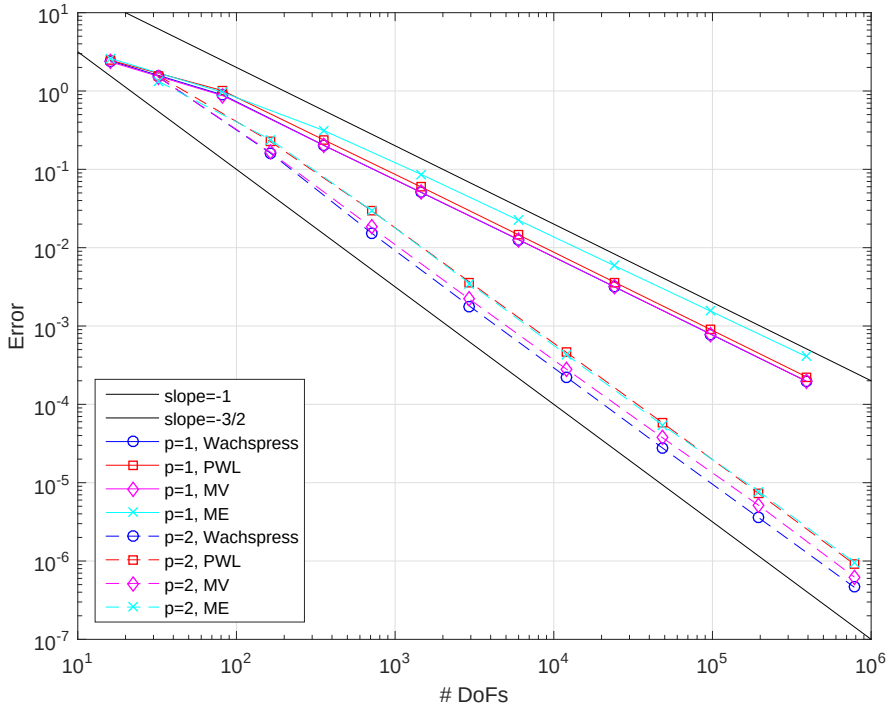


Figure 3.30: Convergence rates for the sinusoid MMS problem on a regular polygonal mesh.

as outlined in Sections 3.1 and 3.2 as well as

3.6 Conclusions

In this chapter, we have presented

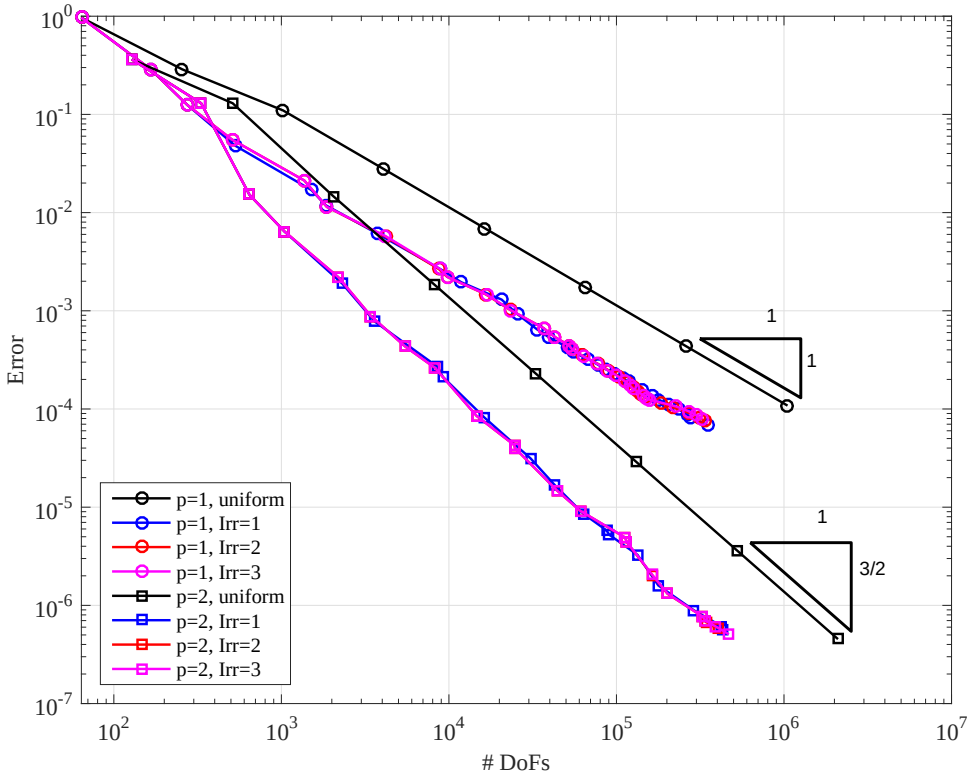


Figure 3.31: Convergence rates for the 2D Gaussian MMS problem using the PWL basis functions.

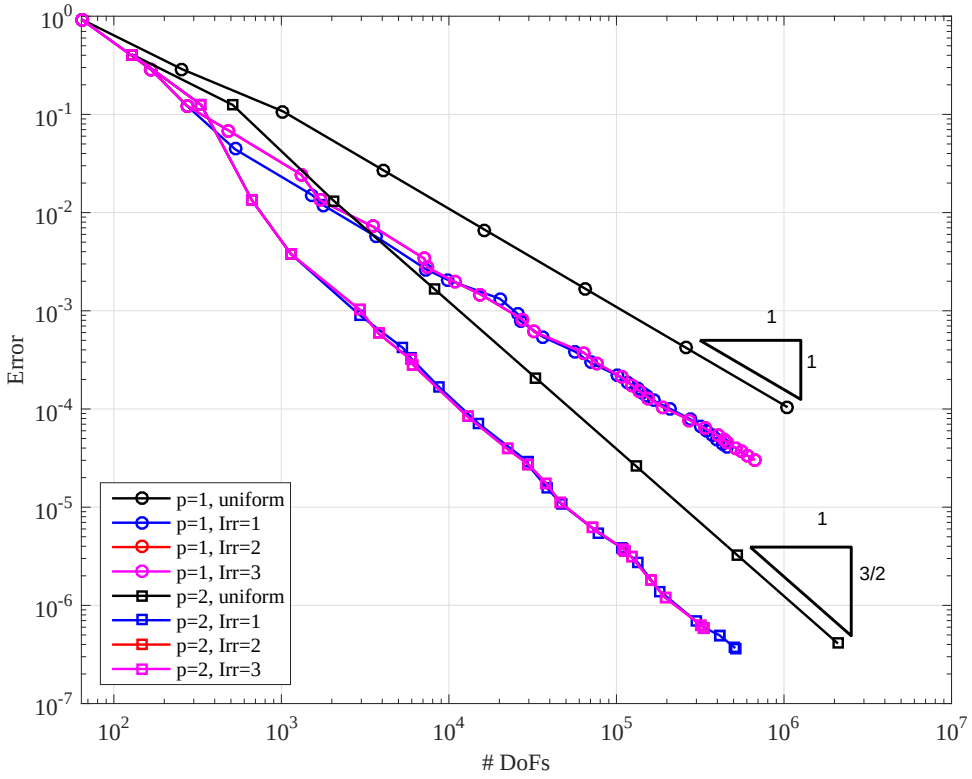


Figure 3.32: Convergence rates for the 2D Gaussian MMS problem using the mean value basis functions.

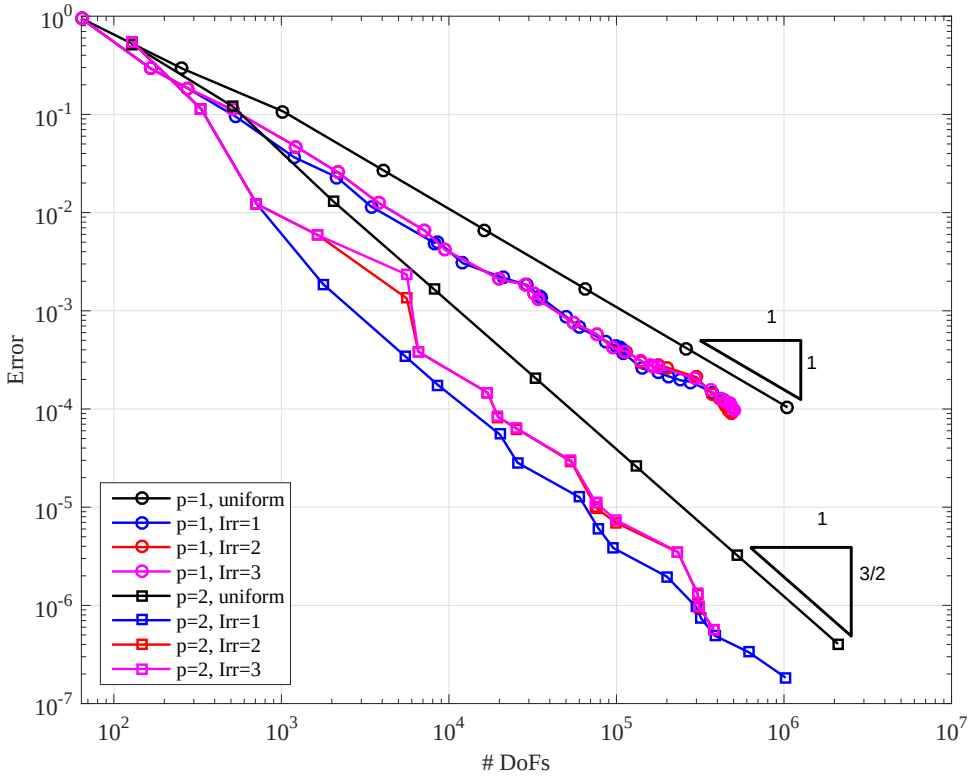


Figure 3.33: Convergence rates for the 2D Gaussian MMS problem using the maximum entropy basis functions.

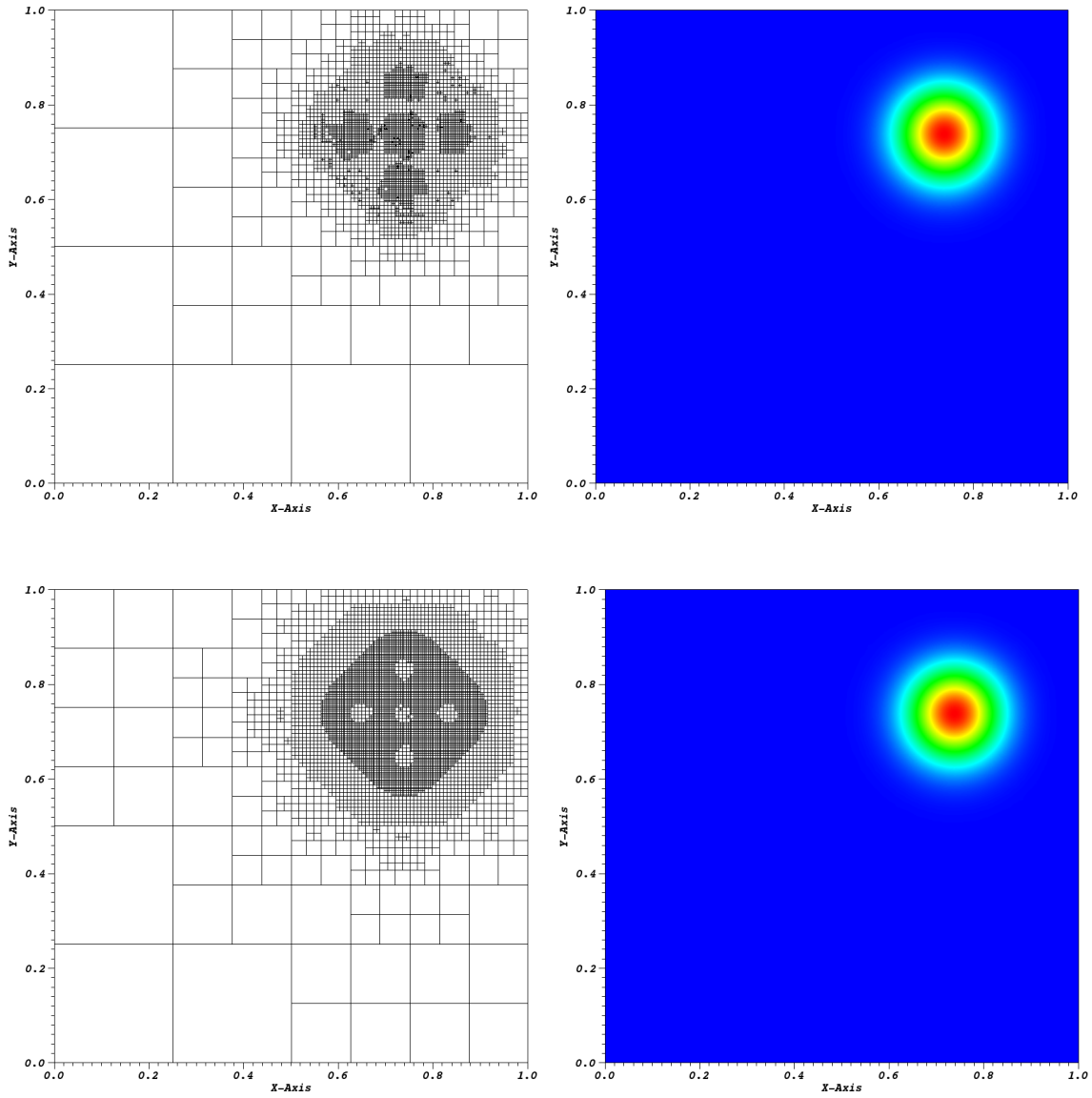


Figure 3.34: AMR meshes and solutions for the gaussian MMS problem using the maximum entropy coordinates: (top) linear basis functions at cycle 15 and (bottom) quadratic serendipity basis functions at cycle 08.

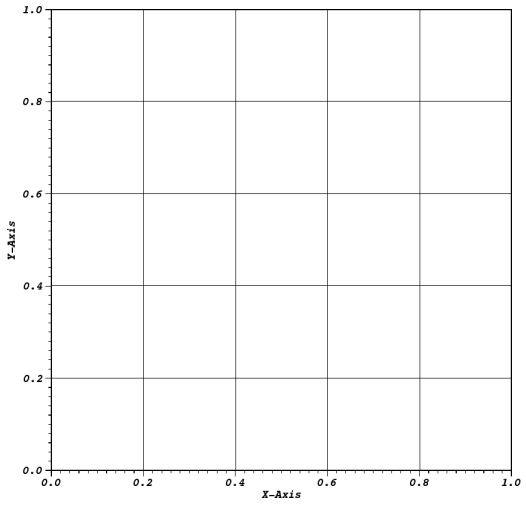


Figure 3.35: Initial mesh configuration for the searchlight problem before any refinement cycles.

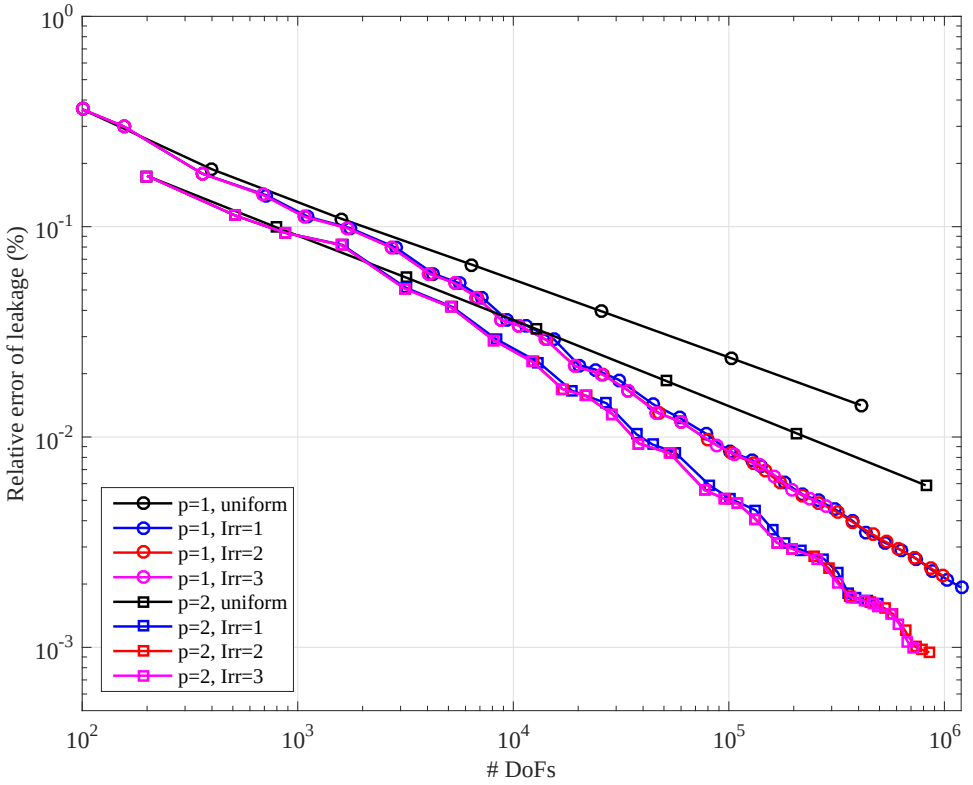


Figure 3.36: blah.

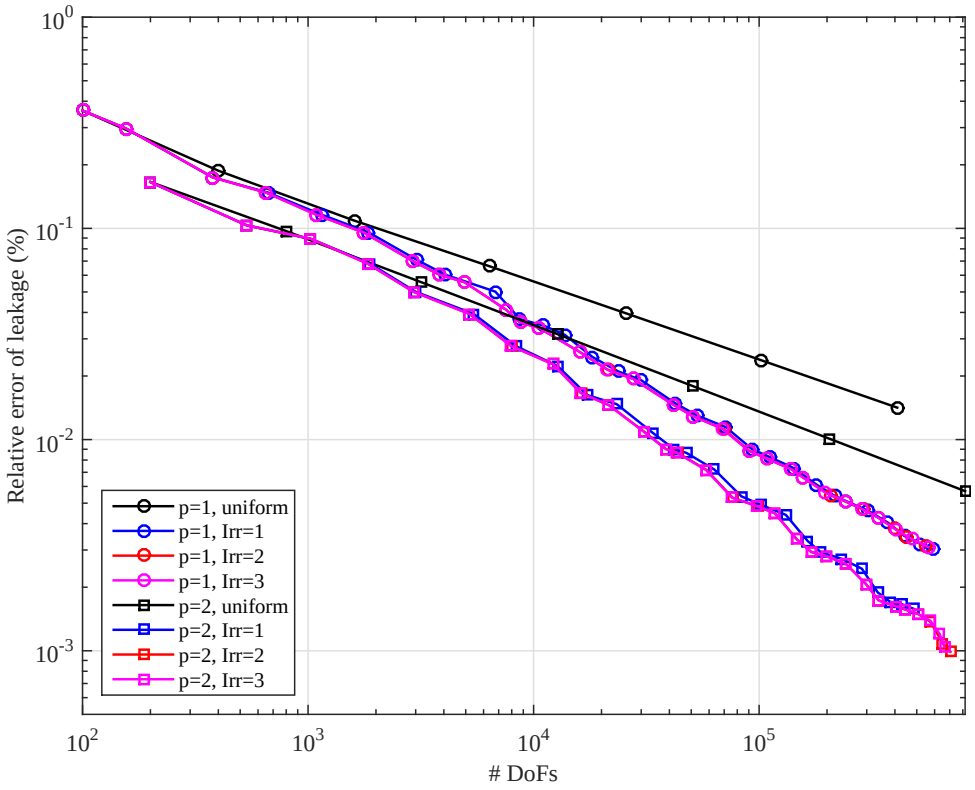


Figure 3.37: blah.

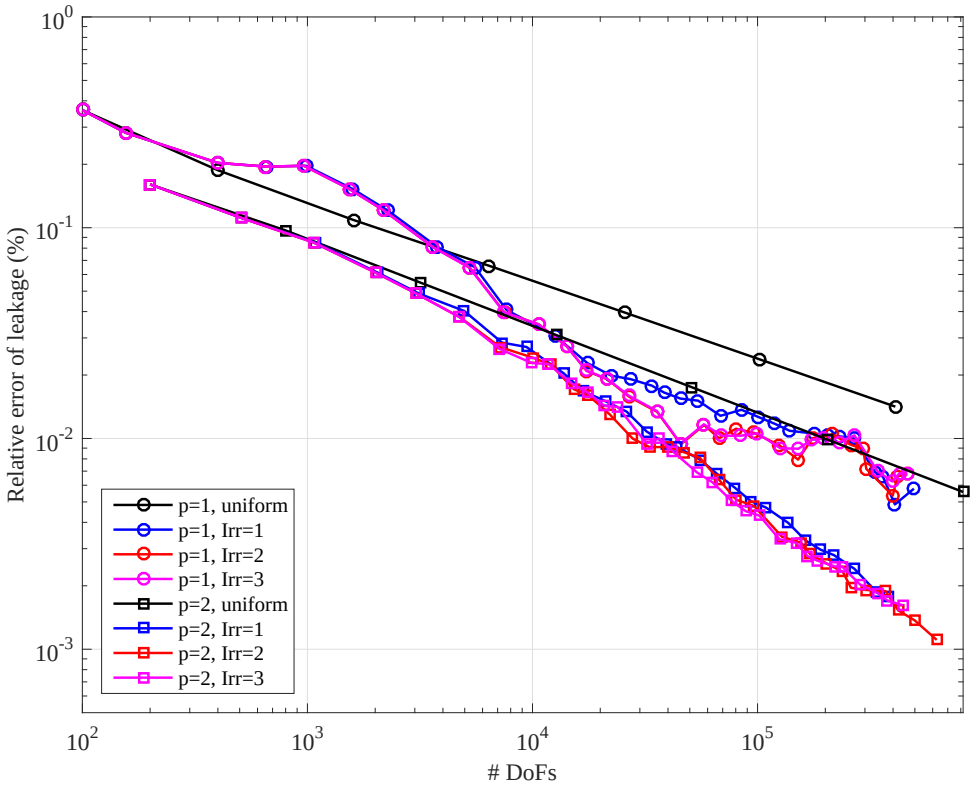
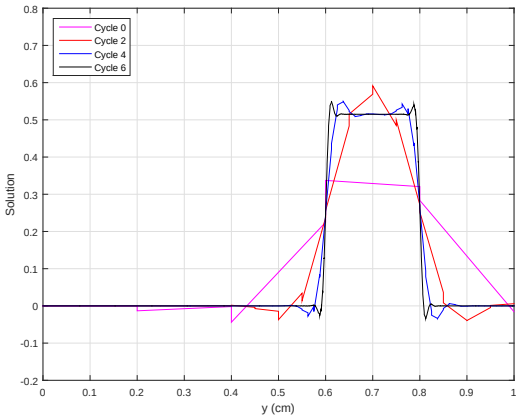
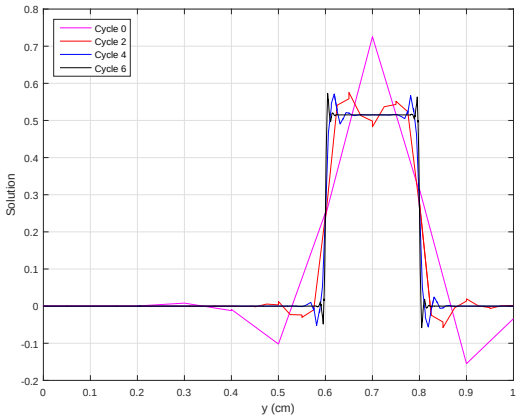


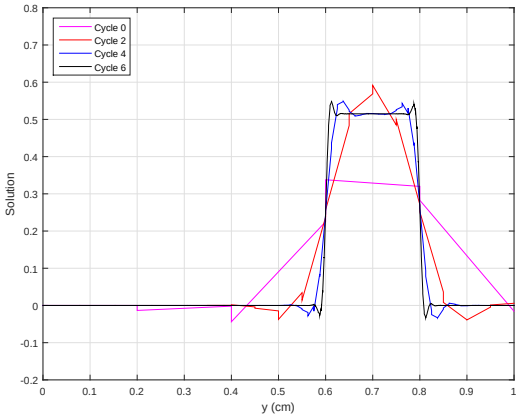
Figure 3.38: blah.



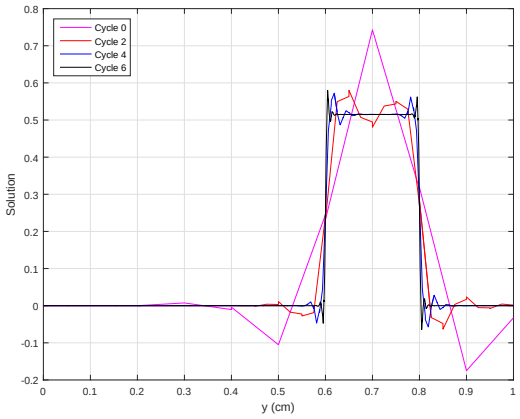
(a) Linear PWL



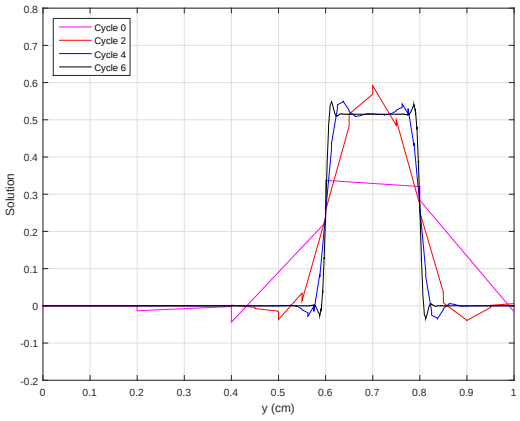
(b) Quadratic PWL



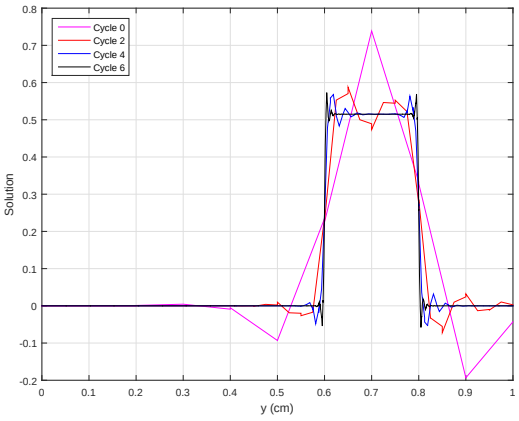
(c) Linear Mean Value



(d) Quadratic Mean Value



(e) Linear Maximum Entropy



(f) Quadratic Maximum Entropy

Figure 3.39: Exiting angular flux on the right boundary with uniform refinement.

4. DIFFUSION SYNTHETIC ACCELERATION FOR DISCONTINUOUS FINITE ELEMENTS ON UNSTRUCTURED GRIDS

4.1 Introduction

In this chapter, we analyze the Modified Interior Penalty (MIP) form of the diffusion equation as a discretization scheme for use with Diffusion Synthetic Acceleration (DSA) of the DFEM S_N transport equation on unstructured grids. Specifically, we wish to analyze its efficacy on massively-parallel computer architectures where scalability of solution times and memory footprints can become burdensome. This chapter is laid out in the following manner. The remainder of this Section will provide an overview of synthetic acceleration techniques as well as a review of DSA schemes. Section 4.2 provides the DSA methodologies that we will employ for 1-group and thermal neutron upscattering acceleration. We then present the discontinuous Symmetric Interior Penalty (SIP) form of the diffuion equation in Section 4.3 as well as the MIP variant that we will use for our DSA analysis in Section 4.4. The numerical procedures that we will use to solve the diffusion system of equations is given in Section 4.5. The theoretical Fourier analysis tool is given in Section 4.6. Results are provided in Section 4.7 and we finish the chapter with some concluding remarks in Section 4.8

4.1.1 Review of Diffusion Synthetic Accleration Schemes

In Section 2.7.2, we described the full-domain transport sweep, and how it can efficiently invert the loss operator. We also provided the parallel implementation details that allow these full-domain transport sweeps to scale out to $O(10^6)$ processors. However, the ability to efficiently invert the transport (streaming and collision) operator does not necessarily mean that transport solutions can be easily obtained.

In general, radiation transport solutions are obtained iteratively. The simplest and widely-used method is a fixed-point scheme (*i.e.* Richardson iteration) ubiquitously called Source Iteration (SI) in the transport community. Unfortunately, the iteration process of SI can converge arbitrarily slowly if the problem is optically thick [29]. This corresponds to long mean free paths for neutronics problems. This also corresponds to time steps and material heat capacities tending to infinity and zero, respectively, for thermal radiative transport (TRT) problems.

For these problem regimes in which solution is prohibitively slow, additional steps should be taken to speed up, or accelerate, solution convergence [29]. The most used methods to assist in solution convergence are often called synthetic acceleration techniques. These techniques were first introduced by Kopp [74] and Lebedev [75, 76, 77, 78, 79, 80, 81] in the 1960's. From Kopp's and Lebedev's work, Gelbard and Hageman then introduced two synthetic acceleration options for the low-order operator: diffusion and S_2 [82]. Their diffusion preconditioning led to efficient convergence properties on fine spatial meshes. Reed then showed that Gelbard and Hageman's diffusion preconditioning would yield a diverging system for coarse meshes [83]. At this point in time, no one knew if an unconditionally efficient acceleration method could be derived.

Then in 1976, Alcouffe proposed a remedy to Gelbard and Reed that he called diffusion synthetic acceleration (DSA) [84, 85, 86]. He showed that if you derived the diffusion operator consistently with the discretized transport operator, then SI could be accelerated with DSA in an efficient and robust manner. Larsen and McCoy then demonstrated that unconditional stability required that consistency be maintained in both spatial and angular discretization in their four-step procedure [87, 88]. However, Adams and Martin then showed that partially-consistent diffusion discretizations could effectively accelerate DFEM discretizations of the neutron transport equation

[89]. Their modified-four-step procedure (M4S), based on Larsen and McCoy's work, was shown to be unconditionally stable for regular geometries, but divergent for unstructured multi-dimensional meshes [90]. In more recent years, alternate discretizations for the diffusion operator have been applied to unstructured multi-dimensional grids. These include the partially consistent Wareing-Larsen-Adams (WLA) DSA [91], the fully consistent DSA (FCDSA) [90], and the partially consistent MIP DSA [92, 93, 94].

Most recently, the partially consistent MIP DSA method has been shown to be an unconditionally stable acceleration method for the 2D DFEM transport equation on unstructured meshes. Wang showed that it acted as an effective preconditioner for higher-order DFEM discretizations on triangles [92, 93]. Turcksin and Ragusa then extended the work to arbitrary polygonal meshes [94]. The MIP diffusion operator is symmetric positive definite (SPD) and was shown to be efficiently invertible with preconditioned conjugate gradient (PCG) and advanced preconditioners such as algebraic multi-grid (AMG) [94].

4.1.2 Synthetic Acceleration Overview

Synthetic acceleration techniques have been widely used in the nuclear engineering community to improve solution convergence for prohibitively slow problems. We now provide a general framework for how synthetic acceleration methods are derived. We begin by expressing our neutron transport equation in the following form,

$$(\mathbf{A} - \mathbf{B})\,\Psi = \mathbf{Q}, \tag{4.1}$$

where $\mathbf{A}$ and $\mathbf{B}$ are both linear operators, Ψ is the full angular flux solution in space, angle, and energy, and $\mathbf{Q}$ is the source or driving function. If we had the ability to efficiently invert $(\mathbf{A} - \mathbf{B})$ directly, then Ψ could be directly computed:

$$\Psi = (\mathbf{A} - \mathbf{B})^{-1} \mathbf{Q}. \tag{4.2}$$

However, since in practice the discretized version of $(\mathbf{A} - \mathbf{B})$ is much more costly to directly invert than the discretized version of $\mathbf{A}$, we instead choose to iteratively solve for Ψ.

To compute Ψ, the following iterative system of equations is usually employed,

$$\mathbf{A}\Psi^{(k+1)} = \mathbf{B}\Psi^{(k)} + \mathbf{Q}, \tag{4.3}$$

where directly solving for $\Psi^{(k+1)}$ yields the following:

$$\Psi^{(k+1)} = \mathbf{A}^{-1}\mathbf{B}\Psi^{(k)} + \mathbf{A}^{-1}\mathbf{Q}. \tag{4.4}$$

For brevity, we define a new operator $\mathbf{C} = \mathbf{A}^{-1}\mathbf{B}$ which is known as the iteration operator. The spectral radius, ρ, of this operator is simply the supremum of the absolute values of its eigenvalues. For this work, we assume that ρ is less than unity to guarantee convergence. We next define the residual, $r^{(k)}$, as the difference between two successive solution iterates,

$$r^{(k)} = \Psi^{(k)} - \Psi^{(k-1)}, \tag{4.5}$$

which can also be written as the following:

$$r^{(k)} = \mathbf{C}r^{(k-1)}. \tag{4.6}$$

With the iteration operator, $\mathbf{C}$, and the residual for iterate k , $r^{(k)}$, defined, we can then write the true, converged solution in terms of the solution at iteration k

and an infinite series of residuals:

$$\Psi = \Psi^{(k)} + \sum_{n=1}^{\infty} r^{(k+n)}. \tag{4.7}$$

Using both Eqs. (4.6) and (4.7), we can rewrite Eq. (4.7) using the iteration operator and the last residual,

$$\Psi = \Psi^{(k)} + \left(\mathbf{I} + \mathbf{C} + \mathbf{C}^2 + ...\right) \mathbf{C} r^{(k)}. \tag{4.8}$$

Since we have assumed that the spectral radius of $\mathbf{C}$ is less than unity, the infinite operator series of Eq. (4.8) converges to $(\mathbf{I} - \mathbf{C})^{-1} \mathbf{C}$. This means that we can succinctly write Eq. (4.8) as the following:

$$\Psi = \Psi^{(k)} + (\mathbf{I} - \mathbf{C})^{-1} \mathbf{C} r^{(k)}. \tag{4.9}$$

By using the definition of $\mathbf{C}$ along with some linear algebra, Eq. (4.9) becomes

$$\Psi = \Psi^{(k)} + (\mathbf{A} - \mathbf{B})^{-1} \mathbf{B} r^{(k)}. \tag{4.10}$$

We would like to use the results of Eq. (4.10) to immediately compute our exact transport solution, Ψ. However, this would require the inversion of $(\mathbf{A} - \mathbf{B})$ which we did not employ originally in Eq. (4.1) because of the difficulty. This means that, in its current form, Eq. (4.10) is no more useful to us than Eq. (4.1). This would then be an exercise in futility if we were restricted to only working with the $(\mathbf{A} - \mathbf{B})$ operator. Instead, suppose that we could define an operator, $\mathbf{W}$, that closely approximates $(\mathbf{A} - \mathbf{B})$ but it is easily invertible. If $\mathbf{W}$ efficiently approximates the slowest converging error modes of $(\mathbf{A} - \mathbf{B})$, then Eq. (4.10) can be modified to form a new iterative procedure.

The new iterative procedure begins by simply taking the half-iterate of Eq. (4.4) instead of its full version: $(k+1/2)$ instead of (k+1). This half-iterate has the form

$$\Psi^{(k+1/2)} = \mathbf{C}\Psi^{(k)} + \mathbf{A}^{-1}\mathbf{Q}. \tag{4.11}$$

We can then express the full-iterate by the suggestion of Eq. (4.10). Using the low-order operator, we express the full-iterate as the following,

$$\Psi^{(k+1)} = \Psi^{(k+1/2)} + \mathbf{W}^{-1}\mathbf{B}r^{(k+1/2)}, \tag{4.12}$$

where $r^{(k+1/2)} = \Psi^{(k+1/2)} - \Psi^{(k)}$. We can also express Eq. (4.12) in terms of just the previous iterate, $\Psi^{(k)}$, and a new operator:

$$\Psi^{(k+1)} = \left[\mathbf{I} - \mathbf{W}^{-1}\left(\mathbf{A} - \mathbf{B}\right)\right]\mathbf{C}\Psi^{(k)} + \left(\mathbf{I} + \mathbf{W}^{-1}\mathbf{B}\right)\mathbf{A}^{-1}\mathbf{Q}. \tag{4.13}$$

Observe in Eq. (4.13) that as $\mathbf{W}$ more closely approximates $(\mathbf{A} - \mathbf{B})$, the operator $\mathbf{W}^{-1}\left(\mathbf{A} - \mathbf{B}\right)$ converges to the identity matrix, $\mathbf{I}$. This means that the spectral radius of this new iteration matrix will approach zero as $\mathbf{W}$ gets closer to $(\mathbf{A} - \mathbf{B})$ and therefore more quickly and efficiently converge to the true solution.

4.2 Diffusion Synthetic Acceleration Methodologies

The procedures outlined in Section 4.1.2 define a general methodology to perform synthetic acceleration on the transport equation. We could utilize any of the acceleration strategies that have been developed over the years including DSA, TSA, BPA, etc. The only difference arises in what form the low-order operator, $\mathbf{W}$, will take. We obviously are focusing on DSA for this dissertation work, and we do so by first describing in Section 4.2.1 a simple 1-group, continuous specification of the synthetic acceleration methodology just presented. Then, we present a generalized description

of the 1-group DSA strategy for the discretized transport equation in Section 4.2.2. We also show how DSA acts a preconditioner for the iterative transport methods. We conclude this section on DSA methodologies by presenting different strategies that can be employed to accelerate thermal neutron upscattering in Section 4.2.3.

4.2.1 Simple 1-Group, Isotropic DSA Strategy

Section 4.1.2 details the general methodology behind synthetic acceleration strategies. We now present a detailed derivation of DSA for a 1-group transport problem with isotropic scattering. This simple transport problem can be described by the following equation,

$$\vec{\Omega} \cdot \vec{\nabla}\psi + \sigma_t \psi = \frac{\sigma_s}{4\pi}\phi + \frac{Q}{4\pi} \tag{4.14}$$

where we do not include the spatial parameter for clarity. If $\mathbb{D}$ is the diameter of the problem domain, then Eq. (4.14) can slowly converge if the problem is optically thick ($\sigma_t \, \mathbb{D} \gg 1$) and there is little absorption in the problem ($\sigma_s \approx \sigma_t$). The Richardson method then calls for the following iterative approach where we use the half-iterate index, $(k + 1/2)$,

$$\vec{\Omega} \cdot \vec{\nabla}\psi^{(k+1/2)} + \sigma_t \psi^{(k+1/2)} = \frac{\sigma_s}{4\pi}\phi^{(k)} + \frac{Q}{4\pi}. \tag{4.15}$$

We could iterate on Eq. (4.15) continuously until we arrive at a converged solution. Each iteration simply adds the contribution of the scattering source from the previous iteration into the total source term. However, this process can be prohibitively slow if the problem is optically thick with little absorption.

Following the methodology of synthetic acceleration from Section 4.1.2, we now need to determine a formulation for the iteration error of this transport problem so

that we can employ the correction procedure of Eq. (4.10). An exact definition for the iteration error can be obtained by taking the difference of Eq. (4.15) from Eq. (**??**). This forms the following error equation,

$$\vec{\Omega} \cdot \vec{\nabla} \left(\psi - \psi^{(k+1/2)} \right) + \sigma_t \left(\psi - \psi^{(k+1/2)} \right) = \frac{\sigma_s}{4\pi} \left(\phi - \phi^{(k)} \right), \tag{4.16}$$

where we note that the distributed source, Q, vanishes. We can then add and subtract the term $\frac{\sigma_s}{4\pi}\phi^{(k+1/2)}$ into the right-hand-side of Eq. (4.16) to form

$$\vec{\Omega} \cdot \vec{\nabla} \left(\psi - \psi^{(k+1/2)} \right) + \sigma_t \left(\psi - \psi^{(k+1/2)} \right) = \frac{\sigma_s}{4\pi} \left(\phi - \phi^{(k+1/2)} \right) + \frac{\sigma_s}{4\pi} \left(\phi^{(k+1/2)} - \phi^{(k)} \right) \tag{4.17}$$

For brevity, we can define succinct terms for the error in the angular and scalar fluxes as

$$\delta\psi^{(k+1/2)} \equiv \psi - \psi^{(k+1/2)}, \tag{4.18}$$

and

$$\delta\phi^{(k+1/2)} \equiv \int_{4\pi} \delta\psi^{(k+1/2)}, \tag{4.19}$$

respectively. Inserting these error terms into Eq. (4.17) leads to the final, compact form for the continuous transport error:

$$\vec{\Omega} \cdot \vec{\nabla} \delta\psi^{(k+1/2)} + \sigma_t \delta\psi^{(k+1/2)} = \frac{\sigma_s}{4\pi} \delta\phi^{(k+1/2)} + \frac{\sigma_s}{4\pi} \left(\phi^{(k+1/2)} - \phi^{(k)} \right). \tag{4.20}$$

If we could efficiently solve for Eq. (4.20), then we would have the exact distribution

of the transport error and could obtain the exact transport solution with

$$\psi = \psi^{(k+1/2)} + \delta\psi^{(k+1/2)}. \tag{4.21}$$

However, solving Eq. (4.20) is just as difficult as the original transport equation. Therefore, we will form an approximate, low-order equation to Eq. (4.20) that is easier to compute.

For this optically thick transport problem that is dominated by scattering, the diffusive error modes that are not attenuated by the transport sweep dominate [29]. Our low-order approximation to Eq. (4.20) then needs to attenuate these diffusive modes. DSA schemes attenuate the low frequency error modes and underestimate the high frequency modes that are efficiently handled by transport sweeps. Therefore, we will approximate Eq. (4.20) with a diffusion equation. We form the standard diffusion equation by taking the continuous transport equation of Eq. (4.14) and performing the following steps:

1. Compute the 0th angular moment of Eq. (4.14).
2. Compute the 1st angular moment of Eq. (4.14).
3. Use the P_1 approximation to evaluate the pressure tensor in the 1st angular moment equation.
4. Represent the error equation from the derived standard diffusion equation.

We first take the 0th angular moment of the continuous transport equation which is simply done by integrating Eq. (4.14) over all angle. This yields

$$\vec{\nabla} \cdot \vec{J} + \sigma_a \phi = Q, \tag{4.22}$$

where we make use of the fact that $\sigma_t = \sigma_a + \sigma_s$ and that the angular current, $\vec{J}$, is defined as

$$\vec{J} = \int_{4\pi} d\Omega \, \vec{\Omega} \, \psi(\vec{\Omega}). \tag{4.23}$$

Next, we take the 1st angular moment of Eq. (4.14) by multiplying the equation by $\vec{\Omega}$ and then integrating over all angles. This then yields

$$\vec{\nabla} \cdot \int_{4\pi} d\Omega \, \vec{\Omega} \, \vec{\Omega} \, \psi(\vec{\Omega}) + \sigma_t \vec{J} = \vec{0}, \tag{4.24}$$

where the first term is a pressure term defined by the tensor product, $\vec{\Omega} \, \vec{\Omega}$, which has the form,

$$\vec{\Omega} \, \vec{\Omega} = \begin{bmatrix} \Omega_x \Omega_x & \Omega_x \Omega_y & \Omega_x \Omega_z \\ \Omega_y \Omega_x & \Omega_y \Omega_y & \Omega_y \Omega_z \\ \Omega_z \Omega_x & \Omega_z \Omega_y & \Omega_z \Omega_z \end{bmatrix}. \tag{4.25}$$

We then evaluate this pressure tensor by using the P_1 approximation on the angular flux. The P_1 expansion of the angular flux is linearly anisotropic and has the form

$$\psi = \frac{1}{4\pi} \left[\phi + 3\vec{\Omega} \cdot \vec{J} \right]. \tag{4.26}$$

Inserting this P_1 approximation into the pressure term and performing the angular integration leads to the following,

$$\begin{aligned}
\vec{\nabla}\cdot\int_{4\pi} d\Omega\,\vec{\Omega}\,\vec{\Omega}\,\psi(\vec{\Omega}) &\approx \frac{1}{4\pi}\vec{\nabla}\cdot\int_{4\pi} d\Omega\,\vec{\Omega}\,\vec{\Omega}\,\left[\phi+3\vec{\Omega}\cdot\vec{J}\right] \\
&= \frac{1}{4\pi}\vec{\nabla}\cdot\left[\phi\int_{4\pi} d\Omega\,\vec{\Omega}\,\vec{\Omega}+3\vec{J}\int_{4\pi} d\Omega\,\vec{\Omega}\,\vec{\Omega}\,\vec{\Omega}\right], \\
&= \frac{1}{4\pi}\vec{\nabla}\cdot\left[\phi\frac{4\pi}{3}\mathbb{I}+\vec{0}\right] \\
&= \frac{1}{3}\vec{\nabla}\phi
\end{aligned} \tag{4.27}$$

where $\mathbb{I}$ is the identity tensor. Inserting the result of Eq. (4.27) into Eq. (4.24) yields the approximate form for the 1st angular moment equation:

$$\frac{1}{3}\vec{\nabla}\phi+\sigma_t\vec{J}=\vec{0}. \tag{4.28}$$

Finally, we solve for the current, $\vec{J}$, in Eq. (4.28) and insert it into the 0th angular moment equation of Eq. (4.22). This leads to the standard reaction-diffusion equation,

$$\vec{\nabla}\cdot D\vec{\nabla}\phi+\sigma_a\phi=Q, \tag{4.29}$$

where the diffusion coefficient, D, has the form: $D=\frac{1}{3\sigma_t}$.

Equation (4.29) can then be represented as the low-order operator by properly inserting the error terms and the source residual. The final form for our low-order diffusion operator is the following:

$$\vec{\nabla}\cdot D\vec{\nabla}\delta\phi^{(k+1/2)}+\sigma_a\delta\phi^{(k+1/2)}=\sigma_s\left(\phi^{(k+1/2)}-\phi^{(k)}\right). \tag{4.30}$$

Equation (4.30) represents the continuous form for the error operator which has not been spatially discretized. There are many such discretization schemes that could

be employed as outlined in Section 4.1.1. We leave the details of the discretization scheme that we will employ in this work until Section 4.3. Once this approximate error distribution, $\delta\phi^{(k+1/2)}$, is computed by any solution algorithm of choice, the full-iterate correction of the scalar flux is given by:

$$\phi^{(k+1)} = \phi^{(k+1/2)} + \delta\phi^{(k+1/2)}. \tag{4.31}$$

Thus far, we have presented a DSA scheme to accelerate the continuous 1-group, isotropic transport equation. We did not prescribe an angular or spatial discretization scheme for the transport and derived low-order diffusion equations. Next in Section 4.2.2, we detail a generalized DSA strategy for the discretized 1-group transport problem using operator notation. We then show how these DSA schemes form preconditioned operators for our Richardson and Krylov iterative methods.

4.2.2 Generalized 1-Group DSA Operators

Section 4.2.1 defined the DSA strategy for the simple case of the continuous, 1-group, isotropic transport equation. We now provide a detailed description of the DSA strategy for a generally-dscretized, 1-group transport problem using operator notation. We do this by again detailing how the error equation is formed from the discretized iterative equations. Then, we detail how the transport error equation can be restricted and prolongated from the coarse-grid, low-order diffusion operator. Finally, we combine all the operators into the appropriate correction equation, that is the discretized analogue to Eq. (4.31).

Recall the operator form of the fully discretized transport equation as defined in Section 2.7.1,

$$\begin{aligned} \mathbf{L}\boldsymbol{\Psi} &= \mathbf{M}\boldsymbol{\Sigma}\boldsymbol{\Phi} + \mathbf{Q} \\ \boldsymbol{\Phi} &= \mathbf{D}\boldsymbol{\Psi} \end{aligned}, \tag{4.32}$$

where $\mathbf{L}$ is the total interaction and streaming operator, $\mathbf{M}$ is the moment-to-discrete operator, $\mathbf{D}$ is the discrete-to-moment operator, $\boldsymbol{\Sigma}$ is the scattering operator, and $\mathbf{Q}$ is the forcing function. In this case, we simply treat this discretized problem as only having 1 energy group, but no restriction on the order of the Spherical Harmonic expansion, N_p. The functional form of the discretized moment-to-discrete and discrete-to-moment operators are

$$M_m \equiv \sum_{p=0}^{N_P} \frac{2p+1}{4\pi} \sum_{n=-p}^{p} Y_{p,n}(\vec{\Omega}_m), \tag{4.33}$$

and

$$D_{p,n} \equiv \sum_{m=1}^{M} w_m Y_{p,n}(\vec{\Omega}_m), \tag{4.34}$$

respectively. This means that the operation $\mathbf{D}\mathbf{L}^{-1}$ corresponds to a full-domain transport sweep followed by the computation of the flux moments from the angular flux. We next apply our half-iterate and previous iterate indices on Eq. (4.32) to form the Richardson iteration procedure for our transport equation:

$$\mathbf{L}\boldsymbol{\Psi}^{(k+1/2)} = \mathbf{M}\boldsymbol{\Sigma}\boldsymbol{\Phi}^{(k)} + \mathbf{Q}. \tag{4.35}$$

We can then use the $\mathbf{D}\mathbf{L}^{-1}$ operator to present our transport equation of Eq. (4.35) in terms of just the half-iterate flux moments,

$$\boldsymbol{\Phi}^{(k+1/2)} = \mathbf{D}\mathbf{L}^{-1}\mathbf{M}\boldsymbol{\Sigma}\boldsymbol{\Phi}^{(k)} + \mathbf{D}\mathbf{L}^{-1}\mathbf{Q}. \tag{4.36}$$

We define the operators $\mathbf{T} \equiv \mathbf{D}\mathbf{L}^{-1}\mathbf{M}\boldsymbol{\Sigma}$ and $\mathbf{b} \equiv \mathbf{D}\mathbf{L}^{-1}\mathbf{Q}$ and insert them into Eq. (4.36) to form

$$\boldsymbol{\Phi}^{(k+1/2)} = \mathbf{T}\boldsymbol{\Phi}^{(k)} + \mathbf{b}. \tag{4.37}$$

Next, the functional form for the iteration error equation is formed by taking the difference of Eq. (4.35) from the first line of Eq. (4.32). This yields the following discretized iteration error equation,

$$\mathbf{L}\delta\boldsymbol{\Psi}^{(k+1/2)} - \mathbf{M}\boldsymbol{\Sigma}\delta\boldsymbol{\Phi}^{(k+1/2)} = \mathbf{M}\boldsymbol{\Sigma}\left(\boldsymbol{\Phi}^{(k+1/2)} - \boldsymbol{\Phi}^{(k)}\right). \tag{4.38}$$

In a similar manner as outlined in Section 4.2.1, the low-order diffusion operator for the transport error of Eq. (4.38) can be formed by taking its 0th and 1st angular moments. The left-hand-side of this error equation, which contains the reaction and diffusion terms, is now given by the single operator $\mathbf{A}$. The diffusion correction to the transport solution after the $(k+1/2)$ Richardson iteration is given by

$$\delta\boldsymbol{\Phi}^{(k+1/2)} = \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\left(\boldsymbol{\Phi}^{(k+1/2)} - \boldsymbol{\Phi}^{(k)}\right), \tag{4.39}$$

where $\mathbf{P}$ and $\mathbf{X}$ are the prolongation and restriction operators, respectively. The restriction operator acts by limiting the contributions to the low-order residual to the 0th (and maybe 1st) moments. Then once the diffusion correction, $\delta\boldsymbol{\Phi}^{(k+1/2)}$, is calculated, the prolongation operator manipulates the error corrections onto the appropriate flux moments. With the definitions of the half-iterate solution and the diffusion correction given in Eqs. (4.36) and (4.39), respectively, we can form the full-iterate operator:

$$
\begin{aligned}
\boldsymbol{\Phi}^{(k+1)} &= \boldsymbol{\Phi}^{(k+1/2)} + \delta\boldsymbol{\Phi}^{(k+1/2)} \\
&= \mathbf{T}\boldsymbol{\Phi}^{(k)} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\left(\boldsymbol{\Phi}^{(k+1/2)} - \boldsymbol{\Phi}^{(k)}\right) + \mathbf{b} \\
&= \mathbf{T}\boldsymbol{\Phi}^{(k)} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\left(\mathbf{T}\boldsymbol{\Phi}^{(k)} + \mathbf{b} - \boldsymbol{\Phi}^{(k)}\right) + \mathbf{b} \\
&= \mathbf{T}\boldsymbol{\Phi}^{(k)} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\left(\mathbf{T} - \mathbf{I}\right)\boldsymbol{\Phi}^{(k)} + \left(\mathbf{I} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\right)\mathbf{b} \\
&= \left[\mathbf{T} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\left(\mathbf{T} - \mathbf{I}\right)\right]\boldsymbol{\Phi}^{(k)} + \left(\mathbf{I} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\right)\mathbf{b}
\end{aligned}
\quad . \tag{4.40}
$$

Equation (4.40) provides the accelerated iteration scheme where the matrix operator $\left[\mathbf{T} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\left(\mathbf{T} - \mathbf{I}\right)\right]$ is the corresponding iteration matrix for the DSA method. This matrix is in contrast to the unaccelerated iteration matrix, $\mathbf{T}$, given in Eq. (4.37). These accelerated and unaccelerated iteration matrices can be analyzed to obtain knowldege of how the transport solutions will converge.

We now quickly show how the DSA iterative procedure given by Eq. (4.40) is exactly a preconditioned Richardson iteration. We do this by first adding and subtracting $\boldsymbol{\Phi}^{(k)}$ to the final line of Eq. (4.40) to form the following equivalent iteration scheme,

$$
\begin{aligned}
\boldsymbol{\Phi}^{(k+1)} &= \left[\mathbf{T} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\left(\mathbf{T} - \mathbf{I}\right)\right]\boldsymbol{\Phi}^{(k)} + \left(\mathbf{I} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\right)\mathbf{b} \\
&= \boldsymbol{\Phi}^{(k)} + \left(\mathbf{T} - \mathbf{I}\right)\boldsymbol{\Phi}^{(k)} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\left(\mathbf{T} - \mathbf{I}\right)\boldsymbol{\Phi}^{(k)} + \left(\mathbf{I} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\right)\mathbf{b} \\
&= \boldsymbol{\Phi}^{(k)} + \left(\mathbf{I} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\right)\left(\mathbf{T} - \mathbf{I}\right)\boldsymbol{\Phi}^{(k)} + \left(\mathbf{I} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\right)\mathbf{b} \\
&= \boldsymbol{\Phi}^{(k)} - \left(\mathbf{I} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\right)\left(\mathbf{I} - \mathbf{T}\right)\boldsymbol{\Phi}^{(k)} + \left(\mathbf{I} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\right)\mathbf{b}
\end{aligned}
\quad . \tag{4.41}
$$

Next, we replace the iteration terms of $\boldsymbol{\Phi}^{(k)}$ and $\boldsymbol{\Phi}^{(k+1)}$ with $\boldsymbol{\Phi}$ in Eq. (4.41) and move the solution terms to the left-hand-side of the equation. This gives the following equation with no iteration indices,

$$\left(\mathbf{I}+\mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\right)\left(\mathbf{I}-\mathbf{T}\right)\boldsymbol{\Phi}=\left(\mathbf{I}+\mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\right)\mathbf{b}. \tag{4.42}$$

Recall from Section 2.7 that the moment-only form of the discretized transport equation is given by

$$\left(\mathbf{I}-\mathbf{T}\right)\boldsymbol{\Phi}=\mathbf{b}. \tag{4.43}$$

This means that the DSA scheme is exactly a preconditioned Richardson iterative scheme where the operator $\left(\mathbf{I}+\mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\right)$ acts as a left-preconditioner. When Krylov schemes are used in place of Richardson iteration, Eq. (4.42) will still describe our preconditioned transport equation. Recall from Section 2.7.1.2 that only minor changes need to made to existing source iteration routines to enable Krylov solvers to properly solve the transport equation. In particular, the subtle differences lie in applying the matrix-free transport sweep operator on some Krylov vector, $\mathbf{x}$ and in building the right-hand-side vector. We can show that we form the appropriate action of the matrix operator, which we denote as $\mathbf{Z}$, in an identical manner to unaccelerated Richardson iteration. Recall, that the unaccelerated matrix operator is computed by differencing the original Krylov vector by the vector following a transport sweep. Therefore, we compute this same difference with the results from our DSA accelerated transport sweep to form:

$$\begin{aligned}\mathbf{Z}\mathbf{x}&=\mathbf{x}-\left[\mathbf{T}+\mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\left(\mathbf{T}-\mathbf{I}\right)\right]\mathbf{x}\\&=\left(\mathbf{I}-\mathbf{T}\right)\mathbf{x}+\mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\left(\mathbf{I}-\mathbf{T}\right)\mathbf{x}\,.\\&=\left(\mathbf{I}+\mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\right)\left(\mathbf{I}-\mathbf{T}\right)\mathbf{x}\end{aligned} \tag{4.44}$$

In Eq. (4.44), we see that we identically recover the left-preconditioned transport operator from Eq. (4.42). Finally, it is easy to show that we can compute the

right-hand-side source of Eq. (4.42). Recall that $\mathbf{b} = \mathbf{D}\mathbf{L}^{-1}\mathbf{Q}$, which is computed by performing a single transport sweep on $\mathbf{Q}$. This means that the accelerated Krylov right-hand-side vector, $\mathbf{b}+\mathbf{P}\mathbf{A}^{-1}\mathbf{X}\boldsymbol{\Sigma}\mathbf{b}$, can be formed by the same correction operation of Eq. (4.40). The only differences lie in the use of $\mathbf{Q}$ instead of the Krylov vector, $\mathbf{x}$, and the use of $\mathbf{b}$ instead of $\left(\boldsymbol{\Phi}^{(k+1/2)} - \boldsymbol{\Phi}^{(k)}\right)$ when computing the residual for the diffusion equation.

4.2.3 DSA Acceleration Strategies for Thermal Neutron Upscattering

We have just provided a detailed description of the DSA methodology for a 1-group transport problem. The transport and low-order diffusion operators were discretized, but left as arbitrary. If desired, this 1-group acceleration methodology could be utilized to precondition transport sweeps for a single group in a multigroup problem (if an energy group has a high scattering ratio). However, for a transport problem with many energy groups that need acceleration, there may be more efficient acceleration strategies. In particular, it is difficult to converge the scattering source for transport problems that are dominated by thermal neutron upscattering. We next present three different DSA strategies to accelerate the neutron upscattering that we will analyze for this work.

4.2.3.1 Standard Two-Grid (TG) Acceleration

The first thermal neutron upscatter acceleration methodology that we will investigate is the standard Two-Grid (TG) acceleration scheme devised by Adams and Morel [95]. They originally derived the scheme to accelerate 1D multigroup transport problems that are dominated by thermal neutron upscattering (physical systems containing a lot of graphite or heavy water). The method can be quickly summarized as the following:

1. Perform a Gauss-Seidel procedure in energy for the thermal groups and con-

verge the inner iterations;

2. Perform a spectral collapse of the transport iteration error into a 1-group diffusion equation;

3. Solve the 1-group diffusion equation for the spatial variation of the transport iteration error;

4. Interpolate the diffusive multigroup error back onto the transport solution.

We now provide fulls details for each of these steps of the TG method.

The first step in the TG method is to perform a Gauss-Seidel procedure in energy across the thermal groups. This means that for every outer iteration (we can also think of these as thermal iterations), we sequentially proceed through the thermal energy groups. Based on the notation of a group set introduced in Section 2.7, we can achieve this Gauss-Seidel iteration scheme by having each thermal group be in its own group set. The TG method then requires full convergence of the within-group inner iterations for each of the group sets. If we have G number of thermal groups, this Gauss-Seidel iteration scheme (with convergence of the inner iterations) leads to the following iteration equation,

$$\mathbf{L_{gg}}\psi_g^{(k+1/2)} = \mathbf{M}\sum_{g'=1}^{g}\mathbf{\Sigma}_{gg'}\phi_{g'}^{(k+1/2)} + \mathbf{M}\sum_{g'=g+1}^{G}\mathbf{\Sigma}_{gg'}\phi_{g'}^{(k)} + \mathbf{Q}_g, \tag{4.45}$$

where the scattering terms still contain some arbitrary number of moments. In operator form, the exact solution and half-iterate equations are given by

$$\mathbf{L\Psi} = \mathbf{M}\left(\mathbf{S_L} + \mathbf{S_D} + \mathbf{S_U}\right)\mathbf{\Phi} + \mathbf{Q}, \tag{4.46}$$

and

$$\mathbf{L}\boldsymbol{\Psi}^{(k+1/2)} = \mathbf{M}\left(\mathbf{S_L} + \mathbf{S_D}\right)\boldsymbol{\Phi}^{(k+1/2)} + \mathbf{M}\mathbf{S_U}\boldsymbol{\Phi}^{(k)} + \mathbf{Q}, \tag{4.47}$$

respectively. $\mathbf{S_L}$, $\mathbf{S_D}$, and $\mathbf{S_U}$ are the strictly-downscattering, diagonal within-group scattering, and strictly upscattering portions of the scattering matrix, respectively. By moving the downscattering and diagonal portions of the scattering operator to the left side of the equation, inverting the $\mathbf{L}$ operator, and applying the discrete-to-moment operator, $\mathbf{D}$, we can rewrite the iteration equation of Eq. (4.47) in terms of only the flux moments:

$$\left[\mathbf{I} - \mathbf{D}\mathbf{L}^{-1}\mathbf{M}\left(\mathbf{S_L} + \mathbf{S_D}\right)\right]\boldsymbol{\Phi}^{(k+1/2)} = \mathbf{D}\mathbf{L}^{-1}\mathbf{M}\mathbf{S_U}\boldsymbol{\Phi}^{(k)} + \mathbf{D}\mathbf{L}^{-1}\mathbf{Q}. \tag{4.48}$$

By inverting the left-side operator, we directly solve for the half-iterate flux moments,

$$\boldsymbol{\Phi}^{(k+1/2)} = \left[\mathbf{I} - \mathbf{D}\mathbf{L}^{-1}\mathbf{M}\left(\mathbf{S_L} + \mathbf{S_D}\right)\right]^{-1}\mathbf{D}\mathbf{L}^{-1}\mathbf{M}\mathbf{S_U}\boldsymbol{\Phi}^{(k)} + \mathbf{b}, \tag{4.49}$$

where the simplified distributed source term, $\mathbf{b}$, is now defined for further clarity:

$$\mathbf{b} = \left[\mathbf{I} - \mathbf{D}\mathbf{L}^{-1}\mathbf{M}\left(\mathbf{S_L} + \mathbf{S_D}\right)\right]^{-1}\mathbf{D}\mathbf{L}^{-1}\mathbf{Q}. \tag{4.50}$$

At this point, Eq. (4.49) provides a formulation for the transport solution at iteration $(k+1/2)$ based on the solution at iteration (k). We now require a formulation for the accompanying error at this iteration step. We subtract Eq. (4.45) from the exact transport solution to form an equation specifying the exact error at iteration $(k+1/2)$,

$$\mathbf{L_{gg}}\delta\psi_g^{(k+1/2)} = \mathbf{M}\sum_{g'=1}^{g}\mathbf{\Sigma}_{gg'}\delta\phi_{g'}^{(k+1/2)} + \mathbf{M}\sum_{g'=g+1}^{G}\mathbf{\Sigma}_{gg'}\delta\phi_{g'}^{(k)}, \tag{4.51}$$

where the angular flux and flux moment error terms have an analogous multigroup form,

$$\begin{aligned}\delta\psi_g^{(k+1/2)} &= \psi_g - \psi_g^{(k+1/2)} \\ \delta\phi_g^{(k+1/2)} &= \mathbf{D}\delta\psi_g^{(k+1/2)}\end{aligned}. \tag{4.52}$$

Next, we add and subtract $\mathbf{M}\sum_{g'=g+1}^{G}\mathbf{\Sigma}_{gg'}\phi_{g'}^{(k+1/2)}$ to Eq. (4.51) to form

$$\mathbf{L_{gg}}\delta\psi_g^{(k+1/2)} = \mathbf{M}\sum_{g'=1}^{G}\mathbf{\Sigma}_{gg'}\delta\phi_{g'}^{(k+1/2)} + \mathbf{M}\sum_{g'=g+1}^{G}\mathbf{\Sigma}_{gg'}\left(\phi_{g'}^{(k+1/2)} - \phi_{g'}^{(k)}\right). \tag{4.53}$$

Equation (4.53) can be recast into its appropriate operator notation,

$$\mathbf{L}\delta\mathbf{\Psi}^{(k+1/2)} - \mathbf{MS}\delta\mathbf{\Phi}^{(k+1/2)} = \mathbf{MS_U}\left(\mathbf{\Phi}^{(k+1/2)} - \mathbf{\Phi}^{(k)}\right), \tag{4.54}$$

where $\mathbf{S} = \mathbf{S_L} + \mathbf{S_D} + \mathbf{S_U}$ is the full scattering operator. Equation (4.53) and the corresponding operator form in Eq. (4.54) provide the complete formulation of the multrigroup iteration error. Just like it was previously mentioned for the 1-group scenario, these error equations are just as difficult to solve as the full transport equation.

We again choose to utilize the diffusion operator as the low-order operator for the iteration error. Taking the 0th and 1st angular moments of Eq. (4.53) and applying Fick's Law, we arrive at the standard multigroup diffusion equation for the error,

$$\vec{\nabla} \cdot D_g \vec{\nabla} \delta\phi_g^{(k+1/2)} + \sigma_{t,g} \delta\phi_g^{(k+1/2)} - \sum_{g'=1}^{G} \sigma_{s,0}^{gg'} \delta\phi_{g'}^{(k+1/2)} = R_g^{(k+1/2)}, \tag{4.55}$$

where the residual, R_g, is only in terms of the 0th-order scattering moment and has the following form:

$$R_g^{(k+1/2)} = \sum_{g'=g+1}^{G} \sigma_{s,0}^{gg'} \left(\phi_{g',0}^{(k+1/2)} - \phi_{g',0}^{(k)} \right). \tag{4.56}$$

We could solve these G coupled multigroup diffusion equations of Eq. (4.55) for the iteration error. However, if the number of thermal groups becomes large, then these coupled equations could be become burdensome to solve. Instead, the TG method opts to perform a spectral collapse of the multigroup diffusion error. First, we factorize the 0th moment of the multigroup error,

$$\delta\phi_{g,0}^{(k+1/2)} = \xi_g \epsilon^{(k+1/2)}(\vec{r}), \qquad \sum_{g=1}^{G} \xi_g = 1, \tag{4.57}$$

into a spatial component, $\epsilon^{(k+1/2)}(\vec{r})$, and an energy component, ξ_g. The spectral shape, ξ_g, is the eigenfunction corresponding to the largest eigenvalue of the infinite medium iteration matrix of Eq. (4.47) with no driving source term. This eigenvalue problem can be succinctly written as,

$$\left(\mathbf{T} - \mathbf{S_{L,0}} - \mathbf{S_{D,0}} \right)^{-1} \mathbf{S_{U,0}} \xi = \rho \xi, \tag{4.58}$$

where $\mathbf{T}$ is the diagonal matrix of total cross sections and $\mathbf{S_{L,0}}$, $\mathbf{S_{D,0}}$, and $\mathbf{S_{U,0}}$ are restricted to the 0th-order moments of the scattering cross sections. Inserting the factorized error of Eq. (4.55) into Eq. (4.55) and summing over energy groups gives

$$\vec{\nabla} \cdot \left\langle D \right\rangle \vec{\nabla} \epsilon^{(k+1/2)} + \vec{\nabla} \cdot \left\langle \vec{D} \right\rangle \epsilon^{(k+1/2)} + \left\langle \sigma \right\rangle \epsilon^{(k+1/2)} = \left\langle R^{(k+1/2)} \right\rangle, \tag{4.59}$$

where the energy-averaged terms are

$$\begin{aligned} \left\langle D \right\rangle &= \sum_{g=1}^{G} D_g \xi_g, \\ \left\langle \vec{D} \right\rangle &= \sum_{g=1}^{G} D_g \vec{\nabla} \xi_g, \\ \left\langle \sigma \right\rangle &= \sum_{g=1}^{G} \left(\sigma_{t,g} \xi_g - \sum_{g'=1}^{G} \sigma_{s,0}^{gg'} \xi_{g'} \right), \\ \left\langle R^{(k+1/2)} \right\rangle &= \sum_{g=1}^{G} R_g^{(k+1/2)}. \end{aligned} \tag{4.60}$$

Equation (4.59) is not the standard diffusion equation because of the drift term containing the gradient of the spectral shape. This term is an artifact of the factorization of the error and it is identically zero in homogeneous regions but undefined at material interfaces. The TG method simply neglects this term, which leads to the final form for our coarse-grid error equation,

$$\vec{\nabla} \cdot \left\langle D \right\rangle \vec{\nabla} \epsilon^{(k+1/2)} + \left\langle \sigma \right\rangle \epsilon^{(k+1/2)} = \left\langle R^{(k+1/2)} \right\rangle. \tag{4.61}$$

If we define the left-hand-side matrix operator of Eq. (4.61) as $\mathbf{A}$, then the operator form of this coarse-grid error equation is

$$\mathbf{A} \epsilon^{(k+1/2)} = \mathbf{X} \mathbf{S}_\mathbf{U} \left(\mathbf{\Phi}^{(k+1/2)} - \mathbf{\Phi}^{(k)} \right), \tag{4.62}$$

where $\mathbf{X}$ is the restriction operator that confines the diffusion problem to the 0th-order moment and performs the sum from Eq. (4.60). Solving Eq. (4.61) gives the

spatial distribution of the iteration error. We can then update the error for the 0th moment flux with the following:

$$\phi_{g,0}^{(k+1)} = \phi_{g,0}^{(k+1/2)} + \xi_g \epsilon^{(k+1/2)}, \qquad g = 1, ..., G. \tag{4.63}$$

Up to this point, we have provided the full details of the TG methodology including the Gauss-Seidel iteration equations, the process to spectrally-collapse the diffusive error, and the addiditve interpolation of the diffusive error. We now go through these steps using compact operator notation to arive at a single matrix form for the TG accelerated transport iterations. First, we express the full phase-space update equation as

$$\mathbf{\Phi}^{(k+1)} = \mathbf{\Phi}^{(k+1/2)} + \delta\mathbf{\Phi}^{(k+1/2)}. \tag{4.64}$$

The half-iterate solution, $\mathbf{\Phi}^{(k+1/2)}$, is given by Eqs. (4.49) and (4.50), and the iteration error, $\delta\mathbf{\Phi}^{(k+1/2)}$, is

$$\delta\mathbf{\Phi}^{(k+1/2)} = \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\mathbf{S}_\mathbf{U}\left(\mathbf{\Phi}^{(k+1/2)} - \mathbf{\Phi}^{(k)}\right), \tag{4.65}$$

where $\mathbf{P}$ is the prolongation operator which interpolates the error back into the full phase-space of the transport equation. For the TG method, this prolongation operator acts on only the 0th-order flux moments (the higher moments are characteristically zero) and appropriately adds the error correction for thermal group g with the appropriate spectral weight, ξ_g. Inserting these definitions into Eq. (4.64) gives

$$\begin{aligned}\mathbf{\Phi}^{(k+1)} = &\left[\mathbf{I} - \mathbf{D}\mathbf{L}^{-1}\mathbf{M}\left(\mathbf{S}_\mathbf{L} + \mathbf{S}_\mathbf{D}\right)\right]^{-1}\mathbf{D}\mathbf{L}^{-1}\mathbf{M}\mathbf{S}_\mathbf{U}\mathbf{\Phi}^{(k)} \\ &+ \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\mathbf{S}_\mathbf{U}\left(\mathbf{\Phi}^{(k+1/2)} - \mathbf{\Phi}^{(k)}\right) + \mathbf{b}\end{aligned}. \tag{4.66}$$

We further define the term $\mathbf{F} \equiv \left[\mathbf{I} - \mathbf{DL}^{-1}\mathbf{M}\left(\mathbf{S_L} + \mathbf{S_D}\right)\right]^{-1}\mathbf{DL}^{-1}$, and use it to re-express Eq. (4.66) into a singular equation for the full-iterate solution,

$$\begin{aligned}
\mathbf{\Phi}^{(k+1)} &= \mathbf{FMS_U}\mathbf{\Phi}^{(k)} + \mathbf{PA}^{-1}\mathbf{XS_U}\left(\mathbf{\Phi}^{(k+1/2)} - \mathbf{\Phi}^{(k)}\right) + \mathbf{b} \\
&= \mathbf{FMS_U}\mathbf{\Phi}^{(k)} + \mathbf{PA}^{-1}\mathbf{XS_U}\left(\ \mathbf{FMS_U}\mathbf{\Phi}^{(k)} + \mathbf{b} - \mathbf{\Phi}^{(k)}\right) + \mathbf{b} \\
&= \mathbf{FMS_U}\mathbf{\Phi}^{(k)} + \mathbf{PA}^{-1}\mathbf{XS_U}\left(\ \mathbf{FMS_U} - \mathbf{I}\right)\mathbf{\Phi}^{(k)} + \left(\mathbf{PA}^{-1}\mathbf{XS_U} + \mathbf{I}\right)\mathbf{b} \\
&= \left[\mathbf{FMS_U} + \mathbf{PA}^{-1}\mathbf{XS_U}\left(\ \mathbf{FMS_U} - \mathbf{I}\right)\right]\mathbf{\Phi}^{(k)} + \left(\mathbf{PA}^{-1}\mathbf{XS_U} + \mathbf{I}\right)\mathbf{b}
\end{aligned}, \quad (4.67)$$

where we note that $\mathbf{b} = \mathbf{FDL}^{-1}\mathbf{Q}$. From Eq. (4.67), the iteration matrix for the TG scheme is given by $\left[\mathbf{FMS_U} + \mathbf{PA}^{-1}\mathbf{XS_U}\left(\ \mathbf{FMS_U} - \mathbf{I}\right)\right]$. The eigenvalues of this iteration matrix will give insight into the convergence properties of the TG method in the asymptotic region.

4.2.3.2 Modified Two-Grid (MTG) Acceleration

The second thermal upscattering acceleration method that we will investigate is a simple modification to the standard TG method of Section 4.2.3.1. At the end of their work involving a TSA variant of the TG method [96], Evans, Clarno, and Morel proposed a modified form for the TG method, which they labeled the Modified Transport Two-Grid (MTTG) method. We wish to adopt their iterative strategy, but use the diffusion equation as our low-order operator. We choose to call this method the Modified Two-Grid (MTG) method. In their work, they proposed to not fully converge the inner iterations for each group in the Gauss-Seidel process. Just like the TG method, we again sequentially proceed through the thermal groups in a Gauss-Seidel manner but only perform 1 transport sweep for each group. This process yields the following iteration scheme,

$$\mathbf{L_{gg}}\psi_g^{(k+1/2)} = \mathbf{M}\sum_{g'=1}^{g-1}\mathbf{\Sigma}_{gg'}\phi_{g'}^{(k+1/2)} + \mathbf{M}\sum_{g'=g}^{G}\mathbf{\Sigma}_{gg'}\phi_{g'}^{(k)} + \mathbf{Q}_g, \tag{4.68}$$

where it differs with Eq. (4.45) in the ending and beginning energy indices for the $(k+1/2)$ and (k) iterations, respectively. In operator notation, the iteration equation of the MTG method is the following:

$$\mathbf{L\Psi}^{(k+1/2)} = \mathbf{MS_L\Phi}^{(k+1/2)} + \mathbf{M}\left(\mathbf{S_D} + \mathbf{S_U}\right)\mathbf{\Phi}^{(k)} + \mathbf{Q}. \tag{4.69}$$

This operator equation for the MTG differs from Eq. (4.47) of the TG method by the locations of the different scattering operators. Solving for the flux moments, we can give the half-iterate and external source equations as,

$$\mathbf{\Phi}^{(k+1/2)} = \left[\mathbf{I} - \mathbf{DL}^{-1}\mathbf{MS_L}\right]^{-1}\mathbf{DL}^{-1}\mathbf{M}\left(\mathbf{S_D} + \mathbf{S_U}\right)\mathbf{\Phi}^{(k)} + \mathbf{b}, \tag{4.70}$$

and

$$\mathbf{b} = \left[\mathbf{I} - \mathbf{DL}^{-1}\mathbf{MS_L}\right]^{-1}\mathbf{DL}^{-1}\mathbf{Q}. \tag{4.71}$$

respectively.

The generation of the spectrally-collapsed diffusion equation for the MTG iteration error is almost identical to the TG method. The only differences lie with generating the spectral shape functions and the residual. Like the TG method, the spectral shape function for each material is the eigenfunction corresponding to the largest eigenvalue of the infinite medium iteration matrix of Eq. (4.68). This eigenproblem is given by,

$$\left(\mathbf{T} - \mathbf{S_{L,0}}\right)^{-1}\left(\mathbf{S_{D,0}} + \mathbf{S_{U,0}}\right)\xi = \rho\xi, \tag{4.72}$$

where the matrix operators remain from before. With this spectral shape, the average diffusion coefficient and average absorption cross section can again be calculated by Eq. (4.60), while the error residual is given by

$$R_g^{(k+1/2)} = \sum_{g'=g}^{G} \sigma_{s,0}^{gg'} \left(\phi_{g',0}^{(k+1/2)} - \phi_{g',0}^{(k)} \right). \tag{4.73}$$

Again, the coarse-grid error equation is given by Eq. (4.61), with a corresponding operator form of

$$\mathbf{A}\epsilon^{(k+1/2)} = \mathbf{X}\left(\mathbf{S_D} + \mathbf{S_U}\right)\left(\mathbf{\Phi}^{(k+1/2)} - \mathbf{\Phi}^{(k)}\right). \tag{4.74}$$

We now provide the full phase-space update equation in a like manner to TG. Just like the TG method, the update equation can be expressed as

$$\mathbf{\Phi}^{(k+1)} = \mathbf{\Phi}^{(k+1/2)} + \delta\mathbf{\Phi}^{(k+1/2)}, \tag{4.75}$$

where we insert the half-iterate and coarse-grid error terms from Eqs. (4.70) and (4.83), respectively. Defining $\mathbf{F} \equiv \left[\mathbf{I} - \mathbf{D}\mathbf{L}^{-1}\mathbf{M}\mathbf{S_L}\right]^{-1}\mathbf{D}\mathbf{L}^{-1}$ for brevity, we can give the singular equation for the MTG method,

$$\begin{aligned}
\mathbf{\Phi}^{(k+1)} =& \mathbf{F}\mathbf{M}\left(\mathbf{S_D} + \mathbf{S_U}\right)\mathbf{\Phi}^{(k)} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\left(\mathbf{S_D} + \mathbf{S_U}\right)\left(\mathbf{\Phi}^{(k+1/2)} - \mathbf{\Phi}^{(k)}\right) + \mathbf{b} \\
=& \mathbf{F}\mathbf{M}\left(\mathbf{S_D} + \mathbf{S_U}\right)\mathbf{\Phi}^{(k)} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\left(\mathbf{S_D} + \mathbf{S_U}\right)\left(\mathbf{F}\mathbf{M}\left(\mathbf{S_D} + \mathbf{S_U}\right) - \mathbf{I}\right)\mathbf{\Phi}^{(k)} \\
&+ \left(\mathbf{P}\mathbf{A}^{-1}\mathbf{X}\left(\mathbf{S_D} + \mathbf{S_U}\right) + \mathbf{I}\right)\mathbf{b} \\
=& \left[\mathbf{F}\mathbf{M}\left(\mathbf{S_D} + \mathbf{S_U}\right) + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\left(\mathbf{S_D} + \mathbf{S_U}\right)\left(\mathbf{F}\mathbf{M}\left(\mathbf{S_D} + \mathbf{S_U}\right) - \mathbf{I}\right)\right]\mathbf{\Phi}^{(k)} \\
&+ \left(\mathbf{P}\mathbf{A}^{-1}\mathbf{X}\left(\mathbf{S_D} + \mathbf{S_U}\right) + \mathbf{I}\right)\mathbf{b}
\end{aligned}, \tag{4.76}$$

where we note that $\mathbf{b} = \mathbf{FDL}^{-1}\mathbf{Q}$ and is unchanged from the TG method ($\mathbf{F}$ just simply has a different definition). Equation (4.76) gives the iteration matrix for the MTG scheme, which can be used to understand the convergence properties of the method in the asymptotic region.

4.2.3.3 Multigroup Jacobi Acceleration

The third and final thermal neutron upscattering acceleration method that we will investigate is markedly different than the TG and MTG methods. With the increasing expansion of parallel computing resources, the sequential Gauss-Seidel approach for the thermal energy groups necessarily becomes a limiting bottleneck. Therefore, we have natural recourse to develop an alternate thermal upscattering acceleration method that can more effectively make use of parallel algorithms and architectures. Therefore, instead of sequentially marching through the thermal energy groups in a Gauss-Seidel, we solve them simultaneously at each outer iteration. This is achieved by placing all the thermal energy groups into a single group set. Then, each outer iteration performs one transport sweep where the thermal scattering source was generated from the sweep of the previous outer iteration. This procedure is identical to a block Jacobi iteration where a single inner iteration is performed for each thermal group. Therefore, we choose to call this methodology the Multigroup Jacobi Acceleration (MJA) method. This process of simultaneously solving all the thermal groups in a transport sweep yields the following iteration scheme:

$$\mathbf{L_{gg}}\psi_g^{(k+1/2)} = \mathbf{M}\sum_{g'=1}^{G}\mathbf{\Sigma}_{gg'}\phi_{g'}^{(k)} + \mathbf{Q}_g. \tag{4.77}$$

Just like the TG and MTG methods, we can express Eq. (4.77) in compact operator notation,

$$\mathbf{L}\boldsymbol{\Psi}^{(k+1/2)} = \mathbf{MS}\boldsymbol{\Phi}^{(k)} + \mathbf{Q}, \tag{4.78}$$

where $\mathbf{S} = \mathbf{S_L} + \mathbf{S_D} + \mathbf{S_U}$ is the full scattering operator that we utilize for brevity. Again, solving for the flux moments yields the half-iterate and external source equations of

$$\boldsymbol{\Phi}^{(k+1/2)} = \mathbf{DL}^{-1}\mathbf{MS}\boldsymbol{\Phi}^{(k)} + \mathbf{b}, \tag{4.79}$$

and

$$\mathbf{b} = \mathbf{DL}^{-1}\mathbf{Q}, \tag{4.80}$$

respectively. We note that Eq. (4.79) has an identical functional form as the half-iterate equation for the 1-group, isotropic scattering iteration operator of Eq. (4.36). Therefore, we can view Eq. (4.79) as the generalized iterative form for this Jacobi iterative procedure with an arbitrary number of energy groups and scattering moments, with Eq. (4.36) being a specialized case.

After each transport sweep, where all thermal groups perform 1 inner iteration, an energy collapsed, 1-group, diffusion acceleration step that is identical to TG and MTG is performed. The diffusion equation is still described by Eq. (4.61), with the only differences again arising with the spectral functions and residual. The spectral shape functions are calculated for each material by the following eigenproblem,

$$\mathbf{T}^{-1}\left(\mathbf{S_{L,0}} + \mathbf{S_{D,0}} + \mathbf{S_{U,0}}\right)\xi = \rho\xi, \tag{4.81}$$

which corresponds to the infinite medium iteration matrix of Eq. (4.78). For this MJA method, the residual corresponds to the full 0th moment scattering residual,

$$R_g^{(k+1/2)} = \sum_{g'=1}^{G} \sigma_{s,0}^{gg'} \left(\phi_{g',0}^{(k+1/2)} - \phi_{g',0}^{(k)} \right). \tag{4.82}$$

With this definition for the residual, the operator form for the coarse-grid error equation is given by,

$$\mathbf{A}\epsilon^{(k+1/2)} = \mathbf{XS} \left(\mathbf{\Phi}^{(k+1/2)} - \mathbf{\Phi}^{(k)} \right), \tag{4.83}$$

where the $\mathbf{A}$ and $\mathbf{X}$ operators are the same from the TG and MTG methods. Finally, using the half-iterate and coarse-grid error equations of Eqs. (4.79) and (4.83), respectively, the full-iterate equation becomes

$$\begin{aligned}
\mathbf{\Phi}^{(k+1)} &= \mathbf{FMS}\mathbf{\Phi}^{(k)} + \mathbf{PA}^{-1}\mathbf{XS} \left(\mathbf{\Phi}^{(k+1/2)} - \mathbf{\Phi}^{(k)} \right) + \mathbf{b} \\
&= \mathbf{FMS}\mathbf{\Phi}^{(k)} + \mathbf{PA}^{-1}\mathbf{XS} \left(\mathbf{FMS}\mathbf{\Phi}^{(k)} + \mathbf{b} - \mathbf{\Phi}^{(k)} \right) + \mathbf{b} \\
&= \mathbf{FMS}\mathbf{\Phi}^{(k)} + \mathbf{PA}^{-1}\mathbf{XS} \left(\mathbf{FMS} - \mathbf{I} \right) \mathbf{\Phi}^{(k)} + \left(\mathbf{PA}^{-1}\mathbf{XS} + \mathbf{I} \right) \mathbf{b} \\
&= \left[\mathbf{FMS} + \mathbf{PA}^{-1}\mathbf{XS} \left(\mathbf{FMS} - \mathbf{I} \right) \right] \mathbf{\Phi}^{(k)} + \left(\mathbf{PA}^{-1}\mathbf{XS} + \mathbf{I} \right) \mathbf{b}
\end{aligned}, \tag{4.84}$$

where $\mathbf{F} \equiv \mathbf{DL}^{-1}$ and we recall that $\mathbf{b} = \mathbf{DL}^{-1}\mathbf{Q}$. Equation (4.84) provides the iteration matrix for the MJA method and its eigenspectrum can be analyzed to provide insight into the method.

4.2.3.4 Summary of the Thermal Neutron Upscattering Acceleration Methods

In this work, we have defined three different methodologies to accelerate thermal neutron upscattering: the Two-Grid method, the Modified Two-Grid method, and the Multigroup Jacobi method. These methods are similar in that we perform a single coarsening step for each iteration where a spectrally-collapsed, 1-group diffusion equation is the low-order error operator. However, there are key differences in the

iterative procedures of the three methods. Both the TG and MTG methods perform a Gauss-Seidel procedure in energy where the thermal energy groups are solved sequentially. However, the inner iterations are not converged for any of the thermal groups with the MTG method. The Multigroup Jacobi method is different from the other two in that the thermal groups are iterated upon simultaneously.

For all three methods, we can define a general expression for the full phase-space solution at each iteration by the following,

$$\begin{aligned}\boldsymbol{\Phi}^{(k+1)} = &\left[\mathbf{FM\Sigma} + \mathbf{PA}^{-1}\mathbf{X\Sigma}\left(\mathbf{FM\Sigma} - \mathbf{I}\right)\right]\boldsymbol{\Phi}^{(k)} \\ &+ \left(\mathbf{PA}^{-1}\mathbf{X\Sigma} + \mathbf{I}\right)\mathbf{FDL}^{-1}\mathbf{Q}\end{aligned}, \tag{4.85}$$

where the differences lie in the $\mathbf{F}$ and $\mathbf{\Sigma}$ terms. These terms are given in Table 4.1, and we can clearly see that the differences in the schemes arise from the ordering of the scattering operators.

Table 4.1: Iteration terms for the different thermal upscatter acceleration methods.

Method	$\mathbf{F}$	$\mathbf{\Sigma}$
Two-Grid	$\left[\mathbf{I} - \mathbf{DL}^{-1}\mathbf{M}\left(\mathbf{S_L} + \mathbf{S_D}\right)\right]^{-1}\mathbf{DL}^{-1}$	$\mathbf{S_U}$
Modified Two-Grid	$\left[\mathbf{I} - \mathbf{DL}^{-1}\mathbf{MS_L}\right]^{-1}\mathbf{DL}^{-1}$	$\mathbf{S_D} + \mathbf{S_U}$
Multigroup Jacobi	$\mathbf{DL}^{-1}$	$\mathbf{S_L} + \mathbf{S_D} + \mathbf{S_U}$

4.3 Symmetric Interior Penalty Form of the Diffusion Equation

So far, we have presented several strategies in Section 4.2 in which DSA can be used to accelerate both within-group scattering and thermal neutron upscattering. We have also simply stated that our low-order operator will be the diffusion equation. However, we have not presented the exact form of the diffusion equation that we will

utilize. In Section 4.4, we present the full form of the modified interior penalty (MIP) form of the diffusion equation that we will use as our low-order operator for DSA calculations. However, we first present in this Section a more generalized version of the interior penalty form that we could use as a stand-alone solver for the standard diffusion equation: the symmetric interior penalty (SIP) form [97, 98, 99].

We begin our discussion of the SIP form by analyzing the standard form of the diffusion equation,

$$-\vec{\nabla} \cdot D(\vec{r})\vec{\nabla}\Phi(\vec{r}) + \sigma\Phi(\vec{r}) = Q(\vec{r}), \qquad \vec{r} \in \mathcal{D}, \tag{4.86}$$

with Dirichlet boundary conditions

$$\Phi(\vec{r}) = \Phi_0(\vec{r}), \qquad \vec{r} \in \partial\mathcal{D}^d, \tag{4.87}$$

Neumann boundary conditions

$$-D\partial_n\Phi(\vec{r}) = J_0(\vec{r}), \qquad \vec{r} \in \partial\mathcal{D}^n, \tag{4.88}$$

and Robin boundary conditions

$$\frac{1}{4}\Phi(\vec{r}) + \frac{D}{2}\partial_n\Phi(\vec{r}) = J^{inc}(\vec{r}), \qquad \vec{r} \in \partial\mathcal{D}^r. \tag{4.89}$$

We then convert Eq. (4.86) into its weak formulation by left-multiplying it with the test function, b, and apply Gauss' theorem to the Laplacian term,

$$\Big(D\vec{\nabla}b, \vec{\nabla}\Phi\Big)_{\mathcal{D}} - \Big\langle b, D\partial_n\Phi\Big\rangle_{\partial\mathcal{D}} + \Big(\sigma b, \Phi\Big)_{\mathcal{D}} = \Big(b, Q\Big)_{\mathcal{D}} \tag{4.90}$$

If we were to use the CFEM form of Eq. (4.90), then there would be no further formu-

lations required except on how to properly apply the boundary term: $\left\langle b, D\partial_n\Phi \right\rangle_{\partial\mathcal{D}}$. For the neumann and robin boundary conditions, this is straightforward since we simply need to insert the definitions of the outgoing currents, $D\partial_n\Phi$, of Eqs. (4.88) and (4.89) into Eq. (4.90). However, this still leaves the question of how to handle the dirichlet boundary conditions. Again, if we were to use CFEM, we could simply strongly enforce these boundary conditions [28].

Instead, we are choosing to utilize a discontinuous form of the diffusion equation. This means that we can employ the same DG finite elements used in the discretization of the transport operator in Section 2.6. With this in mind, we recast Eq. (4.90) to only contain the appropriate inner products for element K,

$$\left(D\vec{\nabla}b, \vec{\nabla}\Phi\right)_K - \left\langle b, D\partial_n\Phi \right\rangle_{\partial K} + \left(\sigma b, \Phi\right)_K = \left(b, Q\right)_K, \tag{4.91}$$

where we use the same notation for the volumetric and surface inner products as Section 2.6. We further decompose the boundary terms for element K into its respective interior ($\partial K \backslash \partial\mathcal{D}$), dirichlet ($\partial K \cap \partial\mathcal{D}^d$), neumann ($\partial K \cap \partial\mathcal{D}^n$), and robin ($\partial K \cap \partial\mathcal{D}^r$) components:

$$\begin{aligned}
&\left(D\vec{\nabla}b, \vec{\nabla}\Phi\right)_K + \left(\sigma b, \Phi\right)_K - \left\langle b, D\partial_n\Phi \right\rangle_{\partial K\backslash\partial\mathcal{D}} \\
&- \left\langle b, D\partial_n\Phi \right\rangle_{\partial K\cap\partial\mathcal{D}^d} - \left\langle b, D\partial_n\Phi \right\rangle_{\partial K\cap\partial\mathcal{D}^n} - \left\langle b, D\partial_n\Phi \right\rangle_{\partial K\cap\partial\mathcal{D}^r} \\
&= \left(b, Q\right)_K
\end{aligned} \tag{4.92}$$

We can then immediately utilize the definitions of the outgoing currents from Eqs. (4.88) and (4.89), and add the neumann and robin boundary contributions from element K:

$$\begin{aligned}\Big(D\vec{\nabla}b,\vec{\nabla}\Phi\Big)_K+\Big(\sigma b,\Phi\Big)_K\\ -\Big\langle b,D\partial_n\Phi\Big\rangle_{\partial K\setminus\partial\mathcal{D}}-\Big\langle b,D\partial_n\Phi\Big\rangle_{\partial K\cap\partial\mathcal{D}^d}+\frac{1}{2}\Big\langle b,\Phi\Big\rangle_{\partial K\cap\partial\mathcal{D}^r}\\ =\Big(b,Q\Big)_K-\Big\langle b,J_0\Big\rangle_{\partial K\cap\partial\mathcal{D}^n}+2\Big\langle b,J^{inc}\Big\rangle_{\partial K\cap\partial\mathcal{D}^r}.\end{aligned}\tag{4.93}$$

Unfortunately, we are now left with the burden of deciding what to do with element K's interior and dirichlet boundary surface terms. In conforming CFEM analysis, we would enforce at least C^0 continuity across the elements, thus not allowing any interelement jumps in the solution. Instead, with the choice of a DG formulation, we have a wide variability in how we wish to weakly express the discontinuous solution between two elements. This choice is extended to the dirichlet boundary conditions as well. Instead of simply strongly-enforcing the dirichlet conditions, we choose to weakly enforce them via a penalty method. The idea of penalty methods can be traced back to [100], where the weakly-enfoced dirichlet conditions now have the form,

$$\Phi(\vec{r})+\frac{1}{\kappa}D\partial_n\Phi(\vec{r})=\Phi_0(\vec{r}),\qquad \vec{r}\in\partial\mathcal{D}^d,\tag{4.94}$$

where κ is known as the penalty coefficient and $\kappa\gg 1$. It is clear that as κ becomes large, the solution, Φ, converges to the dirichlet value, Φ_0. In his work [101], Nitsche further proposed a consistent formulation with this penalty method via symmetrization. This led to the following weak formulation of the Laplacian term with dirichlet boundary conditions,

$$\begin{aligned}\Big(D\vec{\nabla}b,\vec{\nabla}\Phi\Big)_{\mathcal{D}}-\Big\langle D\partial_n b,\Phi\Big\rangle_{\partial\mathcal{D}^d}-\Big\langle b,D\partial_n\Phi\Big\rangle_{\partial\mathcal{D}^d}\\ +\Big\langle\kappa b,(\Phi-\Phi_0)\Big\rangle_{\partial\mathcal{D}^d}=-\Big\langle D\partial_n b,\Phi_0\Big\rangle_{\partial\mathcal{D}^d}\end{aligned}\tag{4.95}$$

where we dropped the reaction and forcing terms. Here the penalty coefficient, κ, has

the form $\kappa = \alpha/h$ and $\alpha > 1$ to ensure stability. Later, Arnold proposed extending the weak enforcement of the dirichlet boundaries by Nitsche onto all interior surfaces [102]. If the same symmetric consistency of Nitsche is utilized and we integrate over all mesh elements, then our weak formulation for the solution across all interior faces becomes,

$$\left\langle \kappa[\![b]\!], [\![\Phi]\!] \right\rangle_{E_h^i} + \left\langle [\![b]\!], \{\{D\partial_n\Phi\}\} \right\rangle_{E_h^i} + \left\langle \{\{D\partial_n b\}\}, [\![\Phi]\!] \right\rangle_{E_h^i} = 0, \tag{4.96}$$

where κ is again a penalizing coefficient to ensure stability. The mean value and the jump of the terms on a face from Eq. (4.96) are defined as

$$\{\{\Phi\}\} \equiv \frac{\Phi^+ + \Phi^-}{2} \qquad \text{and} \qquad [\![\Phi]\!] \equiv \Phi^+ - \Phi^-, \tag{4.97}$$

respectively. The directionality of the terms across a face can be defined in negative, Φ^-, and positive, Φ^+ directions by their trace:

$$\Phi^\pm \equiv \lim_{s \to 0^\pm} \Phi(\vec{r} + s\vec{n}), \tag{4.98}$$

where the face's unit normal direction, $\vec{n}$, has been arbitrarily chosen.

These weak formulations for the dirichlet boundary conditions and the interior faces can now be inserted into Eq. (4.93). From there, we sum the remaining inner products besides the interior face terms across all elements. With this completed, the SIP form of the diffusion equation can be succinctly written as

$$a^{SIP}(b, \Phi) = b^{SIP}(b), \tag{4.99}$$

with the following bilinear matrix:

$$
\begin{aligned}
a^{SIP}(b,\Phi) = \Big(D\vec{\nabla}b, \vec{\nabla}\Phi\Big)_{\mathcal{D}} + \Big(\sigma b, \Phi\Big)_{\mathcal{D}} + \frac{1}{2}\Big\langle b, \Phi\Big\rangle_{\partial\mathcal{D}^r} \\
+\Big\langle \kappa_e^{SIP}[\![b]\!], [\![\Phi]\!]\Big\rangle_{E_h^i} + \Big\langle [\![b]\!], \{\{D\partial_n\Phi\}\}\Big\rangle_{E_h^i} + \Big\langle \{\{D\partial_n b\}\}, [\![\Phi]\!]\Big\rangle_{E_h^i}, \\
+\Big\langle \kappa_e^{SIP} b, \Phi\Big\rangle_{\partial\mathcal{D}^d} - \Big\langle b, D\partial_n\Phi\Big\rangle_{\partial\mathcal{D}^d} - \Big\langle D\partial_n b, \Phi\Big\rangle_{\partial\mathcal{D}^d}
\end{aligned}
\tag{4.100}
$$

and with the following linear right-hand-side:

$$
\begin{aligned}
b^{SIP}(b) = \Big(b, Q\Big)_{\mathcal{D}} - \Big\langle b, J_0\Big\rangle_{\partial\mathcal{D}^n} + 2\Big\langle b, J^{inc}\Big\rangle_{\partial\mathcal{D}^r}. \\
+\Big\langle \kappa_e^{SIP} b, \Phi_0\Big\rangle_{\partial\mathcal{D}^d} - \Big\langle D\partial_n b, \Phi_0\Big\rangle_{\partial\mathcal{D}^d}
\end{aligned}
\tag{4.101}
$$

As previously stated, the general penalty term, κ needs to have sufficient positive measure to ensure stability. From previous investigations [92, 93, 94], we choose the SIP penalty coefficient to be face dependent with the following form,

$$
\kappa_f^{SIP} = \begin{cases} \frac{c}{2}\left(\frac{D^+}{h^+} + \frac{D^-}{h^-}\right) & f \in E_h^i \\ c\frac{D^-}{h^-} & f \in \partial\mathcal{D} \end{cases}, \tag{4.102}
$$

for interior, E_h^i, and boundary, $\partial\mathcal{D}$, faces respectively. In Eq. (4.102), $h^{\pm}$ is the orthogonal projection of the face, f, into the cells defined by the trace in Eq. (4.98). Turcksin and Ragusa, [94], defined $h^{\pm}$ for 2D polygons, whose definitions can be seen in Table 4.2. The orthogonal projection for both triangles and quadrilaterals can be explicitly defined from simple geometric relationships. However, for polygons with > 4 faces, there is no explicit geometric relationship to define the orthogonal projection. Instead, the polygon is approximated as regular, and the orthogonal projection is no longer face-dependent. For polygons with an even number of faces greater than 4, the orthogonal projection is twice the apothem, which is the line

segment between the polygon's center and the midpoint of each polygon's side. For odd number of faces greater than 4, the polygon's orthogonal projection becomes the sum of the apothem and the circumradius.

In a similar manner to the 2D orthogonal projections defined in Table 4.2, we define our choice for the extension of the orthogonal projections to 3D in Table 4.3. Like triangles and quadrilaterals in 2D, the orthogonal projections for tetrahedra and hexahedra can be explicitly defined from the volume equations for pyramids and parallelepipeds, respectively. For cells that are not tetrahedra or hexahedra, we introduce an approximation similar to 2D where we treat the cell as a regular polyhedron. In 3D there is no compact formula that can be given, unlike the definitions of the apothem and circumradius for 2D. Instead, we take the geometric limit of a polyhedra as the number of faces tends to infinity (a sphere). In this limiting case, the orthogonal projection simply becomes the sphere's diameter. We can then define the sphere's diameter with geometric information that would also be available to polyhedra by dividing a sphere's volume (the polyhedral volume) by its surface area (the sum of the areas of the polyhedral faces). While this leads to an approximation of the orthogonal projection for polyhedra that are not tetrahedra or hexahedra, it will provide appropriate geometric measure, especially for strictly convex polyhedra.

Table 4.2: Orthogonal projection, h, for different polygonal types: A_K is the area of cell K, L_f is the length of face f, and P_K is the perimeter of cell K.

Number of Vertices	3	4	> 4 and even	> 4 and odd
h	$2\frac{A_K}{L_f}$	$\frac{A_K}{L_f}$	$4\frac{A_K}{P_K}$	$2\frac{A_K}{P_K} + \sqrt{\frac{2A_K}{N_K \sin(\frac{2\pi}{N_K})}}$

Table 4.3: Orthogonal projection, h, for different polyhedral types: V_K is the volume of cell K, A_f is the area of face f, and SA_K is the surface area of cell K.

Number of Faces	4	6	otherwise
h	$3\frac{V_K}{A_f}$	$\frac{V_K}{A_f}$	$6\frac{V_K}{SA_K}$

4.3.1 Elementary Stiffness Matrices

In Eqs. (4.100) and (4.101), there are two additional elementary matrix types required other than those presented in Chapter 2. The first has the form of $\left(D\vec{\nabla}b, \vec{\nabla}\Phi\right)_K$ and is referred to as the stiffness matrix [28]. For a cell K, we define the stiffness matrix $\mathbf{S}$ as

$$\mathbf{S}_K = \int_K \vec{\nabla}\mathbf{b}_K \cdot \vec{\nabla}\mathbf{b}_K^T \, dr, \tag{4.103}$$

which has dimensionality (N_KxN_K). Just like the cell-wise elementary matrices presented in Chapter 2, it is possible that these integrals can be computed analytically, depending on the FEM basis functions used. However, for most of the 2D basis functions presented in Chapter 3, this cannot be done. Instead, we can again employ a numerical quadrature set, $\left\{\vec{x}_q, w_q^K\right\}_{q=1}^{N_q}$, for cell K, consisting of N_q points, $\vec{x}_q$, and weights, w_q^K. Using this quadrature set, the stiffness matrix can be calculated by the following

$$\mathbf{S}_K = \sum_{q=1}^{N_q} w_q^K \vec{\nabla}\mathbf{b}_K(\vec{x}_q) \cdot \vec{\nabla}\mathbf{b}_K^T(\vec{x}_q). \tag{4.104}$$

In this case, the local cell-wise stiffness matrix has the full matrix form:

$$\mathbf{S}_K = \begin{bmatrix} \int_K \vec{\nabla} b_1 \cdot \vec{\nabla} b_1 & \dots & \int_K \vec{\nabla} b_1 \cdot \vec{\nabla} b_j & \dots & \int_K \vec{\nabla} b_1 \cdot \vec{\nabla} b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_K \vec{\nabla} b_i \cdot \vec{\nabla} b_1 & \dots & \int_K \vec{\nabla} b_i \cdot \vec{\nabla} b_j & \dots & \int_K \vec{\nabla} b_i \cdot \vec{\nabla} b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_K \vec{\nabla} b_{N_K} \cdot \vec{\nabla} b_1 & \dots & \int_K \vec{\nabla} b_{N_K} \cdot \vec{\nabla} b_j & \dots & \int_K \vec{\nabla} b_{N_K} \cdot \vec{\nabla} b_{N_K} \end{bmatrix}, \quad (4.105)$$

where an individual matrix entry is of the form:

$$S_{i,j,K} = \int_K \vec{\nabla} b_i \cdot \vec{\nabla} b_j. \quad (4.106)$$

4.3.2 Elementary Surface Gradient Matrices

The second new elementary matrix corresponds to the integrals of the product of the basis functions with their gradients on a given surface. For a given face f, these matrices are of the general form: $\left\langle D\partial_n b, \Phi \right\rangle_f$. These terms are analagous to the cell streaming matrix but are computed on the cell boundary with dimensionality $(d-1)$. Analyzing a single face, f, in cell K, the analytical surface gradient matrix is of the form,

$$\mathbf{N}_{f,K} = \int_f \vec{n}(s) \cdot \vec{\nabla} \mathbf{b}_K \, \mathbf{b}_K^T \, ds, \quad (4.107)$$

where the surface normal, $\vec{n}$, is directed outwards from cell K and is allowed to vary along the cell face. From the analytical integral, we can see that these matrices have dimensionality, (N_KxN_K). We include the operation of the dot product between the outward normal and the basis function gradient for two reasons. First, it reduces the dimensionality of the matrices. Second, because the interior face terms in the SIP

bilinear form are independent of the orientation of the normal unit vector along the face, we do not need to perform any additional handling of the face normals. We can see that the bilinear form is independent of the face normal orientation by observing the following relations:

$$\begin{aligned} \Big\langle [\![u]\!], \{\{\partial_n v\}\} \Big\rangle_f &= -\Big\langle \{\{\vec{n}u\}\}, \{\{\vec{n}\partial_n v\}\} \Big\rangle_f \\ \Big\langle \{\{\partial_n u\}\}, [\![v]\!] \Big\rangle_f &= -\Big\langle \{\{\vec{n}\partial_n u\}\}, \{\{\vec{n}v\}\} \Big\rangle_f \end{aligned}. \tag{4.108}$$

If the direction of the normal is changed from $\vec{n}$ to $-\vec{n}$ in Eq. (4.108), we can see that these terms are not modified.

Similar to the surface matrix defined in Chapter 2, it is possible that the gradients of the basis functions cannot be integrated analytically along a cell face. Using the same face-wise quadrature notation as before, $\left\{\vec{x}_q, w_q^f\right\}_{q=1}^{N_q^f}$, we can numerically calculate the surface gradient matrix for face f along cell K:

$$\mathbf{N}_{f,K} = \sum_{q=1}^{N_q^f} w_q^f \vec{n}(\vec{x}_q)\, \vec{\nabla}\mathbf{b}_K(\vec{x}_q)\, \mathbf{b}_K^T(\vec{x}_q). \tag{4.109}$$

In this case, the local face-wise surface gradient matrix for face f has the full matrix form,

$$\mathbf{N}_{f,K} = \begin{bmatrix} \int_f \vec{n}\cdot\vec{\nabla}b_1\, b_1 & \dots & \int_f \vec{n}\cdot\vec{\nabla}b_1\, b_j & \dots & \int_f \vec{n}\cdot\vec{\nabla}b_1\, b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_f \vec{n}\cdot\vec{\nabla}b_i\, b_1 & \dots & \int_f \vec{n}\cdot\vec{\nabla}b_i\, b_j & \dots & \int_f \vec{n}\cdot\vec{\nabla}b_i\, b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_f \vec{n}\cdot\vec{\nabla}b_{N_K}\, b_1 & \dots & \int_f \vec{n}\cdot\vec{\nabla}b_{N_K}\, b_j & \dots & \int_f \vec{n}\cdot\vec{\nabla}b_{N_K}\, b_{N_K} \end{bmatrix}, \tag{4.110}$$

where an individual matrix entry is of the form:

$$N_{i,j,f,K} = \int_f \vec{n} \cdot \vec{\nabla} b_i \, b_j. \tag{4.111}$$

4.4 Modified Interior Penalty Form of the Diffusion Equation used for Diffusion Synthetic Acceleration Applications

In Section 4.3, we presented the SIP form of the diffusion equation that uses discontinuous Galerkin finite elements. This form can be used as a general solver for the diffusion equation that contains boundary conditions of the first three kinds: dirichlet, neumann, and robin. From the SIP form, we simply need to make some modifications to account for the boundary conditions that arise from the transport solution error as detailed in Section 4.2. The two types of transport conditions that we have considered in this work are dirichlet type conditions (incoming incident and vacuum) and neumann type conditions (reflecting). Since there is no iteration error associated with the incident boundary conditions, we can express the corresponding diffusion boundary condition as a zero dirichlet condition ($\delta\Phi_0 = 0$). Conversely, reflecting transport boundary conditions yield neumann diffusion boundary conditions that we express as δJ^{inc}. However, depending on the mesh and sweep ordering employed, we are not guaranteed to have this reflecting boundary condition error be strictly zero. If we seek to accelerate the k iterate, then the error in the incoming current, δJ^{inc}, is given by

$$\delta J^{inc} = \sum_{\vec{\Omega}_m \cdot \vec{n} > 0} w_m \left(\vec{\Omega}_m \cdot \vec{n} \right) \delta\Psi_m^{(k)}. \tag{4.112}$$

Using these modifications to the SIP diffusion form with the appropriate boundary conditions required to express the transport solution error, we write the MIP diffusion

form as

$$a^{MIP}(b,\delta\Phi) = b^{MIP}(b), \tag{4.113}$$

with the following bilinear matrix,

$$\begin{aligned} a^{MIP}(b,\delta\Phi) &= \left(D\vec{\nabla}b, \vec{\nabla}\delta\Phi\right)_{\mathcal{D}} + \left(\sigma b, \delta\Phi\right)_{\mathcal{D}} \\ +\left\langle \kappa_e^{MIP}[\![b]\!], [\![\delta\Phi]\!]\right\rangle_{E_h^i} &+ \left\langle [\![b]\!], \{\{D\partial_n\delta\Phi\}\}\right\rangle_{E_h^i} + \left\langle \{\{D\partial_n b\}\}, [\![\delta\Phi]\!]\right\rangle_{E_h^i}, \\ +\left\langle \kappa_e^{MIP} b, \delta\Phi\right\rangle_{\partial\mathcal{D}^d} &- \frac{1}{2}\left\langle b, D\partial_n\delta\Phi\right\rangle_{\partial\mathcal{D}^d} - \frac{1}{2}\left\langle D\partial_n b, \delta\Phi\right\rangle_{\partial\mathcal{D}^d} \end{aligned} \tag{4.114}$$

and with the following linear right-hand-side,

$$b^{MIP}(b) = \left(b, Q\right)_{\mathcal{D}} + \left\langle b, \delta J^{inc}\right\rangle_{\partial\mathcal{D}^{ref}}. \tag{4.115}$$

The MIP penalty coefficient also needs to be of a different form than the one used for SIP. From Eq. (4.102), we can see that as the orthogonal projection, h, grows large compared to the diffusion coefficient, D, the SIP penalty coefficient can become arbitrarily small. Wang and Ragusa demonstrated in [47] that if the penalty coefficient becomes too small, then MIP used as a DSA diffusion form becomes unstable. Instead, they limited κ^{MIP} to the maximum of either κ^{SIP} or 1/4. The value of 1/4 arises as the constant in the terms $\left\langle [\![b]\!], [\![\delta\Phi]\!]\right\rangle_{E_h^i}$ and $\left\langle b, \delta\Phi\right\rangle_{\partial\mathcal{D}^d}$ when the *diffusion conforming form* (DCF) of the diffusion equation is consistently derived from the DGFEM transport equation [92]. This new definition of κ_e^{MIP} can be succinctly written as

$$\kappa_e^{MIP} = \max\left(\kappa_e^{SIP}, \frac{1}{4}\right). \tag{4.116}$$

Just like the SIP penalty coefficient, this definition of κ_e^{MIP} ensures that the MIP bilinear form of Eq. (4.114) is SPD. We next describe the procedures used to efficiently invert the MIP system matrix for our work.

4.5 Solving the MIP Diffusion Problem

Equations (4.114) and (4.115) form the system matrix and right-hand-side, respectively, that we need to solve for a given DSA step. Just like the system of equations for the transport problem is too large to solve in a direct manner, we again employ an iterative scheme. Because the MIP bilinear form is SPD, we have natural recourse to use the simple Preconditioned Conjugate Gradient (PCG) method [34]. If the system of equations you are trying to solve for is SPD, then Conjugate Gradient (CG) is the most light-weight iterative method possible. It has a low memory footprint and is guaranteed to converge in $(N_{dof}/2)$ iterations, even if it is ill-conditioned. With a good preconditioner, the iteration counts with PCG can be reduced substantially further. If we define $\mathbf{A}$ as the MIP system matrix to be inverted and $\mathbf{b}$ as the right-hand-side vector, then the CG algorithm acts by minimizing the residual $\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x}$. This is accomplished by taking successive operations of matrix-vector multiplications on conjugate krylov vectors, $\mathbf{p}_k$. The PCG algorithm performs one additional step by solving the equation $\mathbf{M}\mathbf{z}_\mathbf{k} = \mathbf{r}_\mathbf{k}$ at each CG iteration, where $\mathbf{M}$ is some preconditioner. We provide the simple pseudocode for PCG in algorithm 2.

There are several choices of preconditioners that can be employed with PCG [34]. Depending on the structure and conditioning of the matrix to be inverted, some simple preconditioners can be effective. We can decompose the MIP system matrix, $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{L}^T$, into its strictly lower-triangular portion, $\mathbf{L}$, its strictly diagonal portion, $\mathbf{D}$, and its strictly lower-triangular portion, $\mathbf{L}^T$. The simplest preconditioner we could employ is Jacobi preconditioning, which is just the strictly diagonal portion

of the system matrix: $\mathbf{M} = \mathbf{D}$. This preconditioner is effective for diagonally-dominant matrices. The next preconditioner would be a simple Symmetric Successive Over-Relaxation (SSOR) method: $\mathbf{M}(\omega) = \frac{\omega}{2-\omega}\left(\frac{1}{\omega}\mathbf{D}+\mathbf{L}\right)\mathbf{D}^{-1}\left(\frac{1}{\omega}\mathbf{D}+\mathbf{L}\right)^{T}$. The PCG algorithm can be simplified with this preconditioner choice by the Eisenstat trick [103]. The final simple preconditioner that we will consider is Incomplete LU Factorization (ILU). Instead of simply decomposing the system matrix, we factorize it into a unit-lower triangular portion, $\tilde{\mathbf{L}}$, and an upper triangular portion, $\tilde{\mathbf{U}}$. These factorized matrices then form our preconditioner: $\mathbf{M} = \tilde{\mathbf{L}}\tilde{\mathbf{U}}$. From this factorization, the preconditioning step to solve $\mathbf{Mz} = \mathbf{r}$ is accomplished by first solving $\tilde{\mathbf{L}}\mathbf{y} = \mathbf{r}$, followed by $\tilde{\mathbf{U}}\mathbf{z} = \mathbf{y}$. While this leads to a second preconditioning step, the factorized matrices are easy to invert since they are triangular.

Besides Jacobi, SSOR and ILU preconditioning, we also seek to investigate multigrid methods to invert the MIP system matrix. Turcksin and Ragusa demonstrated in [94] that multigrid preconditioning methods can efficiently invert the MIP operator. Both Algebraic MultiGrid (AMG) [104, 105] and AGgregation-based algebraic MultiGrid (AGMG) [106, 107, 108, 109] were shown to be effective. We leave the general details of multigrid methods to the previously defined references.

4.6 Fourier Analysis

With the acceleration methodologies and diffusion discretization scheme described in detail, we now present the Fourier Analysis (FA) tool. Fourier Analysis is commonly used to analyze the performance characteristics of acceleration schemes for the transport equation. It is a powerful tool because it allows us to express the iteration error in terms of a Fourier series. Then, because of the orthogonality of the terms in the series, each Fourier mode can be analyzed independently. If these modes were not independent, then we would have no ability to simultaneously solve the infinite

Algorithm 2 PCG Algorithm

1: Compute $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$
2: **for** k=1,2,... **do**
3: Solve: $\mathbf{M}\mathbf{z}_{i-1} = \mathbf{r}_{i-1}$
4: $\rho_{i-1} = \mathbf{r}_{i-1}^T \mathbf{z}_{i-1}$
5: **if** $k = 1$ **then**
6: $\mathbf{p}_i = \mathbf{z}_{i-1}$
7: **else**
8: $\mathbf{p}_i = \mathbf{z}_{i-1} + \frac{\rho_{i-1}}{\rho_{i-2}} \mathbf{p}_{i-1}$
9: **end if**
10: $\mathbf{q}_i = \mathbf{A}\mathbf{p}_i$
11: $\alpha_i = \frac{\rho_{i-1}}{\mathbf{p}_i^T \mathbf{q}_i}$
12: $\mathbf{x}_i = \mathbf{x}_{i-1} + \alpha_i \mathbf{p}_i$
13: $\mathbf{r}_i = \mathbf{r}_{i-1} - \alpha_i \mathbf{q}_i$
14: **if** $\frac{||\mathbf{r}_i||}{||\mathbf{b}||} <$ tol **then**
15: Exit
16: **end if**
17: **end for**

spectrum of Fourier modes except for an extremely small set of idealized problems where analytical analysis can be performed.

For the Fourier Analysis of this work, we will only investigate geometries with tensor-based mesh cells (quadrilaterals and hexahedra). For completeness, some examples of 1D FA is provided in Appendix **??**. The FA domains are composed of regular Cartesian geometries defined by the global domain size and cell widths in each dimension. The domain size is given by (X, Y) in 2D and (X, Y, Z) in 3D. The cell layout is described by its N_x cell widths $(\Delta x_1, \Delta x_2, ..., \Delta x_{N_x})$ in the x-dimension, its N_y cell widths $(\Delta y_1, \Delta y_2, ..., \Delta y_{N_y})$ in the y-dimension, and its N_z cell widths $(\Delta z_1, \Delta z_2, ..., \Delta z_{N_z})$ in the z-dimension. This simple, yet structured, geometric layout is shown in Figure 4.1 for a 2D domain. The analogous 3D geometric layout can be formed by extruding this 2D grid. For FA, periodic boundary conditions are applied on the domain boundary. A graphical depiction of periodic boundary

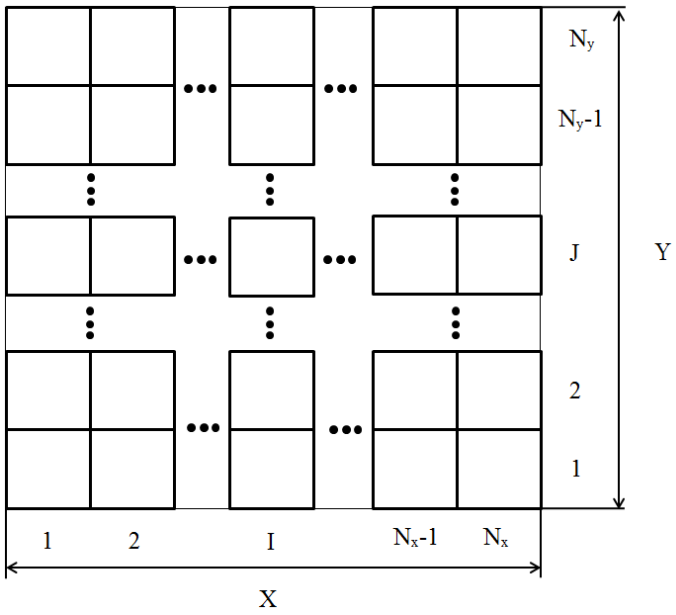


Figure 4.1: Fourier domain for 2D quadrilateral cells or an axial slice of 3D hexahedral cells in a regular grid.

conditions is given for a 2D, 2-cell domain in Figure 4.2. The periodic boundary conditions work by actually representing boundary faces as extensions to the interior faces. In other words, boundary faces are simply additional "interior" faces. Figure 4.2 shows this with the translocations of the neighboring cells to the exterior. For example, the periodic condition for the left face of cell 1 states that the degrees of freedom that we employ actually come from cell 2 (which we denote as 2'). Likewise, the periodic condition for the right face of cell 2 states that the degrees of freedom that we employ actually come from cell 1 (which we denote as 1'). The cells 1' and 2' can be viewed as virtual cells for use as an implementation detail.

Once the geometric domain is specified, we can then expand the solution vectors in terms of the domain's Fourier modes. The wave numbers of the Fourier modes, $\vec{\lambda}$, span the interval $[0, \frac{2\pi}{X}) \otimes [0, \frac{2\pi}{Y})$ in 2D and $[0, \frac{2\pi}{X}) \otimes [0, \frac{2\pi}{Y}) \otimes [0, \frac{2\pi}{Z})$ in 3D. The wave numbers have the definition of $\vec{\lambda} = [\lambda_x, \lambda_y]$ in 2D and $\vec{\lambda} = [\lambda_x, \lambda_y, \lambda_z]$ in 3D. Given a

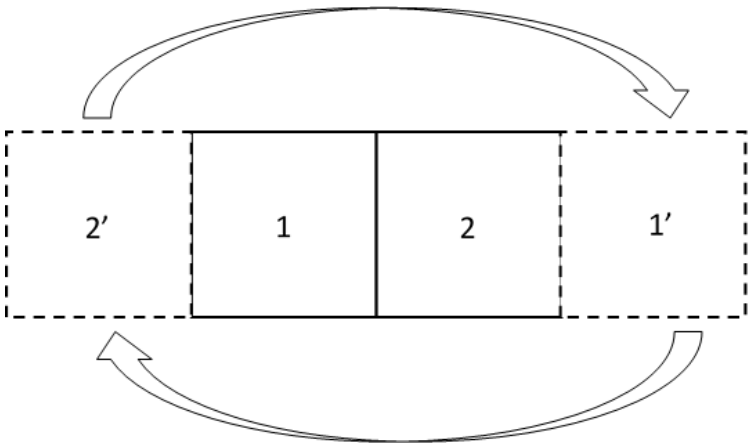


Figure 4.2: Representation of the periodic boundary conditions used for Fourier analysis for a 2-cell geometric layout.

particular wave number, $\vec{\lambda}$, the error modes for the angular flux can be given by the following Fourier ansatz,

$$\Psi^{(k)}(\vec{r},\vec{\Omega}) = \hat{\Psi}^{(k)}(\vec{\Omega})\, e^{i\vec{\lambda}\cdot\vec{r}}, \tag{4.117}$$

where $i = \sqrt{-1}$. Also, the error modes for the discretized angular flux along direction m and the flux moments can be given by

$$\Psi_m^{(k)}(\vec{r}) = \hat{\Psi}_m^{(k)}\, e^{i\vec{\lambda}\cdot\vec{r}}, \tag{4.118}$$

and

$$\Phi^{(k)}(\vec{r}) = \hat{\Phi}^{(k)} e^{i\vec{\lambda}\cdot\vec{r}}, \tag{4.119}$$

respectively. In Eqs. (4.117) - (4.119), we see that the term $e^{i\vec{\lambda}\cdot\vec{r}}$ acts to transform the spatial parameter into the complex space. For a collection of error modes (*e.g.,* the degrees of freedom of a mesh cell) corresponding to the spatial locations $(\vec{r}_1, \vec{r}_2, \vec{r}_3, \vec{r}_4)$,

we can express Eq. (4.119) by

$$\Phi^{(k)}(\vec{r}) = \mathbb{P}(\vec{\lambda})\hat{\Phi}^{(k)}, \tag{4.120}$$

where the diagonal phase matrix has the form:

$$\mathbb{P}(\vec{\lambda}) = \begin{bmatrix} e^{i\vec{\lambda}\cdot\vec{r}_1} & 0 & 0 & 0 \\ 0 & e^{i\vec{\lambda}\cdot\vec{r}_2} & 0 & 0 \\ 0 & 0 & e^{i\vec{\lambda}\cdot\vec{r}_3} & 0 \\ 0 & 0 & 0 & e^{i\vec{\lambda}\cdot\vec{r}_4} \end{bmatrix}. \tag{4.121}$$

One can also utilize an alternative version of Eq. (4.121) by using exponent laws and physical offsets $(\Delta x, \Delta y)$ from some starting base point.

Now that we have defined our periodic heterogeneous domain configuration and expressed the expansion of the flux solutions in terms of their Fourier modes, we can now detail how to compile the FA iteration matrix. Recall from Section 4.2 the different iteration matrices expressing our accelerated transport iterations. Equations (4.41), (4.67), (4.76), and (4.84) give the accelerated transport iteration matrices for the 1-group DSA scheme, the Two-Grid scheme, the Modified Two-Grid scheme, and Multigroup Jacobi Acceleration scheme, respectively. Each of these matrices can be expressed in the Fourier phase space by appropriate application of the phase matrices, $\mathbb{P}(\vec{\lambda})$. For example, let's consider the simple case of the 1-group DSA scheme. The FA iteration matrices for unaccelerated and accelerated Richardson iterations are given by

$$\mathbf{D}\tilde{\mathbf{L}}^{-1}\mathbf{M}\tilde{\mathbf{\Sigma}}, \tag{4.122}$$

and

$$\mathbf{D}\tilde{\mathbf{L}}^{-1}\mathbf{M}\tilde{\mathbf{\Sigma}} + \mathbf{P}\tilde{\mathbf{A}}^{-1}\mathbf{X}\tilde{\mathbf{\Sigma}}\left(\mathbf{D}\tilde{\mathbf{L}}^{-1}\mathbf{M}\tilde{\mathbf{\Sigma}} - \mathbf{I}\right), \tag{4.123}$$

respectively. In Eqs. (4.122) and (4.123), the $\tilde{\mathbf{L}}$, $\tilde{\mathbf{\Sigma}}$, and $\tilde{\mathbf{A}}$ operators have had phase transformations applied, which we denote with the overhead tilde, $\tilde{}$. During the assembly of the transport and diffusion operators, the phase transformations can be applied with the right-multiplication of the appropriate phase matrix on the corresponding elementary matrices. We note that when assembling the face coupling terms on the domain boundary, the base phase of the cell on the other side of the periodic boundary is not used. Instead, the phase corresponding to the virtual cell is used. For example, let us analyze the left face of cell 1 in Figure 4.2. When applying the periodic condition, we use the degrees of freedom corresponding to cell 2, but use the phase matrix corresponding to cell 2'. Finally, we note that the FA iteration matrices for the thermal neutron upscattering acceleration methods are computed in an identical manner.

Once the FA iteration matrix is assembled for a given wave number, $\vec{\lambda}$, we compute all of its corresponding eigenvalues. The largest of these eigenvalues is the spectral radius for the given wave number. If we compute the spectral radii for all the possible wave numbers (*e.g.,* $[0, \frac{2\pi}{X}) \otimes [0, \frac{2\pi}{Y})$ in 2D) for a given problem configuration, then the maximum corresponds to the global spectral radius for the problem. For Richardson iteration in the asymptotic regime, we will converge to our transport solution more rapidly with a smaller spectral radius that is strictly less than 1. Likewise, our scheme would be unstable if the spectral radius is larger than 1.

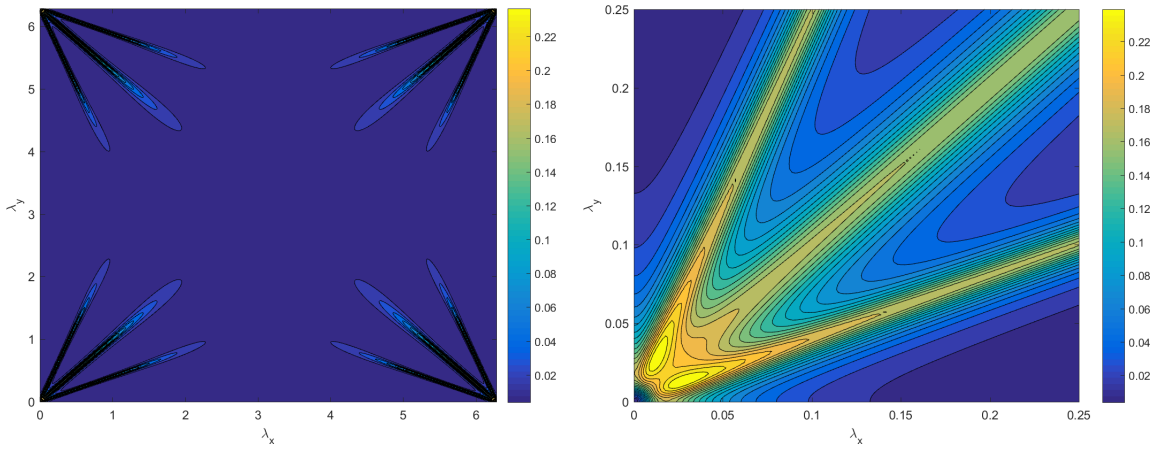
For this work, all fourier analysis was performed in MATLAB. All the eigenmodes corresponding to a fourier wave number for a given iteration matrix can be easily computed with MATLAB's built-in *eig* function. The maximum eigenvalue is found

over the wave number space by use of the Nelder-Mead simplex algorithm [110]. We stress that some sort of minimization algorithm must be employed because some problem configurations can have extremely narrow local maxima. These difficult to find local maxima can correspond to the global maximum which is our desired spectral radius that we wish to compute.

We illustrate the necessity for a minimization algorithm in Figure 4.3. We have modeled a single 2D square mesh cell with dimensions $X = 1$ and $Y = 1$. The total cross section, σ_t, is set to 10^{-2} and the scattering ratio, c, set to 0.9999. We use the $S4$ level-symmetric quadrature set. The left image of Figure 4.3 has the 2D fourier wave number span the full domain space of $[\lambda_x, \lambda_y] = [0, 2\pi]^2$. The right image zooms in on the wave number ranging: $[\lambda_x, \lambda_y] = [0, 1/4]^2$. From the right image, we can see two extremely narrow local maxima. We can qualitatively ascertain that these local maxima would be difficult to find if we had simply laid a grid of wave number points over $[0, 2\pi]^2$. It would require a very fine wave number grid that is both expensive to compute and not guaranteed to locate the maximum. Chang and Adams presented an even more extreme example of a difficult to find global maximum using transport synthetic acceleration [111].

4.7 Numerical Results

We now present all theoretical and applied results pertaining to the described DSA schemes. First, we demonstrate that the 2D linear and quadratic basis functions defined in Chapter 3 can capture the thick diffusion limit in Section 4.7.1. Next, we provide results in Section 4.7.2 on the efficacy of the SIP form as a diffusion solver. Section 4.7.3 provides all theoretical and numerical results pertaining the 1-group DSA scheme. Then, Section 4.7.4 demonstrates the scalability of the MIP diffusion form on massively-parallel computer architectures. Finally, Section 4.7.5 gives the



(a) (b)

Figure 4.3: 2D fourier wave form for MIP, with 1 square cell with 1e-2 mfp, with the PWL coordinates, with the LS4 quadrature, and where the wave numbers range from: (a) $[\lambda_x, \lambda_y] = [0, 2\pi]^2$ and (b) $[\lambda_x, \lambda_y] = [0, 1/4]^2$.

results pertaining to the analysis performed for the thermal neutron upscattering acceleration methods.

4.7.1 Transport Solutions in the Thick Diffusive Limit

We present our first numerical example by demonstrating that the various 2D polygonal finite element basis functions provided in Chapter 3 satisfy the thick diffusion limit. We investigate the transport problem with isotropic scattering and an isotropic distributed source given by the following:

$$\vec{\Omega} \cdot \vec{\nabla}\Psi + \sigma_t \Psi = \frac{\sigma_s}{4\pi}\Phi + \frac{Q_0}{4\pi}. \tag{4.124}$$

As the transport problem becomes more optically thick, the total mean free paths of the neutroncs increases. In the thick diffusion limit, the domain mean free path approaches infinity. If we fix the physical dimensions of the problem to some finite value, then we can scale the cross sections and the source term to reflect the properties

of the thick diffusion limit. In the thick diffusion limit the total and scattering cross sections tend to infinity while the absorption cross section and the source term tend to zero. If we introduce a scaling parameter, ϵ, then we can write the scaled terms as,

$$\begin{aligned} \sigma_t &\rightarrow \frac{\sigma_t}{\epsilon} \\ \sigma_a &\rightarrow \epsilon\sigma_t \\ \sigma_s &\rightarrow \left(\frac{1}{\epsilon} - \epsilon\right)\sigma_t \\ \frac{Q_0}{4\pi} &\rightarrow \epsilon\frac{Q_0}{4\pi} \end{aligned}. \tag{4.125}$$

Inserting these scaled cross sections and source term into Eq. (4.124) leads to the following scaled transport equation:

$$\vec{\Omega}\cdot\vec{\nabla}\Psi + \frac{\sigma_t}{\epsilon}\Psi = \sigma_t\left(\frac{1}{\epsilon} - \epsilon\right)\frac{\Phi}{4\pi} + \epsilon\frac{Q_0}{4\pi}. \tag{4.126}$$

We can also use the scaled terms of Eq. (4.125) to give the corresponding scaled diffusion equation. If we take the 0th and 1st moments of Eq. (4.126) and assume that the P1 terms obey Fick's Law, then the scaled diffusion equation is

$$\epsilon\vec{\nabla}\cdot\frac{1}{3\sigma_t}\vec{\nabla}\Phi + \epsilon\sigma_t\Phi = \epsilon Q_0. \tag{4.127}$$

One can immediately see that Eq. (4.127) does not truly scale because there is an ϵ for each term. This is the desired behavior we want to see from the diffusion equation because, as $\epsilon \rightarrow 0$, the transport equation will converge to its diffusive limit and satisfy a diffusion equation.

For the sake of analysis, we seek to simplify Eqs. (4.126) and (4.127) by normalization. We choose to set σ_t and Q_0 to unity which gives the final transport and

diffusion equations as

$$\vec{\Omega} \cdot \vec{\nabla}\Psi + \frac{1}{\epsilon}\Psi = \left(\frac{1}{\epsilon} - \epsilon\right)\frac{\Phi}{4\pi} + \frac{\epsilon}{4\pi}, \tag{4.128}$$

and

$$\frac{\epsilon}{3}\nabla^2\Phi + \epsilon\Phi = \epsilon, \tag{4.129}$$

respectively.

4.7.2 SIP used as a Diffusion Solver

We first wish to know how an interior penalty form of the diffusion equation will perform on unstructured polyhedral grids. The SIP diffusion formulation has previously been analyzed for use as a DFEM diffusion solver for unstructured 2D polygonal grids [98]. The MIP DSA form has also been successfully utilized for unstructured 2D polygonal grids [94]. Here, we first seek to extend the efficacy of the SIP form as a diffusion solver on polyhedral mesh cellls. We will do this by analyzing the following two problem types:

1. An exactly-linear solution to determine if linear basis functions will span the solution space;

2. The Method of Manufactured Solutions (MMS) to test basis function convergence rates.

The polyhedral mesh types that we employ for this analysis are presented in Section 4.7.2.1. Next, the exactly-linear solution analysis is performed in Section 4.7.2.2. Finally, the MMS analysis to confirm the second order convergence rates of the 3D PWL basis functions is presented in Section 4.7.2.3.

4.7.2.1 Geometry Specification for the SIP Problems

To analyze the SIP diffusion form on 3D polyhedral grids, we will utilize many of the 2D polygonal grids that were used for previous analysis in Chapter 3. For this analysis we will resuse the cartesian, ordered-triangular, polygonal sinusoidal, and polygonal-z meshes. We will also use purely-randomized polygonal grids formed from Voronoi mesh generation as outlined in Section **??**. To form the needed 3D polyhedral grids, we will take these 2D grids and simply extrude the meshes in the z-dimension.

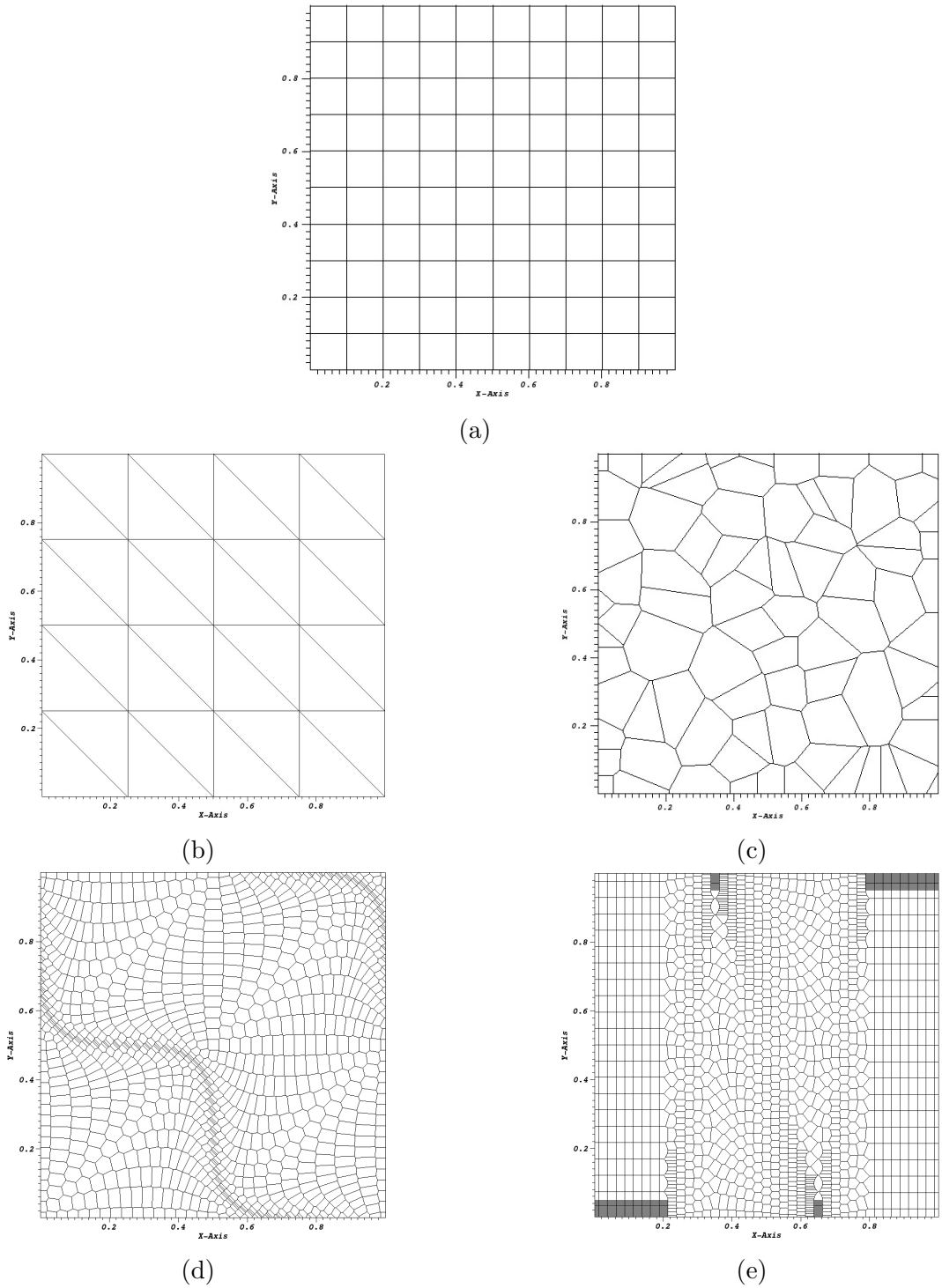
Figure 4.4 provides the 2D mesh types that will be utilized in this analysis. Figure 4.5 then provides the same meshes after they have been extruded in the z-dimension. We note that it will not simply be these exact grids that are employed. For the MMS analysis in Section 4.7.2.3, various refinements of these meshes will be utilized.

4.7.2.2 Exactly-Linear Solution

We first test SIP by enforcing a sytem that yields an exactly-linear diffusion solution. Linear finite elements should then theoretically fully-span the solution space. We can achieve this mathematically by setting the cross-section and right-hand-source terms to zero, $\sigma = Q = 0$. Robin boundary conditions are imposed on opposite faces in 1 dimension, with homogeneous Neumann boundary conditions on all other faces. If the Robin boundaries are chosen in the y-direction, with $y \in (0, L)$, then the analytical solution for the problem will be

$$\Phi(x, y, z) = \frac{4J^{inc}}{L + 4D} \left(L + 2D - y\right), \tag{4.130}$$

with the following boundary conditions in the y-direction:



(a)

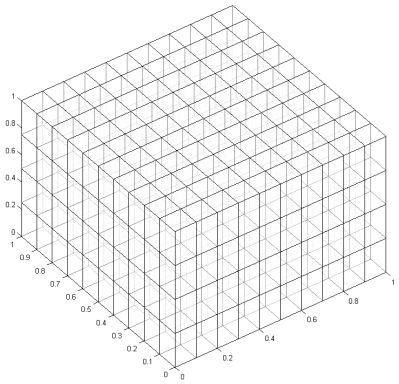
(b)

(c)

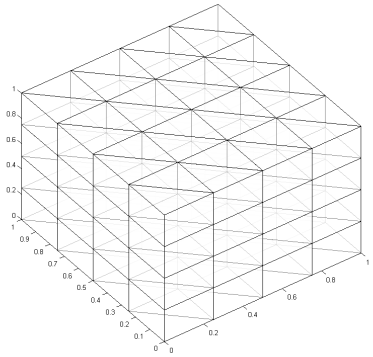
(d)

(e)

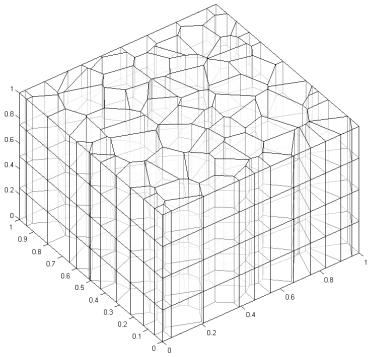
Figure 4.4: 2D grids to be extruded of the different mesh types: (a) cartesian, (b) ordered triangles, (c) random polygons, (d) sinusoidal polygons, and (e) polygonal z-mesh.



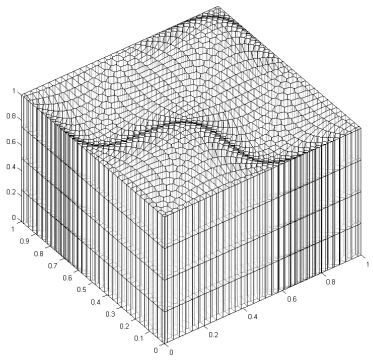
(a)



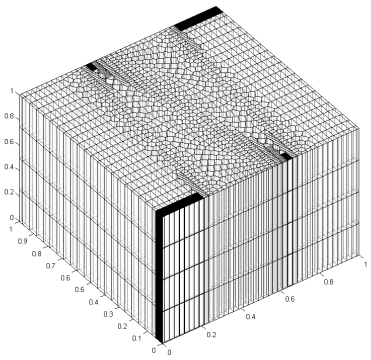
(b)



(c)



(d)



(e)

Figure 4.5: Extrusion of the different mesh types: (a) cartesian, (b) ordered triangles, (c) random polygons, (d) sinusoidal polygons, and (e) polygonal z-mesh.

$$\begin{aligned} \Phi - 2D\partial_y\Phi =& 4J^{inc}, & \forall(x,z), y = 0 \\ \Phi + 2D\partial_y\Phi =& 0, & \forall(x,z), y = L \end{aligned} . \tag{4.131}$$

In Eqs. (4.130) and (4.131), D is once again the standard diffusion coefficient and J^{inc} is the incoming current on the $y = 0$ boundary. For this analysis, we choose D to be 2, J^{inc} to be 9, and L to be 1. Using Eq. (4.130), our solution has a value of 20 at $y = 0$ and linearly-decreases to 16 at $y = 1$.

Using the 3D PWL basis functions, which were previously used for SIP on 2D polygons [98], the linear solutions are presented in Figure 4.6 at the midplane axial slice of $z = L/2$. We can see from the exact contour lines in the plots that SIP can capture an exactly-linear solution even on some highly distorted polyhedral grids.

4.7.2.3 Method of Manufactured Solutions

We next test to see if the SIP diffusion form can capture the appropriate second order convergencerates with the 3D PWL basis functions. These basis functions were previously shown to capture the appropriate convergence rates for the DGFEM transport equation on 3D hexahedral grids [70].

$$\Phi^{quad}(x,y,z) = x(1-x)y(1-y)z(1-z) \tag{4.132}$$

$$\Phi^{gauss}(x,y,z) = \Phi^{quad}(x,y,z)\exp(-(\vec{r}-\vec{r}_0)\cdot(\vec{r}-\vec{r}_0)^T) \tag{4.133}$$

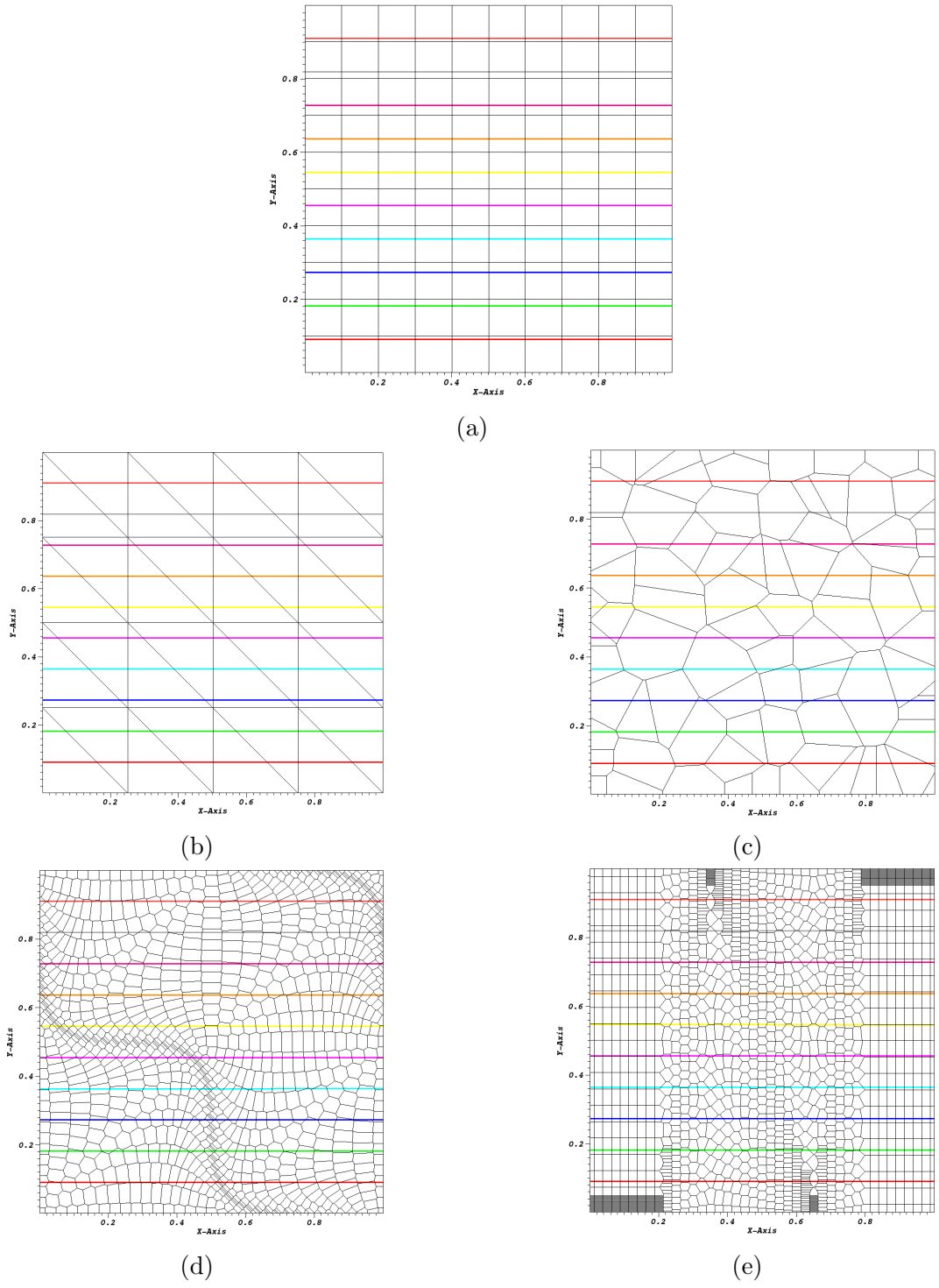


Figure 4.6: Axial slice showing the contours for the linear solution of the different mesh types: (a) cartesian, (b) ordered triangles, (c) random polygons, (d) sinusoidal polygons, and (e) polygonal z-mesh.

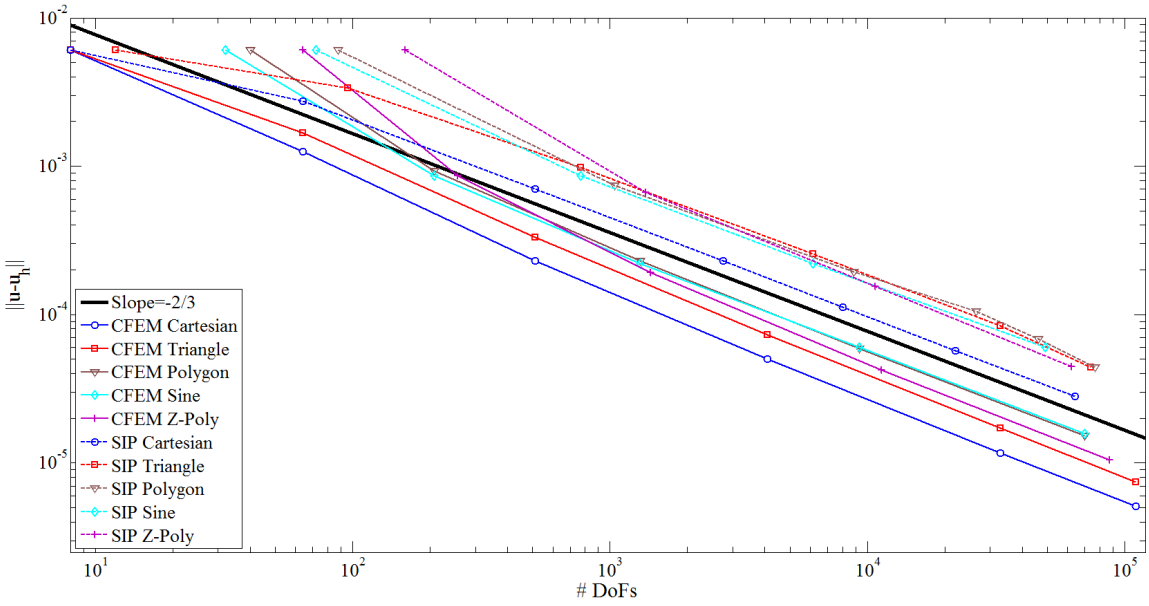


Figure 4.7: blah

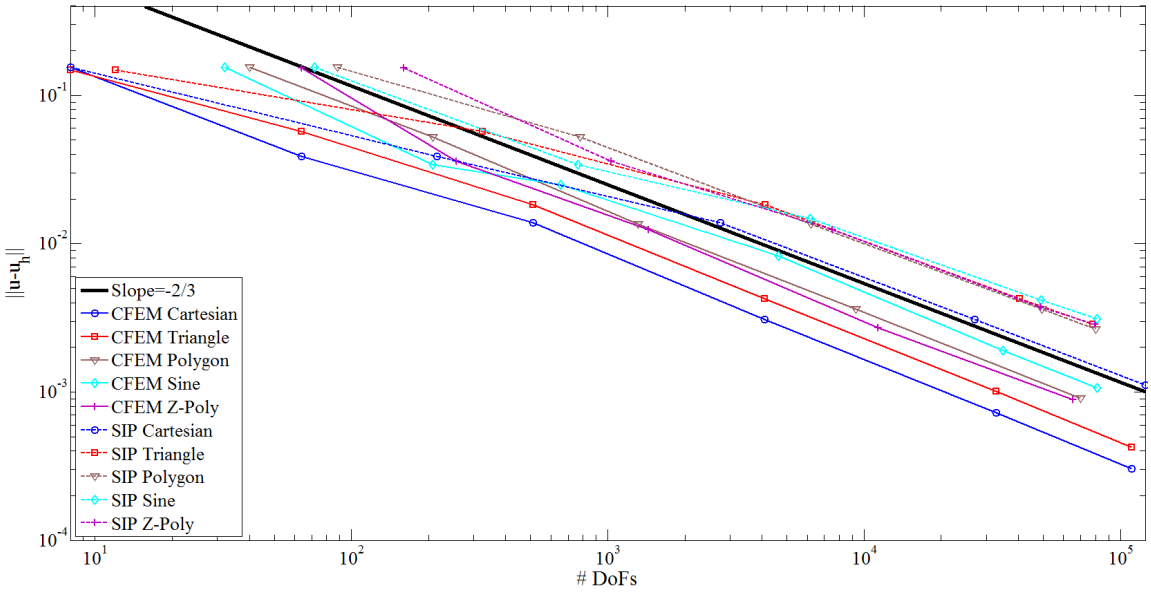


Figure 4.8: blah

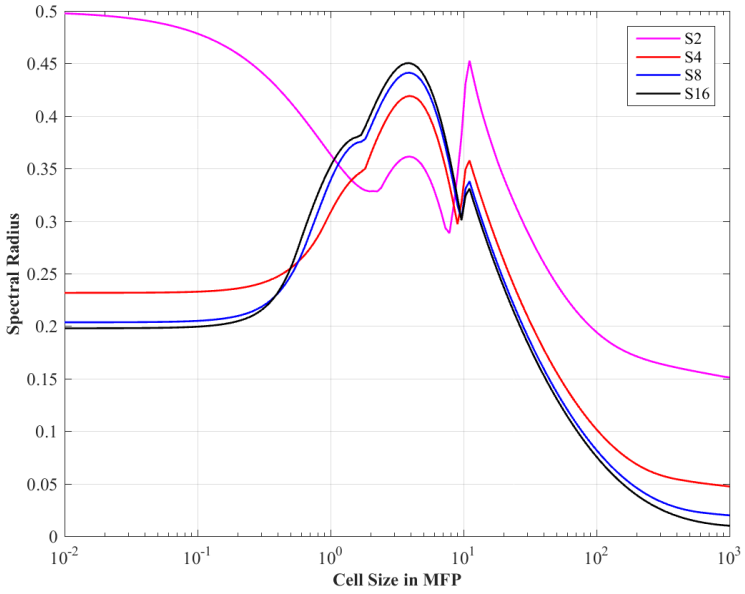


Figure 4.9: Fourier analysis of the 2D MIP form with $c = 4$ and using the linear Wachspress coordinates for the homogeneous infinite medium case as a function of the mesh optical thickness.

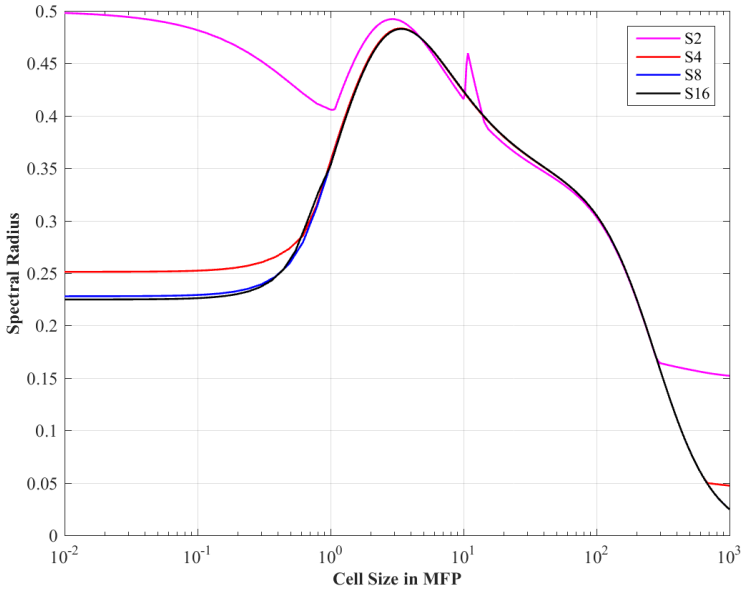


Figure 4.10: Fourier analysis of the 2D MIP form with $c = 4$ and using the linear PWL coordinates for the homogeneous infinite medium case as a function of the mesh optical thickness.

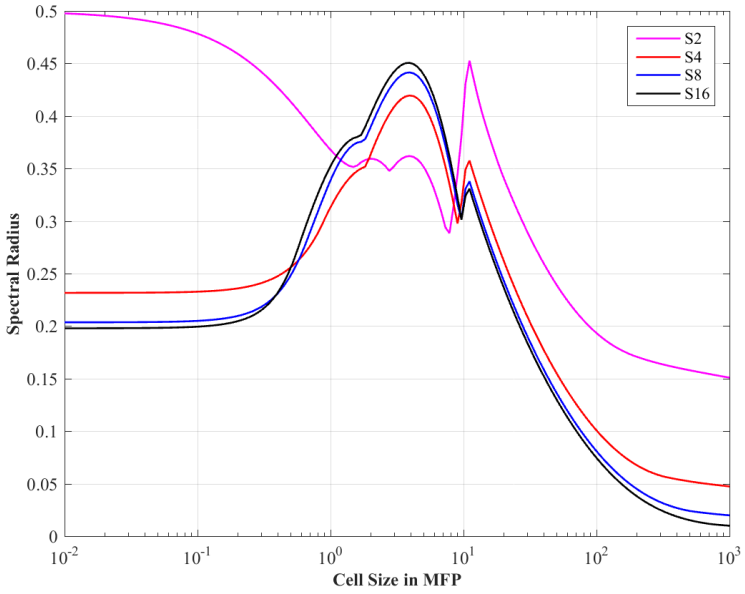


Figure 4.11: Fourier analysis of the 2D MIP form with $c = 4$ and using the linear mean value coordinates for the homogeneous infinite medium case as a function of the mesh optical thickness.

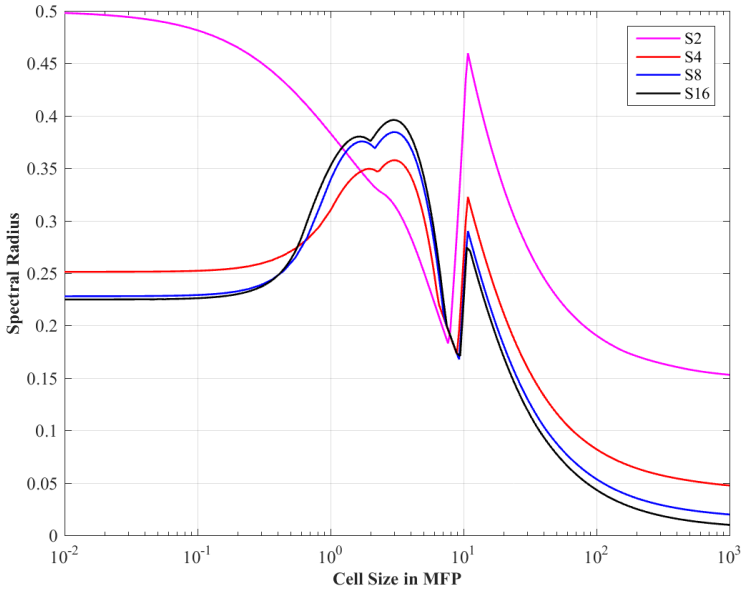
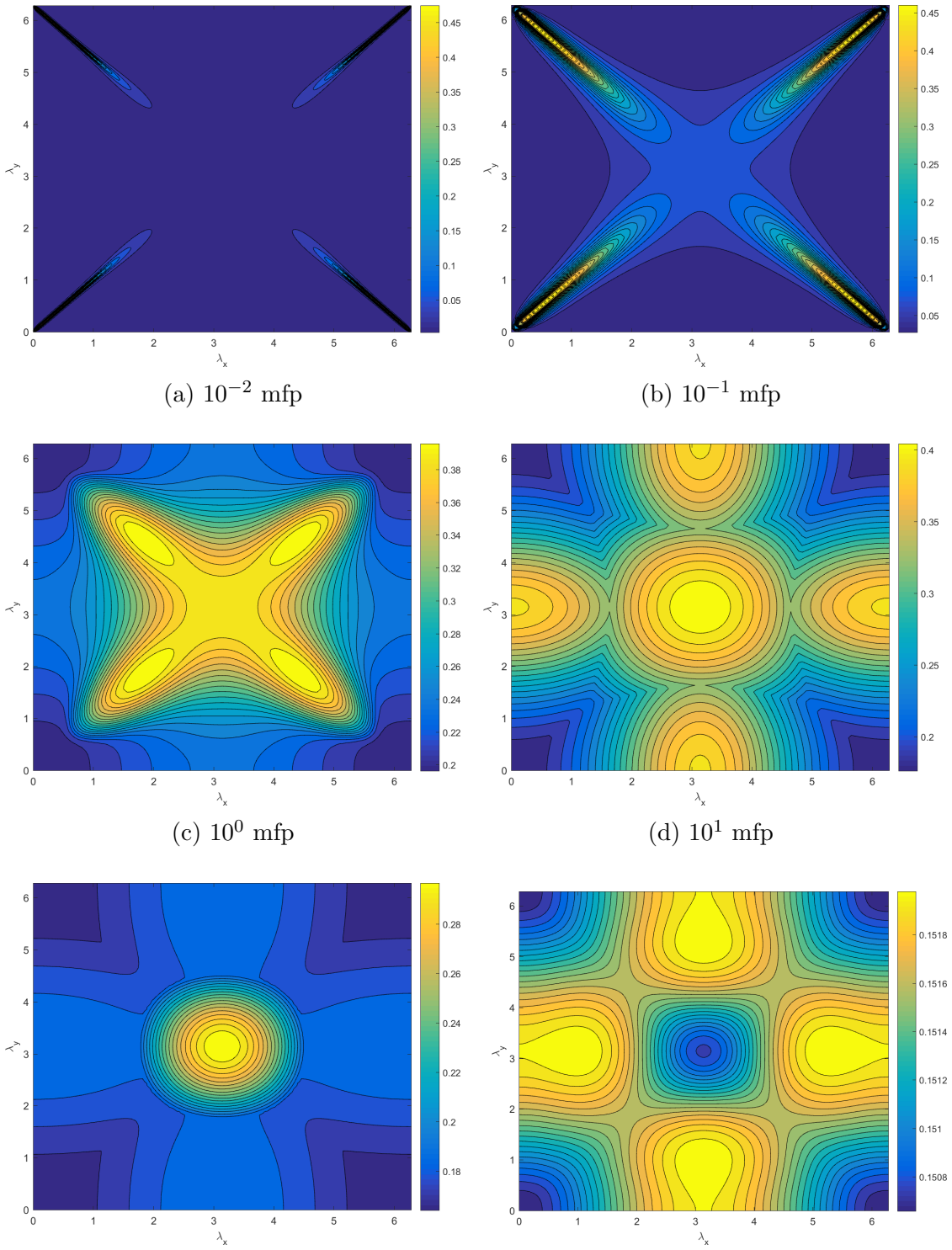


Figure 4.12: Fourier analysis of the 2D MIP form with $c = 4$ and using the linear maximum entropy coordinates for the homogeneous infinite medium case as a function of the mesh optical thickness.



(a) 10^{-2} mfp

(b) 10^{-1} mfp

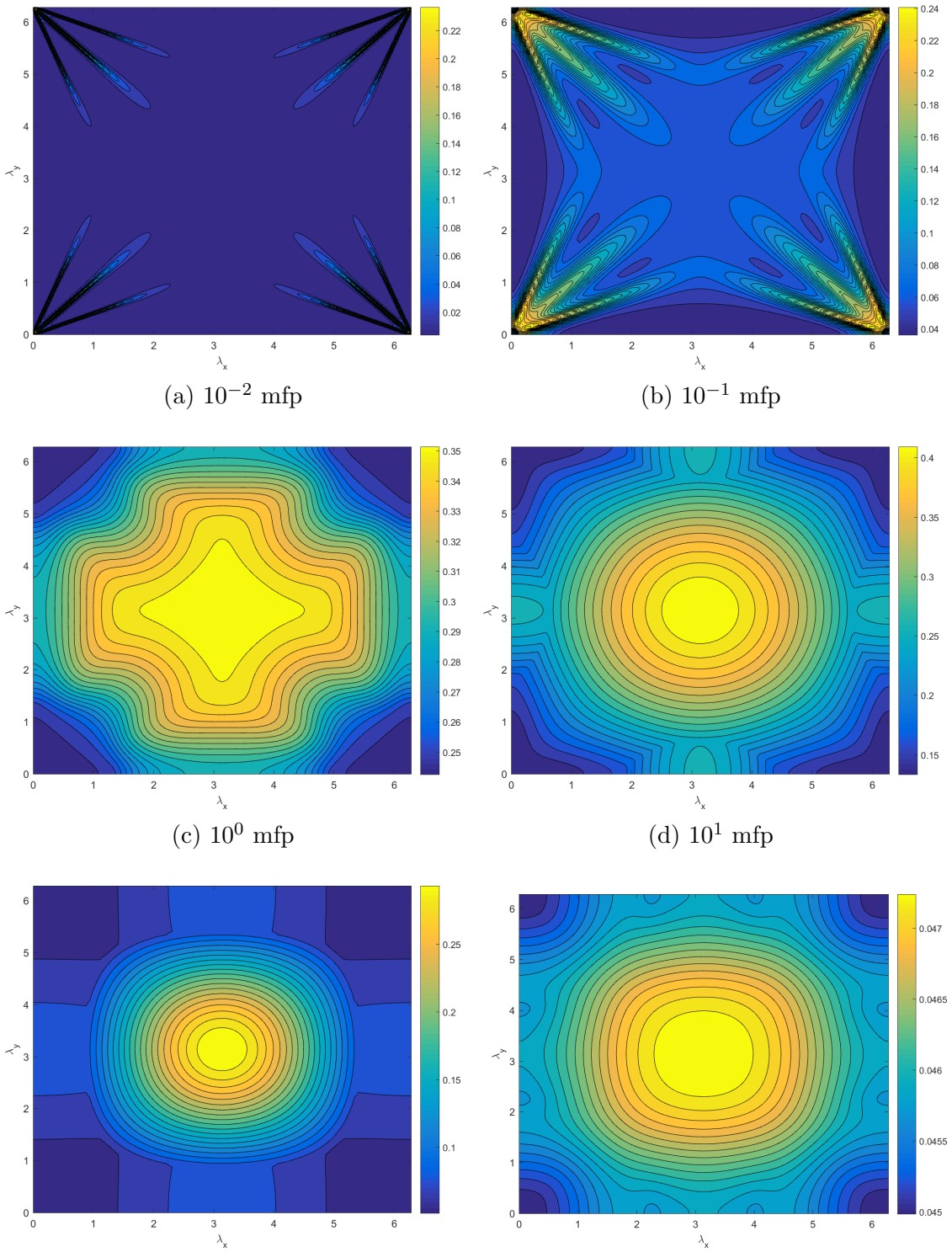
(c) 10^{0} mfp

(d) 10^{1} mfp

(e) 10^{2} mfp

(f) 10^{3} mfp

Figure 4.13: Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL coordinates and LS2 quadrature.



(a) 10^{-2} mfp

(b) 10^{-1} mfp

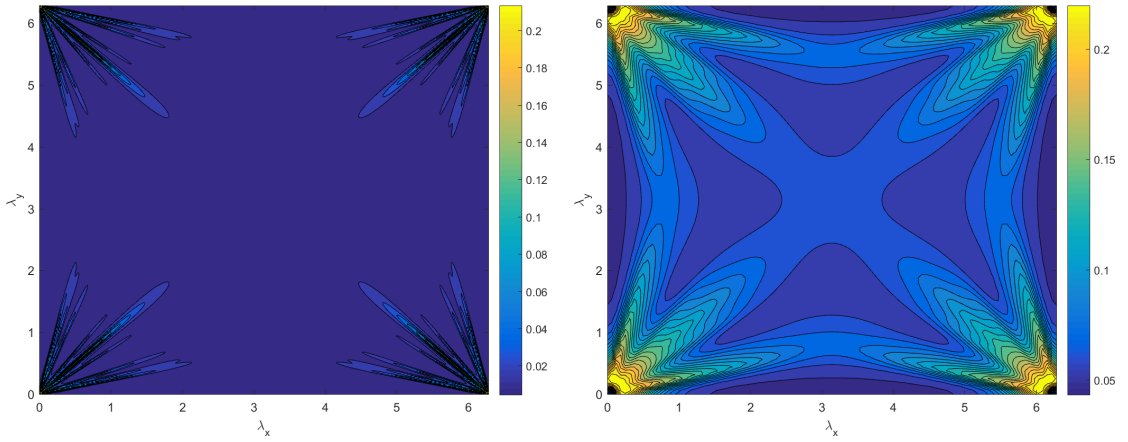
(c) 10^{0} mfp

(d) 10^{1} mfp

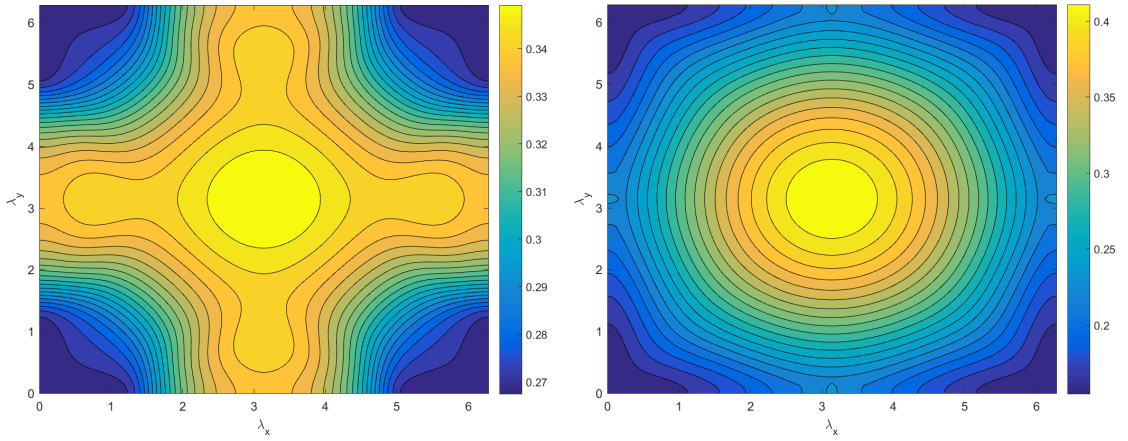
(e) 10^{2} mfp

(f) 10^{3} mfp

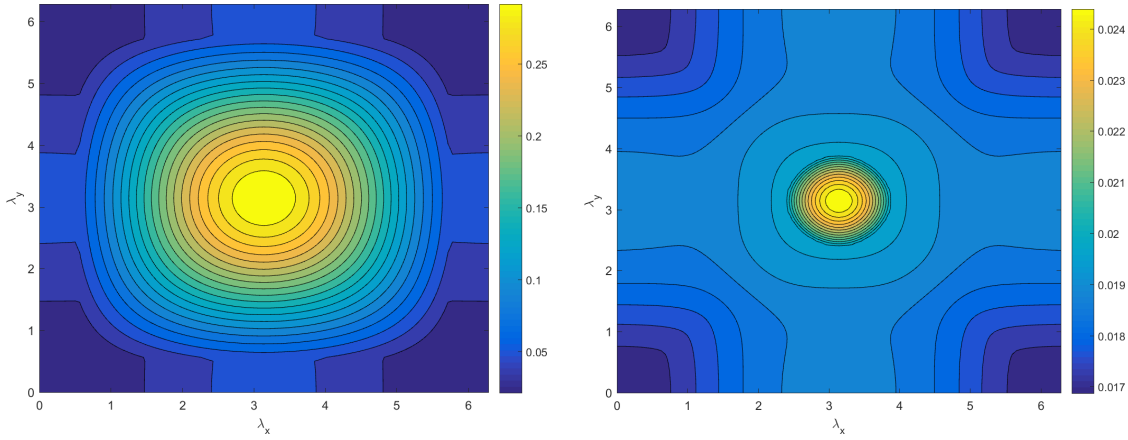
Figure 4.14: Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL coordinates and LS4 quadrature.



(a) 10^{-2} mfp (b) 10^{-1} mfp



(c) 10^{0} mfp (d) 10^{1} mfp



(e) 10^{2} mfp (f) 10^{3} mfp

Figure 4.15: Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL coordinates and LS8 quadrature.

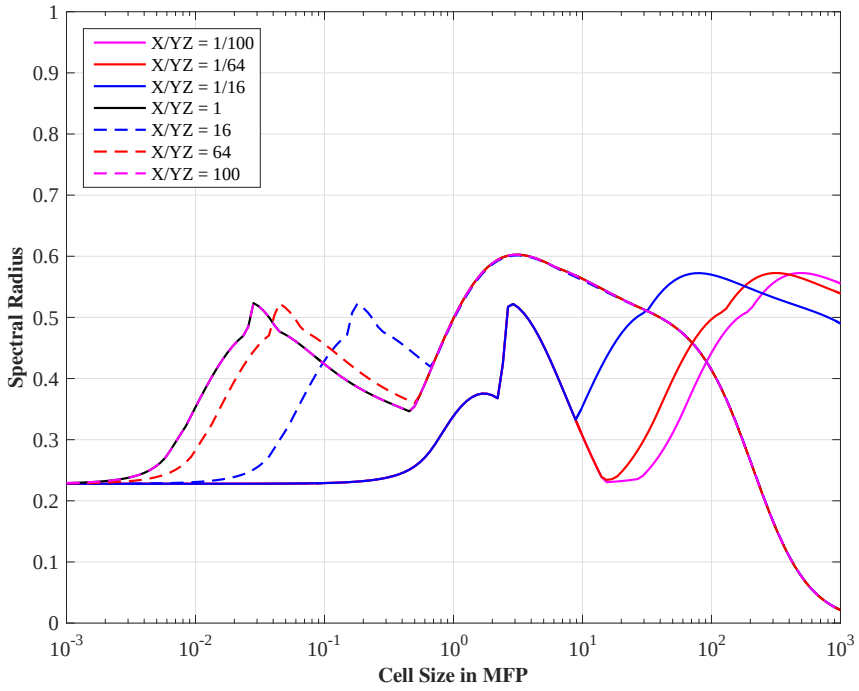


Figure 4.16: Fourier spectral radii for MIP with 3D PWL coordinates with different aspect ratios and $c = 1$.

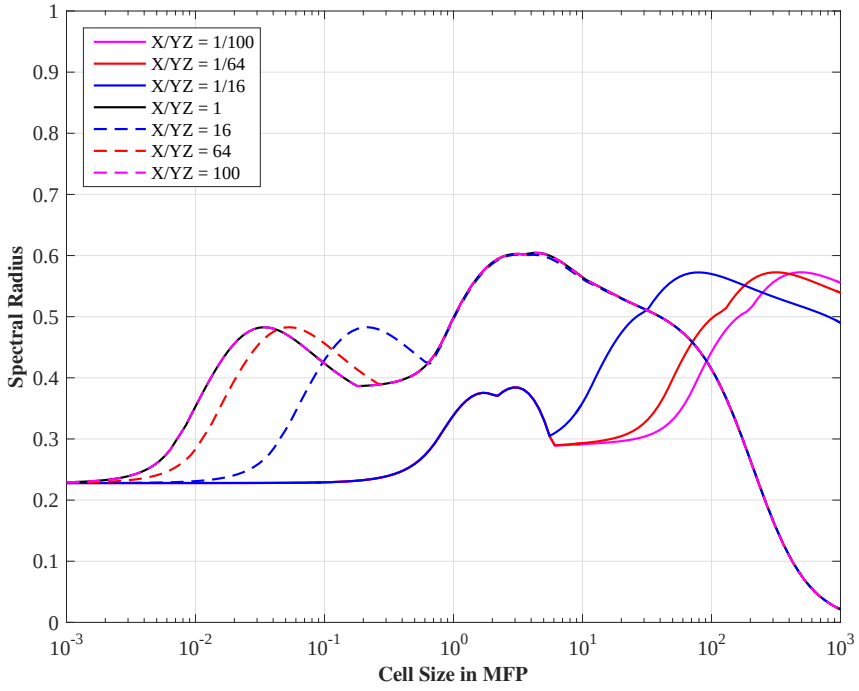


Figure 4.17: Fourier spectral radii for MIP with 3D PWL coordinates with different aspect ratios and $c = 4$.

4.7.3 1 Group DSA Analysis

4.7.3.1 2D Homogeneous Medium Case

4.7.3.2 3D Homogeneous Medium Case

4.7.3.3 Periodic Horizontal Interface Problem

When DSA is applied to multidimensional problems (2D and 3D), the preconditioning of the transport operators can degrade in the presence of heterogeneous configurations with large material discontinuities [112]. The Periodic Horizontal Interface (PHI) problem is considered a litmus test for heterogeneous DSA techniques. This problem consists of horizontal strips of alternating optically thick and optically thin materials that are 1 cell in depth. We define σ_1 as the optically thick total cross section and σ_2 as the optically thin total cross section. We then define a tuning parameter, σ, and allow the region cross sections to have the following form:

$$\begin{aligned} \sigma_1 &= \sigma \\ \sigma_2 &= \frac{1}{\sigma} \end{aligned}. \tag{4.134}$$

We can see that increasing the value of σ will increase the magnitude of difference between the two region cross sections. Therefore, as σ grows large, the material discontinuities will grow which could potentially reduce the performance of our DSA scheme.

From the analysis presented in Section 4.7.3.1, we showed that our different 2D linear and quadratic basis functions were robust and stable, even for mesh cells with large aspect ratios. For this PHI analysis, we concentrate on analyzing just the linear PWL coordinates as our basis functions. Just like before, we will also examine different level-symmetric quadrature sets and their effects problems with

varying optical thickness. The optical thickness and diffusivity of the problem is increased by varying both σ and the scattering ratio, c. This study was conducted with the sequence, $\sigma = [10, 20, 40, 80, 160, 320, 640]$, and with following scattering ratios: $c = [0.9, 0.99, 0.999, 0.9999, 0.99999, 0.999999]$.

The full results of this PHI analysis are presented in Tables 4.4 - 4.7 for the LS2, LS4, LS8, and LS16 quadratures, respectively. From these tables, we can see that MIP DSA loses its effectiveness as the heterogeneity and overall diffusivity of the problem increases. The theoretical spectral radii are greater than any of the values presented for the homogeneous case from Figure 4.10. This result is true even for the example with the smallest heterogeneity and smallest diffusivity ($\sigma = 10$ and $c = 0.9$). We can gain some more knowledge of the DSA degradation by observing the eigenvalue dependency on the fourier wave numbers for our problems. Figure 4.18 provides the eigenvalue distribution based off the fourier wave numbers for the different quadrature sets for $\sigma = 10$ and $c = 0.9$. Figure 4.19 then provides the same information for $\sigma = 640$ and $c = 0.9999$.

4.7.3.4 Performance of MIP DSA on Polygons with Adaptive Mesh Refinement

For the final theoretical analysis of DSA with the MIP form, we analyze the acceleration performance when AMR is utilized on a sufficiently optically thick transport with material discontinuities. The 2D transport problem that will be examined is similar to the Iron-Water problem [113]. It was modified by Wang and Ragusa for use with higher-order basis functions on triangular meshes with hanging nodes [92]. We will reexamine their work on degenarate polygonal grids and not use hanging nodes. The complete geometric desciption of our problem including boundary conditions and material distributions is given in Figure 4.20. The material properties for each region which include the total cross section, scattering ratio, and source strength are

Table 4.4: Spectral radius for the 2D PHI problem with the PWL basis functions and LS2 quadrature.

	Scattering ratios					
σ	0.9	0.99	0.999	0.9999	0.99999	0.99999
10	0.77091	0.89908	0.91279	0.91417	0.91431	0.91432
20	0.82038	0.94425	0.95859	0.96003	0.96018	0.96019
40	0.84803	0.96525	0.97982	0.98129	0.98144	0.98145
80	0.86188	0.97491	0.98955	0.99102	0.99117	0.99119
160	0.87134	0.98003	0.99409	0.99557	0.99571	0.99573
320	0.87833	0.98353	0.99626	0.99774	0.99789	0.99790
640	0.88187	0.98533	0.99732	0.99880	0.99894	0.99896

Table 4.5: Spectral radius for the 2D PHI problem with the PWL basis functions and LS4 quadrature.

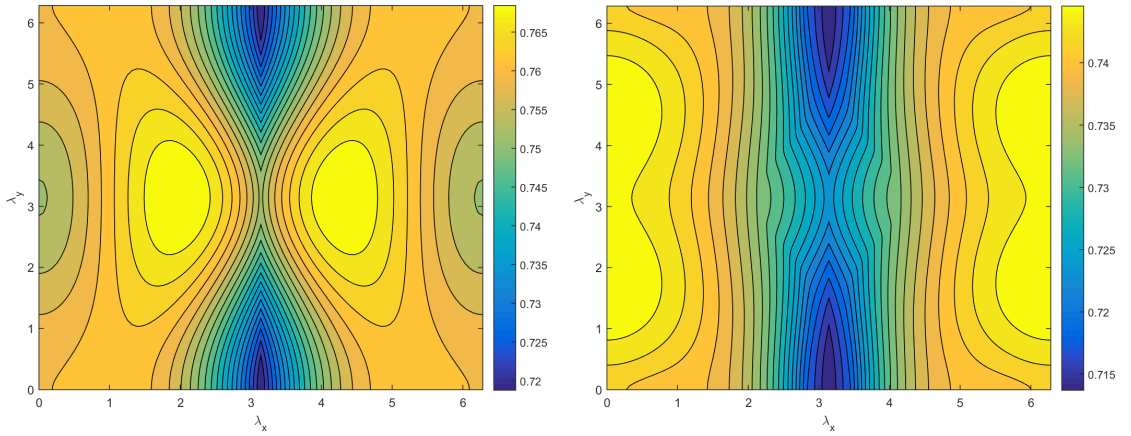
	Scattering ratios					
σ	0.9	0.99	0.999	0.9999	0.99999	0.99999
10	0.74609	0.87284	0.88970	0.89148	0.89166	0.89167
20	0.81135	0.93358	0.95001	0.95179	0.95197	0.95199
40	0.84353	0.96104	0.97612	0.97781	0.97799	0.97800
80	0.85865	0.97342	0.98785	0.98943	0.98960	0.98961
160	0.86998	0.97913	0.99335	0.99482	0.99497	0.99499
320	0.87767	0.98307	0.99593	0.99739	0.99753	0.99755
640	0.88166	0.98500	0.99717	0.99863	0.99878	0.99879

Table 4.6: Spectral radius for the 2D PHI problem with the PWL basis functions and LS8 quadrature.

	Scattering ratios					
σ	0.9	0.99	0.999	0.9999	0.99999	0.99999
10	0.75532	0.87917	0.89780	0.89989	0.90010	0.90012
20	0.81754	0.93694	0.95380	0.95581	0.95602	0.95604
40	0.84681	0.96337	0.97793	0.97969	0.97988	0.97990
80	0.86032	0.97459	0.98898	0.99042	0.99057	0.99058
160	0.86980	0.98007	0.99394	0.99540	0.99554	0.99556
320	0.87795	0.98354	0.99623	0.99768	0.99783	0.99784
640	0.88204	0.98524	0.99732	0.99878	0.99892	0.99894

Table 4.7: Spectral radius for the 2D PHI problem with the PWL basis functions and LS16 quadrature.

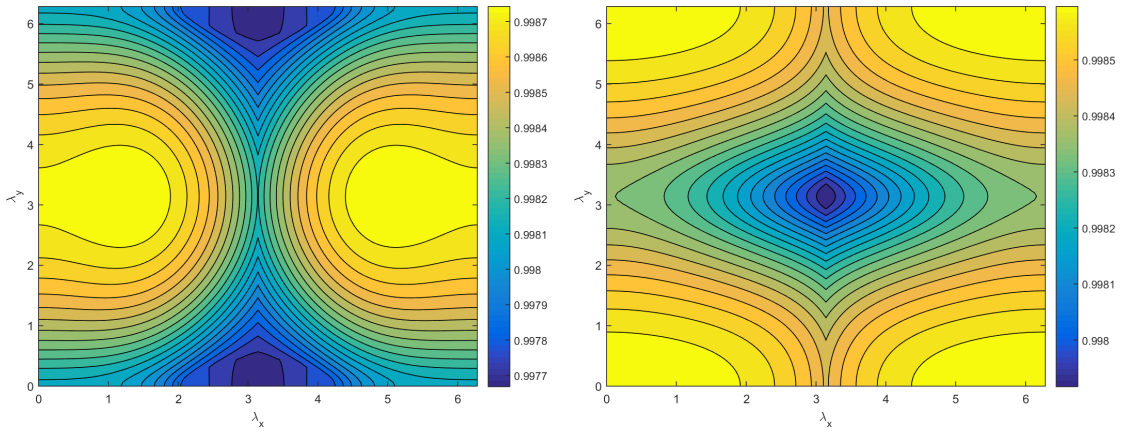
	Scattering ratios					
σ	0.9	0.99	0.999	0.9999	0.99999	0.99999
10	0.75796	0.88052	0.89888	0.90097	0.90119	0.90121
20	0.81903	0.93790	0.95455	0.95651	0.95671	0.95673
40	0.84758	0.96387	0.97842	0.98015	0.98033	0.98035
80	0.86070	0.97485	0.98924	0.99069	0.99083	0.99085
160	0.87025	0.98029	0.99407	0.99553	0.99567	0.99569
320	0.87817	0.98365	0.99629	0.99775	0.99790	0.99791
640	0.88216	0.98530	0.99735	0.99881	0.99896	0.99897



(a) LS2 (b) LS4

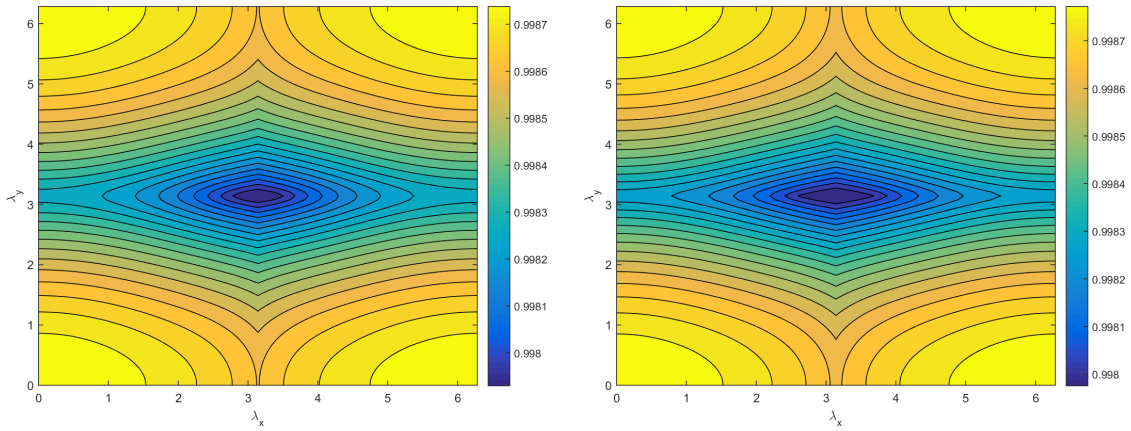
(c) LS8 (d) LS16

Figure 4.18: Fourier wave number distribution for the 2D PHI problem with $\sigma = 10$ and $c = 0.9$ and different level-symmetric quadratures.



(a) LS2

(b) LS4



(c) LS8

(d) LS16

Figure 4.19: Fourier wave number distribution for the 2D PHI problem with $\sigma = 640$ and $c = 0.9999$ and different level-symmetric quadratures.

Table 4.8: Material definitions and physical properties for the Iron-Water problem.

Region	σ_t (cm^{-1})	c	S ($cm^{-3}sec^{-1}$)
I	1.0	0.90	1.0
II	1.5	0.96	0.0
III	1.0	0.30	0.0

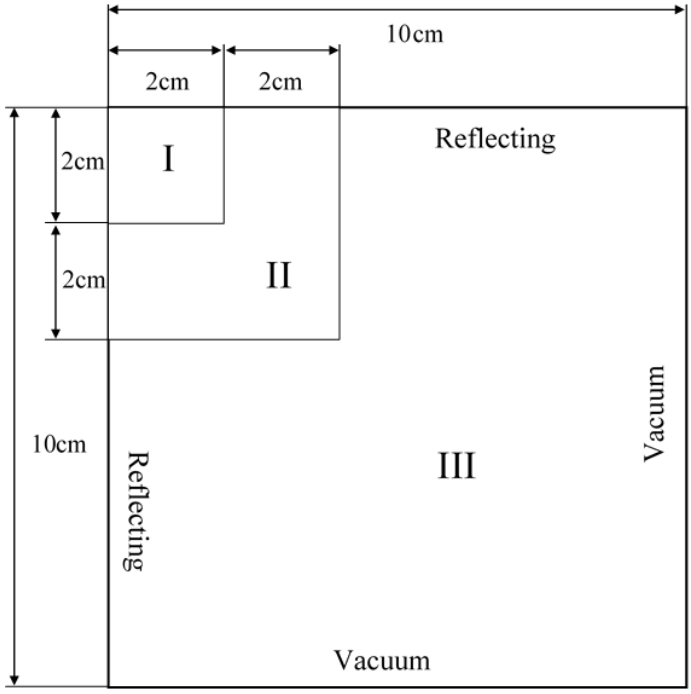


Figure 4.20: Geometry description for the Iron-Water problem.

given in Table 4.8. Scattering is isotropic.

4.7.4 Scalability of the MIP DSA Preconditioner

So far, we have presented a detailed theoretical analysis of DSA preconditioning with the MIP diffusion form. We have shown that the different 2D and 3D basis functions provided in Chapter 3 are robust and stable, even on mesh cells with high aspect ratios. We also demonstrated that problems with large heterogeneous

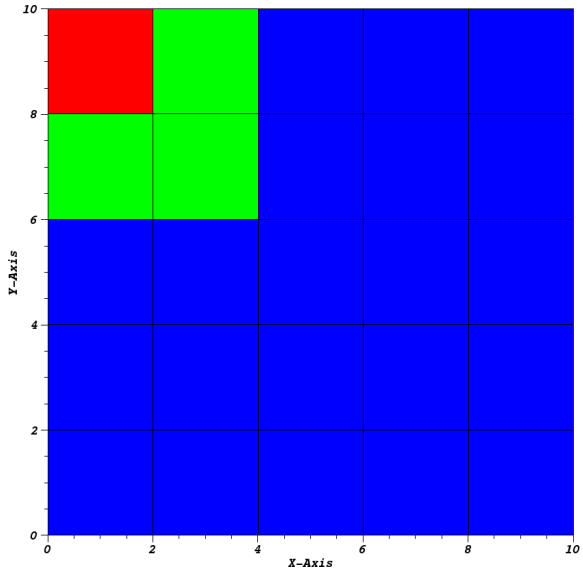
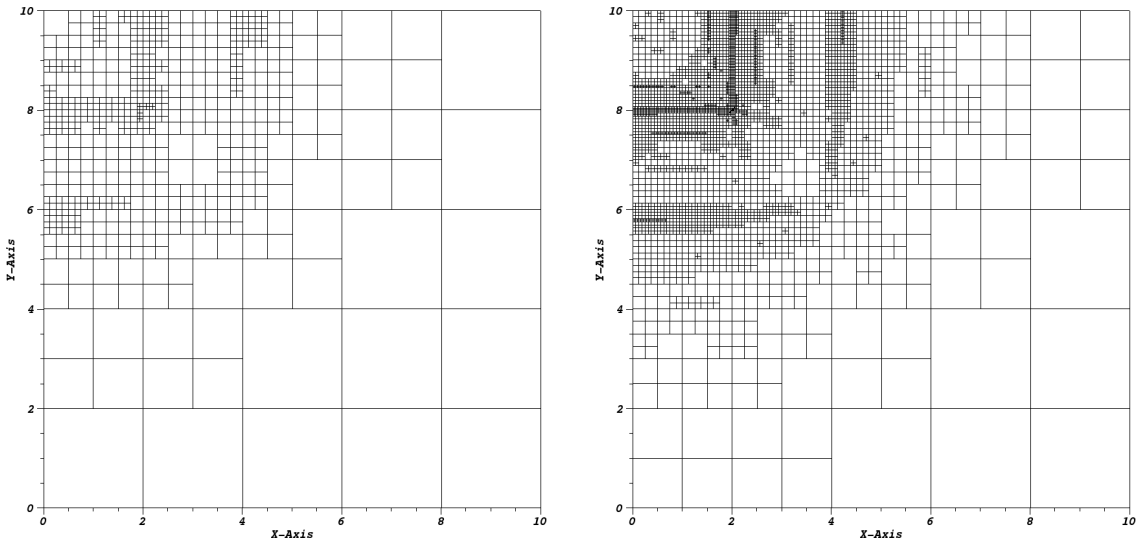


Figure 4.21: Initial mesh for the Iron-Water problem.

configurations can diminish the effectiveness of MIP DSA.

Now, we need to demonstrate the scalability of MIP DSA preconditioning onto large massively-parallel computer architectures. Our ability to utilize this transport acceleration form would be greatly minimized if the DSA solve times scaled at a much worse rate compared to the transport sweep times. We specifically use the low-order diffusion operator because it is supposed to be easier to invert than the full transport operator.

From the results of the Iron-Water problem in Section 4.7.3.4, we observed that the PCG iteration counts did not appreciably grow when utilizing AMG as the preconditioner for the diffusion solve. This was in direct contrast to the simpler preconditioners (Jacobi, GS, and ILU) that had their iteration counts grow rapidly as the number of unknowns increased. Motivated by the efficiency of AMG methods with the MIP diffusion form for the simple Iron-Water AMR problem, we will continue to use AMG as the diffusion preconditioner for our massively-parallel calculations.



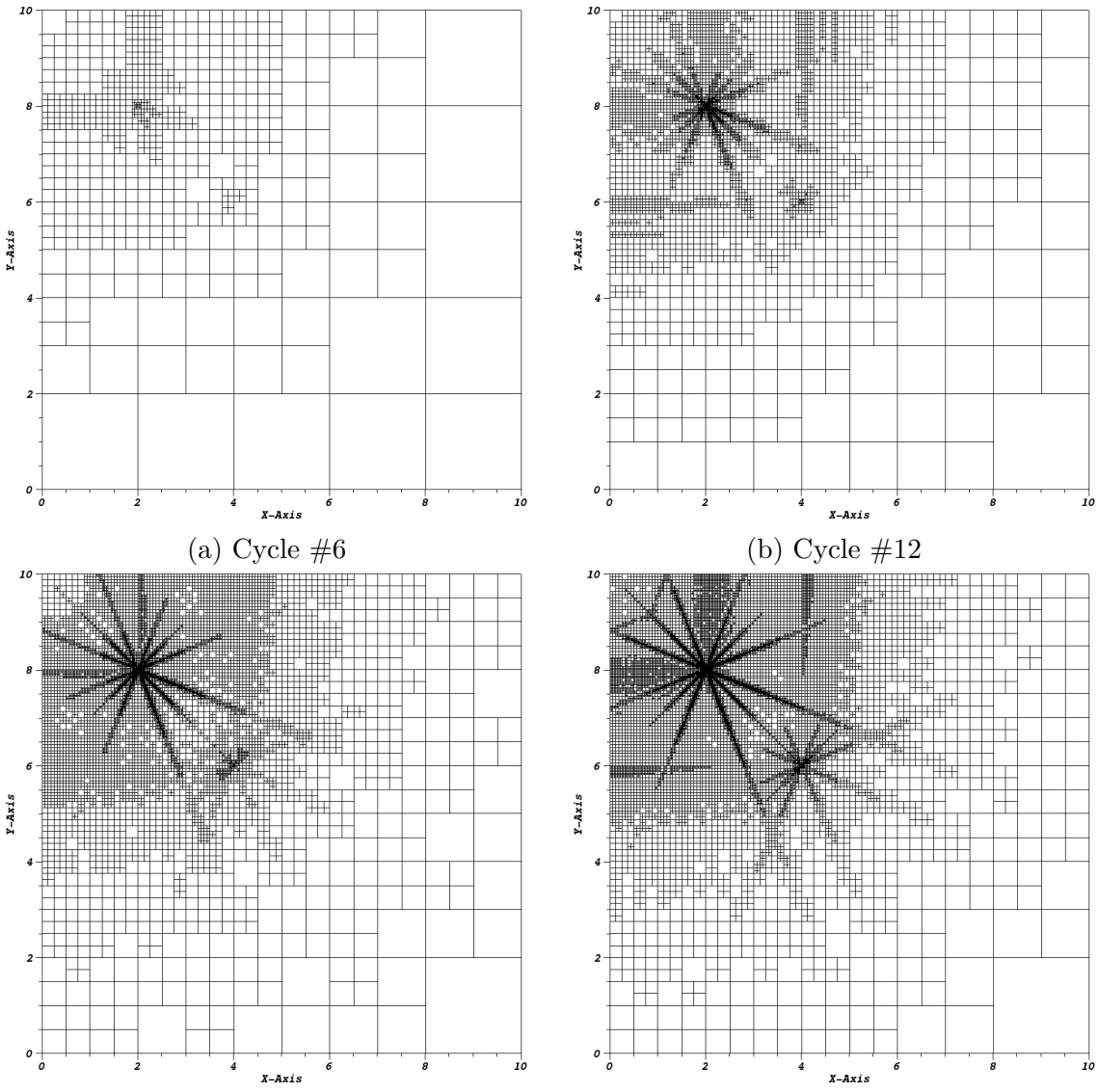
(a) Cycle #6

(b) Cycle #12

(c) Cycle #18

(d) Cycle #24

Figure 4.22: Meshes for the Iron-Water problem using the linear PWL coordinates and LS4 quadrature.



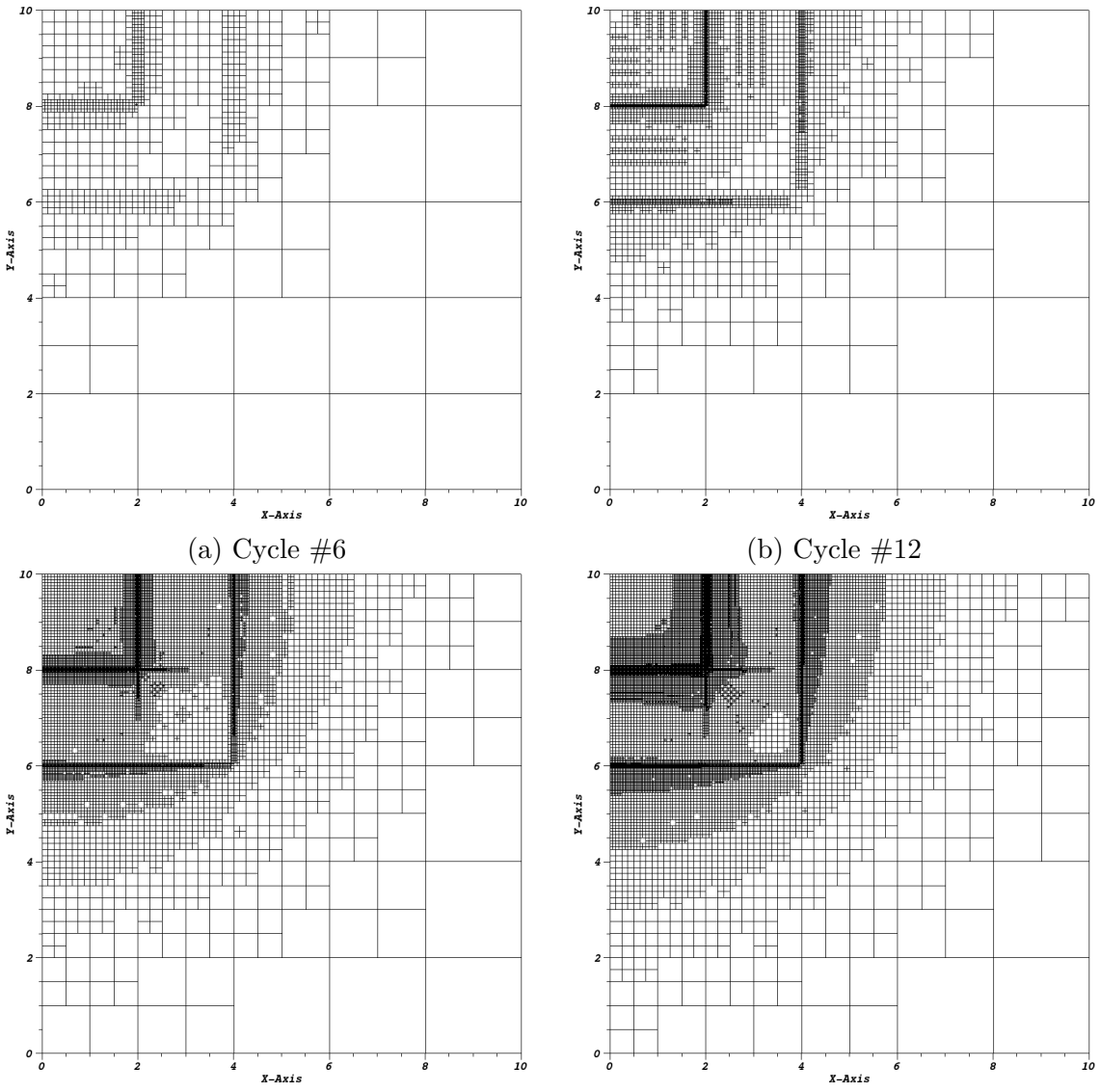
(a) Cycle #6

(b) Cycle #12

(c) Cycle #18

(d) Cycle #24

Figure 4.23: Meshes for the Iron-Water problem using the quadratic PWL coordinates and LS4 quadrature.



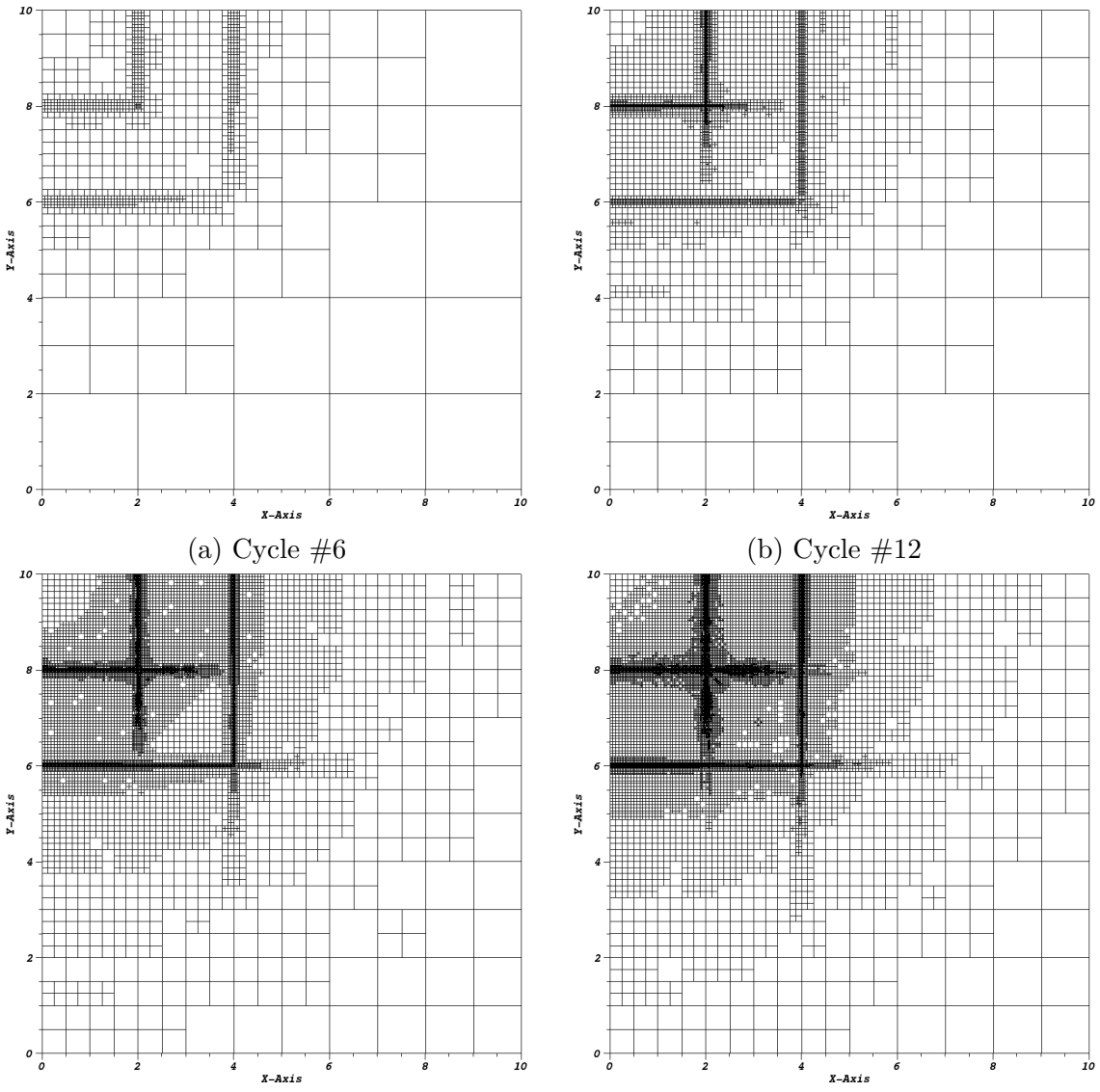
(a) Cycle #6

(b) Cycle #12

(c) Cycle #18

(d) Cycle #24

Figure 4.24: Meshes for the Iron-Water problem using the linear PWL coordinates and S_{24}^2 PGLC quadrature.



(a) Cycle #6

(b) Cycle #12

(c) Cycle #18

(d) Cycle #24

Figure 4.25: Meshes for the Iron-Water problem using the quadratic PWL coordinates and S_{24}^2 PGLC quadrature.

We have implemented the 1-group and thermal upscattering DSA methodologies of Section 4.2 with the MIP form into Texas A&M University's PDT code. It is a massively-parallel DGFEM S_N transport code that has had good sweep scaling efficiency out to $O(10^5) - O(10^6)$ processes [114, 115]. BoomerAMG of the HYPRE library is used as the AMG diffusion preconditioner [116, 117]. We next analyze the scalability of HYPRE's PCG solver with BoomerAMG preconditioning and how this performs in comparison to PDT's transport sweeping.

4.7.4.1 Weak Scaling with a Homogeneous Zerr Problem

We first test MIP's scaling with HYPRE in PDT by analyzing a simple homogenized version of the Zerr scaling problem [35]. The problem configuration is a 3D cube that spans $[0, 16]^3$ in dimension and uses strictly orthogonal hexahedral mesh cells. The total cross section is set to $\sigma_t = 10$ with a scattering ratio of $c = 0.9999$ and a distributed source of $1 \frac{n}{cm^3\, s}$. LS8 quadrature is employed and vacuum boundary conditions are used for all boundaries. A residual tolerance of 10^{-8} is used for the Richardson transport iterations, and a residual tolerance of 10^{-3} is used for the HYPRE PCG iterations.

Our weak scaling study is performed with a fixed 4096 spatial cells per processor as we increase the number of processors. We do not change the physical dimensions of the problem. Instead, we simply increase the number of cells in a given coordinate direction whenever we allocate more processors in that dimension. The partitioning of the processor layout (P_x, P_y, and P_z), the task aggregation (A_x, A_y, and A_z), and the number of mesh cells per dimension (N_x, N_y, and N_z) are selected to minimize the time per sweep in PDT. These parallel parameters are given for the full scaling suite in Table 4.9.

The timing data for this weak scaling study is given in Figure 4.26. As a function

Table 4.9: Partitioning factors, aggregation factors, and cell counts per dimension for the 3D homogeneous Zerr problem run on Vulcan.

P_{tot}	P_x	P_y	P_z	A_x	A_y	A_z	N_x	N_y	N_z
1	1	1	1	16	16	1	16	16	16
8	2	2	2	16	16	1	32	32	32
64	8	4	2	8	16	1	64	64	64
512	16	16	2	8	8	1	128	128	128
1024	32	16	2	4	8	1	128	128	256
2048	32	32	2	8	4	1	256	128	256
4096	64	32	2	4	8	1	256	256	256
8192	64	64	2	4	4	1	256	256	512
16384	128	64	2	4	4	1	512	256	512
32768	128	128	2	4	4	1	512	512	512
65536	256	128	2	2	4	1	512	512	1024
131072	256	256	2	4	2	1	1024	512	1024

of processors used, we plot the overall solve time, the overall sweep time, the overall MIP DSA time, the time to build the MIP system matrix, the time to perform the HYPRE BoomerAMG setup call, and the time per HYPRE PCG iteration. The sweep time only constitutes the time taken to perform the actual sweep. This discounts the time needed to build the transport sources and prepare for each sweep. There are a lot of results that can be observed from this plot. We first note that all the solve times increase at the low processor counts because the number of sweeps increases from about 20 to 35 as the mesh optical thicknesses move into the intermediate range. The second note is that while the sweep times remain about the same, the various HYPRE times begin to scale more poorly as we get to $O(10^4)$ processors and higher. We can see that the HYPRE setup time especially begins to increase at a rapid pace for the highest core counts. Finally we observe that at certain processor values, the HYPRE times seem to 'spike'. This occurs at 1024, 8192, and 65536 processor counts. By looking at the problem's parallel characteristics from Table

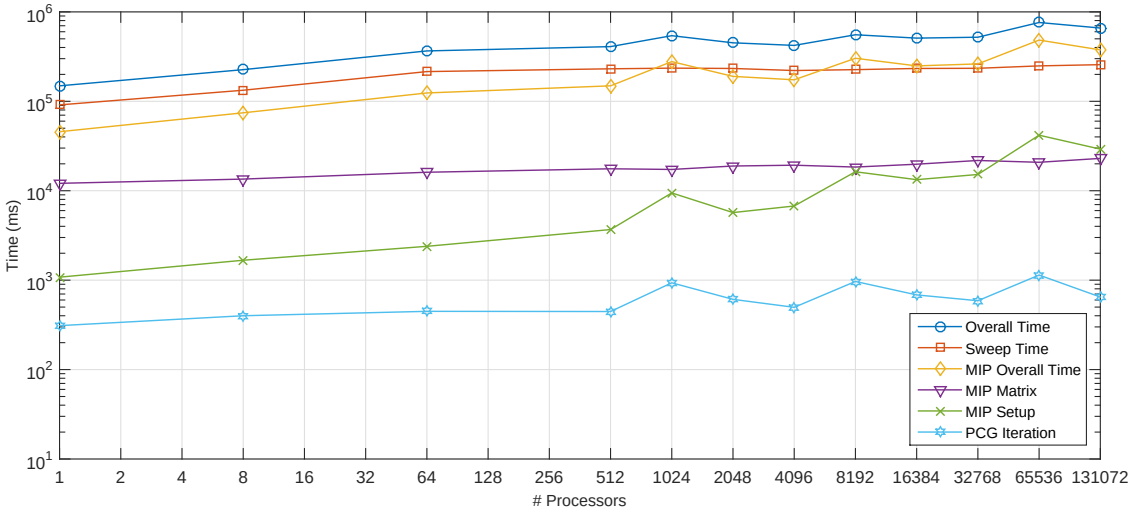


Figure 4.26: blah

4.9, we can see that these processor counts occur when the number of mesh cells in the z-dimension is doubled. This means that the distribution of the cell sets on a processor look like an even longer and skinnier column. This behavior is analyzed in greater detail shortly in Section 4.7.4.2.

From the timing results of Figure 4.26, one must wonder about the scalability of solving the MIP equation at even higher processor counts out to $O(10^6)$ or $O(10^7)$. For this simple problem, our diffusion solver was taking about 50%-60% of the solve time. However, our transport problem was extremely coarse in angle and only 1 energy group. This leads to low parallel concurrency for the phase-space that we solve for in one transport sweep. This means that for a much larger 3D transport problem with many energy groups ($O(10^2)$) and many quadrature angles ($O(10^3)$), the fraction of time that is spent performing DSA calculations will become negligibly small. In fact, we will show a little later that our energy-collapsed thermal neutron upscattering methodologies yield DSA solve times that are not noticible.

4.7.4.2 Aggregation and Partitioning Effects on the HYPRE PCG Algorithm

4.7.5 Thermal Neutron Upscattering Acceleration

We conclude the results of this chapter by presenting analysis on the ability to use DSA to accelerate the convergence of multigroup transport problems that are dominated by thermal neutron upscattering.

4.8 Conclusions

In this chapter, we analyzed the Modified Interior Penalty form of the diffusion equation for use as the diffusion solver for DSA preconditioning of the DGFEM S_N transport equation.

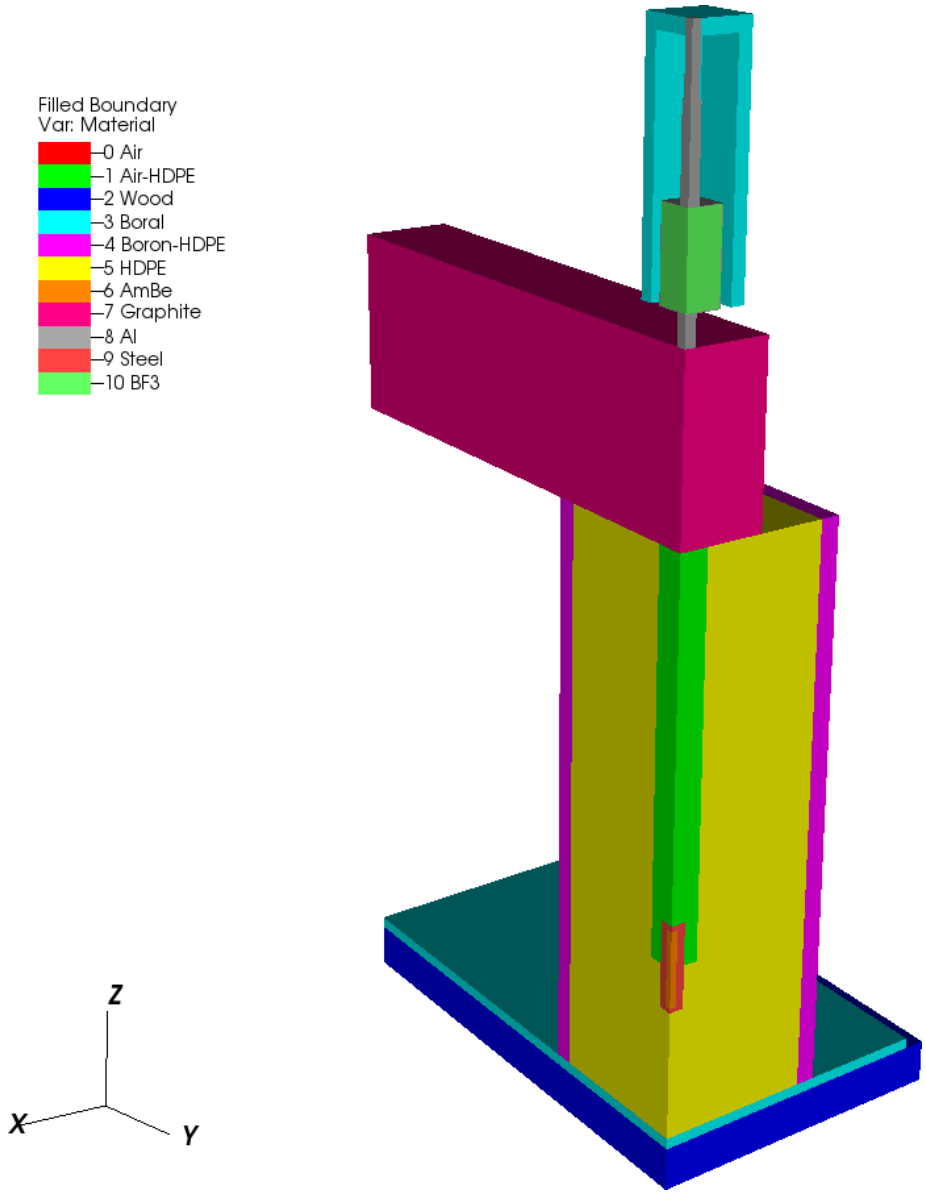
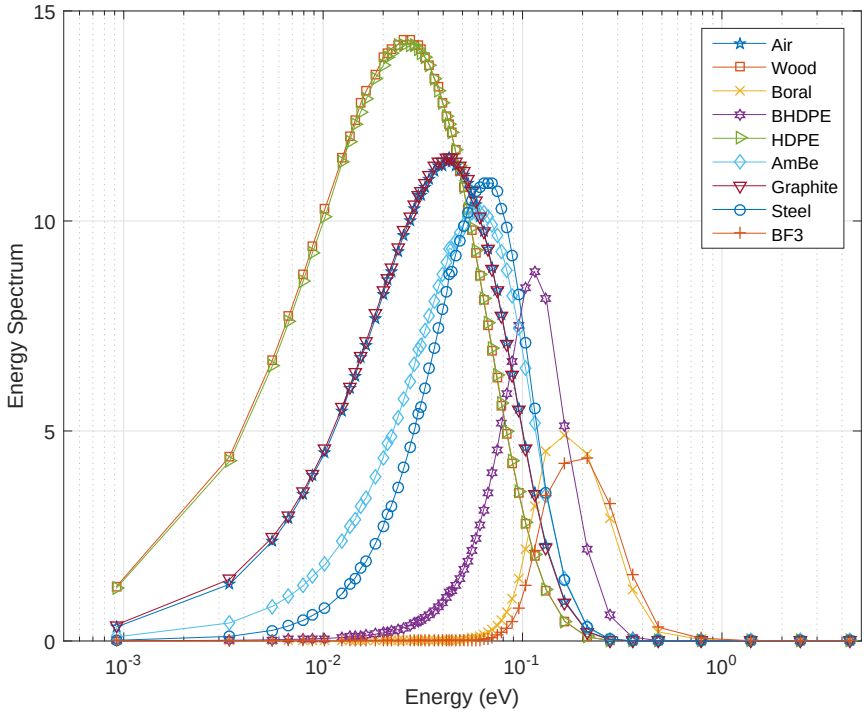
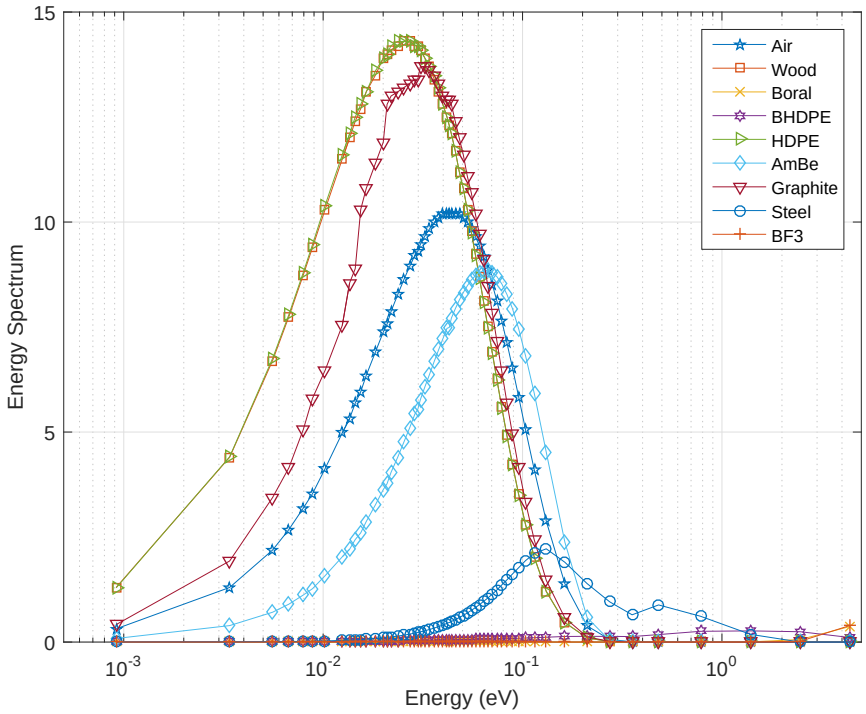


Figure 4.27: Configuration of the IM1 problem with the outer layers of air removed.



(a) Two-Grid



(b) Modified Two-Grid

Figure 4.28: Spectral shape of the infinite medium iteration matrices for the IM1 problem materials.

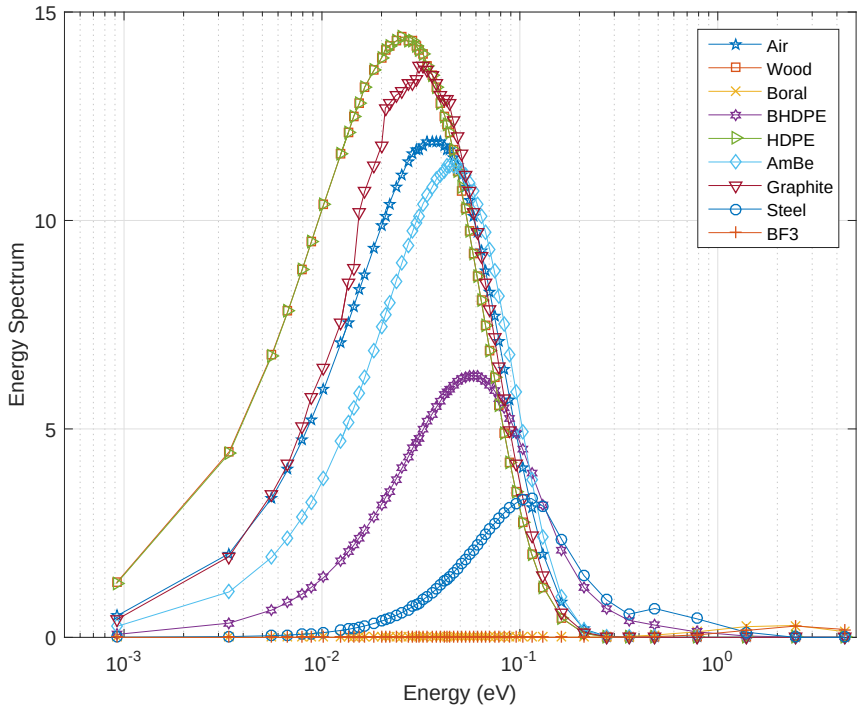
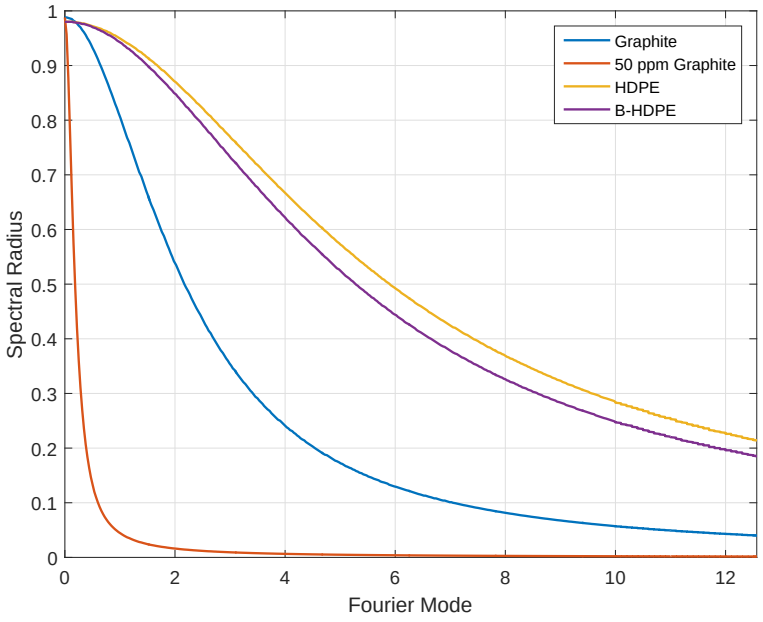
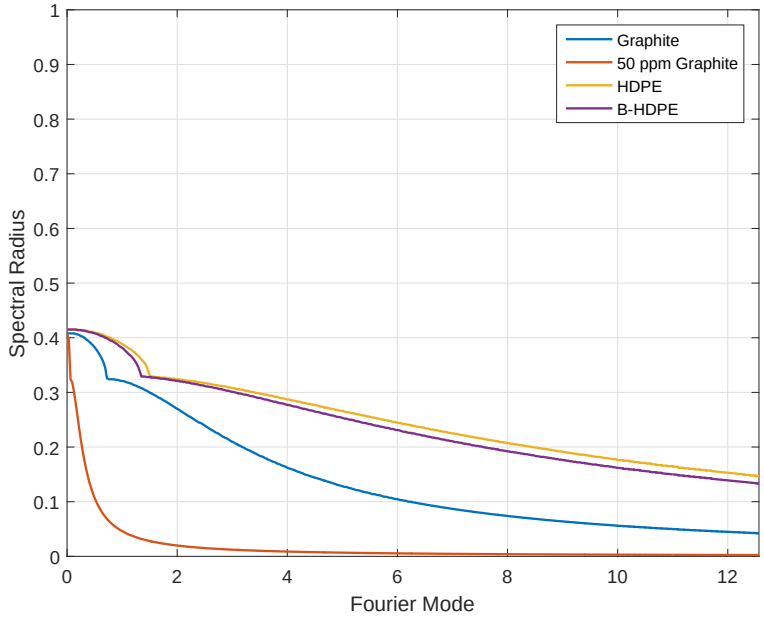


Figure 4.29: Spectral shape of the infinite medium iteration matrices for the Multigroup Jacobi Acceleration Method for the IM1 problem materials.



(a) Gauss-Seidel



(b)

Figure 4.30: 1D fourier analysis for some IM1 materials of interest. (a)

5. CONCLUSIONS

5.1 Conclusions

In this dissertation

5.2 Open Items

REFERENCES

[1] STAR-CCM+, "http://www.cd-adapco.com," (2015).

[2] A. ERN and J.-L. GUERMOND, *Theory and practice of finite elements*, vol. 159, Springer Science & Business Media (2013).

[3] J. R. SHEWCHUK, "Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator," in "Applied computational geometry towards geometric engineering," Springer, pp. 203–222 (1996).

[4] J. R. SHEWCHUK, "Delaunay refinement algorithms for triangular mesh generation," *Computational geometry*, **22**, *1*, 21–74 (2002).

[5] H. SI, "TetGen, a Delaunay-based quality tetrahedral mesh generator," *ACM Transactions on Mathematical Software (TOMS)*, **41**, *2*, 11 (2015).

[6] C. GEUZAINE and J.-F. REMACLE, "Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities," *International Journal for Numerical Methods in Engineering*, **79**, *11*, 1309–1331 (2009).

[7] R. LERNER and G. TRIGG, *Encyclopaedia of Physics*, 2 ed. (1991).

[8] C. PARKER, *McGraw Hill Encyclopaedia of Physics*, 2 ed. (1994).

[9] J. J. DUDERSTADT and W. R. MARTIN, *Transport theory*, John Wiley & Sons (1979).

[10] K. OTT and W. BEZELLA, *Introductory Nuclear Reactor Statics*, American Nuclear Society (1989).

[11] J. J. DUDERSTADT and L. J. HAMILTON, *Nuclear reactor analysis*, Wiley (1976).

[12] E. E. LEWIS and W. F. MILLER, *Computational methods of neutron transport*, John Wiley and Sons, Inc., New York, NY (1984).

[13] B. CARLSON and K. LATHROP, *Computing methods in reactor physics: Transport Theory - The Method of Discrete Ordinates*, Gordon and Breach Science Publishers, Inc. (1968).

[14] B. CARLSON, "On a more precise definition of discrete ordinates methods," in "Proc. Second Conf. Transport Theory," U.S. Atomic Energy Commision (1971), pp. 348–390.

[15] I. ABU-SHUMAYS, "Compatible product angular quadrature for neutron transport in xy geometry," *Nuclear Science and Engineering*, **64**, *2*, 299–316 (1977).

[16] M. ABRAMOWITZ, I. A. STEGUN, ET AL., "Handbook of mathematical functions," *Applied Mathematics Series*, **55**, 62 (1966).

[17] M. ABRAMOWITZ and I. A. STEGUN, *Handbook of mathematical functions: with formulas, graphs, and mathematical tables*, 55, Courier Corporation (1964).

[18] T. A. WAREING, J. M. MCGHEE, J. E. MOREL, and S. D. PAUTZ, "Discontinuous finite element S n methods on three-dimensional unstructured grids," *Nuclear science and engineering*, **138**, *3*, 256–268 (2001).

[19] P. LESAINT and P.-A. RAVIART, "On a finite element method for solving the neutron transport equation," *Mathematical aspects of finite elements in partial differential equations*, , *33*, 89–123 (1974).

[20] P. HOUSTON, C. SCHWAB, and E. SÜLI, "Stabilized hp-finite element methods for first-order hyperbolic problems," *SIAM Journal on Numerical Analysis*,

37, *5*, 1618–1643 (2000).

[21] P. HOUSTON, C. SCHWAB, and E. SÜLI, "Discontinuous hp-finite element methods for advection-diffusion-reaction problems," *SIAM Journal on Numerical Analysis*, **39**, *6*, 2133–2163 (2002).

[22] Y. WANG and J. C. RAGUSA, "On the convergence of DGFEM applied to the discrete ordinates transport equation for structured and unstructured triangular meshes," *Nuclear Science and Engineering*, **163**, *1*, 56–72 (2009).

[23] E. MAGENES and J.-L. LIONS, *Problèmes aux limites non homogènes et applications* (1968).

[24] C. JOHNSON and J. PITKÄRANTA, "Convergence of a fully discrete scheme for two-dimensional neutron transport," *SIAM journal on numerical analysis*, **20**, *5*, 951–966 (1983).

[25] R. H. MACNEAL and R. L. HARDER, "Eight nodes or nine?" *International journal for numerical methods in engineering*, **33**, *5*, 1049–1058 (1992).

[26] D. N. ARNOLD and G. AWANOU, "The serendipity family of finite elements," *Foundations of Computational Mathematics*, **11**, *3*, 337–344 (2011).

[27] O. ZEINKIEWICZ, R. TAYLOR, and J. ZHU, *The finite element method: its basis and fundamentals*, SElsevier Butterworth-Heinemann (2005).

[28] J. AKIN, *Application and implementation of finite element methods*, Academic Press, Inc. (1982).

[29] M. ADAMS and E. LARSEN, "Fast iterative methods for discrete-ordinates particle transport calculations," *Progress in nuclear energy*, **40**, *1*, 3–159 (2002).

[30] S. OLIVEIRA and Y. DENG, "Preconditioned Krylov subspace methods for transport equations," *Progress in Nuclear Energy*, **33**, *1*, 155–174 (1998).

[31] B. GUTHRIE, J. P. HOLLOWAY, and B. W. PATTON, "GMRES as a multi-step transport sweep accelerator," *Transport theory and statistical physics*, **28**, *1*, 83–102 (1999).

[32] B. W. PATTON and J. P. HOLLOWAY, "Application of preconditioned GMRES to the numerical solution of the neutron transport equation," *Annals of Nuclear Energy*, **29**, *2*, 109–136 (2002).

[33] Y. SAAD and M. H. SCHULTZ, "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM Journal on scientific and statistical computing*, **7**, *3*, 856–869 (1986).

[34] Y. SAAD, *Iterative methods for sparse linear systems*, Siam (2003).

[35] R. J. ZERR, *Solution of the within-group multidimensional discrete ordinates transport equations on massively parallel architectures*, Ph.D. thesis, The Pennsylvania State University (2011).

[36] A. GERASOULIS and I. NELKEN, "Static scheduling for linear algebra DAGs," in "Proc. of 4th Conf. on Hypercubes, Monterey," (1989), vol. 1, pp. 671–674.

[37] T. PLEWA, T. LINDE, and V. G. WEIRS, *Adaptive Mesh Refinement - Theory and Applications*, Springer Science & Business Media (2005).

[38] G. F. CAREY, *Computational Grids: Generations, Adaptation & Solution Strategies*, CRC Press (1997).

[39] E. RAMM, E. RANK, R. RANNACHER, K. SCHWEIZERHOF, E. STEIN, W. WENDLAND, G. WITTUM, P. WRIGGERS, and W. WUNDERLICH,

Error-controlled adaptive finite elements in solid mechanics, John Wiley & Sons (2003).

[40] G. KARNIADAKIS and S. SHERWIN, *Spectral/hp element methods for computational fluid dynamics*, Oxford University Press (2013).

[41] C. SCHWAB, *p-and hp-finite element methods: Theory and applications in solid and fluid mechanics*, Oxford University Press (1998).

[42] P. SOLIN, K. SEGETH, and I. DOLEZEL, *Higher-order finite element methods*, CRC Press (2003).

[43] C. FÜHRER and G. KANSCHAT, "A posteriori error control in radiative transfer," *Computing*, **58**, *4*, 317–334 (1997).

[44] R. HARTMANN, *Adaptive finite element methods for the compressible euler equations*, Ph.D. thesis, University of Heidelberg (2002).

[45] A. DEDNER and P. VOLLMÖLLER, "An adaptive higher order method for solving the radiation transport equation on unstructured grids," *Journal of Computational Physics*, **178**, *2*, 263–289 (2002).

[46] R. HARTMANN and P. HOUSTON, "Adaptive discontinuous Galerkin finite element methods for nonlinear hyperbolic conservation laws," *SIAM Journal on Scientific Computing*, **24**, *3*, 979–1004 (2003).

[47] Y. WANG and J. C. RAGUSA, "A two-mesh adaptive mesh refinement technique for SN neutral-particle transport using a higher-order DGFEM," *Journal of computational and applied mathematics*, **233**, *12*, 3178–3188 (2010).

[48] Y. WANG and J. C. RAGUSA, "Standard and goal-oriented adaptive mesh refinement applied to radiation transport on 2D unstructured triangular meshes," *Journal of Computational Physics*, **230**, *3*, 763–788 (2011).

[49] R. VERFÜRTH, *A review of a posteriori error estimation and adaptive mesh-refinement techniques*, John Wiley & Sons Inc (1996).

[50] M. AINSWORTH and J. T. ODEN, *A posteriori error estimation in finite element analysis*, vol. 37, John Wiley & Sons (2011).

[51] L. DEMKOWICZ, *Computing with hp-Adaptive Finite Elements: Volume 1 One and Two Dimensional Elliptic and Maxwell problems*, CRC Press (2006).

[52] E. L. WACHSPRESS, "A Rational Finite Element Basis," *Mathematics in science and engineering* (1975).

[53] H. G. STONE and M. L. ADAMS, "A piecewise linear finite element basis with application to particle transport," in "Proc. ANS Topical Meeting Nuclear Mathematical and Computational Sciences Meeting," (2003).

[54] H. G. STONE and M. L. ADAMS, "New Spatial Discretization Methods for Transport on Unstructured Grids," in "Proc. ANS Topical Meeting Mathematics and Computation, Supercomputing, Reactor Physics and Biological Applications," (2005).

[55] M. S. FLOATER, "Mean value coordinates," *Computer aided geometric design*, **20**, *1*, 19–27 (2003).

[56] K. HORMANN and M. S. FLOATER, "Mean value coordinates for arbitrary planar polygons," *ACM Transactions on Graphics (TOG)*, **25**, *4*, 1424–1441 (2006).

[57] N. SUKUMAR, "Construction of polygonal interpolants: a maximum entropy approach," *International Journal for Numerical Methods in Engineering*, **61**, *12*, 2159–2181 (2004).

[58] M. ARROYO and M. ORTIZ, "Local maximum-entropy approximation schemes: a seamless bridge between finite elements and meshfree methods," *International journal for numerical methods in engineering*, **65**, *13*, 2167–2202 (2006).

[59] K. HORMANN and N. SUKUMAR, "Maximum entropy coordinates for arbitrary polytopes," in "Computer Graphics Forum," Wiley Online Library (2008), vol. 27, pp. 1513–1520.

[60] S. KULLBACK and R. A. LEIBLER, "On information and sufficiency," *The annals of mathematical statistics*, pp. 79–86 (1951).

[61] E. T. JAYNES, "Information theory and statistical mechanics," in "Statistical Physics," (1963), vol. 3, pp. 181–218.

[62] P. SILVESTER, "Symmetric quadrature formulae for simplexes," *Mathematics of Computation*, **24**, *109*, 95–100 (1970).

[63] D. DUNAVANT, "High degree efficient symmetrical Gaussian quadrature rules for the triangle," *International journal for numerical methods in engineering*, **21**, *6*, 1129–1148 (1985).

[64] S. WANDZURAT and H. XIAO, "Symmetric quadrature rules on a triangle," *Computers & Mathematics with Applications*, **45**, *12*, 1829–1840 (2003).

[65] J. N. LYNESS and R. COOLS, "A survey of numerical cubature over triangles," in "Proceedings of Symposia in Applied Mathematics," (1994), vol. 48, pp. 127–150.

[66] R. COOLS and A. HAEGEMANS, "Construction of minimal cubature formulae for the square and the triangle, using invariant theory," (1987).

[67] M. NOOIJEN, G. TE VELDE, and E. BAERENDS, "Symmetric numerical integration formulas for regular polygons," *SIAM journal on numerical analysis*, **27**, *1*, 198–218 (1990).

[68] G. DASGUPTA, "Integration within polygonal finite elements," *Journal of Aerospace Engineering*, **16**, *1*, 9–18 (2003).

[69] S. MOUSAVI, H. XIAO, and N. SUKUMAR, "Generalized Gaussian quadrature rules on arbitrary polygons," *International Journal for Numerical Methods in Engineering*, **82**, *1*, 99–113 (2010).

[70] T. S. BAILEY, *The piecewise linear discontinuous finite element method applied to the RZ and XYZ transport equations*, Ph.D. thesis, Texas A&M University (2008).

[71] A. SHESTAKOV, J. HARTE, and D. KERSHAW, "Solution of the diffusion equation by finite elements in lagrangian hydrodynamic codes," *Journal of Computational Physics*, **76**, *2*, 385–413 (1988).

[72] A. SHESTAKOV, D. KERSHAW, and G. ZIMMERMAN, "Test problems in radiative transfer calculations," *Nuclear science and engineering*, **105**, *1*, 88–104 (1990).

[73] D. S. KERSHAW, "Differencing of the diffusion equation in Lagrangian hydrodynamic codes," *Journal of Computational Physics*, **39**, *2*, 375–395 (1981).

[74] H. KOPP, "Synthetic method solution of the transport equation," *Nuclear Science and Engineering*, **17**, 65 (1963).

[75] V. LEBEDEV, "The Iterative *KP* Method for the Kinetic Equation," in "Proc. Conf. on Mathematical Methods for Solution of Nuclear Physics Problems," (1964).

[76] V. LEBEDEV, "The KP-method of accelerating the convergence of iterations in the solution of the kinetic equation," *Numerical methods of solving problems of mathematical physics, Nauka, Moscow*, pp. 154–176 (1966).

[77] V. LEBEDEV, "On Finding Solutions of Kinetic Problems," *USSR Comp. Math. and Math. Phys.*, **6**, 895 (1966).

[78] V. LEBEDEV, "An Iterative *KP* Method," *USSR Comp. Math. and Math. Phys.*, **7**, 1250 (1967).

[79] V. LEBEDEV, "Problem of the Convergence of a Method of Estimating Iteration Deviations," *USSR Comp. Math. and Math. Phys.*, **8**, 1377 (1968).

[80] V. LEBEDEV, "Convergence of the *KP* Method for Some Neutron Transfer Problems," *USSR Comp. Math. and Math. Phys.*, **9**, 226 (1969).

[81] V. LEBEDEV, "Construction of the *P* Operation in the *KP* Method," *USSR Comp. Math. and Math. Phys.*, **9**, 762 (1969).

[82] E. GELBARD and L. HAGEMAN, "The synthetic method as applied to the SN equations," *Nucl. Sci. Eng*, **37**, *2*, 288 (1969).

[83] W. H. REED, "The effectiveness of acceleration techniques for iterative methods in transport theory," *Nucl. Sci. Eng*, **45**, *3*, 245 (1971).

[84] R. ALCOUFFE, "Stable diffusion synthetic acceleration method for neutron transport iterations," Los Alamos Scientific Lab., NM (1976), vol. 23.

[85] R. ALCOUFFE, "The Diffusion Synthetic Acceleration Method Applied to Two-Dimenaionsl Neutron Transport Problems," Los Alamos Scientific Lab., NM (1977), vol. 27.

[86] R. E. ALCOUFFE, "Diffusion synthetic acceleration methods for the diamond-differenced discrete-ordinates equations," *Nuclear Science and Engineering*, **64**, *2*, 344–355 (1977).

[87] E. W. LARSEN, "Unconditionally stable diffusion-synthetic acceleration methods for the slab geometry discrete ordinates equations. Part I: Theory," *Nucl. Sci. Eng*, **82**, 47 (1982).

[88] D. R. MCCOY and E. W. LARSEN, "Unconditionally stable diffusion-synthetic acceleration methods for the slab geometry discrete ordinates equations. Part II: Numerical results," *Nucl. Sci. Eng*, **82**, 64 (1982).

[89] M. L. ADAMS and W. R. MARTIN, "Diffusion synthetic acceleration of discontinuous finite element transport iterations," *Nucl. Sci. Eng*, **111**, 145–167 (1992).

[90] J. S. WARSA, T. A. WAREING, and J. E. MOREL, "Fully consistent diffusion synthetic acceleration of linear discontinuous S n transport discretizations on unstructured tetrahedral meshes," *Nuclear Science and Engineering*, **141**, *3*, 236–251 (2002).

[91] E. L. T.A. WAREING and M. ADAMS, "Diffusion Accelerated Discontinuous Finite Element Schemes for the S_N Equations in Slab and X-Y Geometries," in "Advances in Mathematics, Computations, and Reactor Physics," (1991), p. 245.

[92] Y. WANG and J. RAGUSA, "Diffusion Synthetic Acceleration for High-Order Discontinuous Finite Element S_n Transport Schemes and Application to Locally Refined Unstructured Meshes," *Nuclear Science and Engineering*, **166**, 145–166 (2010).

[93] Y. WANG, *Adaptive mesh refinement solution techniques for the multigroup SN transport equation using a higher-order discontinuous finite element method*, Ph.D. thesis, Texas A&M University (2009).

[94] B. TURCKSIN and J. C. RAGUSA, "Discontinuous diffusion synthetic acceleration for Sn transport on 2D arbitrary polygonal meshes," *Journal of Computational Physics*, **274**, 356–369 (2014).

[95] B. ADAMS and J. MOREL, "A two-grid acceleration scheme for the multigroup Sn equations with neutron upscattering," *Nuclear science and engineering*, **115**, *3*, 253–264 (1993).

[96] T. M. EVANS, K. T. CLARNO, and J. E. MOREL, "A transport acceleration scheme for multigroup discrete ordinates with upscattering," *Nuclear Science and Engineering*, **165**, *3*, 292–304 (2010).

[97] D. N. ARNOLD, F. BREZZI, B. COCKBURN, and L. D. MARINI, "Unified analysis of discontinuous Galerkin methods for elliptic problems," *SIAM journal on numerical analysis*, **39**, *5*, 1749–1779 (2002).

[98] J. C. RAGUSA, "Discontinuous finite element solution of the radiation diffusion equation on arbitrary polygonal meshes and locally adapted quadrilateral grids," *Journal of Computational Physics*, **280**, 195–213 (2015).

[99] M. HACKEMACK and J. RAGUSA, "A DFEM Formulation of the Diffusion Equation on Arbitrary Polyhedral Grids," in "Transactions of the American Nuclear Society," (2014).

[100] J. LIONS, "Problemes aux Limites non Homogenes a Donnees Irregulieres," *Numerical Analysis of Partial Differential Equations*, pp. 283–292 (2011).

[101] J. NITSCHE, "Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind," in "Abhandlungen aus dem mathematischen Seminar der Universität Hamburg," Springer (1971), vol. 36, pp. 9–15.

[102] D. N. ARNOLD, "An interior penalty finite element method with discontinuous elements," *SIAM journal on numerical analysis*, **19**, *4*, 742–760 (1982).

[103] S. C. EISENSTAT, "Efficient implementation of a class of preconditioned conjugate gradient methods," *SIAM Journal on Scientific and Statistical Computing*, **2**, *1*, 1–4 (1981).

[104] J. RUGE and K. STÜBEN, "Algebraic multigrid," *Multigrid methods*, **3**, 73–130 (1987).

[105] W. L. BRIGGS, S. F. MCCORMICK, ET AL., *A multigrid tutorial*, Siam (2000).

[106] Y. NOTAY, "Users Guide to AGMG," *Electronic Transactions on Numerical Analysis*, **37**, 123–146 (2010).

[107] Y. NOTAY, "An aggregation-based algebraic multigrid method," *Electronic transactions on numerical analysis*, **37**, *6*, 123–146 (2010).

[108] A. NAPOV and Y. NOTAY, "An algebraic multigrid method with guaranteed convergence rate," *SIAM journal on scientific computing*, **34**, *2*, A1079–A1109 (2012).

[109] Y. NOTAY, "Aggregation-based algebraic multigrid for convection-diffusion equations," *SIAM journal on scientific computing*, **34**, *4*, A2288–A2316 (2012).

[110] J. A. NELDER and R. MEAD, "A simplex method for function minimization," *The computer journal*, **7**, *4*, 308–313 (1965).

[111] J. CHANG and M. ADAMS, "Analysis of transport synthetic acceleration for highly heterogeneous problems," in "Proc. of M&C Topical Meeting," (2003).

[112] Y. AZMY, "Unconditionally stable and robust adjacent-cell diffusive preconditioning of weighted-difference particle transport methods is impossible," *Journal of Computational Physics*, **182**, *1*, 213–233 (2002).

[113] H. KHALIL, "A nodal diffusion technique for synthetic acceleration of nodal S_n calculations," *Nuclear Science and Engineering*, **90**, *3*, 263–280 (1985).

[114] M. A. L. R. N. A. WILLIAM HAWKINS, TIMMIE SMITH and M. ADAMS, "Efficient Massively Parallel Transport Sweeps," in "Transactions of the American Nuclear Society," (2012), vol. 107, pp. 477–481.

[115] M. P. ADAMS, M. L. ADAMS, W. D. HAWKINS, T. SMITH, L. RAUCHWERGER, N. M. AMATO, T. S. BAILEY, and R. D. FALGOUT, "Provably Optimal Parallel Transport Sweeps on Regular Grids," in "Proc. International Conference on Mathematics and Computational Methods Applied to Nuclear Science & Engineering, Idaho," (2013).

[116] R. D. FALGOUT and U. M. YANG, "hypre: A Library of High Performance Preconditioners," in P. SLOOT, A. HOEKSTRA, C. TAN, and J. DONGARRA, editors, "Computational Science ICCS 2002," Springer Berlin Heidelberg, *Lecture Notes in Computer Science*, vol. 2331, pp. 632–641 (2002).

[117] V. HENSON and U. YANG, "BoomerAMG: a parallel algebraic multigrid solver and preconditioner," *Applied Numerical Mathematics*, **41**, *5*, 155–177 (2002).