

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

Copyright 2016 Michael Wayne Hackemack

ABSTRACT

In this dissertation, we develop improvements to the discrete ordinates (S_N) neutron transport equation using a Discontinuous Galerkin Finite Element Method (DGFEM) spatial discretization on arbitrary polytope (polygonal and polyhedral) grids compatible for massively-parallel computer architectures. Polytope meshes are attractive for multiple reasons, including their use in other physics communities and their ease in handling local mesh refinement strategies. In this work, we focus on two topical areas of research. First, we discuss higher-order basis functions compatible to solve the DGFEM S_N transport equation on arbitrary polygonal meshes. Second, we assess Diffusion Synthetic Acceleration (DSA) schemes compatible with polytope grids for massively-parallel transport problems.

We first utilize basis functions compatible with arbitrary polygonal grids for the DGFEM transport equation. We analyze four different basis functions that have linear completeness on polygons: the Wachspress rational functions, the PWL functions, the mean value coordinates, and the maximum entropy coordinates. We then describe the procedure to extend these polygonal linear basis functions into the quadratic serendipity space of functions. These quadratic basis functions can exactly interpolate monomial functions up to order 2. Both the linear and quadratic sets of basis functions preserve transport solutions in the thick diffusion limit. Maximum convergence rates of 2 and 3 are observed for regular transport solutions for the linear and quadratic basis functions, respectively. For problems that are limited by the regularity of the transport solution, convergence rates of $3/2$ (when the solution is continuous) and $1/2$ (when the solution is discontinuous) are observed. Spatial Adaptive Mesh Refinement (AMR) achieved superior convergence rates than uniform

refinement, even for problems bounded by the solution regularity. We demonstrated accuracy in the AMR solutions by allowing them to reach a level where the ray effects of the angular discretization are realized.

Next, we analyzed DSA schemes to accelerate both the within-group iterations as well as the thermal upscattering iterations for multigroup transport problems. Accelerating the thermal upscattering iterations is important for materials (*e.g.*, graphite) with significant thermal energy scattering and minimal absorption. All of the acceleration scheme analyzed, use a DGFEM discretization of the diffusion equation that is compatible with arbitrary polytope meshes: the Modified Interior Penalty Method (MIP). MIP uses the same DGFEM discretization as the transport equation. The MIP form is Symmetric Positive Definite (SPD) and efficiently solved with Preconditioned Conjugate Gradient (PCG) with Algebraic MultiGrid (AMG) preconditioning. The analysis from previous work was extended to show MIP's stability and robustness for accelerating 3D transport problems. MIP DSA preconditioning was implemented in the Parallel Deterministic Transport (PDT) code at Texas A&M University and linked with the HYPRE suite of linear solvers. Good scalability was numerically verified out to around 131K processors. The fraction of time spent performing DSA operations was small for problems with sufficient work performed in the transport sweep ($O(10^3)$ angular directions). Finally, we have developed a novel methodology to accelerate transport problems dominated by thermal neutron upscattering. Compared to historical upscatter acceleration methods, our method is parallelizable and amenable to massively parallel transport calculations. Speedup factors of about 3-4 were observed with our new method.

DEDICATION

For the greater glory of God (AMDG).

“Good ideas are not adopted automatically. They must be driven into practice with courageous impatience. Once implemented, they can be easily overturned or subverted through apathy or lack of follow-up - so a continuous effort is required.”

- Admiral Hyman G. Rickover

ACKNOWLEDGEMENTS

I would like to thank my graduate adviser and committee chair, Dr. Jean Ragusa, for all of his guidance towards the completion of this research endeavor. I would also like to thank my other committee members: Dr. Marvin Adams, Dr. Jim Morel, Dr. Nancy Amato, and Dr. Troy Becker for all of their support.

A special thank you goes to my fellow graduate students that I have collaborated with over the past five years: Jordan Evans, Joshua Hansel, Dr. Andrew Till, Donald Bruss, Carolyn McGraw, and Jonathan Madsen. We had many years of fun and frustrating, yet rewarding, research together.

I am also deeply grateful for the support of my family and friends who encouraged me along every step in this process. My parents, Jane and Norwood Hackemack, instilled in me a thirst for knowledge at a young age and always stressed dedication and perseverance in my endeavors. Most importantly, I am thankful for my wife, Johanna, and son, Peter. Their love and support was a constant motivation for me in this work. Finally, I wish to thank Peter's babysitter, Brittany Uhlenbrock. The ability to spend all day and almost every day at work was essential for the completion of this work.

This research was performed under appointment to the Rickover Graduate Fellowship Program in Nuclear Engineering sponsored by the Naval Reactors Division of the United States Department of Energy.

TABLE OF CONTENTS

	Page
ABSTRACT	ii
DEDICATION	iv
ACKNOWLEDGEMENTS	v
TABLE OF CONTENTS	vi
LIST OF FIGURES	ix
LIST OF TABLES	xix
1. INTRODUCTION	1
1.1 Motivation and Purpose of the Dissertation	1
1.2 Current State of the Problem	4
1.2.1 Background on the Multigroup DGFEM S_N Transport Equation	5
1.2.2 Diffusion Synthetic Acceleration	7
1.2.3 Polytope Grid Generation	8
1.3 Organization of the Dissertation	10
2. THE DGFEM FORMULATION OF THE MULTIGROUP S_N EQUATIONS	12
2.1 Fundamental Aspects of the Transport Equation	12
2.2 The Neutron Transport Equation	13
2.3 Energy Discretization	16
2.4 Angular Discretization	19
2.4.1 Level Symmetric Quadrature Set	26
2.4.2 Product Gauss-Legendre-Chebyshev Quadrature Set	31
2.5 Boundary Conditions	33
2.6 Spatial Discretization	37
2.6.1 Convergence Rates of the DGFEM S_N Equation	41
2.6.2 Elementary Matrices on an Arbitrary Spatial Cell	47
2.7 Solution Procedures	52
2.7.1 Angle and Energy Iteration Procedures	53
2.7.2 Spatial Solution Procedures	61
2.8 Conclusions	74

3.	FEM BASIS FUNCTIONS FOR UNSTRUCTURED POLYTOPES	76
3.1	Linear Basis Functions on 2D Polygons	77
3.1.1	Wachspress Rational Basis Functions	79
3.1.2	Piecewise Linear (PWL) Basis Functions	82
3.1.3	Mean Value Basis Functions	90
3.1.4	Maximum Entropy Basis Functions	92
3.1.5	Summary of 2D Linear Basis Functions on Polygons	103
3.2	Converting the Linear Polygonal Basis Functions to the Quadratic Serendipity Space of Functions	104
3.3	Integrating the Arbitrary 2D Polygonal Elements	115
3.4	Linear Basis Functions on 3D Polyhedra	122
3.5	Numerical Results	127
3.5.1	Two-Dimensional Exactly-Linear Transport Solutions	128
3.5.2	Two-Dimensional Exactly-Quadratic Transport Solutions	135
3.5.3	Convergence Rate Analysis by the Method of Manufactured Solutions	137
3.5.4	Convergence Rate Analysis in a Purely-Absorbing Medium	146
3.5.5	Transport Solutions in the Thick Diffusive Limit	167
3.5.6	Searchlight Problem	170
3.6	Conclusions	178
4.	DIFFUSION SYNTHETIC ACCELERATION FOR DISCONTINUOUS FINITE ELEMENTS ON UNSTRUCTURED GRIDS	185
4.1	Introduction	185
4.1.1	Review of Diffusion Synthetic Acceleration Schemes	186
4.1.2	Synthetic Acceleration Overview	188
4.2	Diffusion Synthetic Acceleration Methodologies	191
4.2.1	Simple 1-Group, Isotropic DSA Strategy	191
4.2.2	Generalized 1-Group DSA Operators	197
4.2.3	DSA Acceleration Strategies for Thermal Neutron Upscattering	201
4.3	Symmetric Interior Penalty Form of the Diffusion Equation	222
4.3.1	Elementary Stiffness Matrices	228
4.3.2	Elementary Surface Gradient Matrices	230
4.4	Modified Interior Penalty Form of the Diffusion Equation used for Diffusion Synthetic Acceleration Applications	231
4.5	Solving the MIP Diffusion Problem	233
4.6	Fourier Analysis	235
4.7	Numerical Results	242
4.7.1	SIP used as a Diffusion Solver	242
4.7.2	1 Group DSA Analysis	250
4.7.3	Scalability of the MIP DSA Preconditioner	285

4.7.4	Thermal Neutron Upscattering Acceleration	289
4.8	Conclusions	304
5.	CONCLUSIONS	306
5.1	Conclusions	306
5.2	Open Items	309
	REFERENCES	312
	APPENDIX A ADDENDUM TO CHAPTER 2	332
A.1	Detailed Description of the Spherical Harmonics Expansion of the Scattering Kernel	332
	APPENDIX B ADDENDUM TO CHAPTER 3	337
B.1	Limits of the Linear Polygonal Basis Functions	337
B.1.1	Limits of the Wachspress Coordinates	337
B.1.2	Limits of the Mean Value Coordinates	341
B.1.3	Limits of the Maximum Entropy Coordinates	346
B.2	Jacobian Transformations of the Reference Element	349
B.3	Analytical Integration of the PWL Basis Functions	352
B.3.1	2D PWL Basis Functions	352
B.3.2	3D PWL Basis Functions	354
	APPENDIX C ADDENDUM TO CHAPTER 4	357
C.1	Extended Fourier Analysis Implementation for MIP in 1D	357
C.2	1D MIP Fourier Analysis Results	362
C.3	Comparison between Modified Interior Penalty and Modified Fourier-Step DSA Schemes	366
C.4	Conservation of the SIP Diffusion Form	373

LIST OF FIGURES

FIGURE	Page
1.1 Local mesh refinement of an initial quadrilateral cell (left) leads to a degenerate pentagonal cell (right) without the use of a hanging node.	3
2.1 Interval structure of the multigroup methodology.	17
2.2 Number of spherical harmonics moments, N_{mom} , in 1D, 2D, and 3D as a function of the expansion order, p	22
2.3 Angular coordinate system for the direction $\vec{\Omega}$	27
2.4 Level-Symmetric angular quadrature sets of order (a) 2, (b) 4, (c) 8, and (d) 16.	29
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.	30
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	34
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.	35
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'	38
2.9 Definition of the trace for the upwinding scheme.	40
2.10 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).	47

2.11	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.	55
2.12	Task dependence graph for the transport sweep for a given direction without cycles.	64
2.13	Task dependence graph for the transport sweep for a given direction with cycles present.	64
2.14	Flow chart for mesh adaptation.	68
2.15	Refinement rules for a quadrilateral mesh cell.	72
2.16	Hierarchical refinement tree for a simple domain with quadrilateral cells. .	73
2.17	Bootstrapping the solution from a single unit square element onto the four daughter elements after refinement.	74
3.1	Arbitrary polygon with geometric properties used for 2D basis function generation.	78
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).	83
3.3	Contour plots of the linear Wachspress 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).	84
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). .	87
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).	88
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).	89
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).	93

3.8	Contour plots of the linear mean value 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) $(1, 0)$	94
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) $(1, 0)$	95
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) $(1, 0)$	100
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) $(1, 0)$	101
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) $(1, 0)$	102
3.13	Contour plots of the different linear basis function on the unit square located at vertex $(0, 1)$	105
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.	106
3.15	Contour plots of the different linear basis functions on the L-shaped domain. The PWL (top), mean value (middle), and maximum entropy (bottom) functions are plotted at vertices $(0, 1)$ (left) and $(1/2, 1/2)$ (right).	107
3.16	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.	108
3.17	Contour plots of the different quadratic serendipity basis function on the unit square located at vertex $(0, 1)$	116
3.18	Contour plots of the different quadratic serendipity basis function on the unit square at a mid-face node located at $(0, 1/2)$	117

3.19	Contour plots of the different quadratic serendipity basis functions on the degenerate square. The PWL (top), mean value (middle), and maximum entropy (bottom) functions are plotted at vertices $(0, 1)$ (left) and $(1/2, 1)$ (right)	118
3.20	Mapping a point on the reference triangle onto a sub-triangle of an arbitrary polygon.	119
3.21	Quadrature sets on the reference triangle of varying order.	123
3.22	Examples of spatial quadrature sets of varying order on a regular pentagon.	124
3.23	Examples of spatial quadrature sets of varying order on a regular hexagon.	125
3.24	Contour plots of the exactly-linear solution with the Wachspress basis functions.	131
3.25	Contour plots of the exactly-linear solution with the PWL basis functions.	132
3.26	Contour plots of the exactly-linear solution with the mean value basis functions.	133
3.27	Contour plots of the exactly-linear solution with the linear maximum entropy basis functions.	134
3.28	Plots of the exactly-quadratic solution with the quadratic serendipity maximum entropy basis functions.	138
3.29	Plots of the error of the exactly-quadratic solution with the quadratic serendipity Wachspress basis functions.	139
3.30	Plots of the error of the exactly-quadratic solution with the quadratic serendipity PWL basis functions.	140
3.31	Plots of the error of the exactly-quadratic solution with the quadratic serendipity mean value basis functions.	141
3.32	Plots of the error of the exactly-quadratic solution with the quadratic serendipity maximum entropy basis functions.	142
3.33	Quadratic solutions containing the x^2y^2 term for different meshes with the quadratic maximum entropy basis functions.	143

3.34 Examples of the mesh refinement for the sinusoid MMS transport problem for Cartesian (top), triangular (middle), and polygonal (bottom) meshes.	147
3.35 Example contour plots for the sinusoid problem using the linear PWL basis functions. The meshes used are those from Figure 3.34.	148
3.36 Example contour plots for the sinusoid problem using the quadratic PWL basis functions. The meshes used are those from Figure 3.34. . .	149
3.37 Convergence rates for the sinusoid MMS problem on a Cartesian mesh.	150
3.38 Convergence rates for the sinusoid MMS problem on an ordered-triangular mesh.	150
3.39 Convergence rates for the sinusoid MMS problem on a regular polygonal mesh.	151
3.40 Convergence rates for the 2D Gaussian MMS problem using the PWL basis functions.	151
3.41 Convergence rates for the 2D Gaussian MMS problem using the mean value basis functions.	152
3.42 Convergence rates for the 2D Gaussian MMS problem using the maximum entropy basis functions.	152
3.43 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. . .	153
3.44 Mesh types used for the purely-absorbing medium problem.	156
3.45 Example solution of the purely-absorbing medium case with left-face incidence and $\sigma_t = 1$ using the linear PWL basis functions.	157
3.46 Example solution of the purely-absorbing medium case with left-face and top-face incidence and $\sigma_t = 1$ using the linear PWL basis functions.	158
3.47 Convergence rates for the pure absorber problem with left-face incidence on triangular meshes with different values of σ_t	159
3.48 Convergence rates for the pure absorber problem with left-face incidence on Cartesian meshes with different values of σ_t	160

3.49	Convergence rates for the pure absorber problem with left-face incidence on polygonal meshes with different values of σ_t	161
3.50	Convergence rates for the pure absorber problem with left-face incidence on split-polygonal meshes with different values of σ_t	162
3.51	Convergence rates for the pure absorber problem with left-face and top-face incidence on triangular meshes with different values of σ_t	163
3.52	Convergence rates for the pure absorber problem with left-face and top-face incidence on Cartesian meshes with different values of σ_t . . .	164
3.53	Convergence rates for the pure absorber problem with left-face and top-face incidence on polygonal meshes with different values of σ_t . . .	165
3.54	Convergence rates for the pure absorber problem with left-face and top-face incidence on split-polygonal meshes with different values of σ_t .	166
3.55	Diffusion solution representing the thick diffusion limit problem with linear (top) and quadratic (bottom) mean value basis functions. . . .	171
3.56	Transport solutions of the thick diffusion limit problem using the Wachspress basis functions for varying values of the scaling parameter, ϵ	172
3.57	Transport solutions of the thick diffusion limit problem using the PWL basis functions for varying values of the scaling parameter, ϵ	173
3.58	Transport solutions of the thick diffusion limit problem using the mean value basis functions for varying values of the scaling parameter, ϵ	174
3.59	Transport solutions of the thick diffusion limit problem using the maximum entropy basis functions for varying values of the scaling parameter, ϵ	175
3.60	Convergence rates for the diffusion limit problem on a Cartesian mesh.	176
3.61	Convergence rates for the diffusion limit problem on a polygonal mesh.	176
3.62	Initial mesh configuration for the searchlight problem before any refinement cycles.	177
3.63	Convergence history for the searchlight problem using the PWL basis functions.	178

3.64	Convergence history for the searchlight problem using the mean value basis functions.	179
3.65	Convergence history for the searchlight problem using the maximum entropy basis functions.	179
3.66	Exiting angular flux on the right boundary with uniform refinement. . .	180
3.67	Exiting angular flux on the right boundary with AMR and the PWL basis functions with different mesh irregularities.	181
3.68	Exiting angular flux on the right boundary with AMR and the mean value basis functions with different mesh irregularities.	182
3.69	Exiting angular flux on the right boundary with AMR and the maximum entropy basis functions with different mesh irregularities.	183
4.1	Fourier domain for 2D quadrilateral cells or an axial slice of 3D hexahedral cells in a regular grid.	237
4.2	Representation of the periodic boundary conditions used for Fourier analysis for a 2-cell geometric layout.	237
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) $[\lambda_x, \lambda_y] = [0, 1/4]^2$	241
4.4	2D polygonal grids to be extruded for the 3D SIP calculations.	244
4.5	Extrusion of the different polygonal meshes.	245
4.6	Axial slice showing the contours for the linear solution of the SIP diffusion form.	247
4.7	Error in the SIP diffusion form. Additional results using a CFEM diffusion form are provided for comparison.	249
4.8	Fourier analysis of the 2D MIP form with $c = 4$ and using the linear (top) and quadratic (bottom) Wachspress basis functions.	252
4.9	Fourier analysis of the 2D MIP form with $c = 4$ and using the linear (top) and quadratic (bottom) PWL basis functions.	253
4.10	Fourier analysis of the 2D MIP form with $c = 4$ and using the linear (top) and quadratic (bottom) mean value basis functions.	254

4.11 Fourier analysis of the 2D MIP form with $c = 4$ and using the linear (top) and quadratic (bottom) maximum entropy basis functions.	255
4.12 Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL basis functions and LS_2 quadrature.	256
4.13 Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL basis functions and LS_4 quadrature.	257
4.14 Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL basis functions and LS_8 quadrature.	258
4.15 Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL basis functions and LS_{16} quadrature.	259
4.16 Fourier spectral radius of the 3D MIP form with $c = 1$ (top) and $c = 4$ (bottom).	260
4.17 Fourier spectral radii for MIP form with LS_8 quadrature on cells with different aspect ratios.	262
4.18 Comparison between the numerical spectral radii and the theoretical Fourier Analysis spectral radii on the unit cube with $c = 1$ (top) and $c = 4$ (bottom).	263
4.19 Fourier wave number distribution for the 2D PHI problem with $\sigma = 10$ and $c = 0.9$ and different level-symmetric quadratures.	267
4.20 Fourier wave number distribution for the 2D PHI problem with $\sigma =$ 10^4 and $c = 0.9999$ and different level-symmetric quadratures.	268
4.21 Geometry description for the Iron-Water problem.	269
4.22 Initial mesh for the Iron-Water problem.	270
4.23 Meshes for the Iron-Water problem using the linear PWL coordinates and LS4 quadrature.	281
4.24 Meshes for the Iron-Water problem using the quadratic PWL coordinates and LS4 quadrature.	282
4.25 Meshes for the Iron-Water problem using the linear PWL coordinates and S_{24}^2 PGLC quadrature.	283

4.26	Meshes for the Iron-Water problem using the quadratic PWL coordinates and S_{24}^2 PGLC quadrature.	284
4.27	Timing data for the MIP DSA implementation in PDT using HYPRE on VULCAN.	287
4.28	Fraction of time spent performing DSA based on number of total angles.	288
4.29	Experimental setup of the IM1 problem.	290
4.30	Spectral shape of the infinite medium iteration matrices of the IM1 problem materials for the TG and MTG schemes.	292
4.31	Spectral shape of the infinite medium iteration matrices of the IM1 problem materials for the MJA and MJIA schemes.	293
4.32	Flat mode eigenvalues for the TG and MTG schemes.	296
4.33	Flat mode eigenvalues for the MJA and MJIA schemes.	297
4.34	Fourier wave form distribution for a PHI-like problem of the TG method with Air and HDPE.	298
4.35	Configuration of the 2D variant of the IM1 problem. The materials are restricted to only air, HDPE, graphite, and an AmBe source.	299
4.36	Configuration of the IM1 problem with the outer layers of air removed.	302
A.1	Legendre polynomials of degrees 0 through 5.	333
B.1	Arbitrary polygon with geometric properties used for 2D basis function generation.	338
B.2	Mapping a point on the reference triangle onto a sub-triangle of an arbitrary polygon.	349
C.1	Spectral radius for the 1D MIP form using $c = 1$	363
C.2	Spectral radius for the 1D MIP form using $c = 2$	364
C.3	Spectral radius for the 1D MIP form using $c = 4$	364
C.4	Spectral radius for the 1D MIP form using $c = 8$	365
C.5	Spectral radius for the 1D MIP form using $c = 64$	365

- C.6 Comparison between the numerical spectral radii and the theoretical Fourier Analysis spectral radii for the M4S scheme on the unit square. 368
- C.7 Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL basis functions and LS_2 quadrature.369
- C.8 Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL basis functions and LS_4 quadrature.370
- C.9 Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL basis functions and LS_8 quadrature.371
- C.10 Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL basis functions and LS_{16} quadrature.372

LIST OF TABLES

TABLE	Page
2.1 2D angle mapping from the first quadrant into the other 3 quadrants.	25
2.2 3D angle mapping from the first octant into the other 7 octants. . .	26
2.3 Average number of iterations required to reduce SI error by 1 order of magnitude for different SR values.	60
3.1 Summary of the properties of the 2D coordinate systems used on polygons.	104
3.2 L_2 -norm of the error in the quadratic solution containing the x^2y^2 term for the different quadratic serendipity basis functions on differ- ent mesh types.	137
4.1 Eigenproblems to compute the spectral shape of each upscatter ac- celeration method.	221
4.2 Iteration terms for the Two-Grid, Modified Two-Grid, and Multi- group Jacobi Acceleration methods.	222
4.3 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	228
4.4 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	228
4.5 Spectral radius for the 2D PHI problem with the PWL basis func- tions and LS2 quadrature.	265
4.6 Spectral radius for the 2D PHI problem with the PWL basis func- tions and LS4 quadrature.	265
4.7 Spectral radius for the 2D PHI problem with the PWL basis func- tions and LS8 quadrature.	266

4.8	Spectral radius for the 2D PHI problem with the PWL basis functions and LS16 quadrature.	266
4.9	Material definitions and physical properties for the Iron-Water problem.	270
4.10	DSA counts based on preconditioner type for the Iron-Water problem with LS_4 quadrature, linear basis functions, and solution reinitialization.	273
4.11	DSA counts based on preconditioner type for the Iron-Water problem with LS_4 quadrature, linear basis functions, and solution bootstrapping.	274
4.12	DSA counts based on preconditioner type for the Iron-Water problem with LS_4 quadrature, quadratic basis functions, and solution reinitialization.	275
4.13	DSA counts based on preconditioner type for the Iron-Water problem with LS_4 quadrature, quadratic basis functions, and solution bootstrapping.	276
4.14	DSA counts based on preconditioner type for the Iron-Water problem with S_{24}^2 PGLC quadrature, linear basis functions, and solution reinitialization.	277
4.15	DSA counts based on preconditioner type for the Iron-Water problem with S_{24}^2 PGLC quadrature, linear basis functions, and solution bootstrapping.	278
4.16	DSA counts based on preconditioner type for the Iron-Water problem with S_{24}^2 PGLC quadrature, quadratic basis functions, and solution reinitialization.	279
4.17	DSA counts based on preconditioner type for the Iron-Water problem with S_{24}^2 PGLC quadrature, quadratic basis functions, and solution bootstrapping.	280
4.18	Infinite medium spectral radii of the IM1 materials for the three thermal neutron upscattering methods. We include both the unaccelerated (U) and accelerated (A) cases.	294
4.19	Sweep count and timing data for the 2D IM1 variant problem using Two-Grid Acceleration.	300

4.20	Sweep count and timing data for the 2D IM1 variant problem using Modified Two-Grid Acceleration.	300
4.21	Sweep count and timing data for the 2D IM1 variant problem using Multigroup Jacobi Acceleration.	300
4.22	Sweep count and timing data for the 3D brick IM1 problem using Two-Grid Acceleration.	303
4.23	Sweep count and timing data for the 3D brick IM1 problem using Modified Two-Grid Acceleration.	303
4.24	Sweep count and timing data for the 3D brick IM1 problem using Multigroup Jacobi Acceleration.	303

1. INTRODUCTION

1.1 Motivation and Purpose of the Dissertation

Accurate solutions of the neutral particle transport equation are important for multiple fields, including medical imaging, radiotherapy, nuclear power, and other industrial applications. As computing systems continue to advance, the fidelity of these solutions continues to increase as well. Currently, computational resources on High-Performance Computers (HPC) are at the petaflop level. At this time, there is currently a motivation within the United States Department of Energy (DOE) to one day achieve exascale levels of computing [1]. However, this move to exascale computing levels will require significantly different computer architectures than what are currently utilized. These changes will include less available memory per process node and will most likely result in a higher frequency of faults or performance fluctuations. Therefore, future parallel algorithms and methods that will make use of exascale computing levels need to be observant of these architectural changes.

Traditionally, solutions of the neutral particle transport equation have required the use of the most advanced computer hardwares (available at a given time) due to the fidelity required for each dimension of their high-dimensional phase space. This means that further development of transport methods on modern HPCs is beholden to the exascale level limitations. At the exascale level, high-fidelity numerical transport solutions could have up to $O(10^9)$ unknowns in space, $O(10^5 - 10^6)$ unknowns in angle, and $O(10^2 - 10^3)$ unknowns in energy. Due to the decreased availability of memory at exascale levels, the limiting case for data storage for the numerical transport calculations will lead to only a small number of mesh cells per process location (all the way down to 1 mesh cell per process). Therefore, it becomes necessary

to maximize the process work per location compared to data passing and retrieval operations.

With these limitations in mind, this dissertation seeks to advance the state-of-the-art concerning the spatial discretization of the transport equation for use on massively-parallel computer architectures. The discretization that will be used in this work is the *Discontinuous Galerkin Finite Element Method* (DGFEM) in space and *discrete ordinates* (S_N) in angle [2, 3]. Specifically, we seek to marry three distinct topical areas together for this work: polytope (polygons and polyhedra in 2D and 3D, respectively) spatial discretizations, higher-order Finite Element Method (FEM) basis functions on polytope grids, and Diffusion Synthetic Acceleration (DSA) schemes compatible with arbitrary meshes. All three of these points can provide higher-fidelity solutions while being amenable to exascale level computing architectures.

We can summarize the benefits of using polytope meshes as the following:

1. Polytope mesh cells are now being employed in other physics communities - most notably computational fluid dynamics (CFD) [4] and solid mechanics [5];
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 simplicial meshing

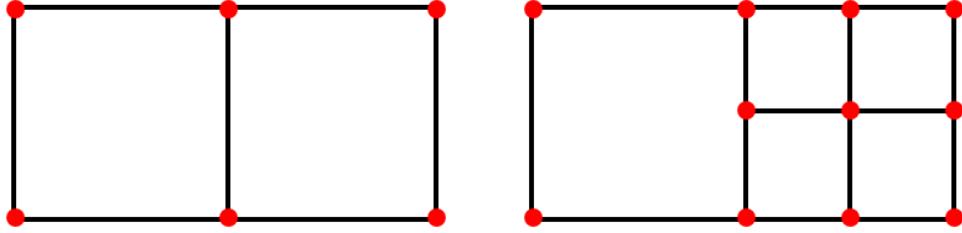


Figure 1.1: Local mesh refinement of an initial quadrilateral cell (left) leads to a degenerate pentagonal cell (right) without the use of a hanging node.

techniques across processor sets in parallel;

6. Hanging nodes from non-conforming meshes, like those that naturally arise from locally refined/adapted meshes as seen in Figure 1.1, are not necessary.

This means that, besides being able to accurately model complicated geometries, polytope meshes have benefits that are amenable to future HPC architectures. Specifically, they can reduce cell counts while maintaining equivalent accuracy. Furthermore, they allow for more general cells on non-conforming grids that can arise from either parallel mesh generation or local refinement procedures.

Higher-order FEM basis functions provide a two-fold benefit when used in conjunction with HPCs. First, they provide a richer interpolatory space for the FEM basis functions resulting in increased convergence rates for the discretized solution [6]. Second, they can maximize the parallel efficiency on these proposed exascale machines. Low-order spatial discretizations of the DGFEM transport equation may require more mesh cells in the computational domain compared to the number of mesh cells of high-order spatial discretizations. For the DGFEM S_N method, this equates to a greater number of independent linear solves for the low-order discretizations. Since the current and future computational bottlenecks for neutron transport are memory-access related, then higher-order FEM basis functions will be more com-

putationally efficient. This is because the higher-order basis functions can perform more on-process work before memory-access routines are needed.

DSA schemes have been an integral component for solving the DGFEM transport equation for highly diffusive problems. These diffusive neutron problems are characterized as being dominated by scattering with minimal leakage and absorption. Many schemes have been proposed over the years to properly discretize the diffusion operator to be consistent with the discretization of the transport equation. In this work, we seek to study a DSA scheme that is compatible with arbitrary polytope meshes and amenable to massively-parallel computations.

In this work, we will answer three specific open questions regarding solutions of the DGFEM S_N transport equations:

1. Can higher-order 2D polygonal basis functions be used to solve the DGFEM transport equation?
2. Can an efficient and robust DSA scheme be used on arbitrary grids while maintaining scalability to high process counts?
3. Can a parallelizable variant of the Two-Grid acceleration method be derived to accelerate problems dominated by thermal neutron upscattering?

The methodology, implementation, and results pertaining to these three items are provided in Chapters 3 and 4.

1.2 Current State of the Problem

We now provide a brief overview of what constitutes the state-of-the-art in the different topical areas related to this dissertation work. First, we give background information on the Multigroup DGFEM S_N transport equation in Section 1.2.1. Then, we provide a brief explanation of the necessity of DSA schemes in Section

1.2.2. Finally, we detail how the polytope meshes in this work will be generated in Section 1.2.3.

1.2.1 Background on the Multigroup DGFEM S_N Transport Equation

The neutral particle transport equation has a high-dimensional phase space, consisting of 3 spatial variables, 1 energy variable, and 2 angular variables. Many different methodologies have been developed over the years to efficiently discretize and solve each of these variables. We now briefly detail the discretization methods that will be utilized in this work.

For the energy variable, we will utilize the multigroup method since it is the only widely-used energy discretization scheme in deterministic transport [7, 8]. With the multigroup approximation, the energy domain is broken up into intervals (groups). Then cross section quantities are computed for each group as weighted integrals with some approximate weighting spectrum. This process requires a combination of nuclear data libraries such as the Evaluated Nuclear Data File (ENDF) [9, 10], the Japanese Evaluated Nuclear Data Library (JENDL) [11], and the Joint Evaluated Fission and Fusion Project (JEFF) [12], as well as high-fidelity processing software such as NJOY [13, 14] and AMPX [15] to form the multigroup cross sections. Work is continuously being performed to quantify uncertainties in these cross sections and their effects to solution accuracy [16, 17].

For the angle variable in the transport equation, there are multiple discretization methods we could employ. For this work, we choose to use the discrete ordinates (S_N) scheme [18, 19]. The S_N method is a collocation discretization scheme where the transport equation is solved along predetermined directions of an angular quadrature set. This method differs strongly from modal expansion schemes such as the P_N or Simplified P_N (SP_N) methods [8, 20]. S_N methods are widely used for a variety of

applications, but they can suffer from ray effects arising from streaming paths in a heterogeneous problem [21]. First-collision source approaches are currently used to mitigate these ray effects [22, 23].

This just leaves the choice for spatial discretization remaining. There have been many spatial schemes for the transport equation that have been developed, including finite element methods, finite difference/volume methods, characteristic methods, collision probability methods, and nodal methods [24, 8, 25, 26, 27, 28]. In this work, we choose to employ the DGFEM discretization for our transport problems [24, 29]. This method was originally derived for neutral particle transport problems in the early 1970's. It was immediately employed to solve the transport equation on triangular meshes in the TRIPLET [30] and TRIDENT [31, 32] codes. TRIPLET utilized approximations with various polynomial orders, but TRIDENT was restricted to only linear DGFEM. Subsequent works in 3D have been largely restricted to linear DGFEM approximations on tetrahedral and hexahedral mesh cells [33, 3]. For the most part, only linear basis functions have been used with the DGFEM transport equation as the researchers wanted to focus on accuracy in the thick diffusive limit and the robustness of the spatial discretization. Recently, the works of Wang and Ragusa have analyzed the convergence rates of the DGFEM S_N equations on unstructured triangular meshes with higher-order basis functions [34, 35].

Traditionally, the DGFEM S_N transport equation has been solved on simplicial (triangles and tetrahedra) or tensor-based meshes (quadrilaterals and hexahedra) using linear basis functions. Very recently, there have been advances made to solve these equations on polygonal and polyhedral grids [36, 37, 38, 39]. However, this has been relegated to only linear basis functions since the use of polytope finite elements is still in its infancy. Within the past 10 years, the applied mathematics communities have made great strides in the fields of interpolatory functions on arbitrary polytopes

[40]. Even more recently, work has begun on higher-order interpolation schemes [41]. This means that the capacity to use higher-order basis functions for the DGFEM S_N transport equation on polytope meshes is now realizable.

1.2.2 *Diffusion Synthetic Acceleration*

For transport calculations involving highly-diffusive media, Diffusion Synthetic Acceleration (DSA) becomes an effective scheme to precondition the DGFEM transport iterations. For these optically thick problems, the traditional Richardson or Source Iteration (SI) schemes become ineffective, and prohibitively slow convergence arises [42]. There are many different synthetic acceleration schemes that have been proposed over the years, but DSA is the most popular and common method for highly-diffusive problems.

Unfortunately, a continuous finite element (CFEM) discretization of the standard reaction-diffusion equation (the simplest option) is ineffective on multi-dimensional, unstructured meshes. This is because we are not guaranteed stability when it is used in conjunction with the DGFEM transport equation due to the inconsistency of the spatial discretizations. It was realized that the discretization of the diffusion equations needed to be consistently derived from the discretized transport equation [43, 44, 45, 46, 47]. Unfortunately, for higher spatial dimensions, this “fully-consistent” DSA scheme constituted a hybrid DGFEM form of the P_1 equations that is computationally expensive to solve. Therefore, much work has been performed to develop “partially-consistent” DSA schemes that are stable and robust for many mesh types [48, 49, 50]. To date, the Modified Interior Penalty (MIP) DSA scheme is the only discontinuous DSA scheme that has been shown to be unconditionally stable and effective on unstructured meshes while still being computationally efficient [50, 51].

1.2.3 Polytope Grid Generation

Since this dissertation work deals with the solution of the transport equation on polytope meshes, we next describe how these grids can be generated. First, we give a brief discussion on the use of bounded Voronoi diagrams to generate polytope meshes. For this work, we will focus on using Voronoi diagrams to form 2D polygonal meshes. Then, these polygonal meshes can be extruded to form 3D polyhedral meshes with prismatic cells. Finally, we give a brief overview on the use of spatial Adaptive Mesh Refinement (AMR) for the DGFEM S_N transport equation. Instead of using hanging nodes, the refined mesh cells will simply form degenerate polytopes.

1.2.3.1 Voronoi Mesh Generation

Traditionally, 2D and 3D FEM calculations have been performed on simplicial (triangles and tetrahedra) and tensor-based meshes (quadrilaterals and hexahedra), respectively. In fact, it is still a standard practice to refer to any type of mesh as a *triangulation* in some communities [6]. Many different mesh generation software have been developed to build these simple grids [52, 53, 54, 55]. However, multiple fields including *computational fluid dynamics* (CFD) and *solid mechanics* are now finding benefits in utilizing polytope meshes for their calculations [4, 5].

However, polytope mesh generation is still in its infancy [5, 56, 57]. In general, polytope meshes have infinite variety in their topological shape and closed-form solutions are impossible. Therefore, polytope mesh generation is defined as a minimization of an energy residual. This is typically performed with Voronoi tessellation (diagram), where the iterative scheme employed is Lloyd's algorithm [58, 59]. There are two main classes of Voronoi mesh generation. The first is based on uniform random sampling known as Maximal Poisson-Disk Sampling (MPS) [57, 60, 61]. MPS inserts Voronoi seed points into the domain without bias and is known as

dart-throwing in computer graphics. The second class of Voronoi mesh generation is known as Centroidal Voronoi Tessellation (CVT) [62, 63]. CVT is a Voronoi meshing methodology where the seed points are the centroids of the Voronoi regions. These Voronoi regions correspond to the elements (cells) in the resulting polytope mesh. A lightweight CVT algorithm written in MATLAB and based on Lloyd’s algorithm is given in [64].

1.2.3.2 Adaptive Mesh Refinement (AMR) for the DGFEM S_N Transport Equation

As mentioned previously, an alternative method to generating polytope meshes is through the use of spatial Adaptive Mesh Refinement (AMR). Spatial AMR calculations can result in the generation of an irregular mesh (one with hanging nodes) at any refinement level [65, 66]. Looking at the right image in Figure 1.1, the colinear vertex along the right face of the unrefined quadrilateral is an example of a hanging node. To the unrefined quadrilateral, this hanging node is not a degree of freedom. However, careful integration must be performed along the face so that the across-face connectivity is preserved for the refined cells. This leads to additional implementation overhead to properly account for the irregular mesh. If we instead use polytope mesh cells, then the unrefined quadrilateral cell with a hanging node can instead be viewed as a degenerate pentagon with two colinear faces. This can ease the implementation burden of AMR methods, though we note that not all basis function types can handle colinear faces.

AMR methods have become commonplace across many scientific and engineering fields [66, 65, 67, 68, 69, 70], but are still in their infancy in their use with the DGFEM transport equation [71, 72, 73, 74, 75, 76]. There are three key factors that are required to yield efficient and successful AMR results:

1. The number of unknowns from the mesh adaptation compared to those result-

ing from uniform refinement is orders of magnitude smaller while still attaining similar order of accuracy.

2. The meshes resulting from the iterative refinement strategy are close to “optimal” and are represented hierarchically as a nest.
3. An efficient *a posteriori* method can be used to accurately determine the relative error distribution of a numerical solution on a mesh.

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 introduction 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 conclude by providing the methodology to be used to solve the full phase-space of the transport problem, including methodology for massively-parallel transport sweeping.

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 chapters.

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 A: addendum to Section 2, corresponding to additional material relating to the multigroup S_N equations.
- Appendix B: addendum to Section 3, corresponding to additional material relating to the various polytope coordinate systems to be utilized as finite element basis functions.
- Appendix C: 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 [77]. 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:

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

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, $\left(\frac{\partial u}{\partial t} \right)_{coll}$ [78].

For neutral particle transport, the following assumptions [79] 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:

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

with the following, general boundary condition:

$$\Psi(\vec{r}, E, \vec{\Omega}, t) = \Psi^{inc}(\vec{r}, E, \vec{\Omega}, t) + \int dE' \int d\Omega' \gamma(\vec{r}, E' \rightarrow E, \vec{\Omega}' \rightarrow \vec{\Omega}, t) \Psi(\vec{r}, E', \vec{\Omega}', t) .$$

for $\vec{r} \in \partial\mathcal{D}^- \left\{ \partial\mathcal{D}, \vec{\Omega} \cdot \vec{n} < 0 \right\}$

(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 note that we have omitted the delayed neutron precursors from Eq. (2.2) because our work is restricted to steady-state problems.

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

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

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

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

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

$$\mathbf{L}\Psi = \mathbf{F}\Psi + \mathbf{S}\Psi + \mathbf{Q}, \quad (2.6)$$

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:

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

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

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

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 left-hand-side and right-hand-side operators of Eq. (2.8). 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 [80].

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 [7, 19]. 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: $E_G < E_{G-1} < \dots < E_1 < E_0$.

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

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

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.

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 analogous 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

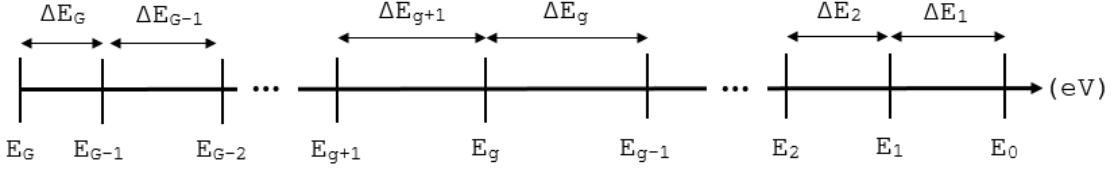


Figure 2.1: Interval structure of the multigroup methodology.

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

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

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

where

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

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 distribution is part of the solution space for which we are trying to solve. 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 :

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

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):

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

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:

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

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 ,

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

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

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

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:

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

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 ,

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

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

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

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 A). 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:

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

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's scattering anisotropy 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,

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

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 simplifies the representation 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

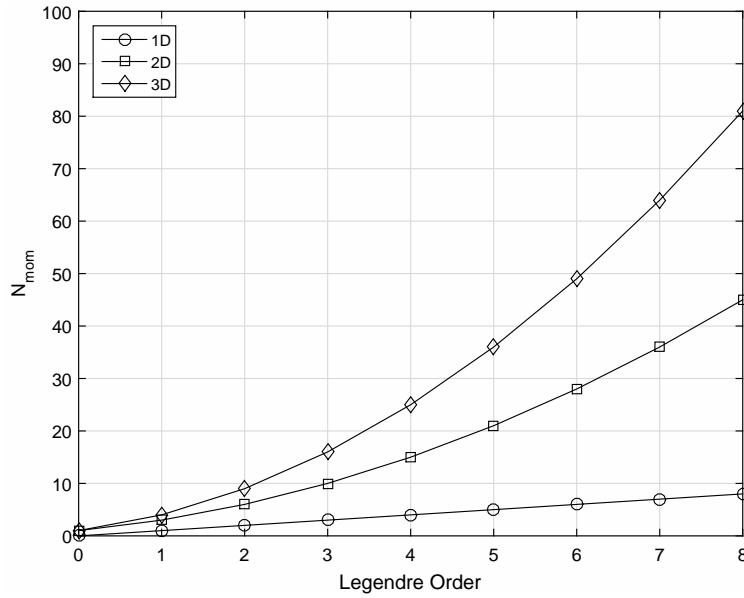


Figure 2.2: Number of spherical harmonics moments, N_{mom} , in 1D, 2D, and 3D as a function of the expansion order, p .

the 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,

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

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

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

as well as the *Lagrange* property,

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

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

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

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

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

$$k = 1, \dots, M$$

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

angular flux moments:

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

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):

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

$$m = 1, \dots, M$$

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 $\vec{0}$: $\sum_m w_m (\vec{\Omega}_m)^n = \vec{0}$ ($n = 1, 3, 5, \dots$).
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 [19, 18]. 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

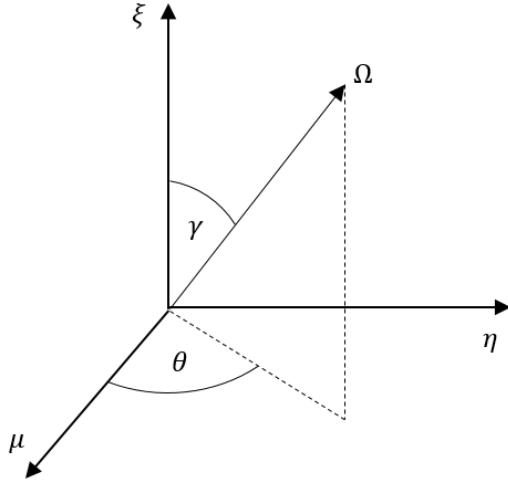


Figure 2.3: Angular coordinate system for the direction $\vec{\Omega}$.

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:

$$\mu_i = \eta_i = \xi_i, \quad i \in (1, N/2) \quad (2.32)$$

and the individual angular directions are composed of combinations of these ordinates.

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 S_2 set ($N = 2$), there is exactly one direction cosine with no degrees of freedom. This requires that $\mu_1^2 = 1/3$ for the S_2 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):

$$\mu_i^2 = \mu_{i-1}^2 + \Delta \quad (2.33)$$

where the spacing constant, Δ , has the form:

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

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. Therefore, there is some flexibility in the level-symmetric quadrature set based on the selection of μ_1 . For this work, we choose to select values of μ_1 in conformance with the LQ_N

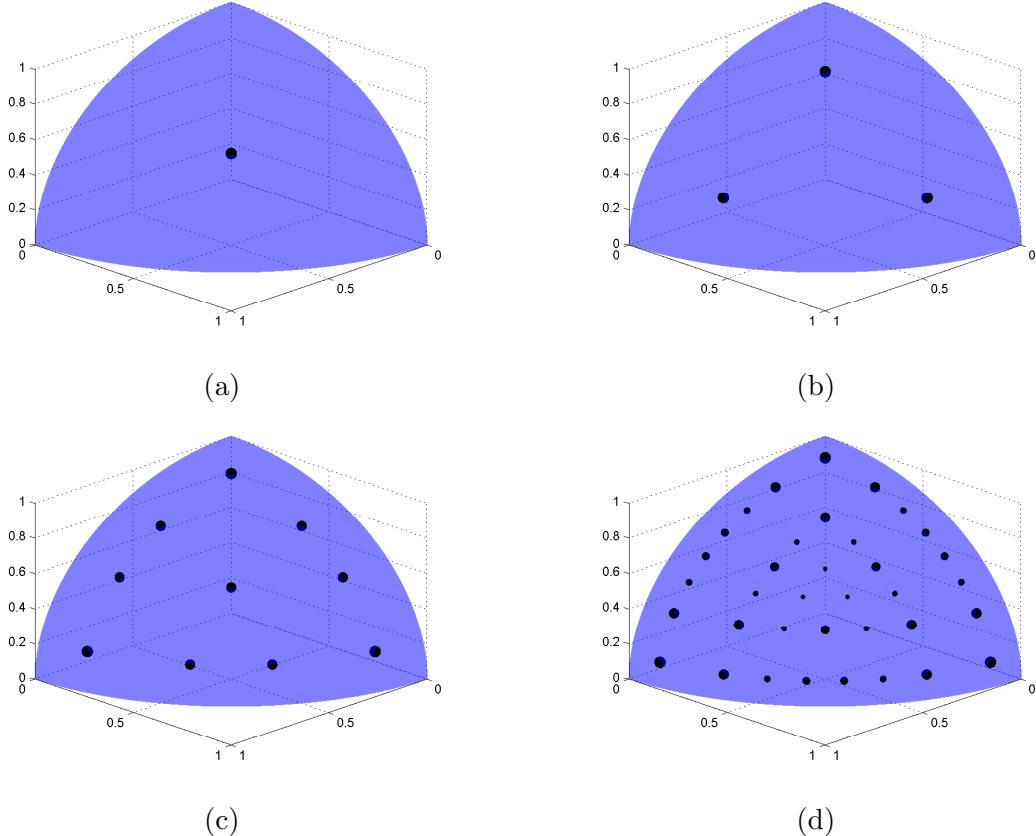


Figure 2.4: Level-Symmetric angular quadrature sets of order (a) 2, (b) 4, (c) 8, and (d) 16.

quadrature set because they can exactly integrate the polynomials of the Legendre expansion of the scattering cross sections [81]. We finally note that the weights of the LQ_N set become negative for $N \geq 20$.

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.

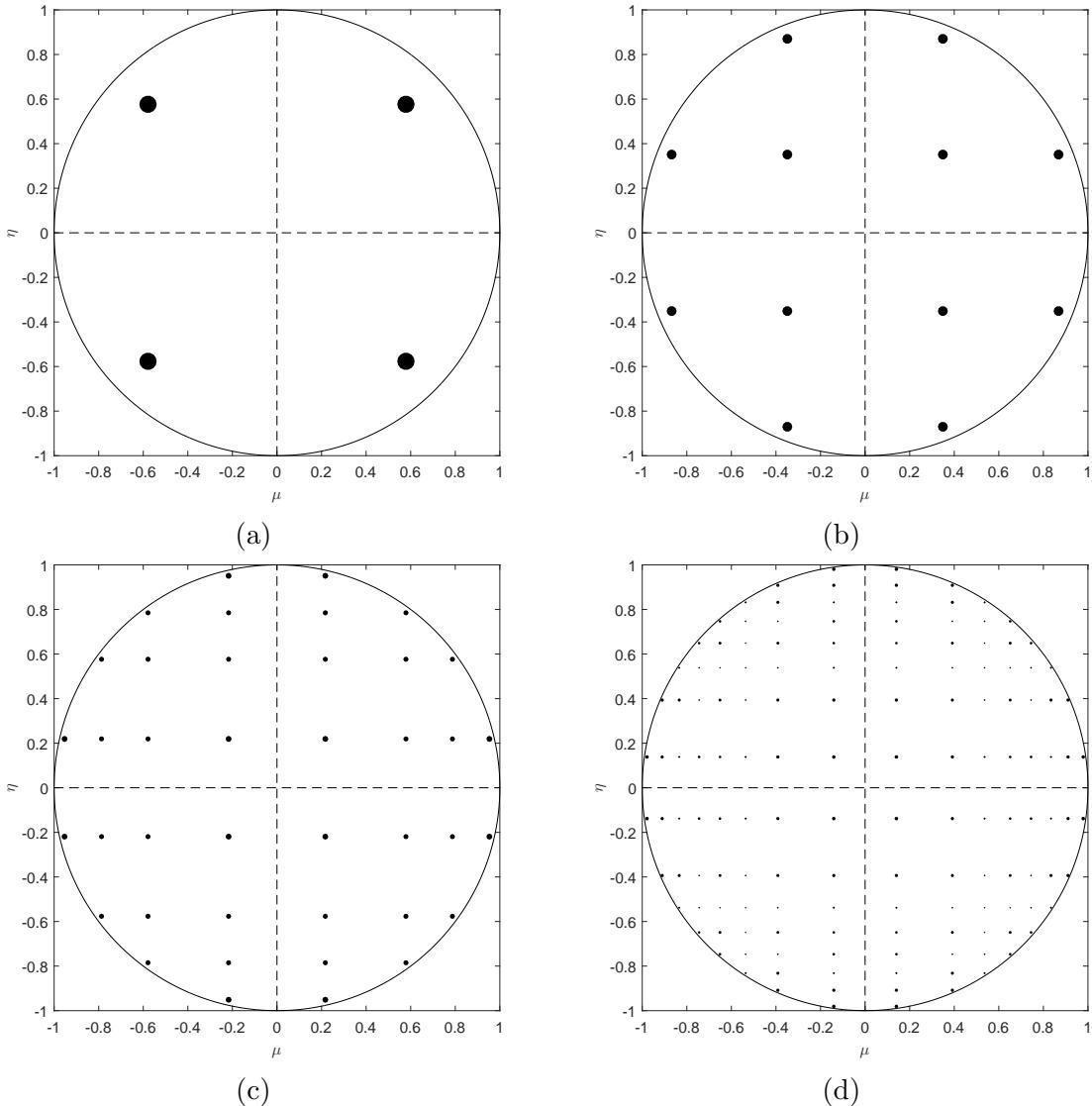


Figure 2.5: Projection of the 3D Level-Symmetric angular quadrature set with orders (a) 2, (b) 4, (c) 8, and (d) 16 onto the x-y space on the unit circle.

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 [82]. 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° rotational invariance within the primary octant; however, we still maintain octant-to-octant symmetry via mapping;
- Has more control over the placement of the angular directions within the primary octant;
- Quadrature weights are aligned with the polar level;
- Has strictly positive weights for all polar and azimuthal combinations;
- Integrates exactly many of the spherical harmonics functions.

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 [83]. 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

$$\theta_m = \frac{2m-1}{4A}\pi \quad \text{and} \quad w_m = \frac{\pi}{2A}, \quad (2.35)$$

respectively.

For the polar direction, a Gauss-Legendre quadrature set is used [84]. 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 cosines of the polar nodes, ξ , of our quadrature set are the positive roots of the $2P$ -order Legendre polynomials taken over the interval $[-1, 1]$. In this case, we simply discard the negative roots. The corresponding Legendre weights are given by the following formula,

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

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,

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

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 :

$$\begin{aligned} (\vec{\Omega}_m \cdot \vec{\nabla} + \sigma_{t,g}) \Psi_{m,g} &= \sum_{g'=1}^G \sum_{p=0}^{N_p} \frac{2p+1}{4\pi} \sigma_{s,p}^{g' \rightarrow g} \sum_{n=-p}^p \Phi_{p,n,g'} Y_{p,n}(\vec{\Omega}_m) \\ &\quad + \frac{\chi_g}{4\pi} \sum_{g'=1}^G \nu \sigma_{f,g'} \Phi_{g'} + Q_{m,g} \end{aligned}, \quad (2.38)$$

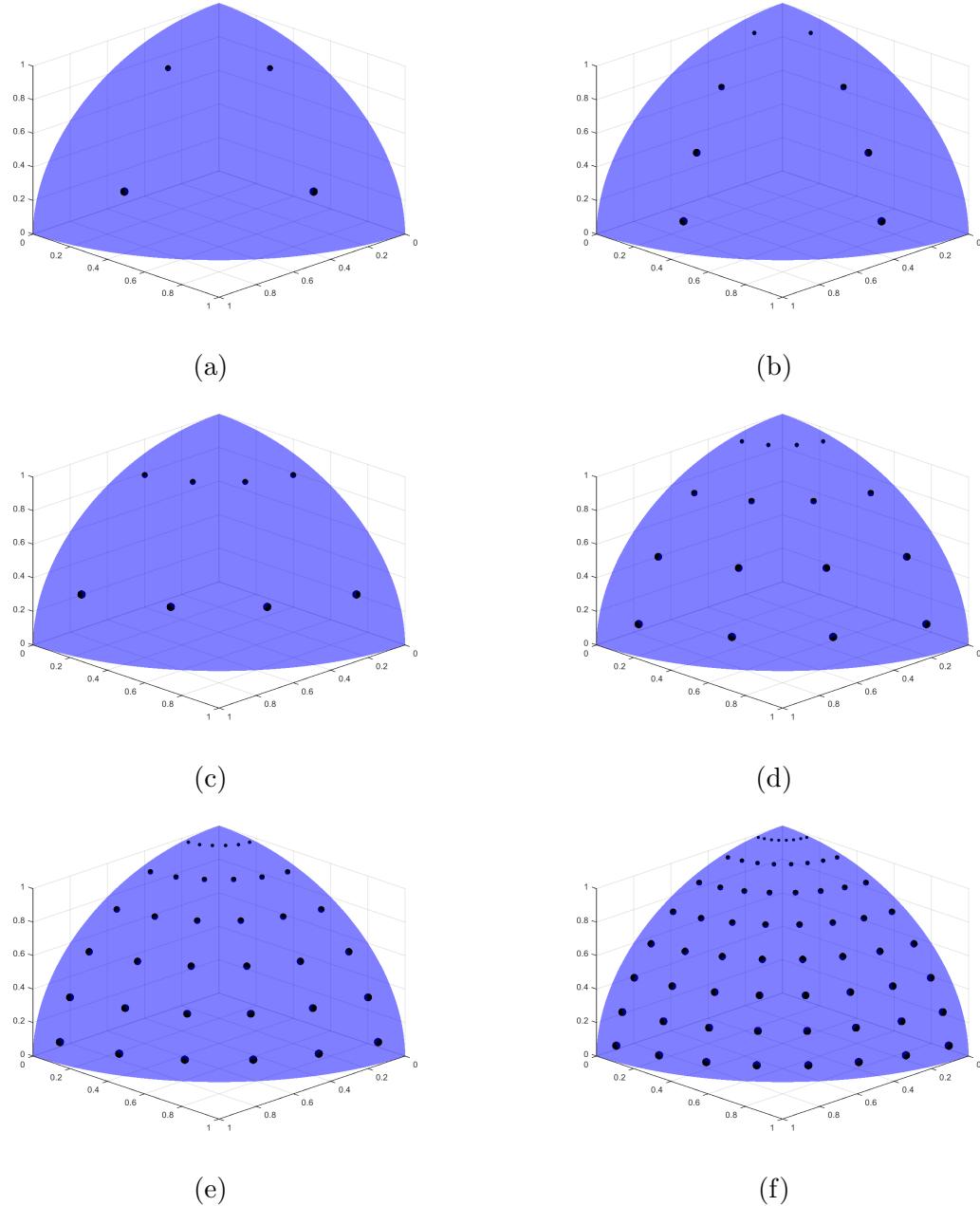


Figure 2.6: Product Gauss-Legendre-Chebyshev angular quadrature set with orders:
 (a) S_2^2 , (b) S_4^4 , (c) S_2^4 , (d) S_4^4 , (e) S_6^6 , and (f) S_8^8 .

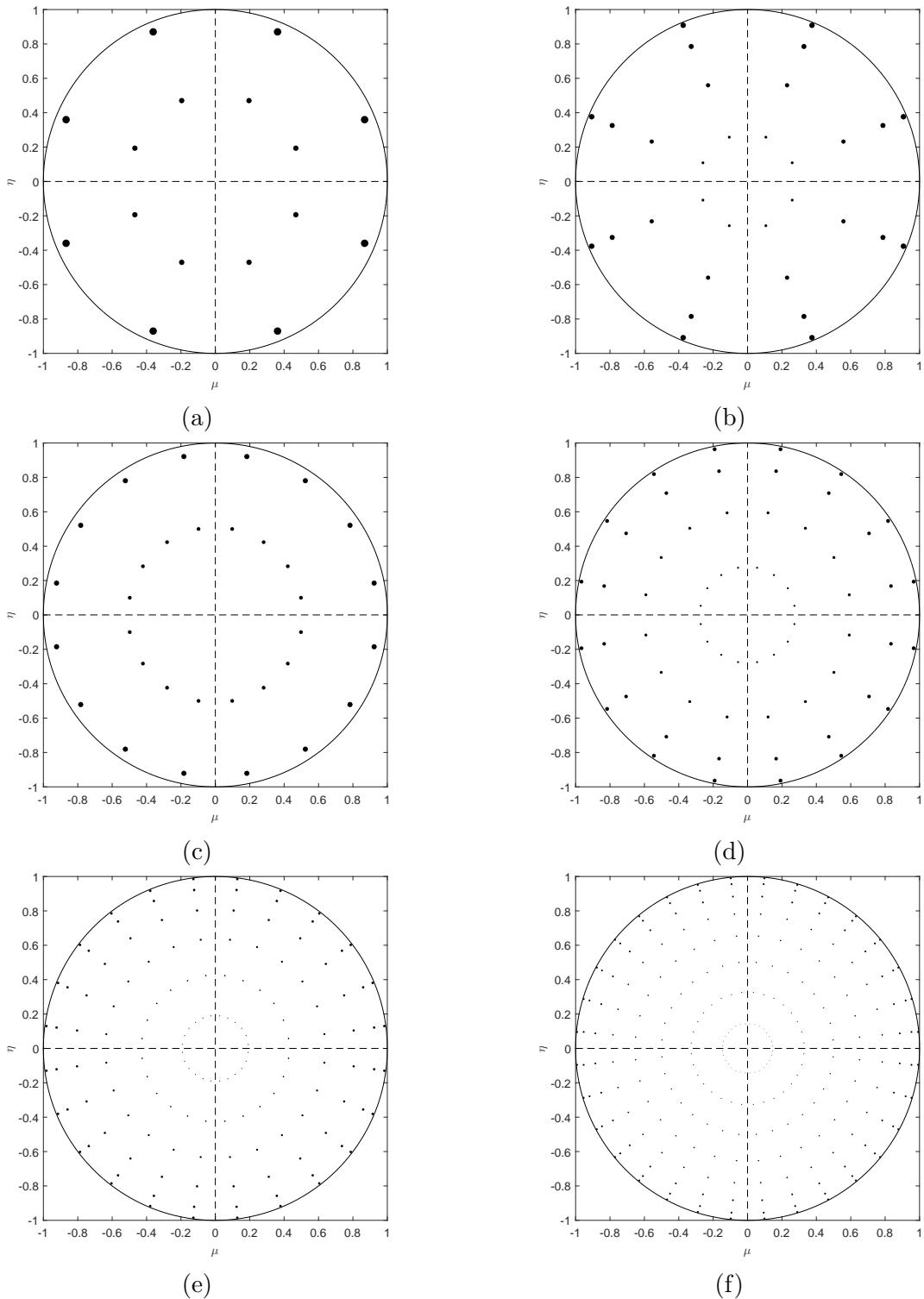


Figure 2.7: Projection of the 3D Product Gauss-Legendre-Chebyshev angular quadrature set with orders: (a) S_2^2 , (b) S_2^4 , (c) S_4^2 , (d) S_4^4 , (e) S_6^6 , and (f) S_8^8 onto the x-y space on the unit circle.

where we have dropped the spatial parameter for clarity and is beholden to the following general, discretized boundary condition:

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

These ($M \times G$) 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

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

where the reflecting angle is $\vec{\Omega}_{m'} = \vec{\Omega}_m - 2(\vec{\Omega}_m \cdot \vec{n})\vec{n}$ 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 moment of the source term and will act similarly to the total interaction term)

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

to form M ($m = 1, \dots, M$) 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 [6, 33], 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.*, $\vec{n}' = -\vec{n}$).

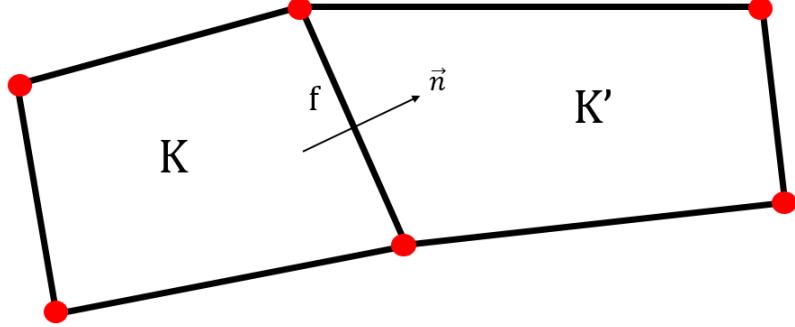


Figure 2.8: Two cells of the spatial discretization with the connecting face, f , with normal direction, \vec{n} , oriented from cell K to cell K' .

Next, we 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 $\vec{\Omega}_m$:

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

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 volume 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: $\partial K^- \setminus \partial \mathcal{D}$; 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:

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

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,

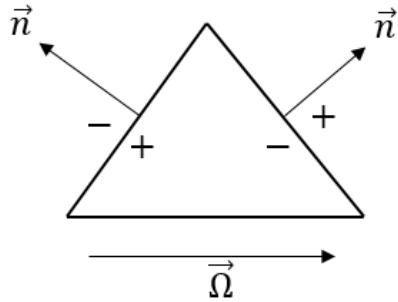


Figure 2.9: Definition of the trace for the upwinding scheme.

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

when the following trace is applied to the angular fluxes :

$$\Psi_m^\pm(\vec{r}) \equiv \lim_{s \rightarrow 0^\pm} \Psi_m\left(\vec{r} + s(\vec{\Omega}_m \cdot \vec{n})\vec{n}\right). \quad (2.47)$$

We give a visual example of this trace in Figure 2.9 on a triangle with a single inflow and a single outflow boundary. For the inflow (left) boundary, the within-cell fluxes are “+” and the out-of-cell fluxes are “-”. Likewise, the outflow (right) boundary has within-cell fluxes designated as “-” and out-of-cell fluxes designated as “+”. Therefore, with this definition of the trace, we always use the flux values corresponding to the “-” direction (as seen in Eq. (2.46)). Now, using the upwind scheme as previously defined, we can write our complete set of DGFEM equations for cell K as

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

We note that fluxes without the trace superscript are all within the cell. Also, the basis functions, b_m , always correspond to 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:

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

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

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

and

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

respectively. The interior faces are designated as the non-repeating set: $E_h^i \in \cup_{K \in \mathbb{T}_h} \partial K \setminus \partial \mathcal{D}$. 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 trace condition condition of Eq. (2.47).

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. (2.53). We have changed the sequence of the summation over directions and over the elements for the boundary

terms.

$$\begin{aligned}
a(\Psi^*, \Psi) &= \sum_{m=1}^M w_m \left[\left(\Psi_m^*, \vec{\Omega}_m \cdot \vec{\nabla} \Psi_m \right)_{\mathcal{D}} + \left(\sigma_t \Psi_m^*, \Psi_m \right)_{\mathcal{D}} \right] \\
&+ \sum_{m=1}^M w_m \left\langle (\vec{\Omega}_m \cdot \vec{n}) \Psi_m^{*+}, [\![\Psi_m]\!] \right\rangle_{E_h^i} - \sum_{f \in \partial \mathcal{D}} \sum_{\vec{\Omega}_m \cdot \vec{n}} \left\langle (\vec{\Omega}_m \cdot \vec{n}) \Psi_m^*, \Psi_m \right\rangle_f \quad (2.53) \\
&+ \sum_{f \in \partial \mathcal{D}^r} \sum_{\vec{\Omega}_m \cdot \vec{n}} \left\langle (\vec{\Omega}_m \cdot \vec{n}) \Psi_m^*, \Psi_{m'} \right\rangle_f - \sum_{p=0}^{N_P} \sum_{n=-p}^p \frac{2p+1}{4\pi} \left(\sigma_{s,p} \Phi_{p,n}^*, \Phi_{p,n} \right)_{\mathcal{D}}
\end{aligned}$$

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

$$a(\Psi^*, \Psi - \Psi_{exact}) = 0, \quad m = 1, \dots, M. \quad (2.54)$$

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

$$\|u\|_{DG} = a(u, u). \quad (2.55)$$

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

$$\|u\|_{DG}^2 = \sum_{K \in \mathbb{T}_h} \left[\|u\|_{L^2(K)}^2 + \frac{1}{2} \|u^+ - u^-\|_{L^2(\partial K)}^2 \right], \quad (2.56)$$

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 ex-

tensively studied [29, 85, 86, 34]. With the discretized flux solutions, $\Psi_h \in W_{\mathcal{D}}^h$ and $\Phi_h \in W_{\mathcal{D}}^h$, 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,

$$\|\Psi - \Psi_{exact}\|_{DG} \leq C \frac{h^q}{(p+1)^q}, \quad (2.57)$$

and flux moments,

$$\|\Phi - \Phi_{exact}\|_{DG} \leq C \frac{h^q}{(p+1)^q}, \quad (2.58)$$

respectively. In Eqs. (2.57) and (2.58), $q = \min(p + 1/2, r - 1/2)$, 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,

$$\|\Phi - \Phi_{exact}\|_{L_2} \leq C \frac{h^{q+1/2}}{(p+1)^q}, \quad (2.59)$$

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. (2.57), (2.58), and (2.59) further, we can see that our DGFEM transport convergence rates are all limited by the regularity, r , of the solution space. The spaces in which the transport equation can reside have been investigated by others in previous works [34, 87, 88]. 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 occurrences, 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,

$$N_{ele} \propto h^{-d}, \quad (2.60)$$

where we assume 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) [89, 90]. 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. (2.59) to yield an L_2 convergence rate in terms of the problem's total number of degrees of freedom,

$$\|\Phi - \Phi_{exact}\|_{L_2} \leq C(p) N_{dof}^{-\frac{q+1/2}{d}}, \quad (2.61)$$

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

$$C(p) = \frac{p^{\frac{q+1/2}{d}}}{(p+1)^q}. \quad (2.62)$$

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

$$\|\Phi - \Phi_{exact}\|_{L_2} \leq C(p) N_{dof}^{-\frac{p+1}{d}}, \quad C(p) = \frac{p^{\frac{p+1}{d}}}{(p+1)^{p+1/2}}. \quad (2.63)$$

The result of Eq. (2.63) 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. (2.63) 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. (2.59) and (2.61) for a 2D problem not bounded by the solution regularity ($q + 1/2 = p + 1$) in Figure 2.10. 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

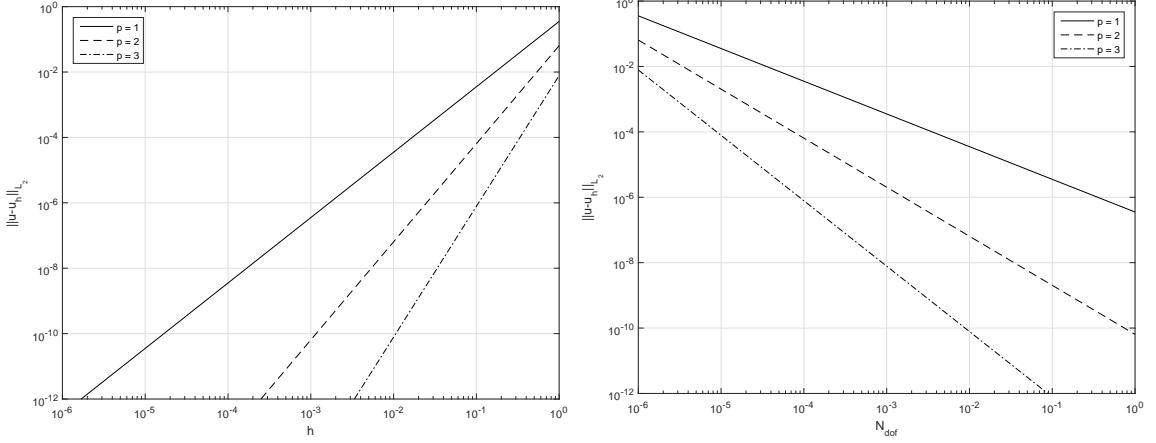


Figure 2.10: Example of the theoretical convergence rates for a DGFEM transport problem that is not bound by solution regularity in terms of the maximum element diameter (left) and number of degrees of freedom (right).

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, $(\sigma b_m, \Psi_m)_K$, streaming, $(\vec{\Omega}_m \cdot \vec{\nabla} b_m, \Psi_m)_K$, and surface, $\left\langle (\vec{\Omega}_m \cdot \vec{n}) b_m, \Psi_m \right\rangle_{\partial K}$. 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: $(\sigma b_m, \Psi_m)_K$ 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 [91, 92]. For cell K , we define the elementary mass matrix, \mathbf{M} , as

$$\mathbf{M}_K = \int_K \mathbf{b}_K \mathbf{b}_K^T dr, \quad (2.64)$$

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. (2.64) 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

$$\mathbf{M}_K = \sum_{q=1}^{N_q} w_q^K \mathbf{b}_K(\vec{x}_q) \mathbf{b}_K^T(\vec{x}_q). \quad (2.65)$$

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 ($N_K \times N_K$) matrix. This matrix is written for completeness of this discussion on the mass matrix:

$$\mathbf{M}_K = \begin{bmatrix} \int_K b_1 b_1 & \dots & \int_K b_1 b_j & \dots & \int_K b_1 b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_K b_i b_1 & \dots & \int_K b_i b_j & \dots & \int_K b_i b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_K b_{N_K} b_1 & \dots & \int_K b_{N_K} b_j & \dots & \int_K b_{N_K} b_{N_K} \end{bmatrix}, \quad (2.66)$$

where an individual matrix entry is of the form:

$$M_{i,j,K} = \int_K b_i b_j. \quad (2.67)$$

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:

$$\vec{\mathbf{G}}_K = \int_K \vec{\nabla} \mathbf{b}_K \mathbf{b}_K^T dr, \quad (2.68)$$

which has dimensionality $(N_K \times N_K \times d)$. We choose to store the elementary streaming matrix in this form and not store M separate $(N_K \times N_K)$ 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 where 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, $\{\vec{x}_q, w_q^K\}_{q=1}^{N_q}$, for cell K to numerically calculate the streaming matrix:

$$\vec{\mathbf{G}}_K = \sum_{q=1}^{N_q} w_q^K \vec{\nabla} \mathbf{b}_K(\vec{x}_q) \mathbf{b}_K^T(\vec{x}_q). \quad (2.69)$$

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

$$\vec{\mathbf{G}}_K = \begin{bmatrix} \int_K \vec{\nabla} b_1 b_1 & \dots & \int_K \vec{\nabla} b_1 b_j & \dots & \int_K \vec{\nabla} b_1 b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_K \vec{\nabla} b_i b_1 & \dots & \int_K \vec{\nabla} b_i b_j & \dots & \int_K \vec{\nabla} b_i b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_K \vec{\nabla} b_{N_K} b_1 & \dots & \int_K \vec{\nabla} b_{N_K} b_j & \dots & \int_K \vec{\nabla} b_{N_K} b_{N_K} \end{bmatrix}, \quad (2.70)$$

where an individual matrix entry is of the form:

$$\vec{G}_{i,j,K} = \int_K \vec{\nabla} b_i b_j. \quad (2.71)$$

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: $\vec{\Omega}_m \cdot \langle \vec{n} b_m, \Psi_m \rangle_{\partial K}$. These terms are analogous 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,

$$\vec{\mathbf{F}}_{f,K} = \int_f \vec{n}(\vec{r}) \mathbf{b}_K \mathbf{b}_K^T ds, \quad (2.72)$$

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. (2.72), we can see that they have dimensionality, $(N_K \times N_K \times d)$. 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 $(N_b^{f,K} \times N_b^{f,K})$.

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 :

$$\vec{\mathbf{F}}_{f,K} = \sum_{q=1}^{N_q^f} w_q^f \vec{n}(\vec{x}_q) \mathbf{b}_K(\vec{x}_q) \mathbf{b}_K^T(\vec{x}_q). \quad (2.73)$$

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,

$$\vec{\mathbf{F}}_{f,K} = \begin{bmatrix} \int_f \vec{n} b_1 b_1 & \dots & \int_f \vec{n} b_1 b_j & \dots & \int_f \vec{n} b_1 b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_f \vec{n} b_i b_1 & \dots & \int_f \vec{n} b_i b_j & \dots & \int_f \vec{n} b_i b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_f \vec{n} b_{N_K} b_1 & \dots & \int_f \vec{n} b_{N_K} b_j & \dots & \int_f \vec{n} b_{N_K} b_{N_K} \end{bmatrix}, \quad (2.74)$$

where an individual matrix entry is of the form:

$$\vec{F}_{i,j,f,K} = \int_f \vec{n} b_i b_j. \quad (2.75)$$

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:

$$\begin{aligned}
& - \left(\vec{\Omega}_m \cdot \vec{\nabla} b_{m,g}, \Psi_{m,g} \right)_K + \left(\sigma_{t,g} b_{m,g}, \Psi_{m,g} \right)_K + \left\langle (\vec{\Omega}_m \cdot \vec{n}) b_{m,g}, \Psi_{m,g}^- \right\rangle_{\partial K^+} \\
& + \left\langle (\vec{\Omega}_m \cdot \vec{n}) b_{m,g}, \Psi_{m,g}^- \right\rangle_{\partial K^- \setminus \partial \mathcal{D}} + \left\langle (\vec{\Omega}_m \cdot \vec{n}) b_{m,g}, \Psi_{m',g}^- \right\rangle_{\partial K^- \cap \partial \mathcal{D}^d} \\
& = \sum_{g'=1}^G \sum_{p=0}^{N_p} \frac{2p+1}{4\pi} \sum_{n=-p}^p Y_{p,n}(\vec{\Omega}_m) \left(\sigma_{s,p}^{g' \rightarrow g} b_{m,g}, \Phi_{p,n,g'} \right)_K \\
& + \left(b_{m,g}, Q_{m,g} \right)_K - \left\langle (\vec{\Omega}_m \cdot \vec{n}) b_{m,g}, \Psi_{m,g}^{inc} \right\rangle_{\partial K^- \cap \partial \mathcal{D}^d}
\end{aligned} . \quad (2.76)$$

All of the notations used in Eq. (2.76) 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, Ψ , with dimensionality of $(G \times M \times N_{dof})$. The angular flux moments, Φ , have dimensionality of $(G \times N_{mom} \times N_{dof})$. 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 $(G \times M \times N_{dof}) \times (G \times M \times N_{dof})$ with a corresponding right-hand-side vector of dimensionality $(G \times M \times N_{dof}) \times 1$, and 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 of the tight 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 2.11 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 2.11 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 $g = 1, \dots, G$ (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 2.11. Then, depending on how the groups are structured, multiple outer iterations may be necessary to fully converge the scattering source.

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 $g \in [8, 10]$.

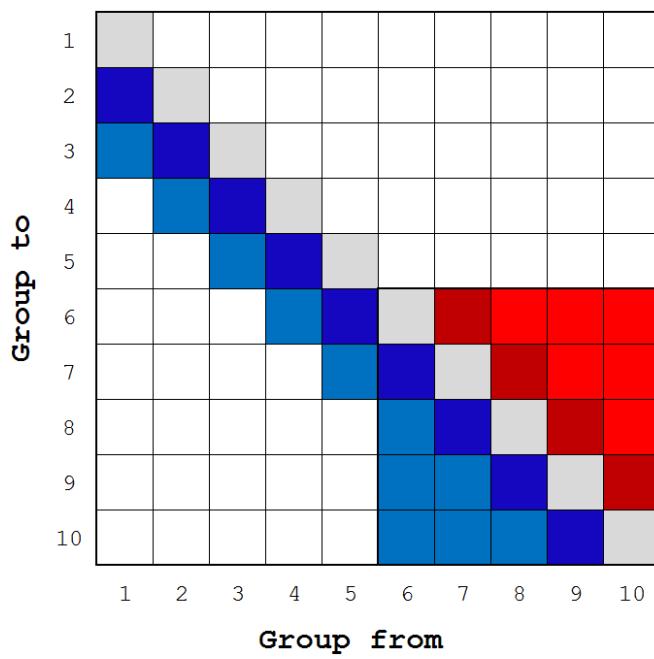
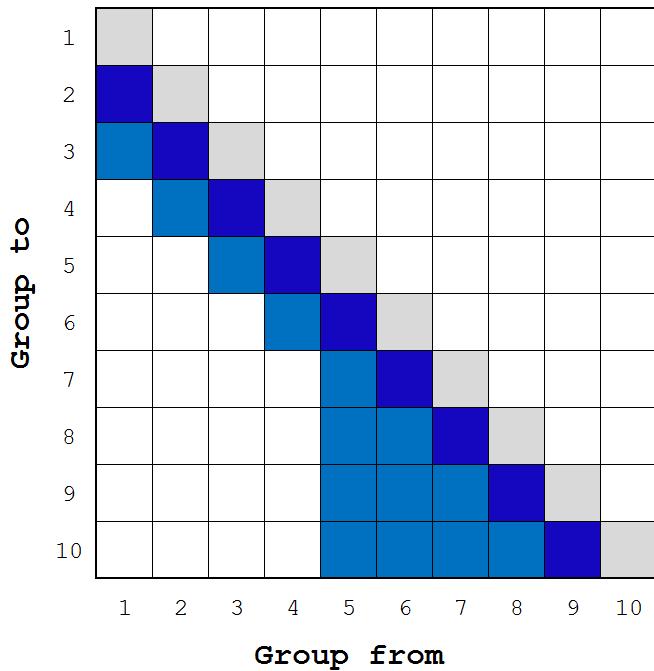


Figure 2.11: Scattering matrices (top) without and (bottom) with upscattering. The gray corresponds to within-group scattering; the blue corresponds to down-scattering in energy; and the red corresponds to up-scattering in energy.

With these group sets defined, we then choose to employ a double iteration loop to converge the scattering source. 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 source 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 iterations 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_{wgs} .

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

$$\mathbf{L}_{gs} \Psi_{gs} = \mathbf{M}_{gs} \Sigma_{gs} \Phi_{gs} + \mathbf{M}_{gs} \sum_{ggs \neq gs} \Sigma_{ggs} \Phi_{ggs} + \mathbf{Q}_{gs}, \quad (2.77)$$

$$gs = 1, \dots, N_{gs}$$

and

$$\Phi_{gs} = \mathbf{D}_{gs} \Psi_{gs}, \quad (2.78)$$

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, Σ 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. (2.77) in terms of only the flux moments. Through algebra and linear algebra techniques, we state the flux-moment-only equation as

$$(\mathbf{I} - \mathbf{T}_{gs}) \Phi_{gs} = \mathbf{D}_{gs} \mathbf{L}_{gs}^{-1} \mathbf{M}_{gs} \sum_{ggs \neq gs} \Sigma_{ggs} \Phi_{ggs} + \mathbf{D}_{gs} \mathbf{L}_{gs}^{-1} \mathbf{Q}_{gs}, \quad (2.79)$$

where we define,

$$\mathbf{T}_{gs} \equiv \mathbf{D}_{gs} \mathbf{L}_{gs}^{-1} \mathbf{M}_{gs} \Sigma_{gs}, \quad (2.80)$$

for further brevity. In Eqs. (2.79) and (2.80), 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. (2.79) in Sections 2.7.1.1 and 2.7.1.2.

Algorithm 1 Iterative Solver in Energy for the Multigroup Transport Problem

```
1: Initialize:  $\Phi_{g,p,n} = 0$ ,  $g = 1, \dots, G; p = 0, \dots, N_p; n = -p, \dots, p$ 
2: for  $a = 1, \dots, I_{ags}$  do
3:   Loop: through group sets
4:   for  $gs = 1, \dots, N_{gs}$  do
5:     for  $k = 1, \dots, I_{wgs}$  do
6:       Perform: residual iteration
7:       Apply: Acceleration for group set  $gs$ 
8:       Check: convergence of group set  $gs$ 
9:     end for
10:    end for
11:    Perform: residual iteration
12:    Apply: Acceleration for across-group-set
13:    Check: across-group-set convergence
14: end for
```

2.7.1.1 Source Iteration

One simple method to invert the $(\mathbf{I} - \mathbf{T})$ operator of Eq. (2.79) 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

$$\mathbf{L}\Psi^{(k+1)} = \mathbf{M}\Sigma\Phi^{(k)} + \mathbf{Q}, \quad (2.81)$$

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. (2.81), we can see that the scattering source at iteration $(k+1)$ is calculated from a previous guess for the flux moments at iteration (k) . This is exactly a Jacobi iteration in energy for the group set. Then, after the application of one transport sweep, we obtain a new guess for the angular flux moments:

$$\begin{aligned}\Psi^{(k+1)} &= \mathbf{L}^{-1} \left(\mathbf{M} \boldsymbol{\Sigma} \Phi^{(k)} + \mathbf{Q} \right) \\ \Phi^{(k+1)} &= \mathbf{D} \Psi^{(k+1)}\end{aligned}. \quad (2.82)$$

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

$$\frac{||\Phi^{(k+1)} - \Phi^{(k)}||}{||\Phi^{(k+1)}||} \leq \text{tol}, \quad (2.83)$$

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 [42]. This spectral radius, ρ , can be estimated in the asymptotic convergence region by the ratio of two successive solution differences:

$$\rho \approx \frac{||\Phi^{(k+1)} - \Phi^{(k)}||}{||\Phi^{(k)} - \Phi^{(k-1)}||} \quad (2.84)$$

We can gather a lot of information about how our transport solution will converge from Eq. (2.84). If $\rho > 1$ in the asymptotic region, then the error in our residual will grow without end, and our solution will diverge. However, in the preasymptotic region, Eq. (2.84) 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

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

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 [42]. 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 [93, 94, 95]. 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 [96, 97]. 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: $\mathbf{b} = \mathbf{DL}^{-1}\mathbf{Q}$. 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 is 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,

$$L_{m,g} = \vec{\Omega}_m \cdot \vec{\nabla} + \sigma_{t,g}, \quad (2.85)$$

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. (2.76) 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 computationally 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) [98].

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 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) [99]. 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

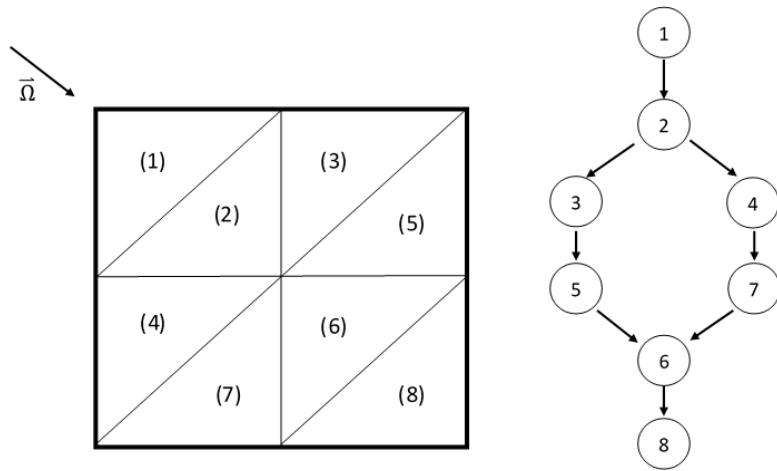


Figure 2.12: Task dependence graph for the transport sweep for a given direction without cycles.

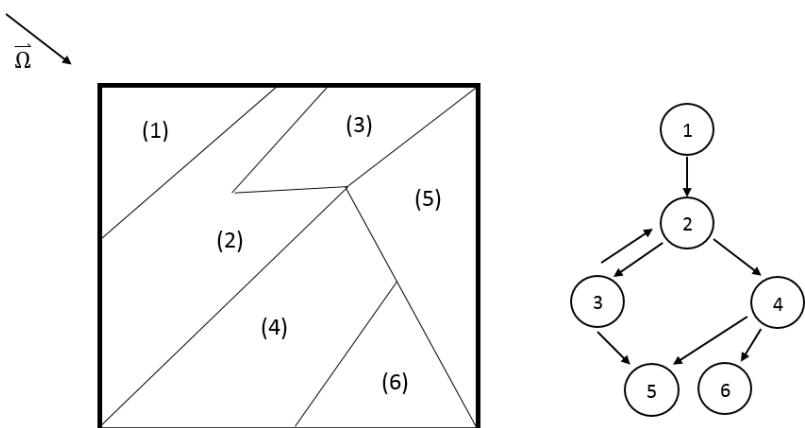


Figure 2.13: Task dependence graph for the transport sweep for a given direction with cycles present.

2.12 shows a DAG without any cycles for the given direction. Figure 2.13 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.

Up to this point, we have provided the basic details on how a transport sweep is performed. We now switch our discussion to recent developments involving provably optimal sweeping algorithms with high efficiencies out to $O(10^5)$ - $O(10^6)$ processors. For over a decade, the Koch-Baker-Alcouffe (KBA) parallel sweeping algorithm was the most prevalent in the computational transport community [100, 101]. The KBA algorithm spatially partitions the problem by assigning each processor a column of cells. During a transport sweep, the cell-by-cell solution of each angular direction is represented as a Task Dependence Graph (TDG). KBA operates by initializing a new TDG as soon as it completes a previous TDG for all directions in an octant-pair. This consecutive execution of TDGs is known as “pipelining”. Unfortunately, KBA’s pipelining penalty grows with processor counts but the TDG width grows only as $P^{2/3}$. Therefore, KBA eventually runs out of parallelism to exploit which led to the common belief that transport sweeps would not scale beyond $O(10^3)$ - $O(10^4)$ processing units.

Now we give an overview of a provably-optimal sweeping algorithm that will be utilized in this work [102, 103]. Consider a transport problem with a $(N_x \times N_y \times N_z)$ spatial grid of Cartesian mesh cells and a $(P_x \times P_y \times P_z)$ processor layout. Assume that N_x/P_x , N_y/P_y , and N_z/P_z are all integer values. Now suppose a group set that we wish to sweep has G groups and M angular directions per octant. This means that each processor must perform $(8MGN_xN_yN_z)/(P_xP_yP_z)$ total calculations for the transport sweep. We then aggregate these calculations into tasks with each task containing A_g groups, A_M directions and $A_xA_yA_z$ spatial cells. Therefore, each

processor must perform N_{tasks} number of tasks which can be given by

$$N_{tasks} = \frac{8MGN_x N_y N_z}{A_g A_M A_x A_y A_z P_x P_y P_z}. \quad (2.86)$$

The complete transport sweep requires a total of N_{stages} stages, given by

$$N_{stages} = N_{tasks} + N_{idle}, \quad (2.87)$$

where N_{idle} is the number of processor idle stages. Using these definitions, the parallel efficiency, ϵ , of the sweep can be written as

$$\epsilon = \frac{1}{\left[1 + \frac{N_{idle}}{N_{tasks}}\right] \left[1 + \frac{T_{comm}}{T_{task}}\right]}, \quad (2.88)$$

where T_{task} is the time required to complete a single task and T_{comm} is the time required to communicate data after a task has completed. Minimizing the terms (as close to unity as possible) in the denominator will lead to higher parallel efficiencies. However, the bracketed terms compete against each other because maximizing N_{tasks} will minimize T_{task} .

For a given set partitioning parameters and aggregation factors, the minimum number of stages is $2N_{fill} + N_{tasks}$, where N_{fill} is the number of stages for a sweepfront to reach the center processors in the domain. If we set $A_x = N_x/P_x$ and $A_y = N_y/P_y$ and define $\delta_u = 0$ or 1 for P_u even or odd, respectively, and $N_k = N_z/(P_z A_z)$, then

$$N_{fill} = \frac{P_x + \delta_x}{2} - 1 + \frac{P_y + \delta_y}{2} - 1 + N_k \left(\frac{P_z \delta_z}{2} - 1 \right) \quad (2.89)$$

$$N_{idle}^{min} = P_x + \delta_x - 2P_y + \delta_y - 2 + N_k(P_z + \delta_z - 2)$$

and

$$N_{stages}^{min} = N_{idle}^{min} + \frac{8MGN_k}{A_M A_g} \quad (2.90)$$

If the minimum stage count could be achieved, then the optimal parallel efficiency, ϵ_{opt} , can be given by the following

$$\epsilon_{opt} = \frac{1}{\left[1 + \frac{(P_x+\delta_x)+(P_x+\delta_x)-4+N_k(P_z+\delta_z-2)}{8MGN_k/(A_M A_g)}\right] \left[1 + \frac{T_{comm}}{T_{task}}\right]}. \quad (2.91)$$

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 [66, 65, 67, 68, 69, 70]. However, while this has become more common practice in other disciplines, the use of AMR for the transport equation is still in its infancy [71, 72, 73, 74, 75, 76]. For this work, we will only utilize *h*-type refinement strategies for 2D transport problems on initial meshes with only quadrilateral cells.

Figure 2.14 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 [104, 105]. 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. 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

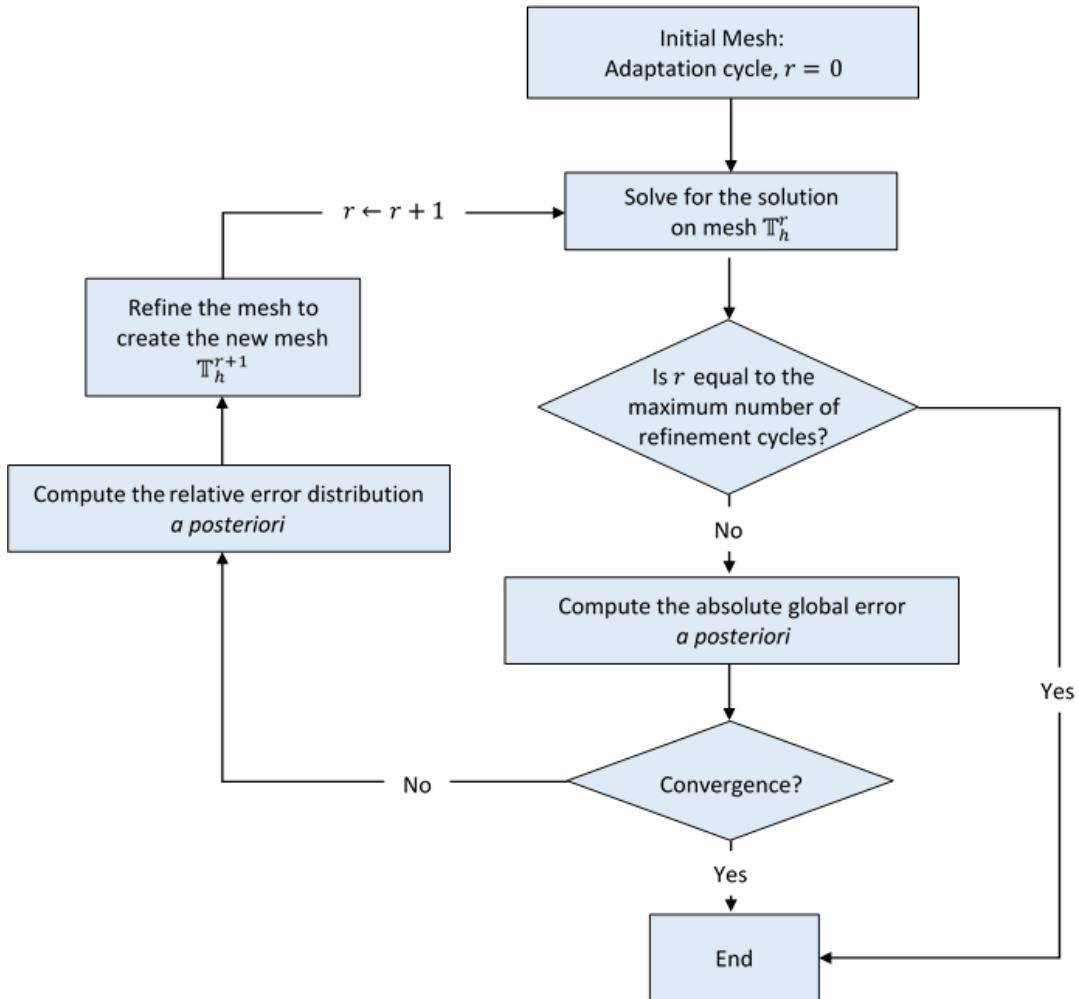


Figure 2.14: Flow chart for mesh adaptation.

estimation of the numerical solution error. The state-of-the-art error estimators rely on either adjoint-based methods [71, 72, 74] or projection-based methods [106]. 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,

$$\eta_K^r = \int_{\partial K} [\Phi^r]^2 ds = \int_{\partial K} \left(\sum_m w_m [\Psi_m^r] \right)^2 , \quad (2.92)$$

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

$$[\Phi(\vec{r})] = \Phi^+(\vec{r}) - \Phi^-(\vec{r}), \quad (2.93)$$

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

$$\Phi^\pm(\vec{r}) = \lim_{s \rightarrow 0^\pm} \Phi(\vec{r} + s\vec{n}). \quad (2.94)$$

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:

$$\Phi^+(\vec{r}) = \begin{cases} \lim_{s \rightarrow 0^+} \Phi(\vec{r} + s\vec{n}) & \vec{r} \notin \partial\mathcal{D} \\ \sum_{\vec{\Omega}_m \cdot \vec{n} > 0} \Psi_m^-(\vec{r}) + \sum_{\vec{\Omega}_m \cdot \vec{n} < 0} \Psi_m^{inc}(\vec{r}) & \vec{r} \in \partial\mathcal{D}^d \\ \Phi^-(\vec{r}) & \vec{r} \in \partial\mathcal{D}^r \end{cases} \quad (2.95)$$

From Eq. (2.95), 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,

$$\eta_K^r \geq \alpha \max_{K' \in \mathbb{T}_h^r} (\eta_{K'}^r), \quad (2.96)$$

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 original 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 2.15. 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.

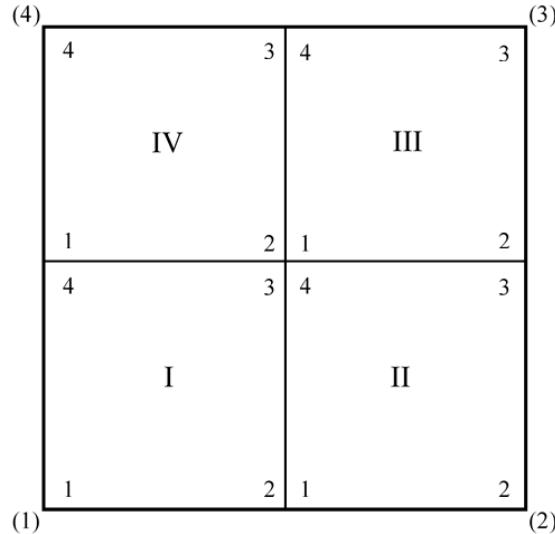


Figure 2.15: Refinement rules for a quadrilateral mesh cell.

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 2.16 for the simple case of a 2-cell domain that undergoes two refinement cycles with 1 cell refined per cycle. From the tree structure in Figure 2.16b, 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 is 0. We note that an element's refinement level is not necessarily the number of refinement cycles that have been performed. In Figure 2.16b, we can see three distinct refinement levels that are represented by the three different vertical levels of nodes.

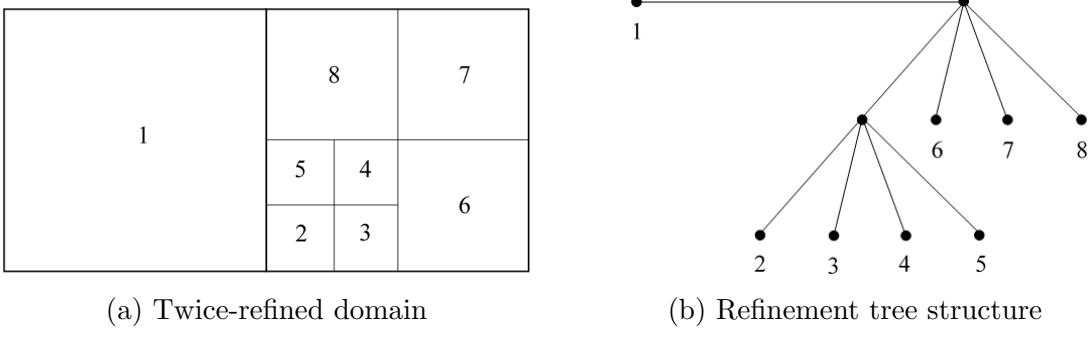


Figure 2.16: Hierarchical refinement tree for a simple domain with quadrilateral cells.

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 2.16 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 $\ell(2) = 2$. 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 next solution on the newly 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

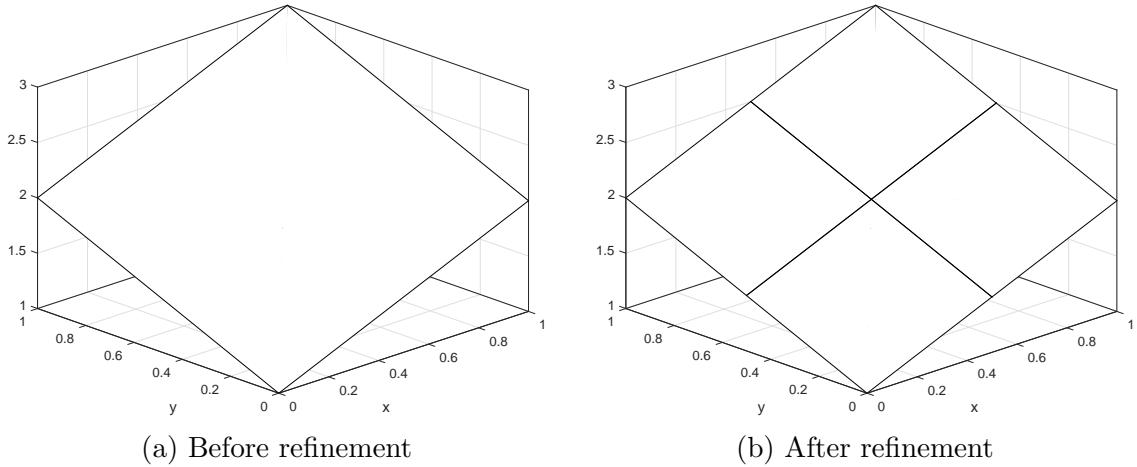


Figure 2.17: Bootstrapping the solution from a single unit square element onto the four daughter elements after refinement.

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 2.17 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 2.17a 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 2.17b shows this bootstrapped solution onto the refined mesh, where we can clearly see that the projected solution lives in 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. From FEM theory [6], the basis functions form a function space with local measure on some subset of elements on a discretized mesh, \mathbb{T}_h . To achieve the maximum possible solution convergence rate for regular solutions of $p + 1$, the basis functions must have polynomial completeness of at least order p . For 2D interpolants, the basis functions are linearly-complete ($p = 1$) if they can exactly interpolate the $\{1, x, y\}$ span of functions. Likewise, 2D basis functions are said to be quadratically-complete if they can exactly interpolate the $\{1, x, y, x^2, xy, y^2\}$ span of functions. This work seeks to analyze the use of the different linearly-complete polygonal basis functions with the DGFEM S_N transport equation. We then seek to extend this analysis to quadratically-complete basis functions, thus achieving higher convergence rates (third-order). We will do this by utilizing the methodology developed by Rand et al. [107] on the different linearly-complete polygonal coordinates.

The remainder of 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 3.3 provides the methodology that will be employed to generate spatial quadrature sets on 2D arbitrary polygons.

Section 3.4 then presents 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. These results include verification examples and demonstrating that the basis functions satisfy transport solutions in the thick diffusion limit. 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 $(1, \dots, N_K)$ 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 $(\vec{x}_j, \vec{x}, \vec{x}_{j+1})$. 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} .

In this dissertation, all linearly-complete, 2D basis functions for an element K will obey the properties for barycentric coordinates. If the element K is composed of N_K vertices, then it contains N_K barycentric coordinates, where each one is located at a vertex. These barycentric coordinates will form a *partition of unity*,

$$\sum_{j=1}^{N_K} b_j(\vec{x}) = 1; \quad (3.1)$$

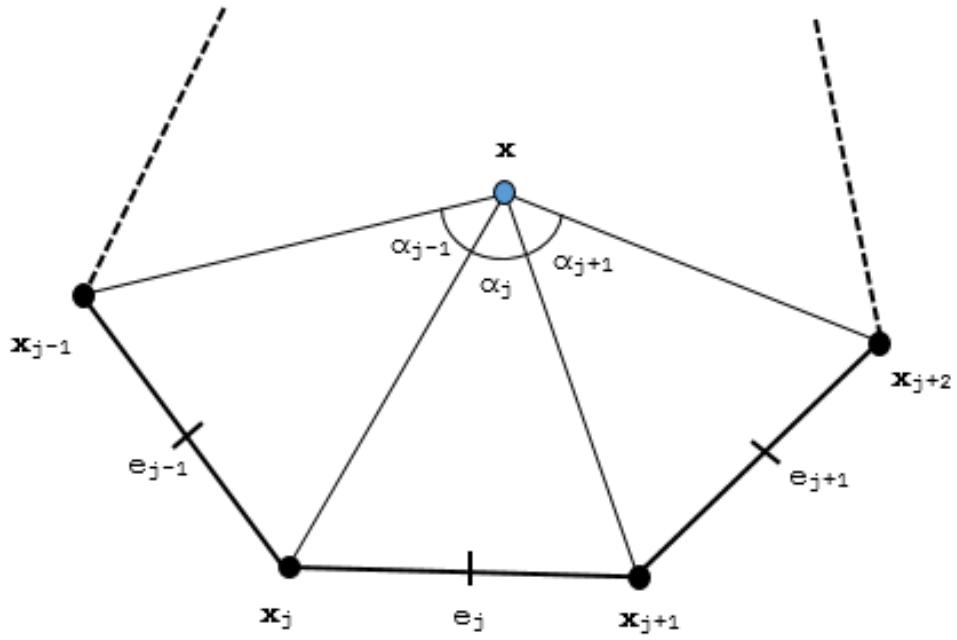


Figure 3.1: Arbitrary polygon with geometric properties used for 2D basis function generation.

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

$$\sum_{j=1}^{N_K} b_j(\vec{x}) \vec{x}_j = \vec{x}; \quad (3.2)$$

and they will satisfy the *Lagrange property*,

$$b_j(\vec{x}_i) = \delta_{ij}. \quad (3.3)$$

They also have piecewise linearity on faces adjacent to their vertex. As an example of this, consider the function at vertex j , b_j , along face e_j . Then the piecewise linearity of the function on the face means that it can interpolate as

$$b_j((1 - \mu)\vec{x}_j + \mu\vec{x}_{j+1}) = (1 - \mu)b_j(\vec{x}_j) + \mu b_j(\vec{x}_{j+1}), \quad \mu \in [0, 1]. \quad (3.4)$$

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

$$\sum_{j=1}^{N_K} b_j(\vec{x}) \vec{c}_{j,1}(\vec{x}) = \vec{q}_1, \quad (3.5)$$

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

$$\vec{c}_{j,1}(\vec{x}) = \begin{bmatrix} 1 \\ x_j - x \\ y_j - y \end{bmatrix} \quad \text{and} \quad \vec{q}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad (3.6)$$

respectively. Equation (3.5) 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 [108]. These rational functions were the first derived for 2D polygons and possess all the properties of the barycentric coordinates previously detailed. However, they are only valid interpolants over strictly-convex polygons. They have zero measure and blow up for weakly-convex and concave polygons, respectively. Also, their values and gradients cannot be directly evaluated on the polygonal boundary. However, they do have a valid limit which we show in Appendix B. The Wachspress coordinates (which we denote as b^W) have the following form

$$b_j^W(\vec{x}) = \frac{w_j(\vec{x})}{\sum_{i=1}^{N_K} w_i(\vec{x})}, \quad (3.7)$$

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

$$w_j(\vec{x}) = \frac{A(\vec{x}_{j-1}, \vec{x}_j, \vec{x}_{j+1})}{A(\vec{x}, \vec{x}_{j-1}, \vec{x}_j) A(\vec{x}, \vec{x}_j, \vec{x}_{j+1})}. \quad (3.8)$$

In Eq. (3.8), the terms $A(\vec{a}, \vec{b}, \vec{c})$ denote the signed area of the triangle with vertices \vec{a} , \vec{b} , and \vec{c} . Each of these signed areas can be computed by

$$A(\vec{a}, \vec{b}, \vec{c}) = \frac{1}{2} \begin{vmatrix} 1 & 1 & 1 \\ x_a & x_b & x_c \\ y_a & y_b & y_c \end{vmatrix}. \quad (3.9)$$

There is an alternative method of expressing the Wachspress weight functions. Warren et al. [109] proposed weight functions that are defined in terms of the perpendicular distance of the point \vec{x} to the polygon's faces. Using the reference polygon of Figure 3.1, the perpendicular distance of the point \vec{x} to the face j is denoted as $h_j(\vec{x})$ and is given by

$$h_j(\vec{x}) = (\vec{x}_j - \vec{x}) \cdot \vec{n}_j = (\vec{x}_{j+1} - \vec{x}) \cdot \vec{n}_j, \quad (3.10)$$

where \vec{n}_j is the outward normal direction of face j . Using these perpendicular distances, the Wachspress coordinates can be calculated using Eq. (3.7) with new function definitions of

$$w_j(\vec{x}) = \frac{|\vec{n}_{j-1} \times \vec{n}_j|}{h_{j-1}(\vec{x}) h_j(\vec{x})}, \quad (3.11)$$

where

$$|\vec{x}_1 \times \vec{x}_2| = \begin{vmatrix} x_1 & x_2 \\ y_1 & y_2 \end{vmatrix}. \quad (3.12)$$

For FEM theory, the basis function gradients are also necessary to compute some of the elementary matrices. The gradients of the Wachspress rational functions are straightforward to calculate by simply taking the partial derivatives of Eq. (3.7). Then, using derivative rules along with some algebra, the Wachspress gradients are given by,

$$\vec{\nabla} b_j^W(\vec{x}) = b_j^W(\vec{x}) \left(\vec{R}_j(\vec{x}) - \sum_i b_i^W(\vec{x}) \vec{R}_i(\vec{x}) \right), \quad (3.13)$$

where the reduced gradient, \vec{R}_j , is defined as

$$\vec{R}_j(\vec{x}) = \frac{1}{w_j} \vec{\nabla} w_j. \quad (3.14)$$

This means that the gradients of the Wachspress coordinates can be calculated by combinations of the all the weight functions and their gradients. The weight function gradients are easy to compute using the perpendicular form. The gradient of the j weight functions is given by

$$\vec{\nabla} w_j(\vec{x}) = w_j(\vec{x}) \left(\frac{\vec{n}_{j-1}}{h_{j-1}(\vec{x})} + \frac{\vec{n}_j}{h_j(\vec{x})} \right). \quad (3.15)$$

This lets us immediately see that \vec{R}_j is simply

$$\vec{R}_j(\vec{x}) = \frac{\vec{n}_{j-1}}{h_{j-1}(\vec{x})} + \frac{\vec{n}_j}{h_j(\vec{x})}. \quad (3.16)$$

We now give a pair of contour plots of the Wachspress coordinates. First, Figure 3.2 provides the contour plots of the four Wachspress functions on the unit square. We see that the functions are smoothly varying within the square with at least C^1 continuity. Then in Figure 3.3, we give the contour plots for a degenerate pentagon which is simply the unit square with a vertex added at point $(1/2, 1)$. We see how the functions fail for this weakly-convex case. The function located at the degenerate vertex has zero measure everywhere within the polygon. Also, the functions located at the vertices adjacent to the degenerate vertex no longer maintain linearity on their adjacent faces. We will not show it here for brevity, but the Wachspress functions on concave polygons will have points in the interior that will result in divide-by-zero operations.

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 [37, 38]. They originally introduced the PWL coordinates to work specifically for the DGFEM transport equation on unstructured quadrilateral and polygonal grids. These coordinates share some similarities with the Wachspress rational functions, but also contain some key differences. The properties of the PWL coordinates that are different from the Wachspress rational functions can be summarized with the following:

1. PWL works with concave polygons;
2. PWL cannot interpolate on curved surfaces;
3. points on the boundary can be directly evaluated;
4. the PWL integrals can be computed analytically;

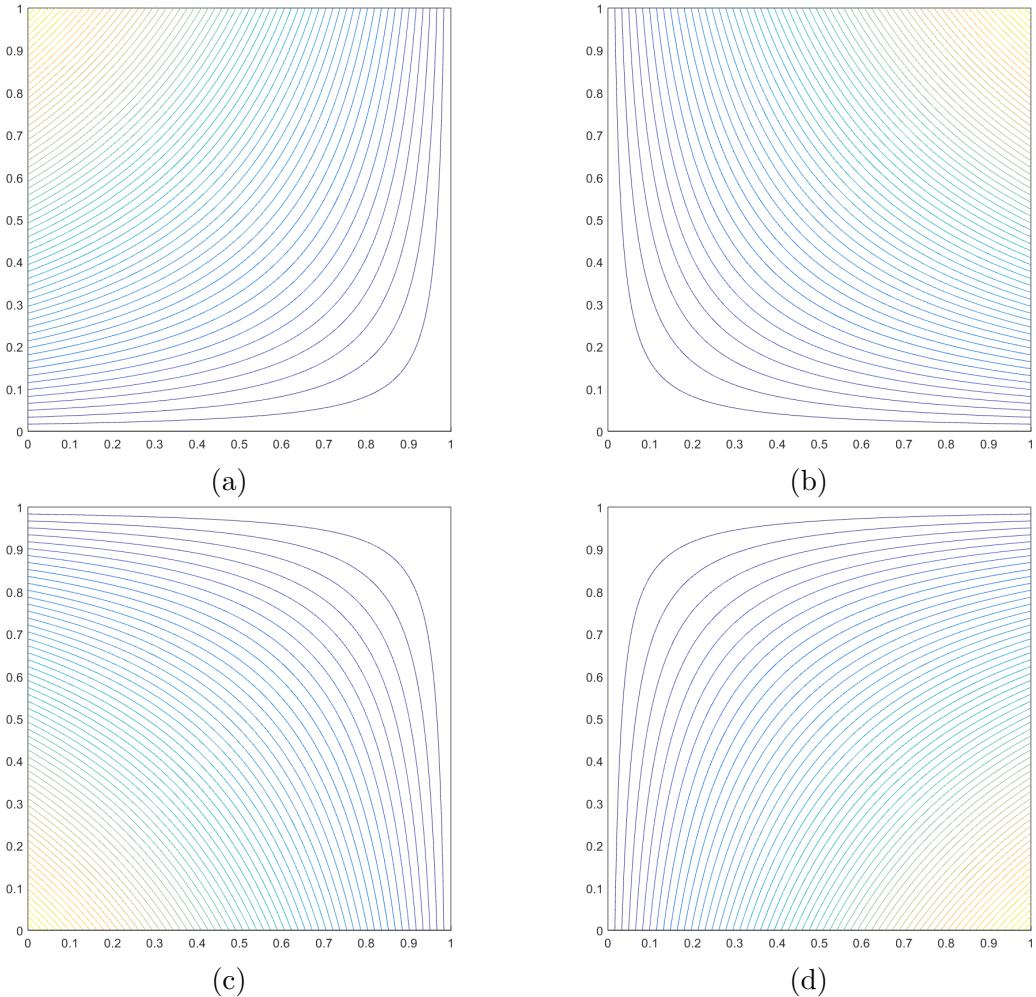


Figure 3.2: Contour plots of the linear Wachspress basis functions on the unit square for the vertices located at: (a) $(0,1)$, (b) $(1,1)$, (c) $(0,0)$, and (d) $(1,0)$.

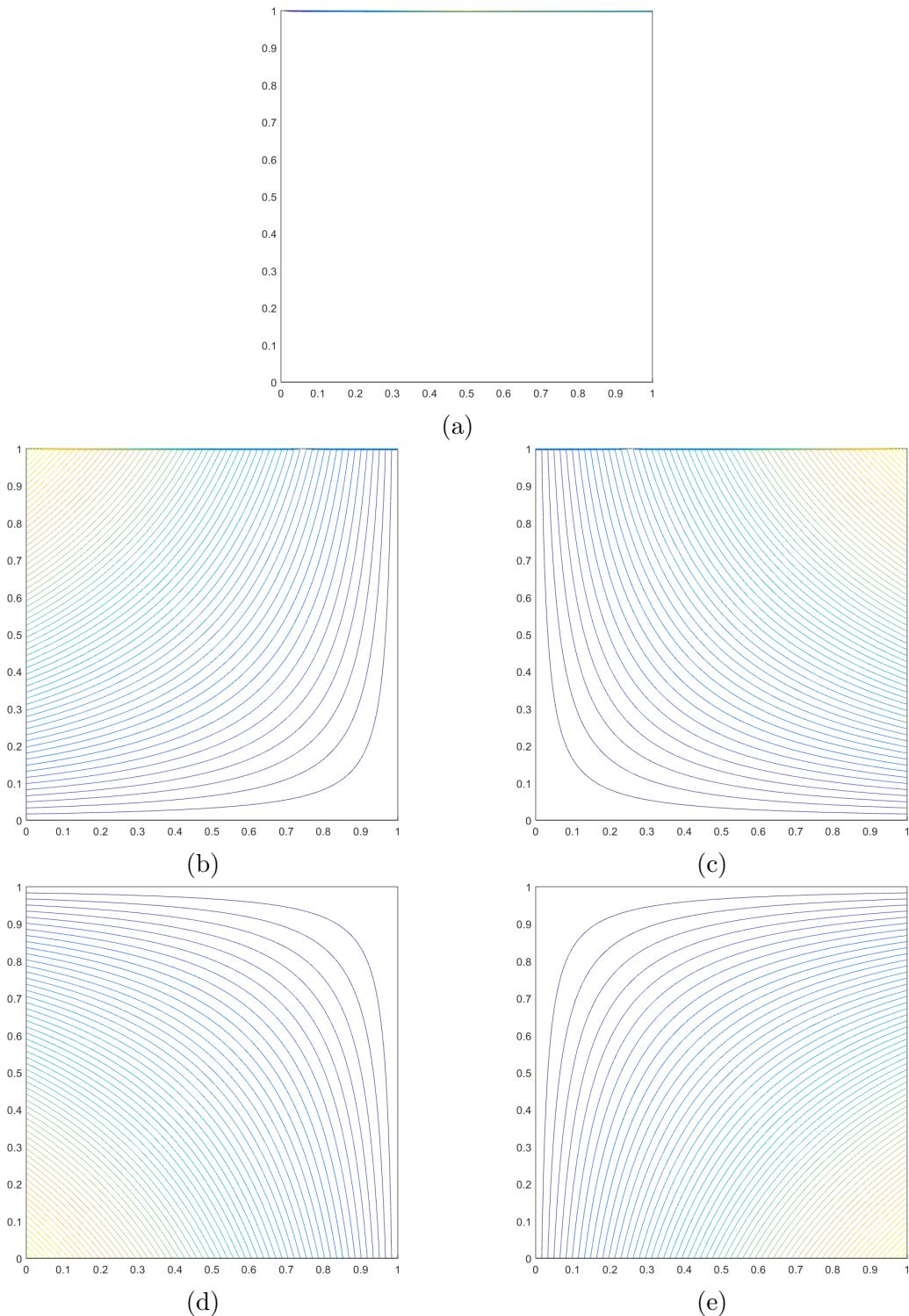


Figure 3.3: Contour plots of the linear Wachspress basis functions on the degenerate pentagon for the vertices located at: (a) $(1/2, 1)$, (b) $(0, 1)$, (c) $(1, 1)$, (d) $(0, 0)$, and (e) $(1, 0)$.

5. the PWL functions are only C^0 continuous: their gradients are discontinuous within the element.

The 2D PWL functions are defined as combinations of linear triangular functions, with some of them only having measure within a subregion of a polygon. These subregions are formed by triangulating the arbitrary 2D polygon into a set of sub-triangles. Each sub-triangle is defined by two adjacent vertices of the polygon (taken in a counter-clockwise ordering to maintain consistency) and the polygon's centroid, \vec{r}_c . Looking at Figure 3.1 as an example, sub-triangle j is defined by the points $\{\vec{x}_j, \vec{x}_{j+1}, \vec{r}_c\}$, which are the polygon's vertices j and $j+1$ and the polygon's centroid. If a polygon K has N_K vertices, then its centroid can be defined by

$$\vec{r}_c = \sum_{j=1}^{N_K} \alpha_j^K \vec{x}_j, \quad (3.17)$$

where α_j^K are the vertex weights functions and,

$$\sum_{j=1}^{N_K} \alpha_j^K = 1. \quad (3.18)$$

For this work, we continue to use the definition for the vertex weight functions from previous works [37, 38, 39],

$$\alpha_j^K = \frac{1}{N_K}. \quad (3.19)$$

This means that the vertex weight functions are the same for every vertex, and the cell centroid simply becomes the average position of all the vertices. However, we note that care must be taken so that the centroid does not lie on the polygon's boundary. This will cause the PWL functions to no longer have piecewise linearity along the boundary. Using these vertex weight functions, the PWL basis function

for vertex j , b_j^{PWL} , is defined as

$$b_j^{PWL}(x, y) = t_j(x, y) + \alpha_j^K t_c(x, y). \quad (3.20)$$

In Eq. (3.20), t_j is the standard 2D linear function with unity at vertex j that linearly decreases to zero at the cell center and each adjoining vertex. t_c is the 2D cell “tent” function located at \vec{r}_c which is unity at the cell center and linearly decreases to zero at each cell vertex. α_j^K is the weight parameter for vertex j in cell K . The functional form of Eq. (3.20) with identical vertex weights means that the PWL function for vertex j , within the domain of K , linearly decreases to a value of $1/N_K$ at the polygonal center. From there, the function linearly decreases to zero on all faces that are not connected to vertex j . In Appendix B, we detail how the 2D PWL coordinates can be analytically integrated using the reference triangle and affine mapping. The gradients of the PWL functions are easy to compute term-by-term in a straightforward manner:

$$\vec{\nabla} b_j^{PWL}(x, y) = \vec{\nabla} t_j(x, y) + \alpha_j^K \vec{\nabla} t_c(x, y). \quad (3.21)$$

We now give some example contour plots of the PWL coordinates over different polygons. First, we provide the contour plots for the four PWL functions on the unit square in Figure 3.4. In this example it is easy to discern the functional form of Eq. (3.20) with the use of constant vertex weights. We clearly see each function linearly decrease from its vertex to the cell center (with a value of $1/N_K$) and then linearly decrease to all non-adjoining faces. Next, Figure 3.5 provides the contour plots for the PWL functions on a degenerate (weakly-convex) pentagon where a fifth vertex was added to the unit square at $(1/2, 1)$. Unlike the Wachspress coordinates, the PWL functions work on weakly-convex polygons. The final example we give

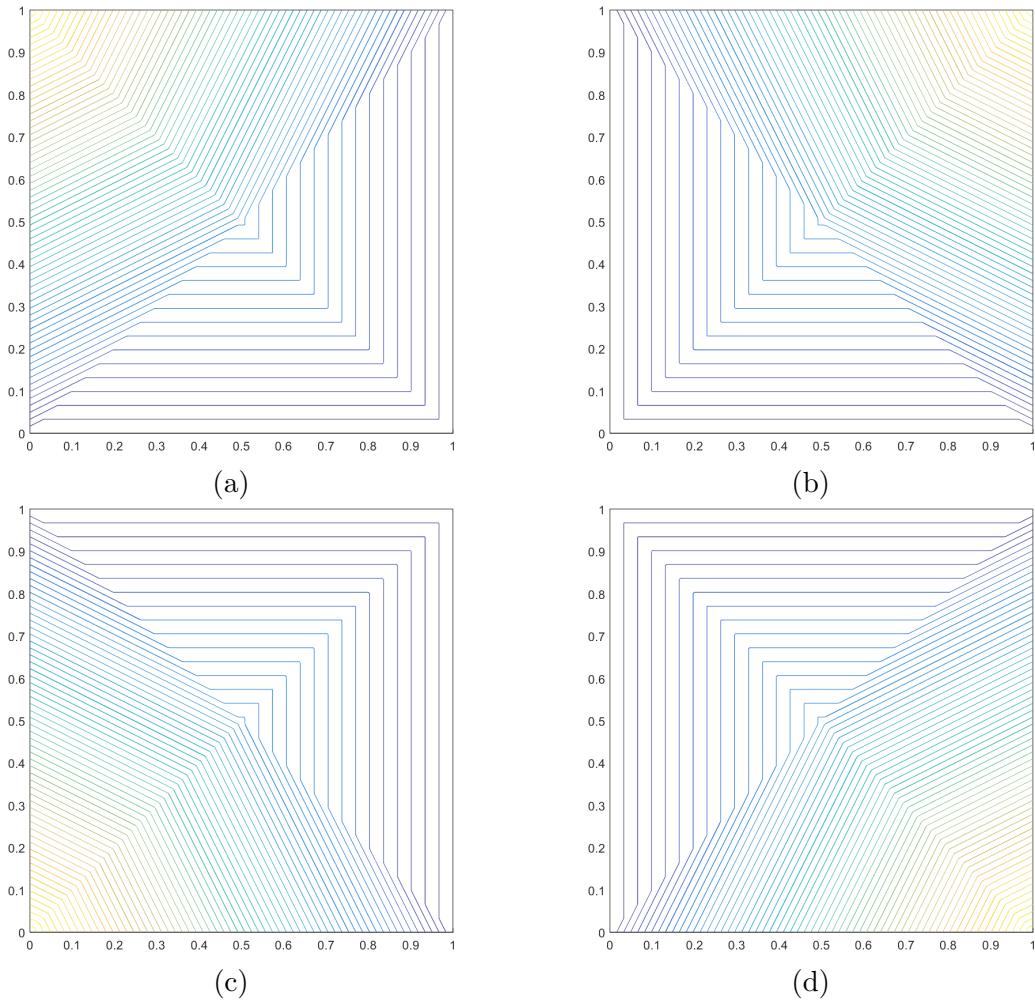


Figure 3.4: Contour plots of the linear PWL basis functions on the unit square for the vertices located at: (a) $(0,1)$, (b) $(1,1)$, (c) $(0,0)$, and (d) $(1,0)$.

in Figure 3.6 is a favorite in the applied mathematics community: the “L-shaped” domain. It provides an example of PWL’s ability to still be linearly-complete on concave polygons. In this example, the cell centroid was forced to be at the point $(1/3, 1/3)$ so that it would be inside the polygon.

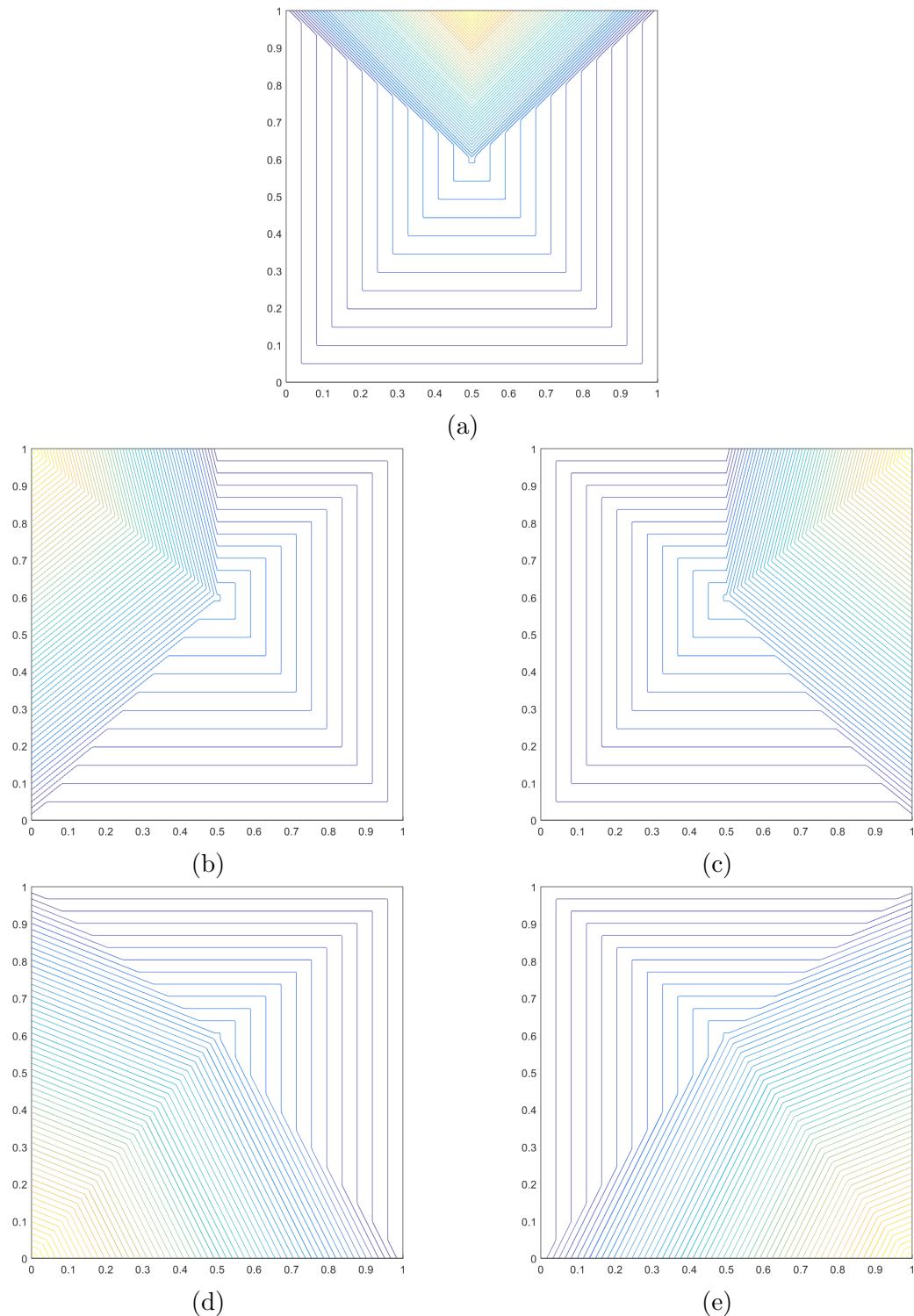


Figure 3.5: Contour plots of the linear PWL basis functions on the degenerate pentagon for the vertices located at: (a) $(1/2, 1)$, (b) $(0, 1)$, (c) $(1, 1)$, (d) $(0, 0)$, and (e) $(1, 0)$.

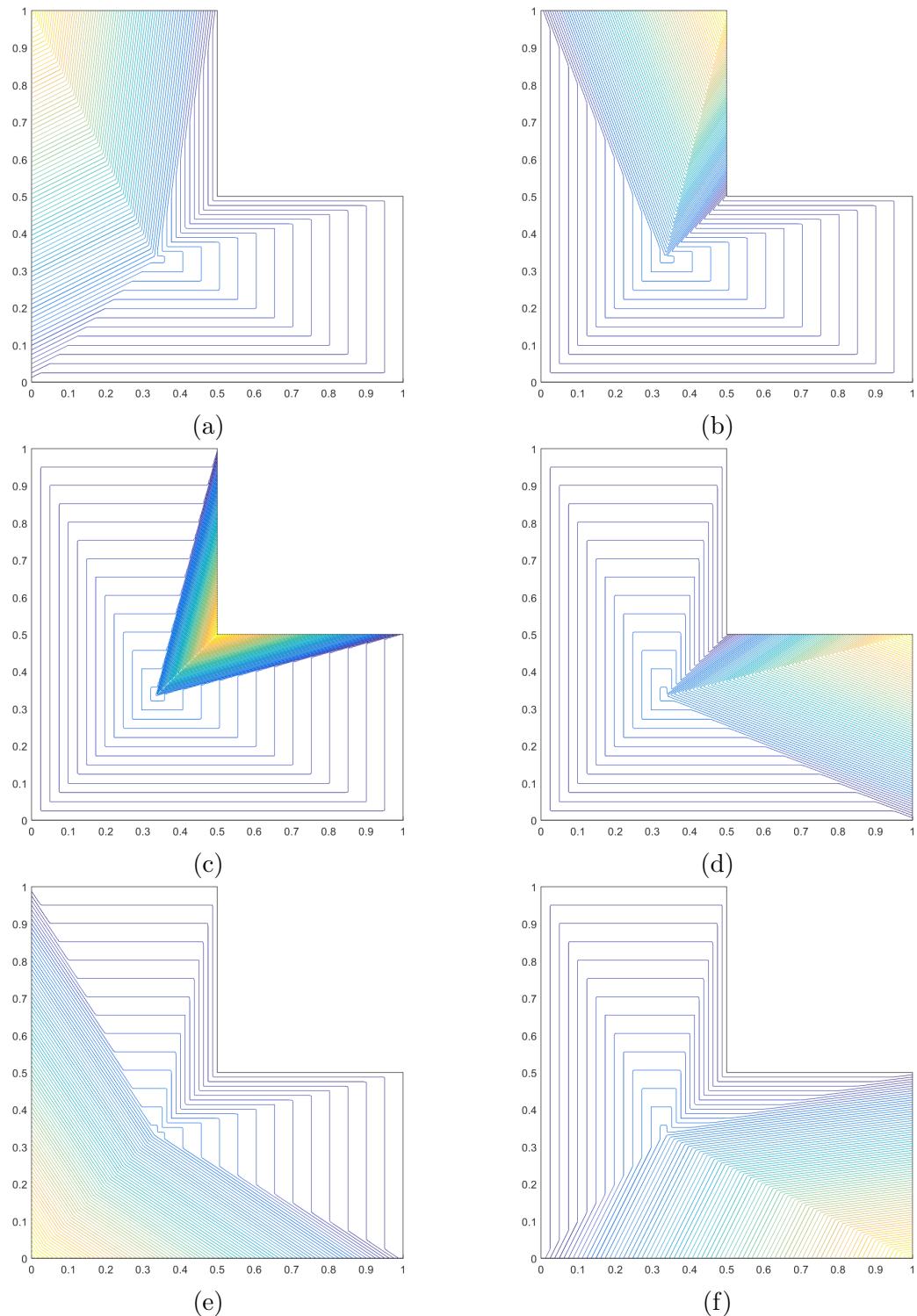


Figure 3.6: Contour plots of the linear PWL basis functions on the L-shaped domain for the vertices located at: (a) $(0,1)$, (b) $(1/2,1)$, (c) $(1/2,1/2)$, (d) $(1,1/2)$, (e) $(0,0)$, and (f) $(1,0)$.

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 [110]. 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. Injectivity is preserved if the interpolatory function is *harmonic* over the piecewise linear maps. This can be shown by expressing a C^2 function u over each sub-triangle, \mathcal{T} , of the triangulated polygon and have it satisfy the Laplace equation,

$$\nabla^2 u = 0, \quad (3.22)$$

where $u(\vec{r}_b) = u_0$ consists of a piecewise linear Dirichlet boundary condition ($\vec{r}_b \in \partial\mathcal{T}$) for each triangulation. Then, by use of the mean value theorem (where this coordinate system got its name), the mean value function at vertex j , b_j^{MV} , for a polygon K with N_K vertices can be given by

$$b_j^{MV}(\vec{x}) = \frac{w_j(\vec{x})}{\sum_{i=1}^{N_K} w_i(\vec{x})}, \quad (3.23)$$

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

$$w_j(\vec{x}) = \frac{\tan(\alpha_{j-1}/2) + \tan(\alpha_j/2)}{|\vec{x}_j - \vec{x}|}. \quad (3.24)$$

This weight simply consists of the addition of the two tangent functions (where the angles are given in Figure 3.1) that is then divided through by the distance between the vertex j and the point of interest, \vec{x} . Through careful observation of Eq. (3.24),

we can see that these MV weights are undefined on certain portions of the polygon's boundary, ∂K , in a similar way to the Wachspress coordinates. However, the limits of the coordinates are bounded on the polygon's faces and vertices, and we rigorously show this in Appendix B.

We now give the form of the mean value gradients. Since the mean value coordinates given by Eq. (3.23) have the same form as the Wachspress coordinates, their gradients can be expressed in an identical manner. The gradients of the mean value coordinates can be expressed as

$$\vec{\nabla} b_j^{MV}(\vec{x}) = b_j^{MV}(\vec{x}) \left(\vec{R}_j(\vec{x}) - \sum_i b_i^{MV}(\vec{x}) \vec{R}_i(\vec{x}) \right), \quad (3.25)$$

where the reduced gradient, \vec{R}_j , still has the same definition from Eq. (3.14). If we define $t_j = \tan(\alpha_j/2)$ and $t_{j-1} = \tan(\alpha_{j-1}/2)$, then after extensive algebra (which we will not show), the mean value reduced gradients are

$$\vec{R}_j = \left(\frac{t_{j-1}}{t_{j-1} + t_j} \right) \frac{\{\vec{c}_{j-1}\}}{\sin(\alpha_{j-1})} + \left(\frac{t_j}{t_{j-1} + t_j} \right) \frac{\{\vec{c}_j\}}{\sin(\alpha_j)} + \frac{\vec{g}_j}{|\vec{x}_i - \vec{x}|}, \quad (3.26)$$

where

$$\vec{c}_j = \frac{\vec{g}_j}{|\vec{x}_j - \vec{x}|} - \frac{\vec{g}_{j+1}}{|\vec{x}_{j+1} - \vec{x}|}, \quad (3.27)$$

and

$$\vec{g}_j = \frac{\vec{x}_j - \vec{x}}{|\vec{x}_j - \vec{x}|}, \quad (3.28)$$

and

$$\{\vec{u}\} = (-u_2, u_1). \quad (3.29)$$

While this direct form for the MV coordinates is more complicated than the last two coordinates presented, it is still easily programmable. The interested reader can look in the appendix of [111] for MATLAB code to compute these gradients.

We again provide example contour plots of the MV coordinates, and we use the same polygonal shapes that we showed for the PWL coordinates. Figure 3.7 gives the MV coordinates on the unit square. Like the Wachspress functions, the MV coordinates are smoothly varying within the domain of the polygon and possess continuous derivatives. Next in Figure 3.8, we give the example contour plots for the degenerate pentagon which is formed by inserting a vertex into the unit square at $(1/2, 1)$. Finally, Figure 3.9 gives the contour plots for the linear mean value coordinates on the L-shaped domain. We see that the linear mean value coordinates are applicable to interpolation on concave polygons.

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) [112, 113, 114]. These coordinates have garnered much interest in different fields because their functional forms can be tailored based on the application. The principle of maximum entropy stems from the concept of *Shannon entropy* [115]. If we define a functional for the Shannon entropy as H , then its maximum will lead to the least-biased statistical inference for some set of testable constraints [116]. For the application of FEM analysis, these testable constraints correspond to those given in Eq. (3.5). For N_K discrete probability functions (corresponding to the N_K vertex functions on polygon K), the functional form for the Shannon entropy can be given by

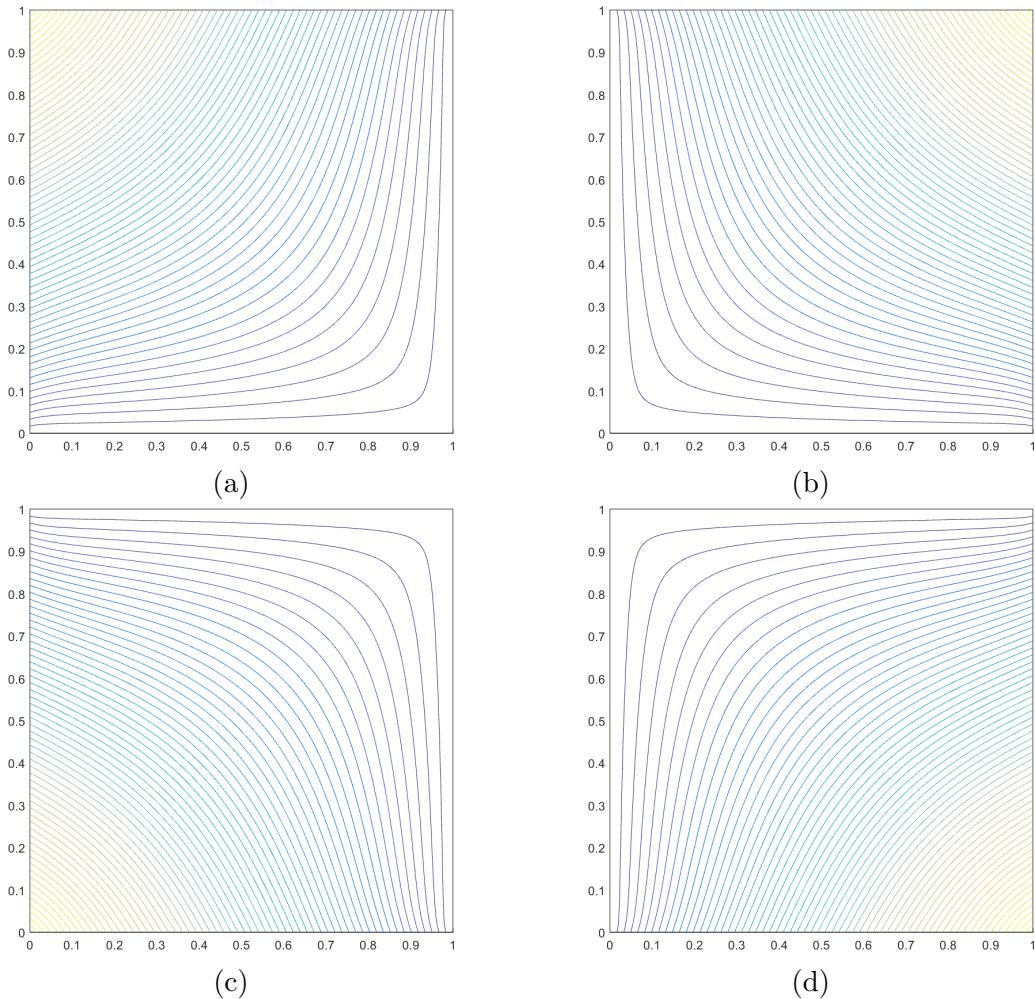


Figure 3.7: Contour plots of the linear mean value basis functions on the unit square for the vertices located at: (a) $(0,1)$, (b) $(1,1)$, (c) $(0,0)$, and (d) $(1,0)$.

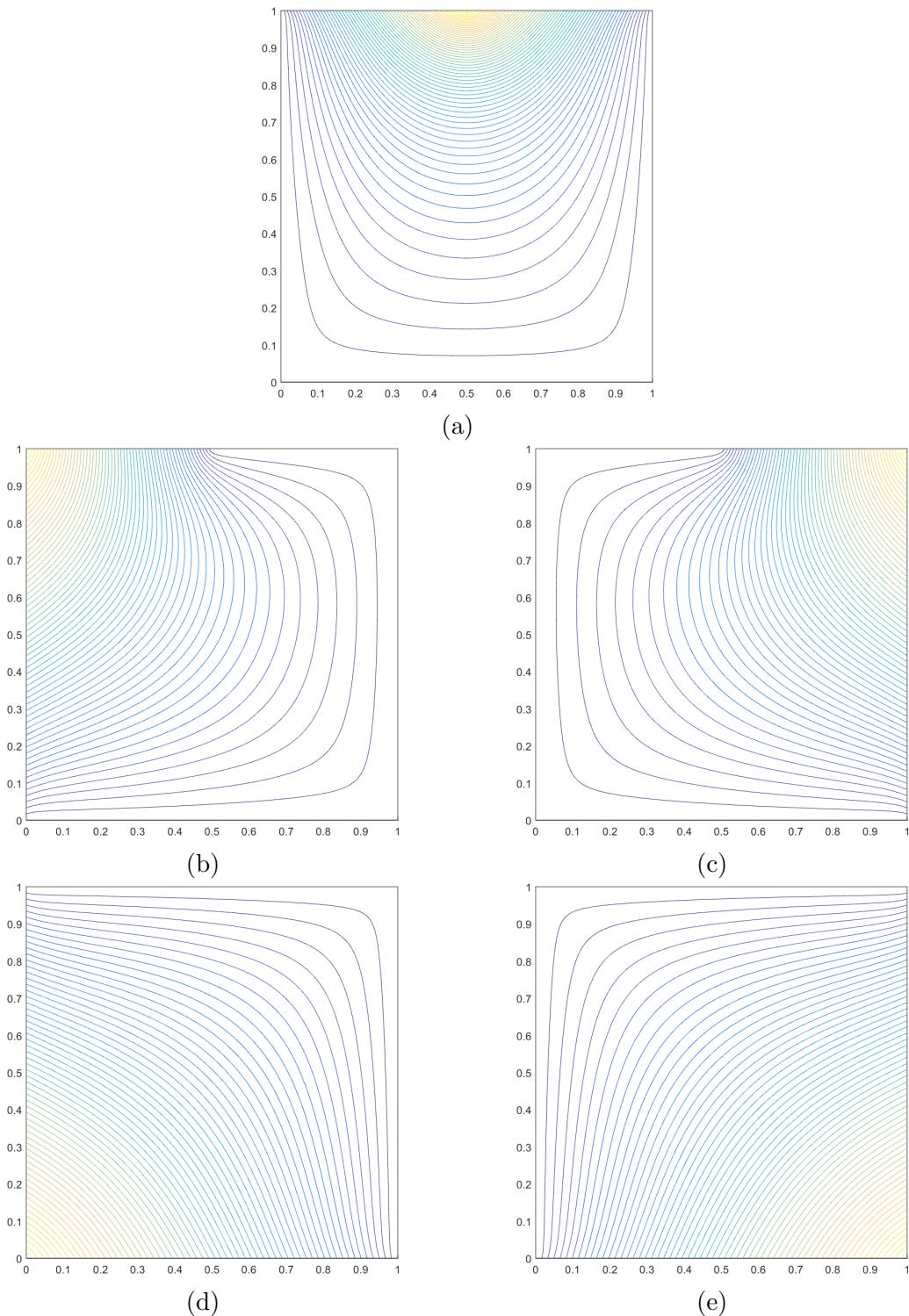


Figure 3.8: Contour plots of the linear mean value basis functions on the degenerate pentagon for the vertices located at: (a) $(1/2, 1)$, (b) $(0, 1)$, (c) $(1, 1)$, (d) $(0, 0)$, and (e) $(1, 0)$.

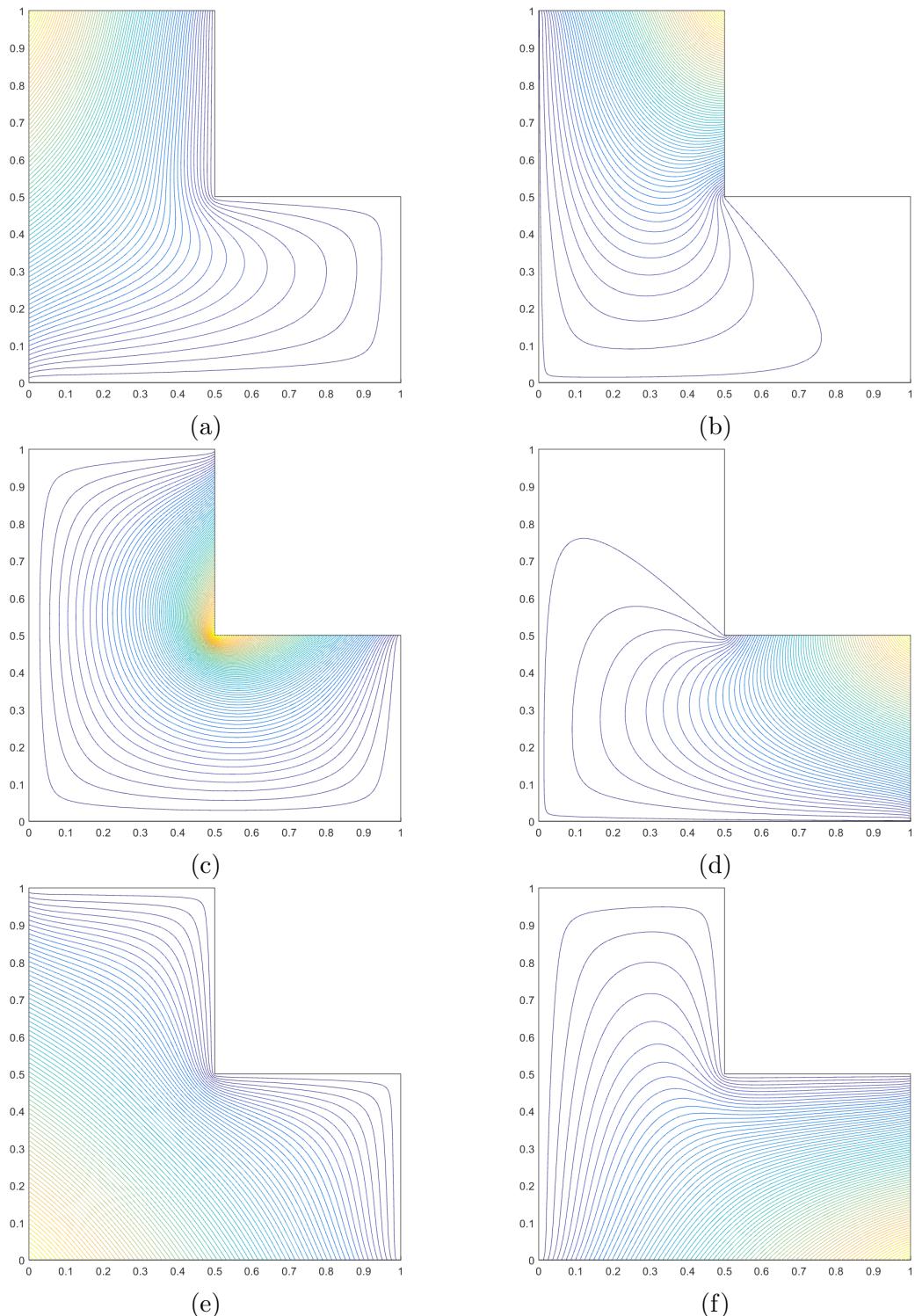


Figure 3.9: Contour plots of the linear mean value basis functions on the L-shaped domain for the vertices located at: (a) $(0,1)$, (b) $(1/2,1)$, (c) $(1/2,1/2)$, (d) $(1,1/2)$, (e) $(0,0)$, and (f) $(1,0)$.

$$H(b, m) = - \sum_{j=1}^{N_K} b_j \log \left(\frac{b_j}{m_j} \right), \quad (3.30)$$

where m_j is called the prior distribution. These prior distributions are a key component of Bayesian inference [117, 118]. If $(b_j^{ME}(\vec{x}), j = 1, \dots, N_K)$ is the solution of the following constrained optimization problem

$$\max_{b(\vec{x})} H(b, m, \vec{x}), \quad (3.31)$$

then the maximum entropy coordinates can be given by

$$b_j^{ME}(\vec{x}) = \frac{w_j(\vec{x})}{\sum_{i=1}^{N_K} w_i(\vec{x})}. \quad (3.32)$$

In Eq. (3.32), the maximum entropy weight function for vertex j , w_j , has the following definition,

$$w_j(\vec{x}) = m_j(\vec{x}) \exp(-\vec{\kappa} \cdot (\vec{x}_j - \vec{x})), \quad (3.33)$$

where $\vec{\kappa}$ is a vector value of dimension d that will be explained shortly. In the context of Eq. (3.33), the prior distribution, m_j , can be viewed as 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. For FEM applications, an appropriate functional form for the prior distribution is given by

$$m_j(\vec{x}) = \frac{\pi_j(\vec{x})}{\sum_{k=1}^{N_K} \pi_k(\vec{x})}, \quad (3.34)$$

where

$$\pi_j(\vec{x}) = \prod_{k \neq j-1, j}^{N_K} \rho_k(\vec{x}) = \rho_1(\vec{x}) \dots \rho_{j-2}(\vec{x}) \rho_{j+1}(\vec{x}) \dots \rho_{N_K}(\vec{x}), \quad (3.35)$$

and

$$\rho_j(\vec{x}) = \|\vec{x} - \vec{x}_j\| + \|\vec{x} - \vec{x}_{j+1}\| - \|\vec{x}_{j+1} - \vec{x}_j\|. \quad (3.36)$$

In Eqs. (B.47) and (B.48), we have defined a new weight function, ρ_j , that corresponds to face j between vertices j and $j + 1$. These face functions are zero along face j , but strictly positive elsewhere due to the triangle inequality. This means that the vertex function π_j is also non-negative and vanishes on all faces that are not adjacent to vertex j . This is important so as to ensure that each of the ME vertex functions are strictly zero on all faces that are not connected to the vertex. However, this also means that once again, the ME coordinates cannot be directly evaluated on the polygon's boundary. We show that the limits of these functions are bounded on the polygon's faces and vertices in Appendix B.

Now that we have provided sufficient details for the basis functions and their weight functions, we can explain how the $\vec{\kappa}$ vector in Eq. (3.33) is computed. We can see that once $\vec{\kappa}$ is known, the ME functions can be directly calculated. To do this, we solve the constrained optimization problem of Eq. (3.31) through the use of Lagrange multipliers with a Newton's method. If we define κ_0 as the Lagrange multiplier for the constant constraint, then the Lagrangian for the problem of Eq. (3.31) is given by

$$\begin{aligned}\mathcal{L}(\vec{b}; \kappa_0, \vec{\kappa}) = & - \sum_{j=1}^{N_K} b_j \log \left(\frac{b_j}{m_j} \right) - \kappa_0 \left(\sum_{j=1}^{N_K} b_j - 1 \right), \\ & - \vec{\kappa} \cdot \left(\sum_{j=1}^{N_K} b_j (\vec{x}_j - \vec{x}) \right)\end{aligned}\quad (3.37)$$

where we omitted the spatial parameter \vec{x} for brevity. If we take the first variation of \mathcal{L} to zero ($\delta\mathcal{L}(\vec{b}; \kappa_0, \vec{\kappa}) = 0$), then we obtain

$$\left[-1 - \log \left(\frac{b_j}{m_j} \right) - \kappa_0 - \vec{\kappa} \cdot (\vec{x}_j - \vec{x}) \right] \delta b_j = 0. \quad (3.38)$$

Since δb_j is arbitrary, everything within the bracket of Eq. (3.38) must be equal to zero. If we use the substitution $\log W = 1 + \kappa_0$ and extensive algebra, then we can rewrite Eq. (3.38) as

$$b_j(\vec{x}) = \frac{m_j(\vec{x}) \exp(-\vec{\kappa} \cdot (\vec{x}_j - \vec{x}))}{W}, \quad (3.39)$$

where $W = \sum_{j=1}^{N_K} w_j(\vec{x})$. We can immediately see that this satisfies the form for our maximum entropy coordinates given in Eq. (3.32). This means that we are just left with describing the non-linear numerical procedure to compute $\vec{\kappa}$. The solution of Eq. (3.39) is equivalent to solving the following dual unconstrained optimization problem [119]:

$$\vec{\kappa}^* = \min_{\vec{\kappa}} F(\vec{\kappa}), \quad F(\vec{\kappa}) = \log W(\vec{\kappa}) \quad (3.40)$$

In Eq. (3.40), the zero of $F(\vec{\kappa})$ is the non-linear function to be solved with Newton's method. We summarize all the steps needed for computation as follows:

1. Compute and store $(\vec{x}_j - \vec{x})$ and the prior functions $m_j(\vec{x})$;

2. Start with iteration counter at $k = 0$ and initialize the Lagrange multiplier:

$$\vec{\kappa}^0 = \vec{0};$$

3. Compute the gradient of F , $\vec{g}_k = \vec{\nabla}_\kappa F(\vec{\kappa}^k)$, and its Hessian, $H_k = \vec{\nabla}_\kappa \vec{\nabla}_\kappa F(\vec{\kappa}^k)$;
4. Determine the Newton search direction: $\Delta \vec{\kappa}^k = -H_k^{-1} \vec{g}_k$;
5. Update the multiplier: $\vec{\kappa}^{k+1} = \vec{\kappa}^k + \alpha \Delta \vec{\kappa}^k$;
6. Check convergence by testing if $\|g_{k+1}\| > \epsilon$.
7. Set $\vec{\kappa}^* = \vec{\kappa}^{k+1}$ and compute $\vec{b}(\vec{x})$

In these computational procedures, α is the damping parameter. If the error at iteration k , $\|\vec{g}_k\|$ is greater than 10^{-4} , then a line search algorithm is used [120]. Otherwise, α can be set to unity as the error decreases. We note that a line search algorithm must be used for certain classes of polygonal shapes. For extremely-distorted concave polygons, this Newton iteration procedure can be unstable without it. Due to the quadratic convergence of Newton's method in the vicinity of the final solution, only 3-7 Newton iterations should be required to obtain accuracies of at least 10^{-10} .

We conclude our discussion of the maximum entropy coordinates by providing example plots over the same polygonal domains that we showed for the PWL and MV coordinates. First, we give the contour plots of the ME coordinates over the unit square in Figure 3.10. Similar to the Wachspress and MV coordinates, the maximum entropy coordinates are smoothly varying within the domain of the polygon and possess continuous derivatives (in contrast to the PWL coordinates). Next in Figure 3.11, we give the contour plots of the degenerate polygon. Finally, Figure 3.12 gives the contour plots for the linear maximum entropy coordinates on the L-shaped domain. We can see that these coordinates can appropriately interpolate functions on concave polygons.

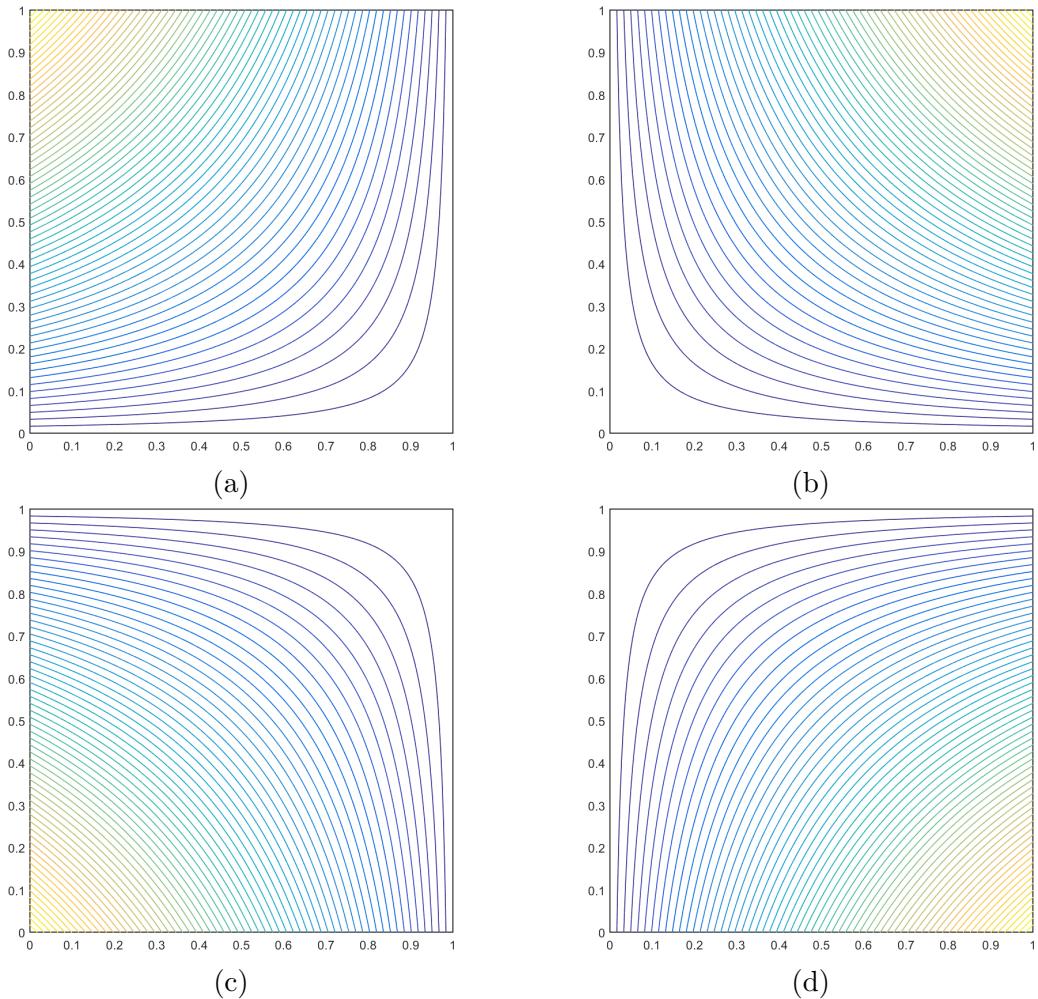


Figure 3.10: Contour plots of the linear maximum entropy basis functions on the unit square for the vertices located at: (a) $(0,1)$, (b) $(1,1)$, (c) $(0,0)$, and (d) $(1,0)$.

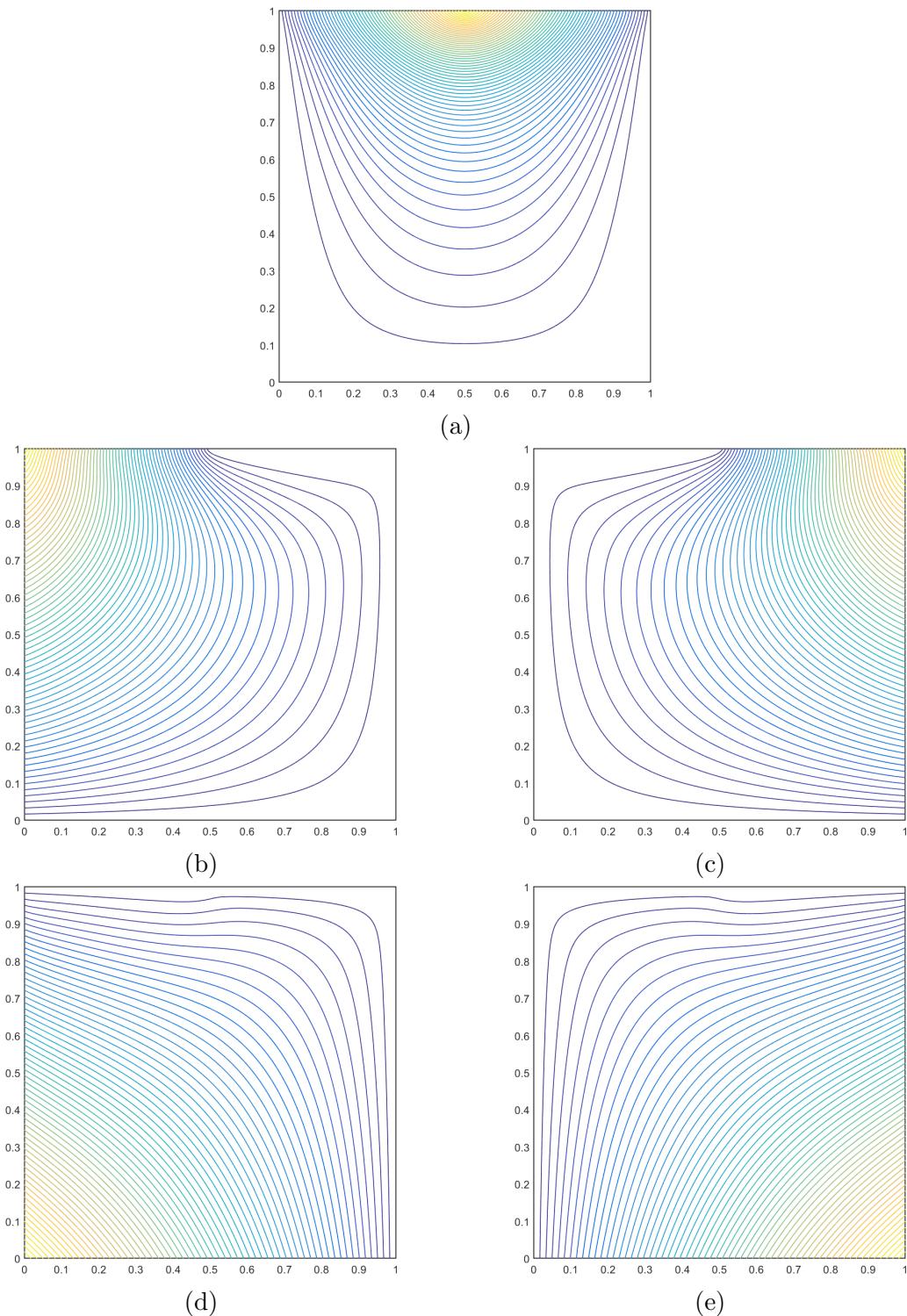


Figure 3.11: Contour plots of the linear maximum entropy basis functions on the degenerate pentagon for the vertices located at: (a) $(1/2, 1)$, (b) $(0, 1)$, (c) $(1, 1)$, (d) $(0, 0)$, and (e) $(1, 0)$.

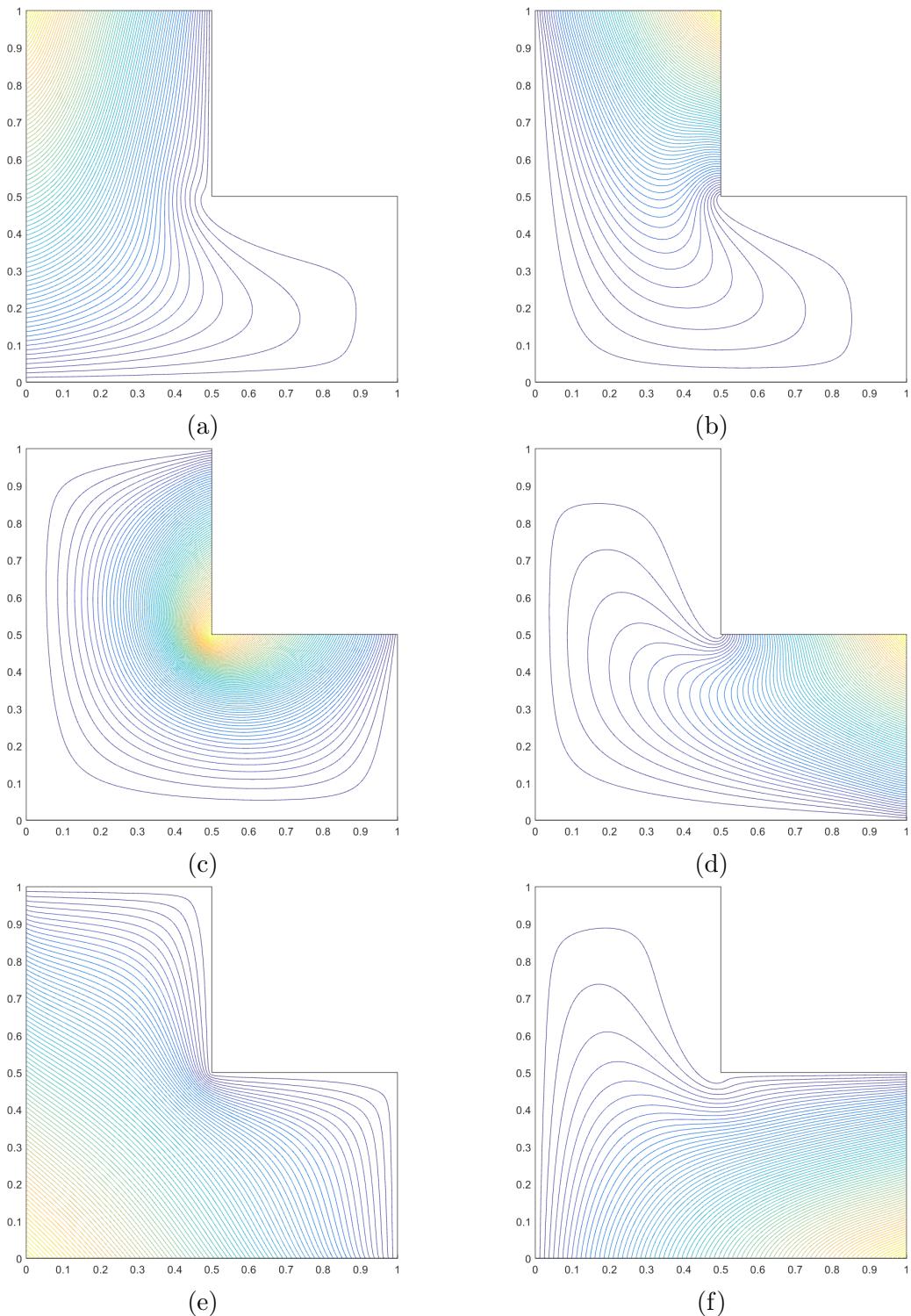


Figure 3.12: Contour plots of the linear maximum entropy basis functions on the L-shaped domain for the vertices located at: (a) $(0,1)$, (b) $(1/2,1)$, (c) $(1/2,1/2)$, (d) $(1,1/2)$, (e) $(0,0)$, and (f) $(1,0)$.

3.1.5 Summary of 2D Linear Basis Functions on Polygons

In Sections 3.1.1, 3.1.2, 3.1.3, and 3.1.4, we presented the linear Wachspress, PWL, mean value, and maximum entropy coordinates, respectively. We gave the functional forms for their values and gradients along with example contour plots for different polygonal elements. Table 3.1 provides a summary of the properties for the different coordinates. The Wachspress, PWL, and ME functions have natural extensions to 3D polyhedra, but the MV functions can only interpolate on polyhedra with triangular facets [121, 122]. The PWL, MV, and ME coordinates can interpolate degenerate-convex and concave polygons while the Wachspress functions can only interpolate strictly-convex polygons. This means that the Wachspress coordinates are not suited for AMR calculations that form degenerate polygons. PWL is the only functional form that can analytically integrate the elementary matrices and directly evaluate its values and gradients on the boundary of the polygon. Finally, every point within the domain can be directly evaluated using the Wachspress, PWL, and MV coordinates. However, the ME coordinates use an iterative approach with Newton's method since their functional form constitutes a non-linear minimization problem.

We conclude this summary discussion on the different linearly-complete 2D polygonal coordinates by again presenting examples of their functional forms. Figure 3.13 provides the contour plots of the different coordinates located at vertex $(0, 1)$ on the unit square. It is easy to see that the Wachspress, MV, and ME coordinates are smoothly varying within the polygon's domain while the PWL coordinates only have C^0 continuity. Figure 3.14 provides the contour plots on the degenerate pentagon that is formed by inserting a vertex at $(1/2, 1)$ into the unit square. This re-emphasizes that the Wachspress coordinates are only valid interpolatory functions on strongly-convex polygons. Finally, Figure 3.15 provides the contour plots of

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

Basis Function	Dimension	Polygon Type	Integration	Evaluation
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

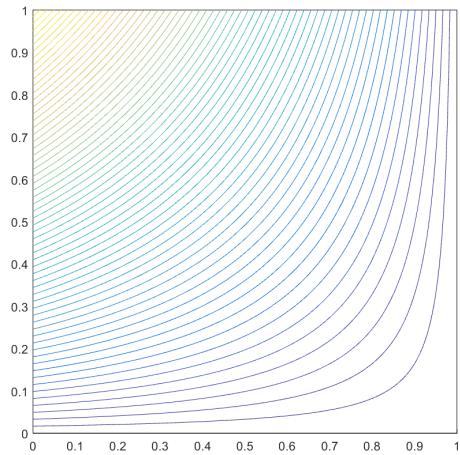
the PWL, MV, and ME functions on the L-shaped domain at the $(0, 1)$ and $(1/2, 1/2)$ vertices. These coordinates can successfully interpolate on concave polygons.

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

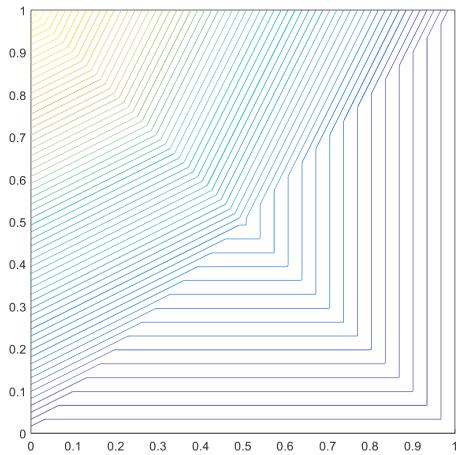
We have given complete details on the linearly-complete generalized barycentric coordinates that we will investigate for this work. Now we describe how to convert any generalized barycentric coordinate into the quadratic serendipity space of functions to yield quadratic (not linear) precision based on the work of Rand et al. [107]. The maximum entropy coordinates were independently converted into the quadratic space [123, 124]. These 2D serendipity coordinates can exactly interpolate the $\{1, x, y, x^2, xy, y^2\}$ span of functions, which can be shown with the first three levels of Pascal's triangle:

$$\begin{array}{ccc}
 & 1 & \\
 & x & y \\
 x^2 & xy & y^2
 \end{array} \tag{3.41}$$

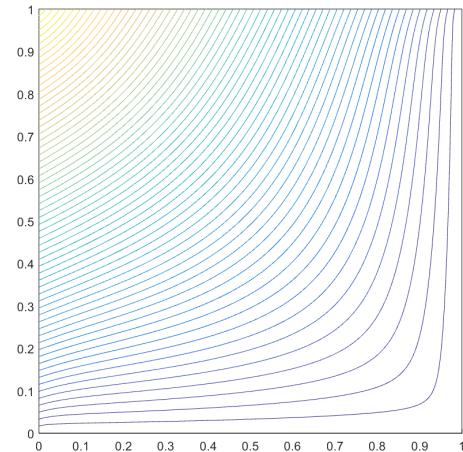
Between the linear and quadratic precision of the generalized barycentric coordinates and the quadratic serendipity extension, a relation can be formed on the functional space of their precision. If we seek a functional space up to order p precision, then for



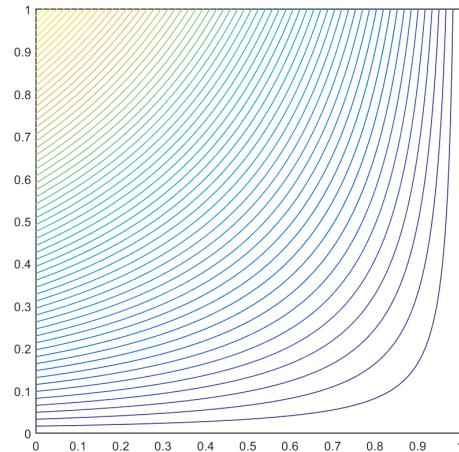
(a) Wachspress



(b) PWL

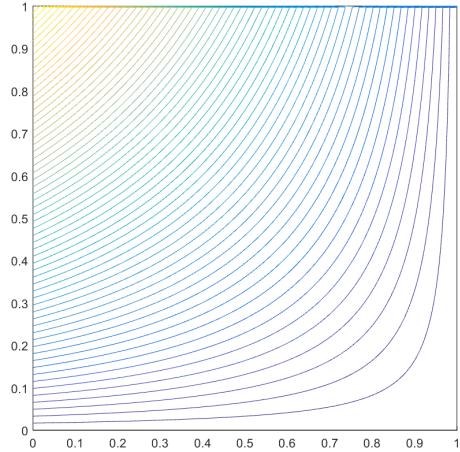


(c) Mean Value

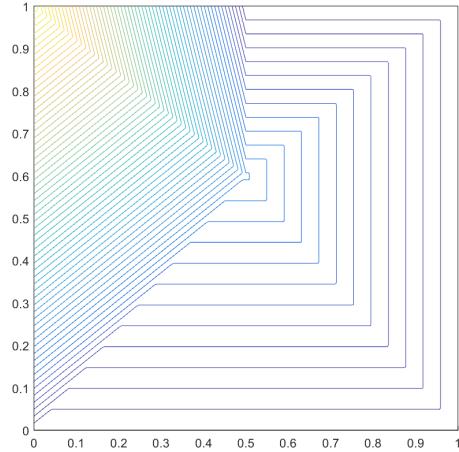


(d) Maximum Entropy

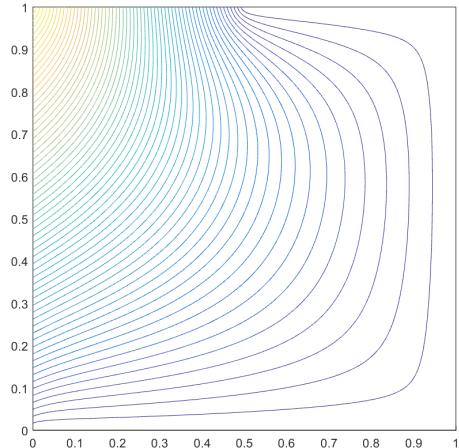
Figure 3.13: Contour plots of the different linear basis function on the unit square located at vertex $(0,1)$.



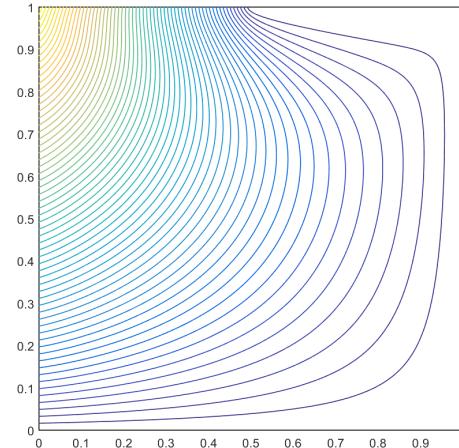
(a) Wachspress



(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.

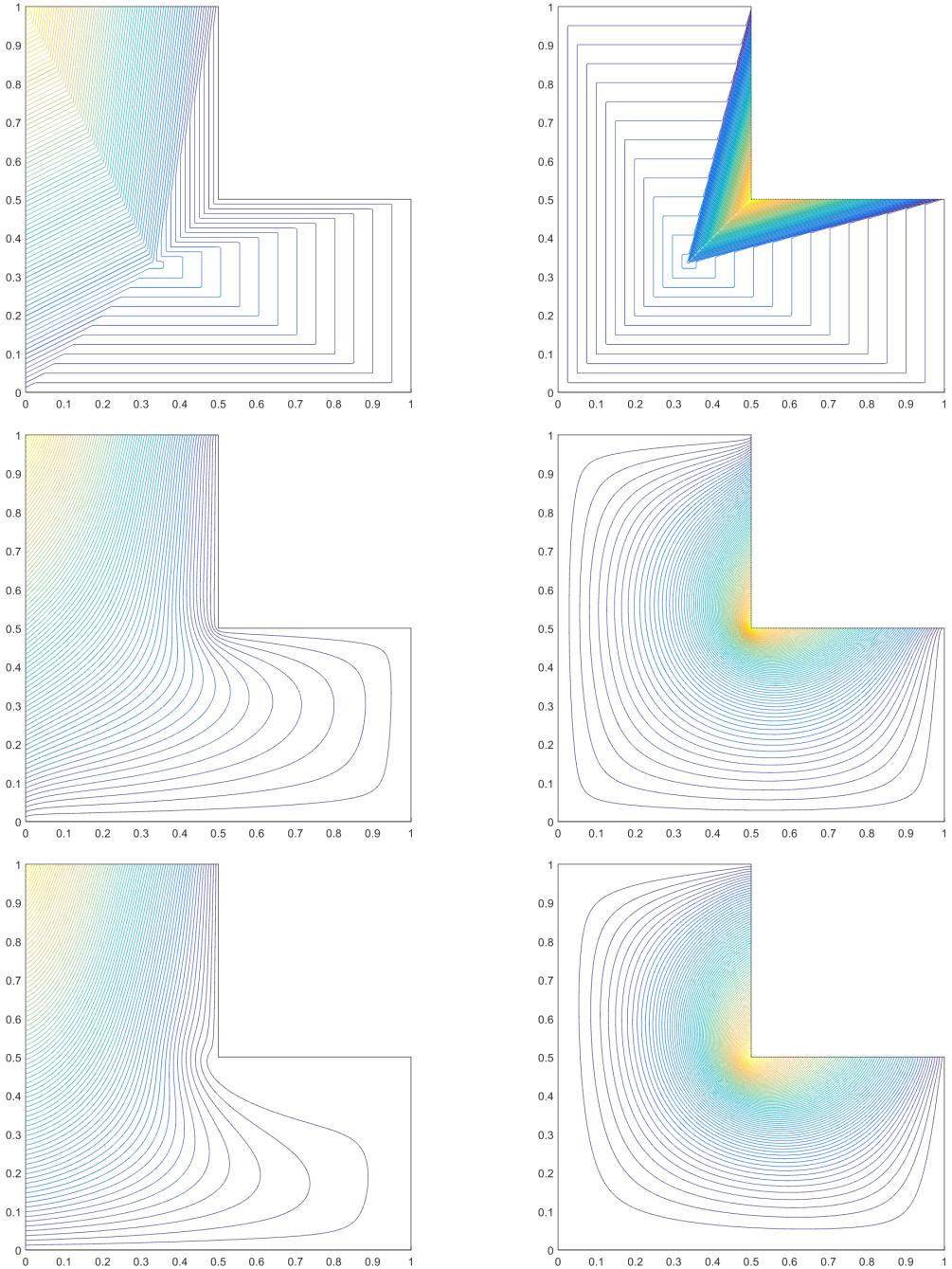


Figure 3.15: Contour plots of the different linear basis functions on the L-shaped domain. The PWL (top), mean value (middle), and maximum entropy (bottom) functions are plotted at vertices $(0, 1)$ (left) and $(1/2, 1/2)$ (right).

$k = 0, \dots, p$, the interpolatory functions must exactly span the following monomial functions,

$$f_{\sigma,\tau}^k(x, y) = x^\sigma y^\tau, \quad (3.42)$$

where $\sigma + \tau = k$. From the first three levels of Pascal's triangle given in Eq. (3.41), we see that $k = 0$ gives the constant function $\{1\}$, that $k = 1$ gives the linear functions $\{x, y\}$, and that $k = 2$ gives the quadratic functions $\{x^2, xy, y^2\}$.

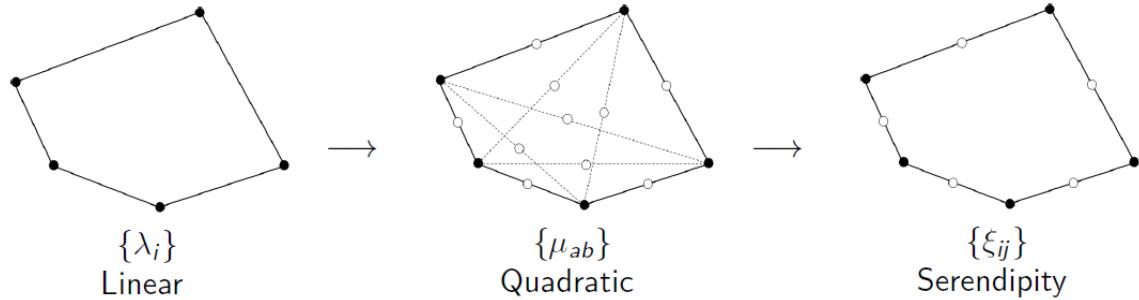


Figure 3.16: Overview of the process to construct the quadratic serendipity basis functions on polygons. The filled dots correspond to basis functions that maintain the Lagrange property while empty dots do not.

Now, we give the full details on converting the linear generalized barycentric coordinates to the quadratic serendipity space of functions. Figure 3.16 gives a visual depiction of this conversion process. For a polygon with N_K vertices, we can summarize this procedure as the following:

1. For a point \vec{x} , compute the N_K linear barycentric functions $\{\lambda_i(\vec{x})\}$ (Wachspress, PWL, mean value, or maximum entropy);
2. Take all non-repeating pairwise products of the linear functions to obtain $\frac{N_K(N_K+1)}{2}$ quadratic functions $\{\mu_{ab}\}$;

3. Form the linear transformation matrix \mathbb{A} through the use of monomial constraint equations;
4. Use the \mathbb{A} matrix to reduce the $\{\mu_{ab}\}$ function set to the $2N_K$ serendipity basis set $\{\xi_{ij}\}$.

We begin by computing the ($i = 1, \dots, N_K$) linear barycentric functions, $\{\lambda_i(\vec{x})\}$, and their gradients, $\{\vec{\nabla} \lambda_i(\vec{x})\}$, for a point \vec{x} . These linearly-complete barycentric functions can be converted immediately to barycentric-like functions with quadratic precision. Taking all non-repeating pairwise products of the linear functions yields a total of $N_Q = \frac{N_K(N_K+1)}{2}$ quadratic functions: $\mu_{ab} = \lambda_a \lambda_b$. Doing this generates functions that either live on the polygon's vertices, mid-face points, or midpoints of the polygon's diagonals between two vertices as seen in Figure 3.16. The N_K vertex functions are denoted as $ab \in V$ ($a = b$), the N_K mid-face (mid-edge) functions are denoted as $ab \in E$ ($|a - b| = 1$), and the $\frac{N_K(N_K-3)}{2}$ diameter (interior) functions are denoted as $ab \in D$ ($|a - b| > 1$). For the mid-edge and interior functions, only 1 combination of ab is kept since $\mu_{ab} = \mu_{ba}$. We also define the abbreviated notation of $\vec{x}_{ab} = \frac{\vec{x}_a + \vec{x}_b}{2}$, so that \vec{x}_{aa} corresponds to a vertex function at \vec{x}_a . Using these various notations, we can write the precision properties of the μ_{ab} functions for the constant constraint,

$$\sum_{aa \in V} \mu_{aa}(\vec{x}) + \sum_{ab \in E \cup D} 2\mu_{ab}(\vec{x}) = 1, \quad (3.43)$$

for the linear constraint,

$$\sum_{aa \in V} \mu_{aa}(\vec{x}) \vec{x}_{aa} + \sum_{ab \in E \cup D} 2\mu_{ab}(\vec{x}) \vec{x}_{ab} = \vec{x}, \quad (3.44)$$

and for the quadratic constraint,

$$\sum_{aa \in V} \mu_{aa}(\vec{x}) (\vec{x}_a \otimes \vec{x}_a) + \sum_{ab \in E \cup D} \mu_{ab}(\vec{x}) (\vec{x}_a \otimes \vec{x}_b + \vec{x}_b \otimes \vec{x}_a) = \vec{x} \otimes \vec{x}. \quad (3.45)$$

In Eq. (3.45), \otimes is the dyadic tensor product. We can immediately see that quadratic precision is ensured. However, the set of these pairwise quadratic functions grows quadratically. This means that as N_K grows large, the number of interpolatory functions grows as order $O(N_K^2)$, but still only maintains precision of the $\{1, x, y, x^2, xy, y^2\}$ span of functions. Therefore, the computational work required to utilize these quadratic barycentric functions can become prohibitive for polygons with large vertex counts.

To minimize the number of interpolatory functions but still maintain the precision of the $\{1, x, y, x^2, xy, y^2\}$ span of functions, we seek to convert the quadratic barycentric functions into the quadratic serendipity space of functions $\{\xi_{ij}\}$. This quadratic serendipity space only contains the vertex and mid-face functions (total of $2N_K$) and has been extensively studied for tensor-based elements in the past [89, 90]. This means that we seek to reduce the $\{\mu_{ab}\}$ set of functions by removing the diagonal functions ($ab \in D$) while still maintaining quadratic precision. If we define ξ_{ii} and $\xi_{i(i+1)}$ as the serendipity functions that live at vertex i and the mid-face point between vertices i and $i+1$, respectively, then we can write the serendipity precision properties for the constant constraint,

$$\sum_{ii \in V} \xi_{ii}(\vec{x}) + \sum_{i(i+1) \in E} 2\xi_{i(i+1)}(\vec{x}) = 1, \quad (3.46)$$

for the linear constraint,

$$\sum_{ii \in V} \xi_{ii}(\vec{x}) \vec{x}_{ii} + \sum_{i(i+1) \in E} 2\xi_{i(i+1)}(\vec{x}) \vec{x}_{i(i+1)} = \vec{x}, \quad (3.47)$$

and for the quadratic constraint,

$$\sum_{ii \in V} \xi_{ii}(\vec{x}) (\vec{x}_i \otimes \vec{x}_i) + \sum_{i(i+1) \in E} \xi_{i(i+1)}(\vec{x}) (\vec{x}_i \otimes \vec{x}_{i+1} + \vec{x}_{i+1} \otimes \vec{x}_i) = \vec{x} \otimes \vec{x}. \quad (3.48)$$

To remove the diagonal functions, we formalize this procedure by recasting it as a linear algebra problem. We seek a matrix \mathbb{A} such that the linear transformation,

$$\{\xi\} = \mathbb{A} \{\mu\}, \quad (3.49)$$

will satisfy the precision properties of Eqs. (3.46 - 3.48). It is easy to see that \mathbb{A} has dimension $(2N_K \times N_Q)$. We wish for this matrix to have constant entries for any point within the polygon's interior so that it does not have to be recalculated for each interpolatory point of interest. To ease the notation, we will assign specific basis orderings for the quadratic and quadratic serendipity functions. The serendipity basis is ordered such that all vertex functions ($ii \in V$) in a counter-clockwise ordering are first, followed by a counter-clockwise ordering of the mid-face nodes ($ij \in E$) starting with the node between vertices 1 and 2. This ordering can be succinctly stated by

$$\{\xi_{ij}\} = \left\{ [\xi_{11}, \xi_{22}, \dots, \xi_{N_K N_K}], [\xi_{12}, \xi_{23}, \dots, \xi_{(N_K-1)N_K}, \xi_{N_K(N_K+1)}] \right\}, \quad (3.50)$$

where $N_K + 1 = 1$. The quadratic basis begins identically to the serendipity basis by first listing the vertex and mid-face functions. Then the diagonal functions are

indexed in lexicographical order. This gives the following ordering for the quadratic functions

$$\{\mu_{ab}\} = \left\{ \begin{aligned} & [\mu_{11}, \mu_{22}, \dots, \mu_{N_K N_K}], [\mu_{12}, \mu_{23}, \dots, \mu_{(N_K-1)N_K}, \mu_{N_K(N_K+1)}], \\ & [\mu_{13}, \dots, (\text{lexicographical}), \dots, \mu_{(N_K-2)N_K}] \end{aligned} \right\}. \quad (3.51)$$

With these basis orderings, we can now denote the entries of \mathbb{A} (given by c_{ab}^{ij}) as

$$\mathbb{A} = \begin{bmatrix} c_{11}^{11} & \dots & c_{ab}^{11} & \dots & c_{(n-2)n}^{11} \\ \dots & \ddots & \vdots & \ddots & \vdots \\ c_{11}^{ij} & \dots & c_{ab}^{ij} & \dots & c_{(n-2)n}^{ij} \\ \dots & \ddots & \vdots & \ddots & \vdots \\ c_{11}^{n(n+1)} & \dots & c_{ab}^{n(n+1)} & \dots & c_{(n-2)n}^{n(n+1)} \end{bmatrix}, \quad (3.52)$$

where $n = N_K$ is used for clarity.

A sufficient set of constraints for the entries in \mathbb{A} to ensure the precision properties of Eqs. (3.46 - 3.48) can be written for the constant constraint,

$$\begin{aligned} \sum_{ii \in V} c_{aa}^{ii} + \sum_{i(i+1) \in E} 2c_{aa}^{i(i+1)} &= 1, \quad \forall aa \in V \\ \sum_{ii \in V} c_{ab}^{ii} + \sum_{i(i+1) \in E} 2c_{ab}^{i(i+1)} &= 2, \quad \forall ab \in E \cup D \end{aligned} \quad (3.53)$$

for the linear constraint,

$$\begin{aligned} \sum_{ii \in V} c_{aa}^{ii} \vec{x}_{ii} + \sum_{i(i+1) \in E} 2c_{aa}^{i(i+1)} \vec{x}_{i(i+1)} &= \vec{x}_{aa}, \quad \forall aa \in V \\ \sum_{ii \in V} c_{ab}^{ii} \vec{x}_{ii} + \sum_{i(i+1) \in E} 2c_{ab}^{i(i+1)} \vec{x}_{i(i+1)} &= 2\vec{x}_{ab}, \quad \forall ab \in E \cup D \end{aligned} \quad (3.54)$$

for the $a \in V$ vertex quadratic constraints

$$\sum_{ii \in V} c_{aa}^{ii} \vec{x}_{ii} \otimes \vec{x}_{ii} + \sum_{i(i+1) \in E} c_{aa}^{i(i+1)} (\vec{x}_i \otimes \vec{x}_{i+1} + \vec{x}_{i+1} \otimes \vec{x}_i) = \vec{x}_{aa} \otimes \vec{x}_{aa}, \quad (3.55)$$

and for the $ab \in E \cup D$ mid-face and diagonal quadratic constraints,

$$\sum_{ii \in V} c_{ab}^{ii} \vec{x}_{ii} \otimes \vec{x}_{ii} + \sum_{i(i+1) \in E} c_{ab}^{i(i+1)} (\vec{x}_i \otimes \vec{x}_{i+1} + \vec{x}_{i+1} \otimes \vec{x}_i) = (\vec{x}_a \otimes \vec{x}_b + \vec{x}_b \otimes \vec{x}_a). \quad (3.56)$$

For $N_K > 3$, there are more coefficients than the six constraint equations. This means that there is flexibility in the construction of the solution to the constraint equations. Therefore, we choose a simple structure for \mathbb{A} that consists of the following,

$$\mathbb{A} = [\mathbb{I} | \mathbb{A}'], \quad (3.57)$$

where \mathbb{I} is the $(2N_K \times 2N_K)$ identity matrix, and \mathbb{A}' is a full $(2N_K \times \frac{N_K(N_K-3)}{2})$ matrix. This means that the vertex and face midpoint serendipity functions, ξ_{ij} , are formed by taking their corresponding quadratic function, μ_{ij} , and adding some linear combination of the interior functions. Therefore, we only need to determine the $\frac{N_K(N_K-3)}{2}$ columns of the \mathbb{A}' matrix to complete this linear transformation.

In their work, Rand et al. proposed a methodology where only six coefficients are chosen to be non-zero and can be directly calculated through geometric expressions. However, their approach is only valid for strictly-convex polygons. This will not work for our analysis since we wish to also analyze degenerate polygons that will arise in our AMR calculations. Therefore, we will use a least squares method to calculate each of the columns of \mathbb{A}' . We note that the coefficients calculated with this least

squares method and that of Rand will be identical only for rectangles. If we isolate the column (ab) from \mathbb{A}' , then we can form the following system of equations,

$$\mathbb{B}\vec{c}_{ab} = \vec{q}_{ab}, \quad (3.58)$$

where \mathbb{B} is a matrix of dimension $(6 \times 2N_K)$, \vec{c}_{ab} is vector of length $2N_K$, and \vec{q}_{ab} is a vector of length 6. The entries of \mathbb{B} correspond to coefficients given in the left-hand-side terms of Eqs. (3.53), (3.54), and (3.56). The values of \vec{q}_{ab} are given by the right-hand-side constants of these same equations. To invert \mathbb{B} , we use the Moore-Penrose pseudoinverse (denoted as \mathbb{B}^*) [125]. For an under-determined system of equations, the pseudoinverse is given by,

$$\mathbb{B}^* = \mathbb{B}^T(\mathbb{B}\mathbb{B}^T)^{-1}. \quad (3.59)$$

We note that we have only tested this methodology on convex and weakly-convex polygons. Once all of the coefficients for the \mathbb{A}' matrix are known, each of the quadratic serendipity functions can be computed by

$$\begin{aligned} \xi_{ij} &= \mu_{i,j} + \sum_{ab \in D} c_{ab}^{ij} \mu_{ab} \\ &= \lambda_i \lambda_j + \sum_{ab \in D} c_{ab}^{ij} \lambda_a \lambda_b \end{aligned}, \quad (3.60)$$

where $i = j$ for vertex functions.

The gradients of the serendipity basis are simple to compute with the chain rule of Calculus. If we take the gradient of Eq. (3.60) and use appropriate derivative rules, then the gradients of the different serendipity functions can be given by

$$\vec{\nabla} \xi_{ij} = \lambda_j \vec{\nabla} \lambda_i + \lambda_i \vec{\nabla} \lambda_j + \sum_{ab \in D} c_{ab}^{ij} \left(\lambda_b \vec{\nabla} \lambda_a + \lambda_a \vec{\nabla} \lambda_b \right). \quad (3.61)$$

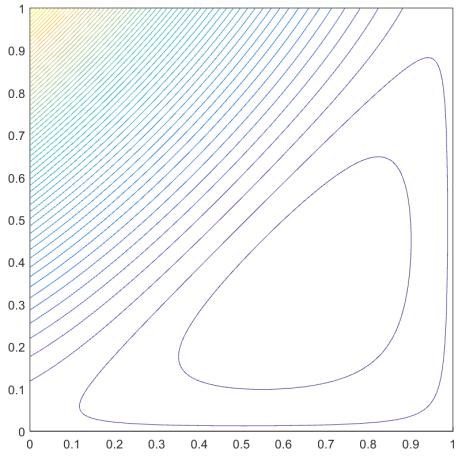
This means that all of the serendipity basis function gradients can be computed from the appropriate values and gradients of the linear barycentric basis functions using the linear transformation of the \mathbb{A} matrix.

We now present some example contour plots for the conversion of the linear barycentric coordinates into the quadratic serendipity space. Figures 3.17 and 3.18 provide the contour plots of the different quadratic serendipity functions located at the upper-left vertex and left side-node, respectively. Then, Figure 3.19 provides some of the contour plots of the PWL, MV, and ME basis functions on the degenerate square that is formed by inserting a vertex at $(1/2, 1)$ onto the unit square.

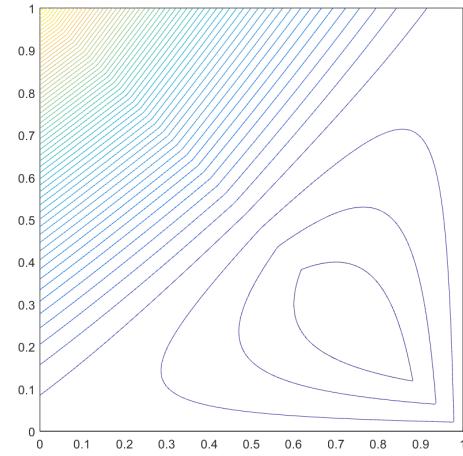
3.3 Integrating the Arbitrary 2D Polygonal Elements

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 performed 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 [126, 127, 128, 129, 130]. These include symmetric rules on triangles and cubature and tensor-product rules on 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 [131, 132, 133].

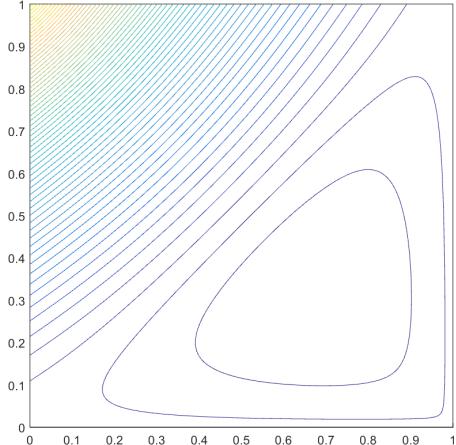
At this time, we are only interested in the accuracy and not the efficiency of the



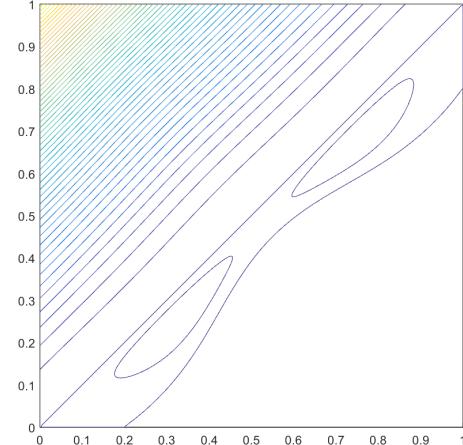
(a) Wachspress



(b) PWL

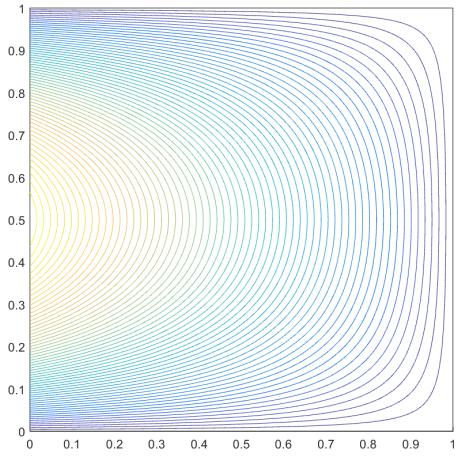


(c) Mean Value

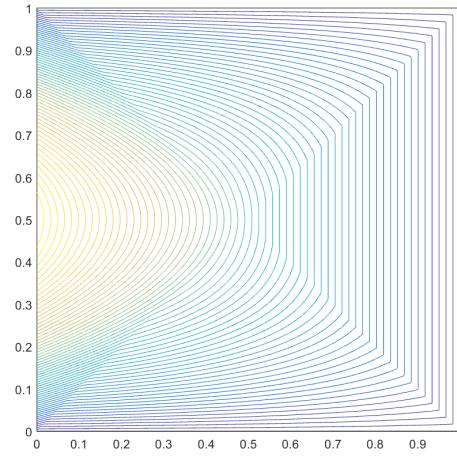


(d) Maximum Entropy

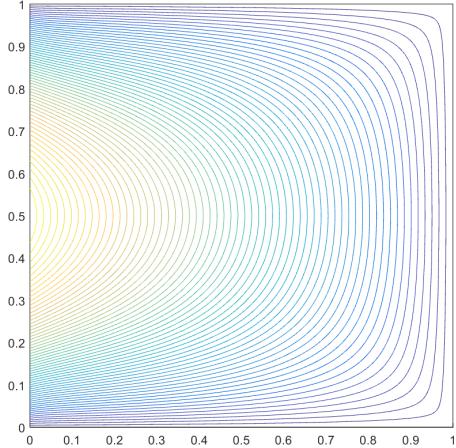
Figure 3.17: Contour plots of the different quadratic serendipity basis function on the unit square located at vertex $(0,1)$.



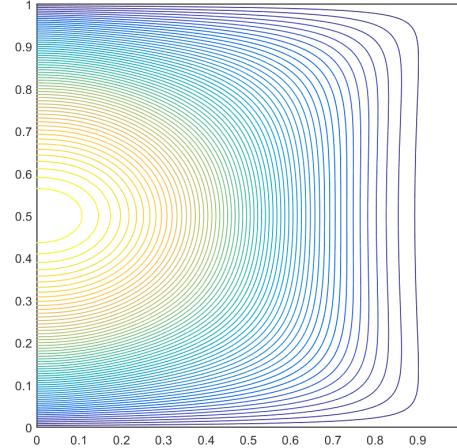
(a) Wachspress



(b) PWL



(c) Mean Value



(d) Maximum Entropy

Figure 3.18: Contour plots of the different quadratic serendipity basis function on the unit square at a mid-face node located at $(0, 1/2)$.

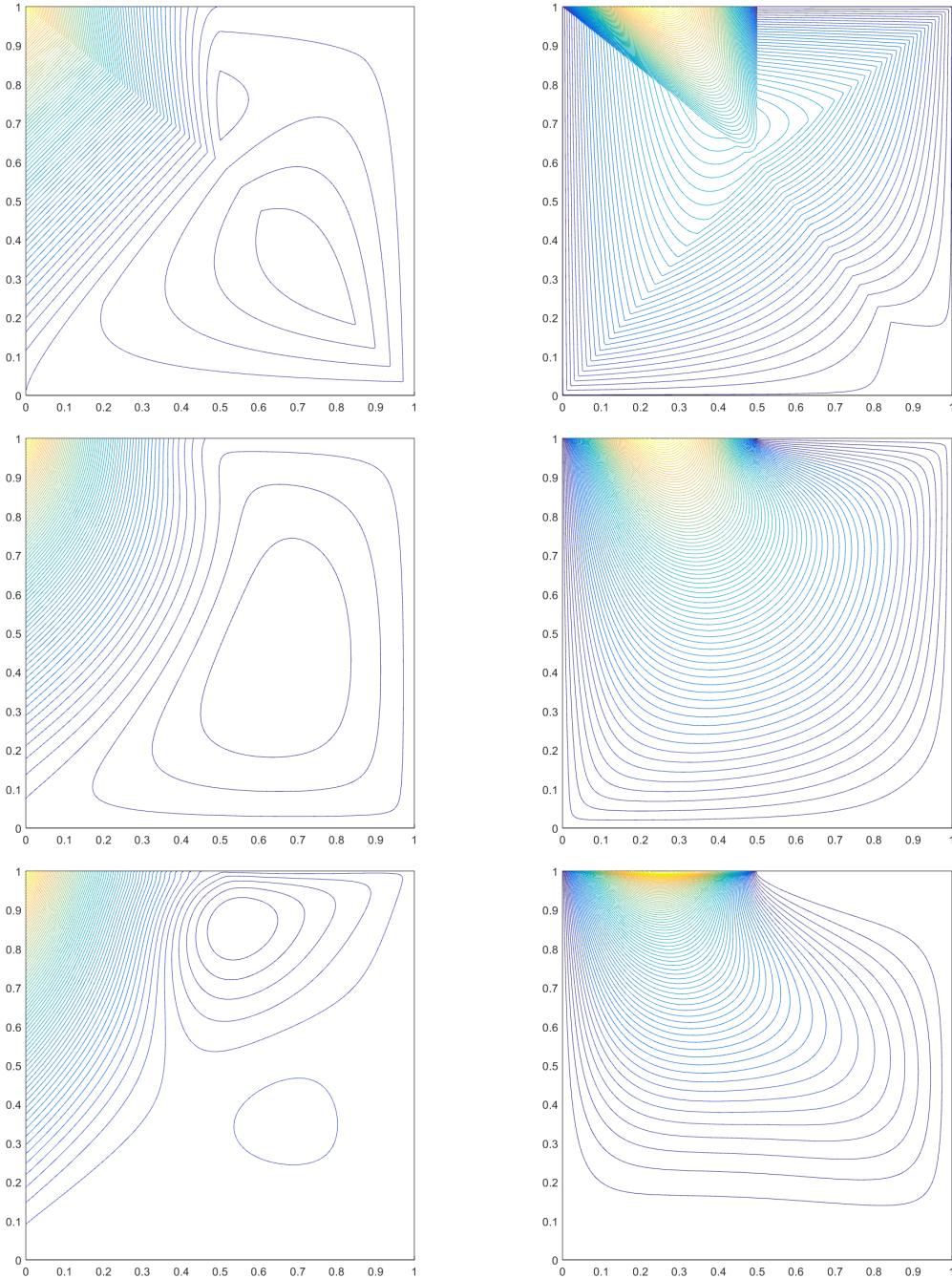


Figure 3.19: Contour plots of the different quadratic serendipity basis functions on the degenerate square. The PWL (top), mean value (middle), and maximum entropy (bottom) functions are plotted at vertices $(0, 1)$ (left) and $(1/2, 1)$ (right)

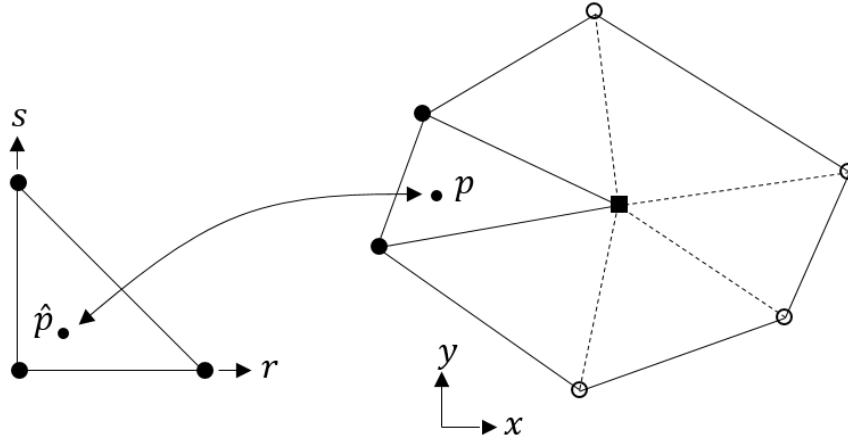


Figure 3.20: Mapping a point on the reference triangle onto a sub-triangle of an arbitrary polygon.

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,

$$\vec{c} = \frac{1}{N_K} \sum_{i=1}^{N_K} \vec{x}_i. \quad (3.62)$$

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 [127]. We denote this reference quadrature rule by $\left\{ \hat{x}_q, \hat{w}_q \right\}_{q=1}^{N_q}$, where the symbol $\hat{\cdot}$ 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,

$$\mathbf{p} = \mathbf{x}_0 + J\hat{\mathbf{p}}, \quad (3.63)$$

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 3.20. 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,

$$J = \begin{bmatrix} x_1 - x_0 & x_2 - x_0 \\ y_1 - y_0 & y_2 - y_0 \end{bmatrix}. \quad (3.64)$$

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,

$$\nabla_{\hat{x}} = J^T \nabla_x, \quad (3.65)$$

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

$$\nabla_x = J^{-T} \nabla_{\hat{x}}. \quad (3.66)$$

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

$$w_{i,q}^K = \hat{w}_q |J_i|. \quad (3.67)$$

In Eq. (3.67), $|J_i|$ is the determinant of the Jacobian matrix corresponding to the transformation of 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,

$$\int_K f = \sum_{i=1}^{N_K} \sum_{q=1}^{N_q} w_{i,q}^K f(\vec{x}_{i,q}), \quad (3.68)$$

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

$$\int_{\mathbb{T}_h} f = \sum_{K \in \mathbb{T}_h} \left(\sum_{i=1}^{N_K} \sum_{q=1}^{N_q} w_{i,q}^K f(\vec{x}_{i,q}) \right), \quad (3.69)$$

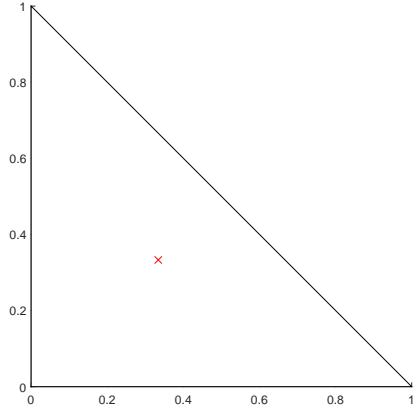
which is clearly just an element-wise sum of Eq. (3.68).

We conclude this section by providing some visual examples of quadrature sets for polygonal elements. Figure 3.21 gives quadrature sets on the reference triangle for orders 1-6 [127]. We can see that our reference quadrature is symmetric about any of the three vertices, though we note that true isoparametric symmetry is only obtained with equilateral triangles. Then Figures 3.22 and 3.23 provide examples of the mapping of the reference quadrature into the global space for a regular pentagon and hexagon, respectively.

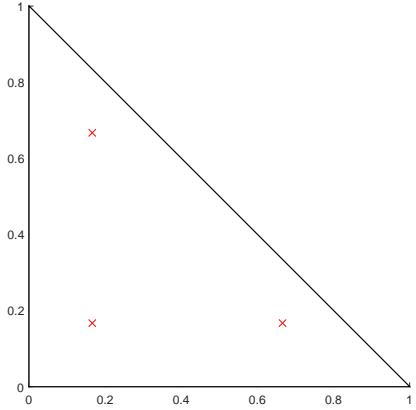
3.4 Linear Basis Functions on 3D Polyhedra

We have defined linearly-complete and quadratically-complete 2D polygonal basis functions for use in FEM analysis of the DGFEM transport equation. Now, we present an efficient coordinate system for arbitrary 3D polyhedra that is linearly-complete. At the time of this work and to the best of our knowledge, no analogous methodology to convert linear coordinates on 3D polyhedra to their serendipity basis exists. However, the 3D serendipity space of functions is well defined for hexahedra [90]. Therefore, we will utilize only a single linearly-complete 3D coordinate system for some of the analysis to be performed in Chapter 4, but we include it here for completeness with the 2D coordinates.

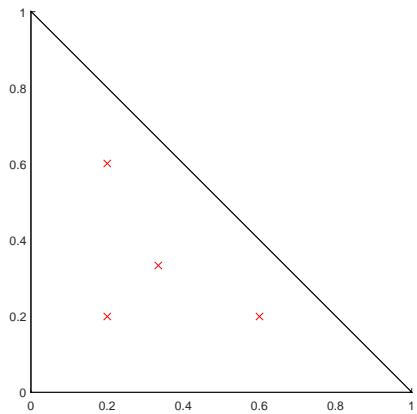
For this work, we will utilize the 3D version of the Piecewise Linear (PWL) coordinates that is suitable for x-y-z geometries [39]. From Table 3.1, we can see some of the properties of the different 2D polygonal coordinates. The PWL functions are the only coordinates with a 3D analogue that allow for direct, analytical integration of the elementary matrices. This means that no spatial quadrature sets are required, though an analogous procedure to Section 3.3 could be employed using sub-tetrahedra instead of sub-triangles. In Appendix B, we detail how these 3D PWL coordinates can be analytically integrated using the reference tetrahedron and



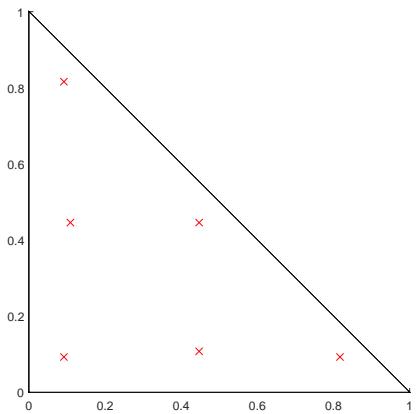
(a) Order 1



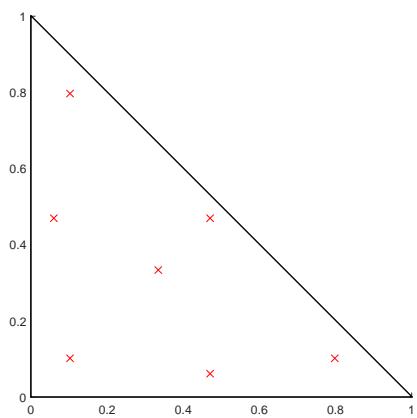
(b) Order 2



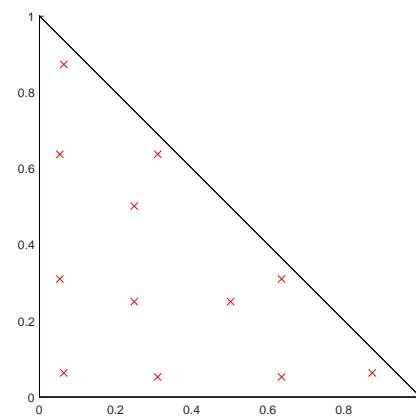
(c) Order 3



(d) Order 4



(e) Order 5



(f) Order 6

Figure 3.21: Quadrature sets on the reference triangle of varying order.

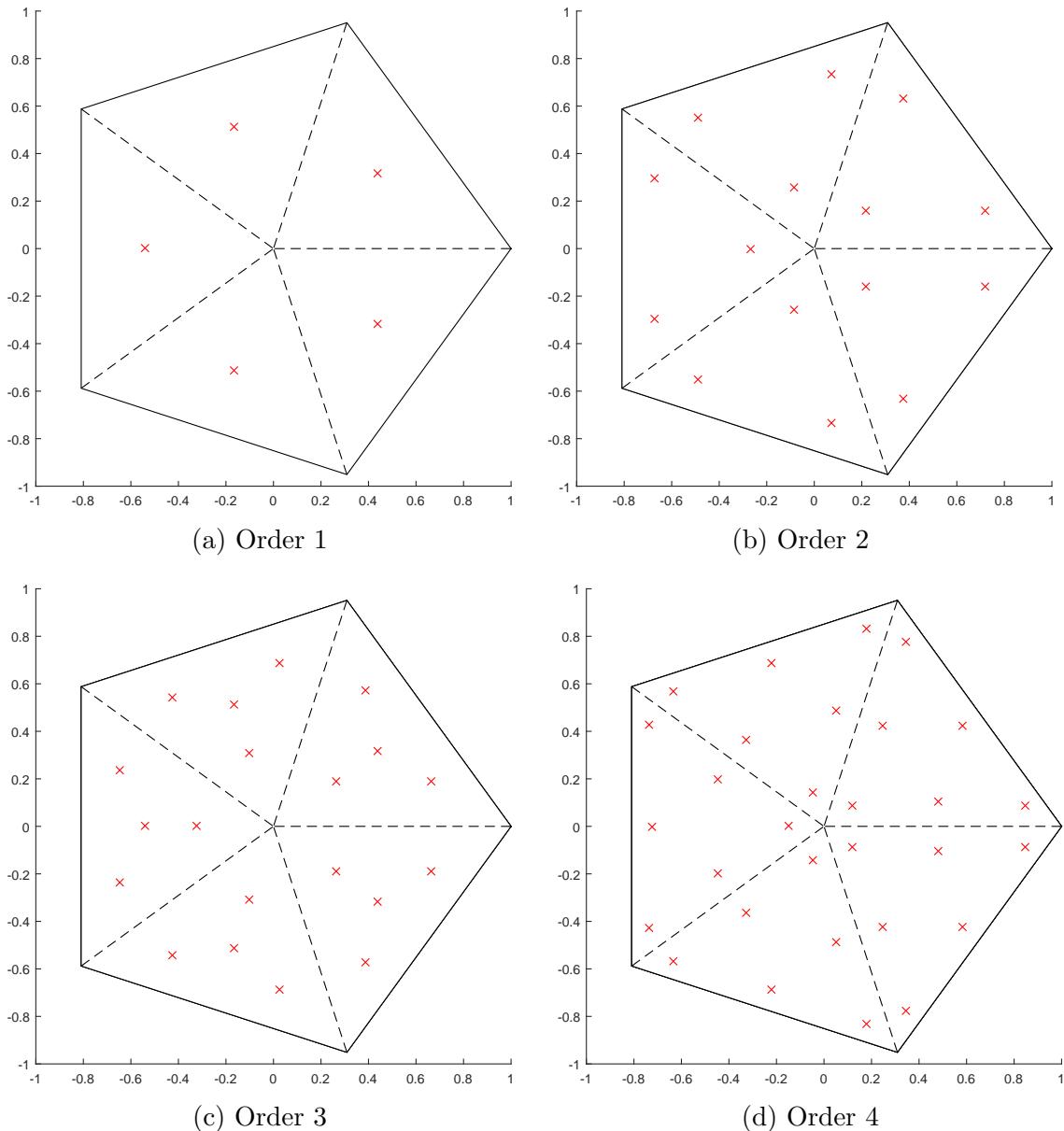


Figure 3.22: Examples of spatial quadrature sets of varying order on a regular pentagon.

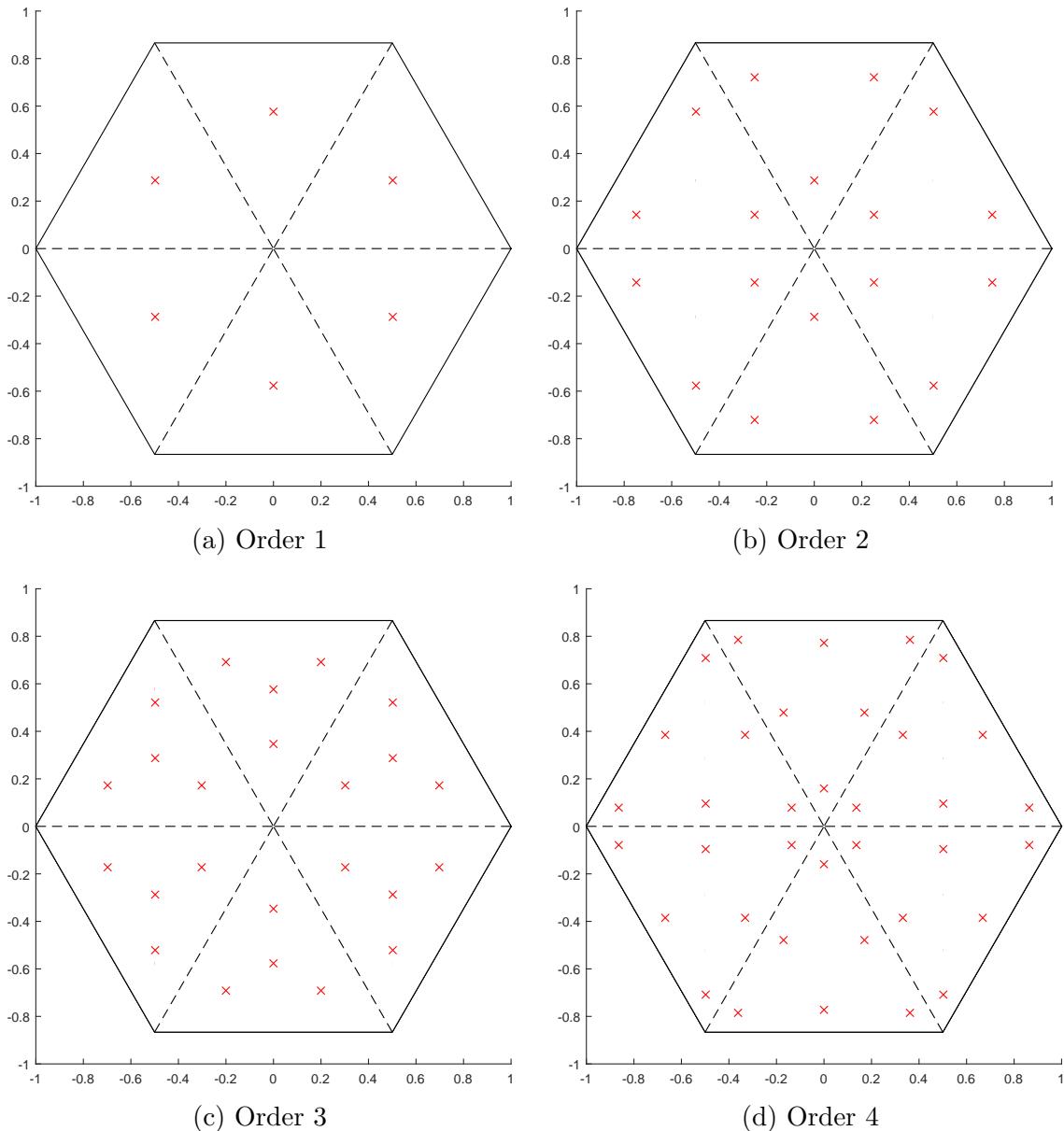


Figure 3.23: Examples of spatial quadrature sets of varying order on a regular hexagon.

affine mapping. The 3D PWL coordinates also allow for extremely-distorted concave polyhedra, though we will not analyze those in this work.

The 3D PWL coordinates have an analogous form to their 2D version from Eq. (3.20). In fact, the 2D PWL coordinates are identical to their 3D version along a polyhedral face (the 3D face centroid acts like the 2D cell centroid). If we define N_K as the number of vertices for cell K and N_f as the number of vertices composing face f , then the cell and face centroids for a strongly-convex polyhedra can be given by

$$\vec{r}_c = \frac{1}{N_K} \sum_{i=1}^{N_K} \vec{x}_i, \quad (3.70)$$

and

$$\vec{r}_f = \frac{1}{N_f} \sum_{i=1}^{N_f} \vec{x}_i, \quad (3.71)$$

respectively. We see that these centroids have the simple definition of the average positions of the cell and face vertices. Using these centroid definitions, the functional form for the 3D PWL coordinates is given by:

$$b_j(x, y, z) = t_j(x, y, z) + \sum_{f=1}^{F_j} \beta_{f,j}^j t_f(x, y, z) + \alpha_j^K t_c(x, y, z). \quad (3.72)$$

In Eq. (3.72), t_j is the standard 3D linear function with unity at vertex j that linearly decreases to zero at 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 located at \vec{r}_c which is unity at the cell center and linearly decreases to zero at each cell vertex and face center. t_f is the face “tent” function for face f located at \vec{r}_f 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 [39], we also choose to assume the cell and face weighting parameters are

$$\alpha_{K,j} = \frac{1}{N_K}, \quad (3.73)$$

and

$$\beta_{f,j} = \frac{1}{N_f}, \quad (3.74)$$

respectively, which leads to constant values of α and β for each cell and face, respectively. This assumption of the cell weight functions remains from the 2D PWL form.

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 linear basis functions can capture an exactly-linear transport solution in Section 3.5.1. We follow that by demonstrating that the quadratic serendipity basis functions can capture an exactly-quadratic transport solution in Section 3.5.2. Next, we present some convergence properties of the basis sets using the method of manufactured solutions (MMS) in Section 3.5.3. Then in Section 3.5.4, we demonstrate how the solution regularity can limit the convergence of our numerical transport solutions to the $H^{1/2}$ and $H^{3/2}$ Hilbert spaces. Then, we demonstrate that all of the 2D linear and quadratic serendipity basis functions can capture the correct transport solution in the thick diffusion limit in Section 3.5.5. We conclude with a

searchlight problem and observe how the basis sets react with adaptive mesh refinement (AMR) to mitigate numerical dispersion through a vacuum in Section 3.5.6.

3.5.1 Two-Dimensional Exactly-Linear Transport Solutions

Our first numerical verification example demonstrates that the linear 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,

$$\mu \frac{\partial \Psi}{\partial x} + \eta \frac{\partial \Psi}{\partial y} + \sigma_t \Psi = Q(x, y, \mu, \eta), \quad (3.75)$$

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:

$$\begin{aligned} \Psi(x, y, \mu, \eta) &= ax + by + c\mu + d\eta + e \\ \Phi(x, y) &= 2\pi(ax + by + e) \end{aligned} . \quad (3.76)$$

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:

$$\sum_q w_q = 2\pi \quad \text{and} \quad \sum_q w_q \begin{bmatrix} \mu_q \\ \eta_q \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \quad (3.77)$$

The sum of the quadrature weights is handled by simply re-normalizing 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. (3.76). Inserting the angular flux solution of Eq. (3.76) into Eq. (3.75), we obtain the distributed source that will produce our exactly-linear solution space:

$$Q(x, y, \mu, \eta) = a\mu + b\eta + \sigma_t(c\mu + d\eta) + \sigma_t(ax + by + e). \quad (3.78)$$

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

$$\begin{aligned} a &= -c\sigma_t, \\ b &= -d\sigma_t. \end{aligned} \quad (3.79)$$

For this work, we chose our parameters so that Eq. (3.79) is not satisfied. Therefore, our source will be angle dependent.

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 [134, 135]. 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 [136]. 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. (3.76) 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 S_8 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, and linear maximum entropy are presented in Figures 3.24, 3.25, 3.26, 3.27, respectively. We can see that for all the polygonal basis functions, an exactly-linear solution is captured as shown by the unbroken nature of the contour lines. This even holds on the highly distorted quadrilateral Shestakov mesh.

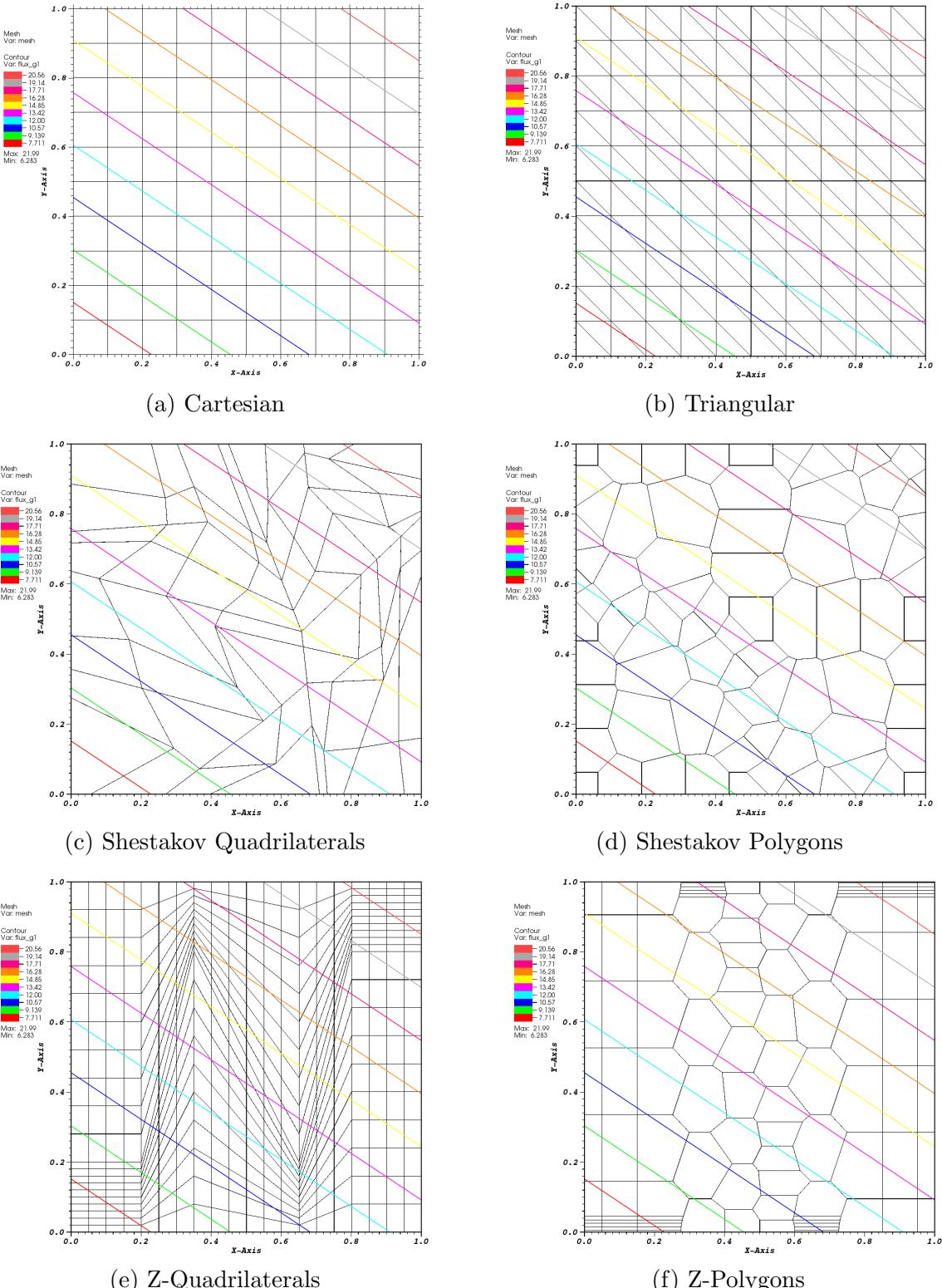


Figure 3.24: Contour plots of the exactly-linear solution with the Wachspress basis functions.

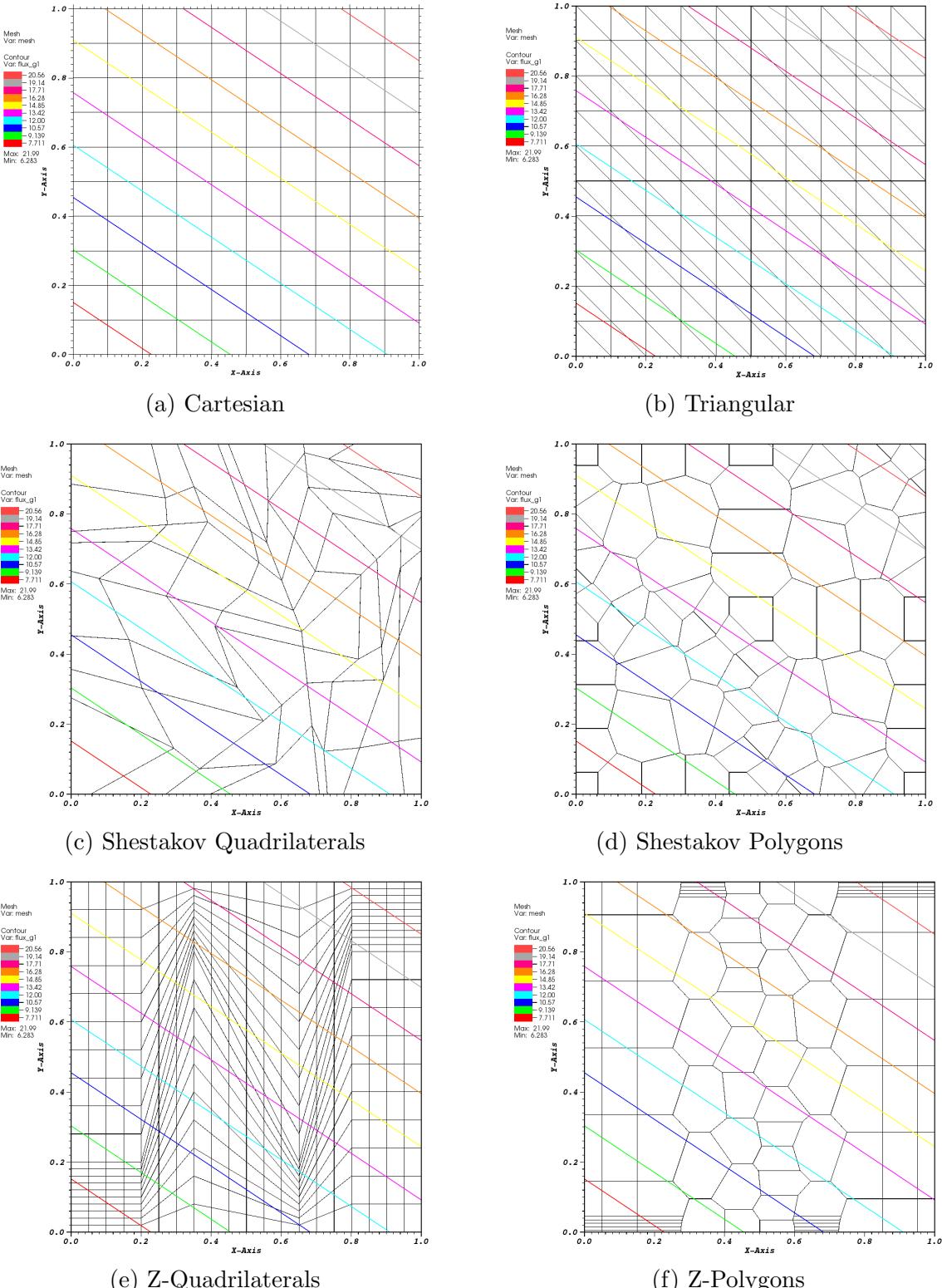


Figure 3.25: Contour plots of the exactly-linear solution with the PWL basis functions.

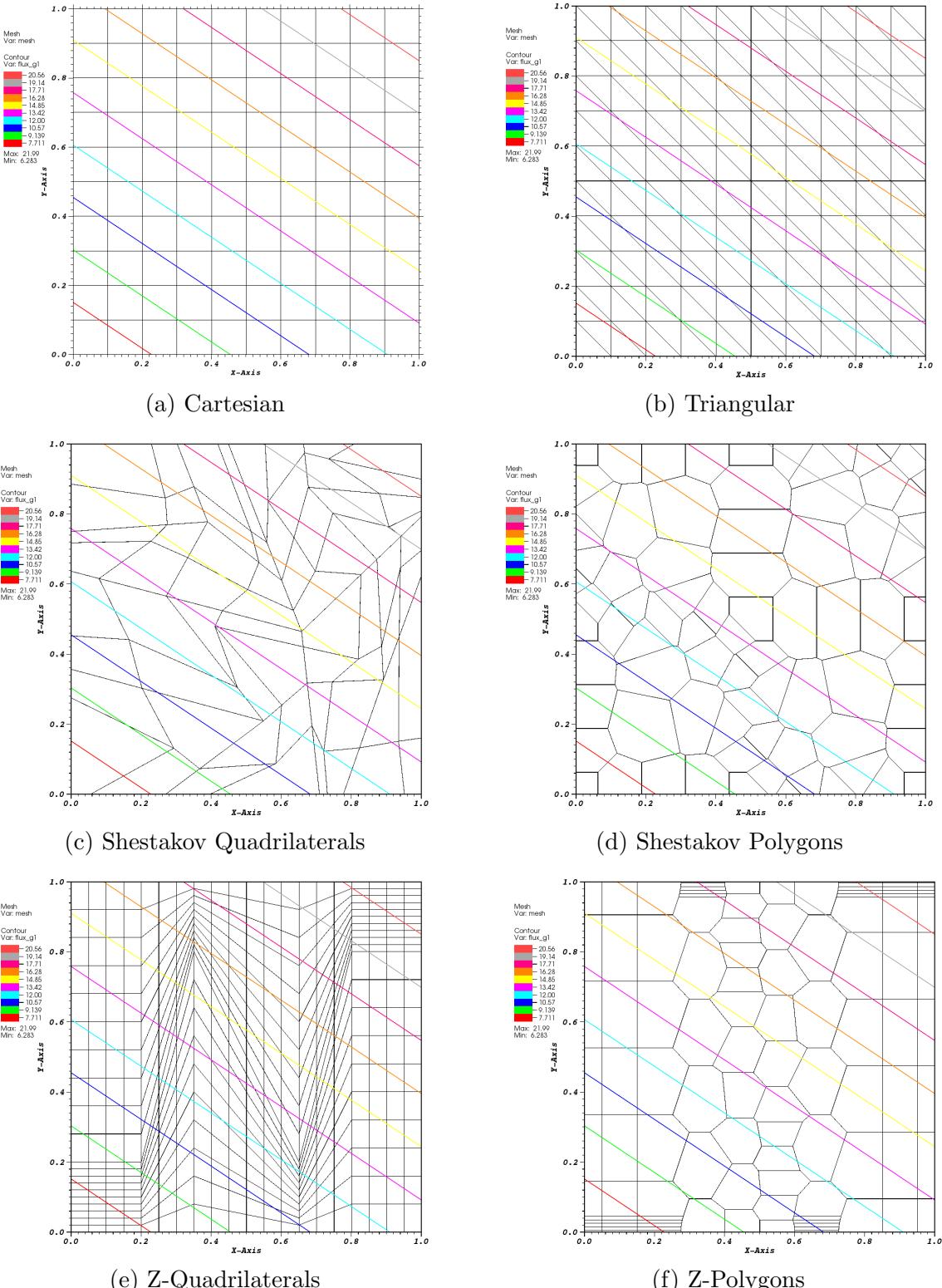


Figure 3.26: Contour plots of the exactly-linear solution with the mean value basis functions.

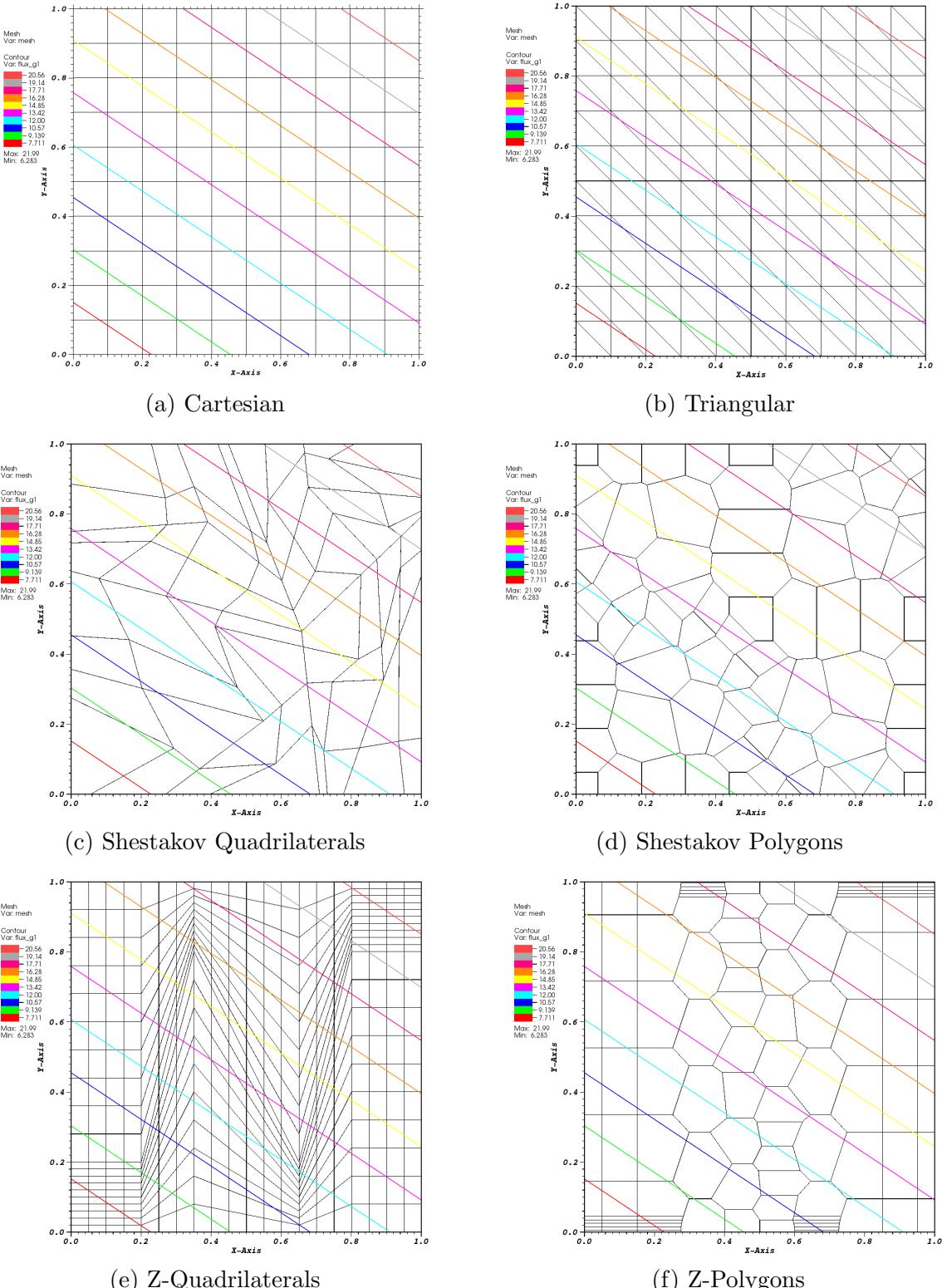


Figure 3.27: Contour plots of the exactly-linear solution with the linear maximum entropy basis functions.

3.5.2 Two-Dimensional Exactly-Quadratic Transport Solutions

In Section 3.2, we stated that the quadratic serendipity basis can only interpolate the $\{1, x, y, x^2, xy, y^2\}$ span of functions. Unlike the \mathbb{Q}_9 elements, the quadratic serendipity basis cannot interpolate the $\{x^2y, xy^2, x^2y^2\}$ span of functions. In this example we show the limit of the interpolatory properties of the quadratic serendipity space by again using MES. We will test two exact solution spaces. The first spans the functions that are captured by the quadratic serendipity space, and the second spans the additional functions captured by the \mathbb{Q}_9 elements (but is not captured by the quadratic serendipity space).

The first problem interpolates the $\{1, x, y, x^2, xy, y^2\}$ span of functions, which we denote with the following general and exactly-quadratic solution, $\{\Psi_q, \Phi_q\}$:

$$\begin{aligned}\Psi_q(x, y, \mu, \eta) &= a + bx + cy + dxy + ex^2 + fy^2 \\ \Phi_q(x, y) &= 2\pi(a + bx + cy + dxy + ex^2 + fy^2)\end{aligned}. \quad (3.80)$$

Inserting the angular flux solution of Eq. (3.80) into the previously-defined Eq. (3.75) yields the following right-hand-side distributed source,

$$\begin{aligned}Q_q(x, y, \mu, \eta) &= (b + dy + 2ex)\mu + (c + dx + 2fy)\eta \\ &\quad + \sigma_t(a + bx + cy + dxy + ex^2 + fy^2)\end{aligned}. \quad (3.81)$$

We clearly see that any combination of positive or negative (non-zero) values for the coefficients $a - f$ will span the quadratic serendipity space. The second problem contains terms up to x^2y^2 . This higher-order functional form has a solution, $\{\Psi_{x2y2}, \Phi_{x2y2}\}$, given by the following

$$\begin{aligned}\Psi_{x2y2}(x, y, \mu, \eta) &= x(L_x - x)y(L_y - y), \\ \Phi_{x2y2}(x, y) &= 2\pi x(L_x - x)y(L_y - y)\end{aligned}\quad (3.82)$$

where L_x and L_y are the dimensions of the domain in x and y . Again, inserting this solution into Eq. (3.80) yields the following right-hand-side distributed source,

$$\begin{aligned}Q_{x2y2}(x, y, \mu, \eta) &= \left[y(L_x - x)(L_y - y) - xy(L_y - y) \right] \mu \\ &\quad + \left[x(L_x - x)(L_y - y) - xy(L_x - x) \right] \eta . \\ &\quad + \sigma_t x(L_x - x)y(L_y - y)\end{aligned}\quad (3.83)$$

For both problems, L_x and L_y are set to 1.0 to form the unit square. For the exactly-quadratic solutions, $\{\Psi_q, \Phi_q\}$, we set the function parameters in Eqs. (3.80) and (3.81) all to be 1.0: $\sigma_t = a = b = c = d = e = f = 1.0$. The meshes used for both problems are the same used for the exactly-linear problem. We give an example of the numerical solutions for the exactly-quadratic solution in Figure 3.28 using the quadratic serendipity maximum entropy coordinates for all the different meshes. Next, we give the spatial distributions of the error for the exactly-quadratic problem in Figures 3.29, 3.30, 3.31, and 3.32 for the Wachspress, PWL, MV, and ME basis functions, respectively. The magnitudes of these spatial distributions are around $10^{-14} - 10^{-15}$ showing that the quadratic serendipity basis functions can capture exactly-quadratic solutions to machine precision. We next present results for the quadratic solution that includes the x^2y^2 term. From the theory for the quadratic serendipity functional space, we should not be able to capture solutions of higher order than x^2 and y^2 . Figure 3.33 gives an example of this solution using the quadratic serendipity maximum entropy coordinates for all the different meshes. Table 3.2 then gives the L_2 -norm of the MMS error for the different basis functions

Table 3.2: L_2 -norm of the error in the quadratic solution containing the x^2y^2 term for the different quadratic serendipity basis functions on different mesh types.

Mesh Type	Basis Functions			
	Wachspress	PWL	Mean Value	Max. Entropy
Cartesian	3.5e-06	2.72e-05	8.29e-06	3.86e-05
Triangular	5.13e-05	5.13e-05	5.13e-05	5.13e-05
Shes. Quad	3.97e-04	3.37e-04	2.81e-04	3.91e-04
Sine Poly	3.75e-05	1.22e-04	7.62e-05	1.39e-04
Z-Poly	2.93e-05	3.07e-05	2.6e-05	3.46e-05
Z-Quad	2.98e-05	8.73e-05	5.08e-05	1.17e-04

and mesh types. We clearly see errors on the order of $O(10^{-6})$ to $O(10^{-4})$ which means that we do not capture the functional space as predicted.

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;

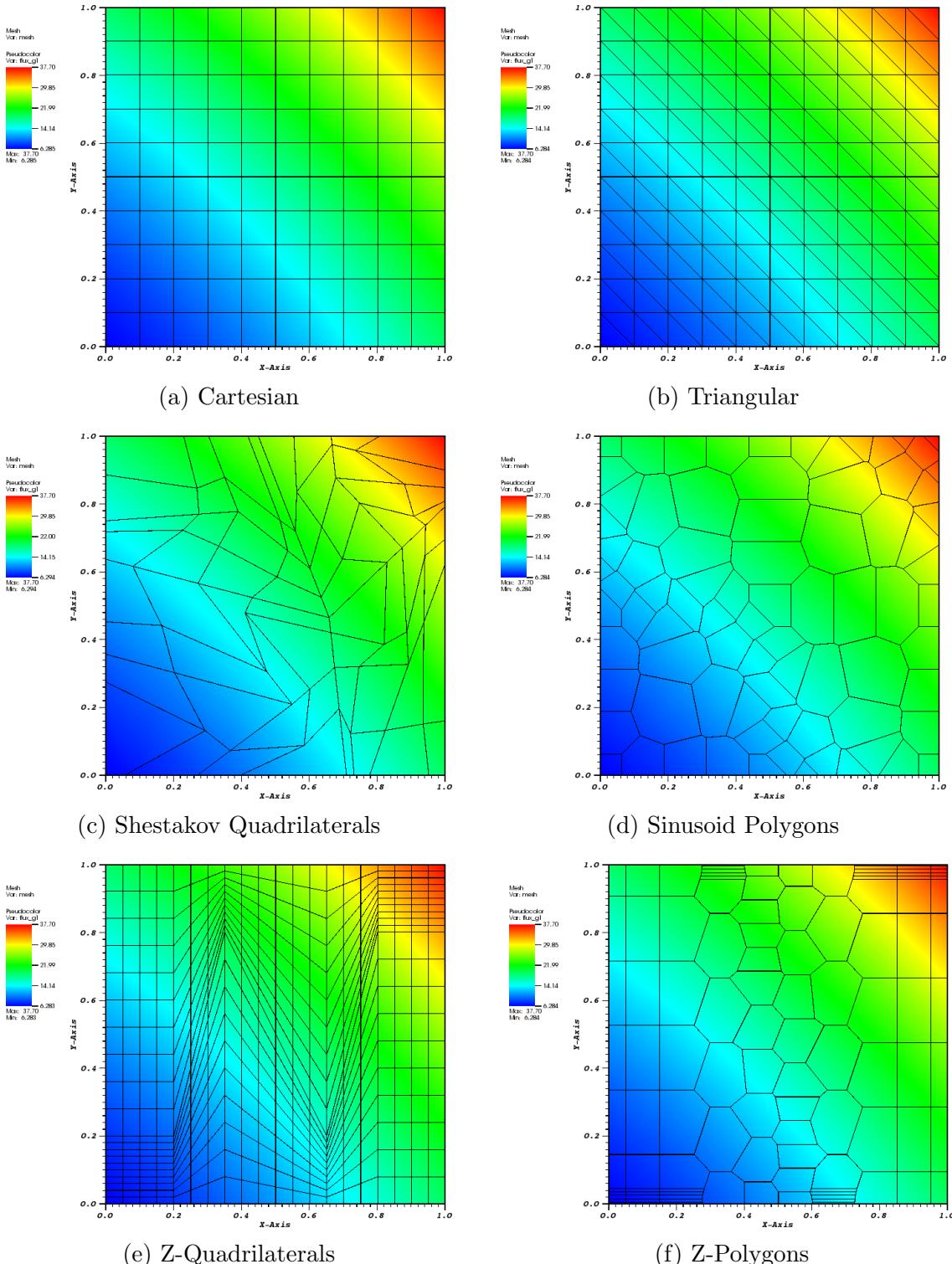


Figure 3.28: Plots of the exactly-quadratic solution with the quadratic serendipity maximum entropy basis functions.

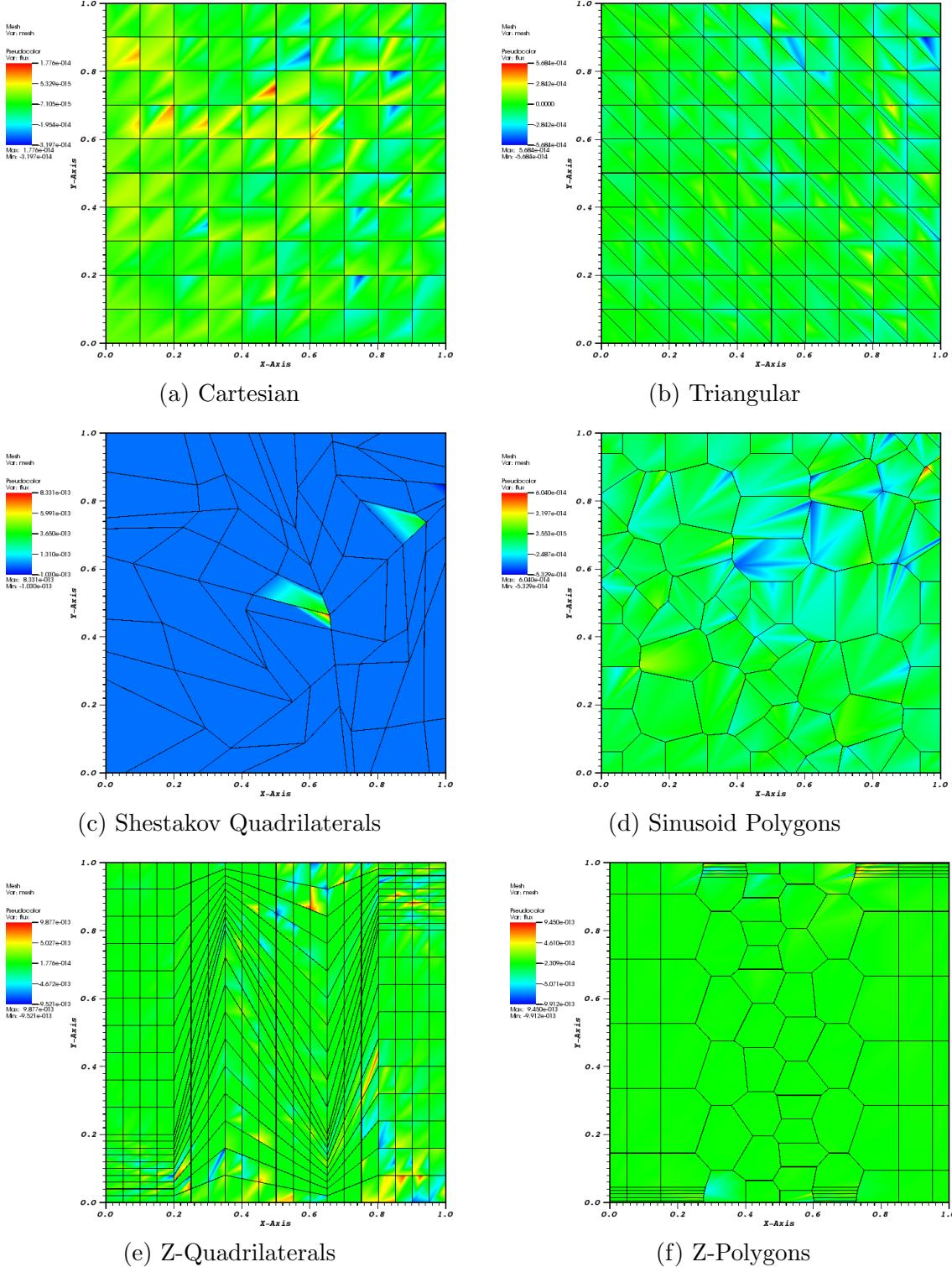


Figure 3.29: Plots of the error of the exactly-quadratic solution with the quadratic serendipity Wachspress basis functions.

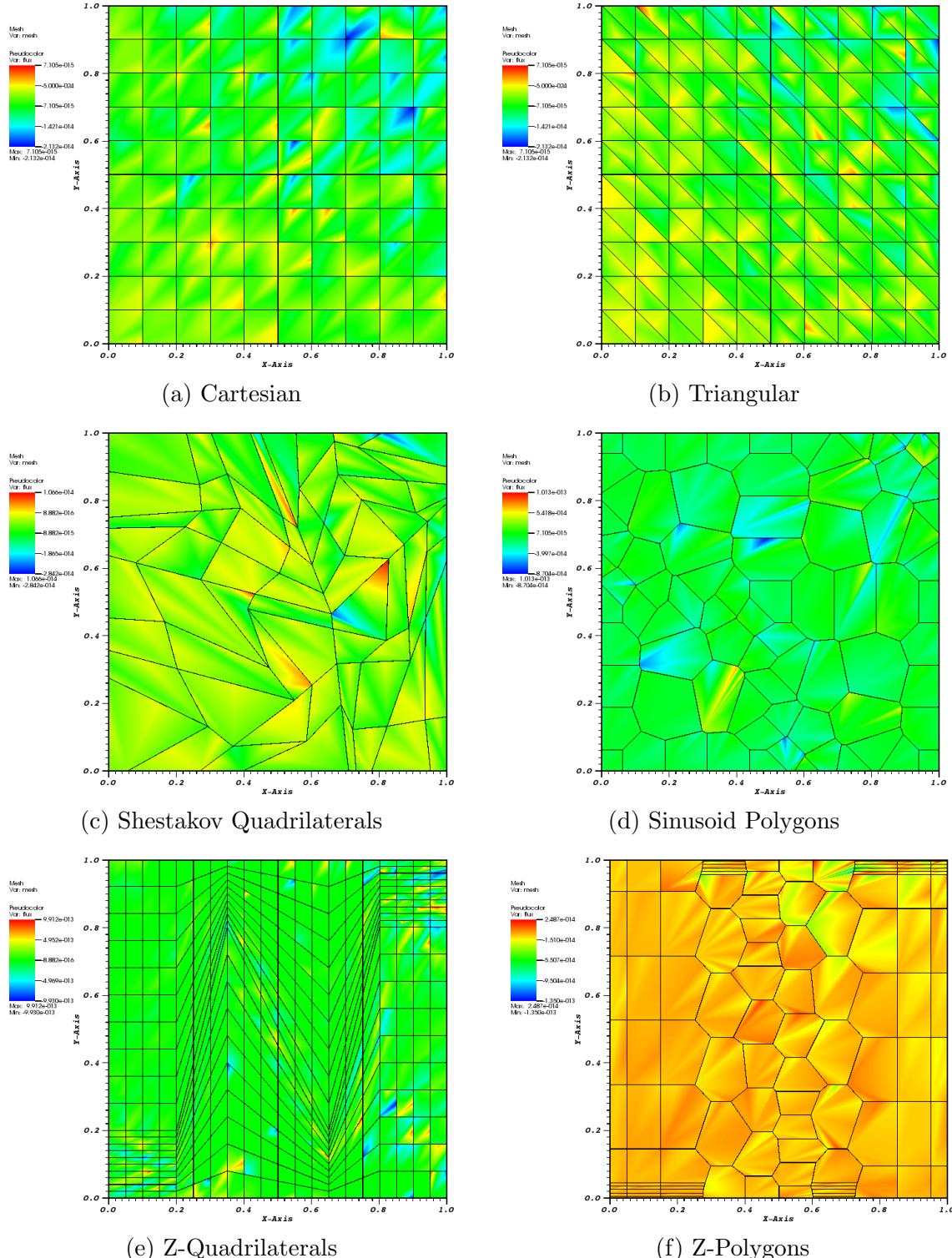
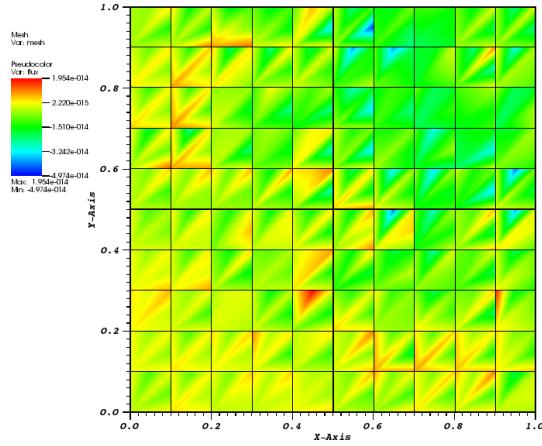
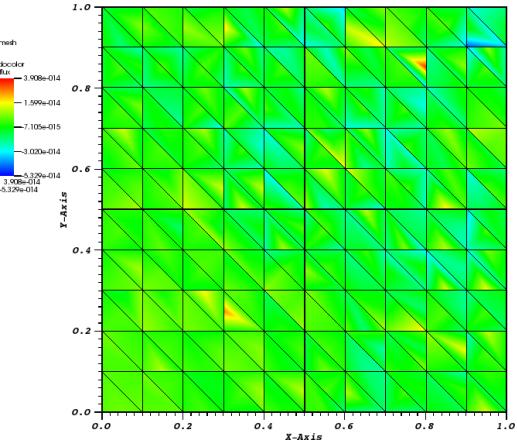


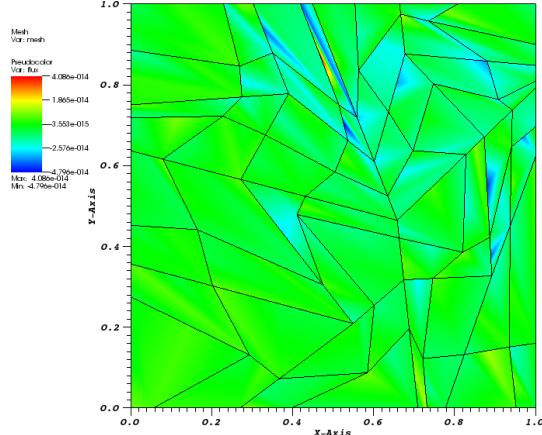
Figure 3.30: Plots of the error of the exactly-quadratic solution with the quadratic serendipity PWL basis functions.



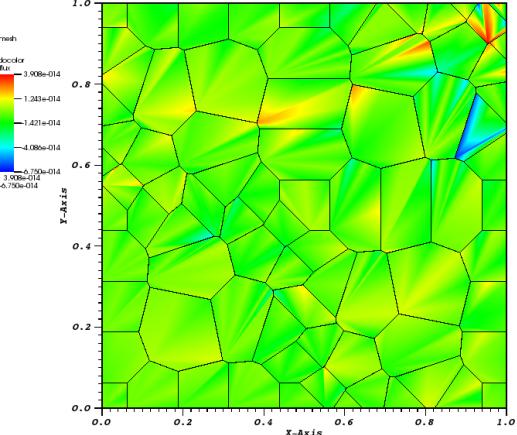
(a) Cartesian



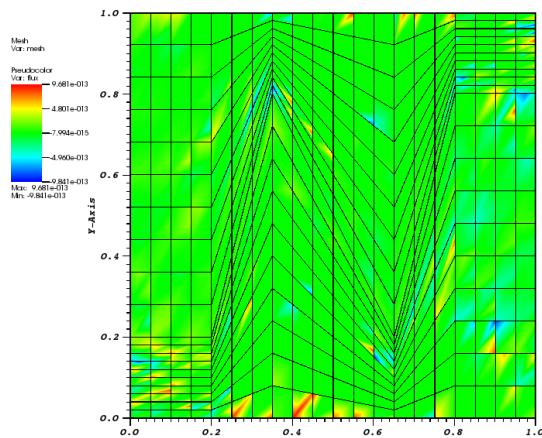
(b) Triangular



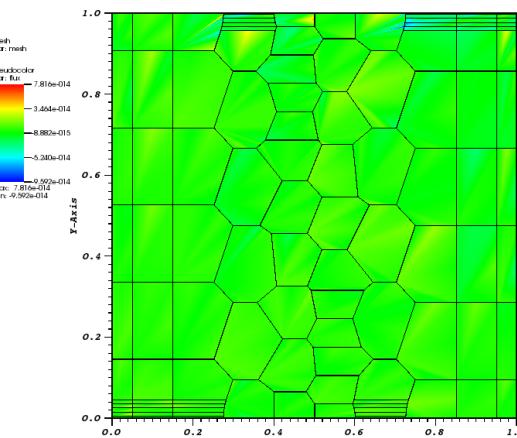
(c) Shestakov Quadrilaterals



(d) Sinusoid Polygons



(e) Z-Quadrilaterals



(f) Z-Polygons

Figure 3.31: Plots of the error of the exactly-quadratic solution with the quadratic serendipity mean value basis functions.

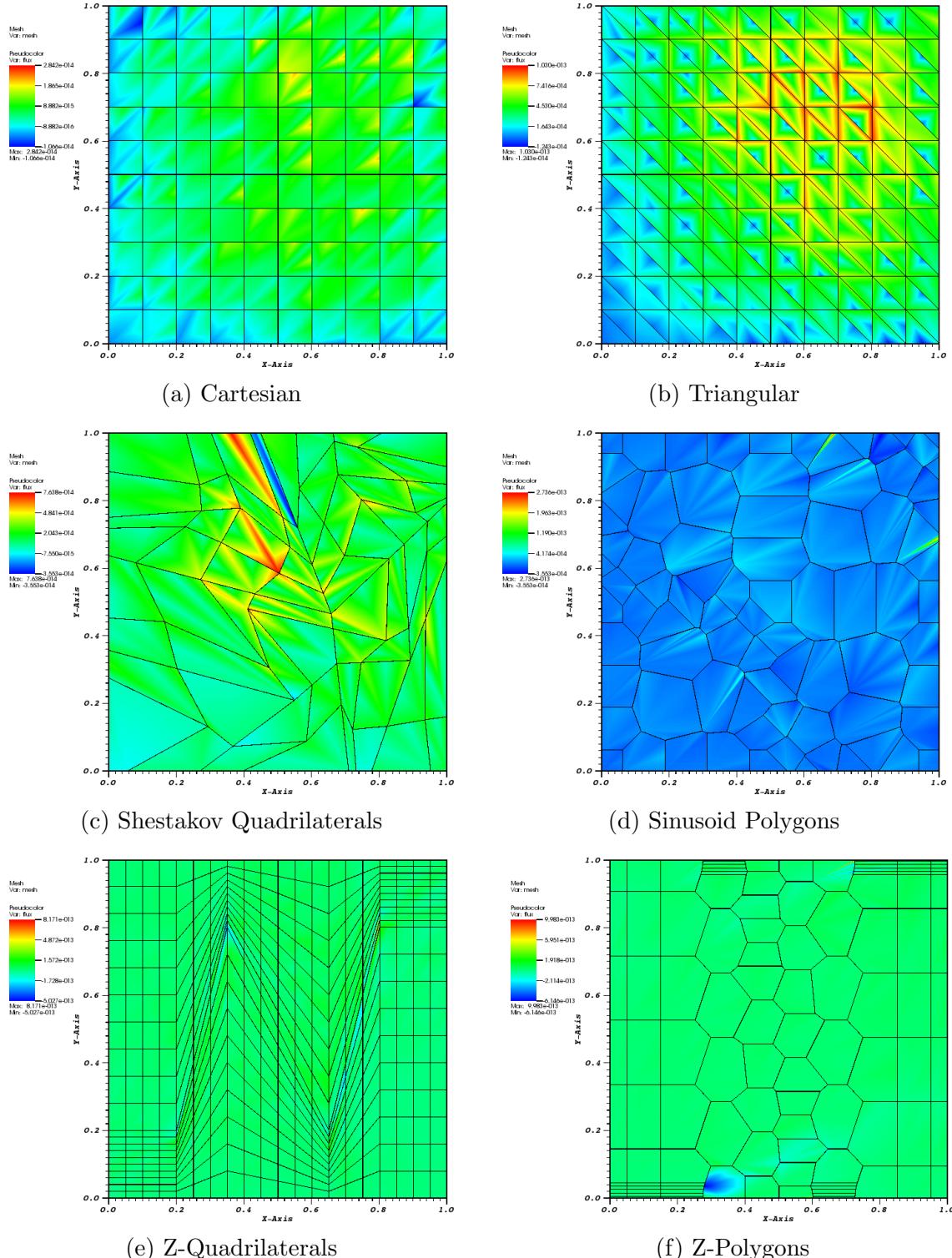


Figure 3.32: Plots of the error of the exactly-quadratic solution with the quadratic serendipity maximum entropy basis functions.

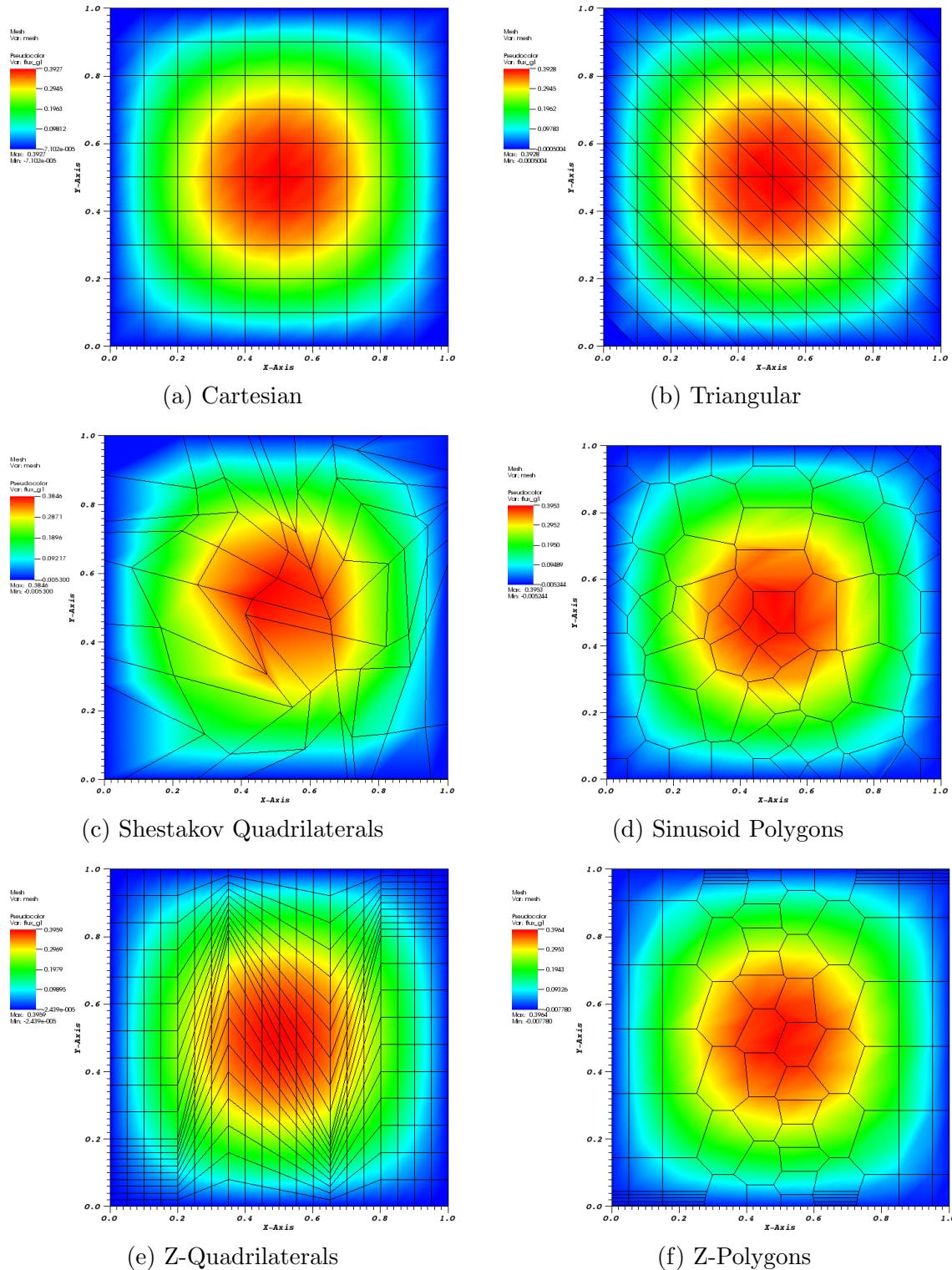


Figure 3.33: Quadratic solutions containing the x^2y^2 term for different meshes with the quadratic maximum entropy basis functions.

2. No scattering to avoid solution discontinuities from the S_N discretization;
3. No solution dependence in angle to avoid introducing angular discretization error;
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 sinusoidal solution with no extreme local maxima. The second solution is a product of the quadratic function from Eq. (3.82) and a Gaussian which yields a significant local maximum.

The sinusoid flux solutions, $\{\Psi_s, \Phi_s\}$, have the following parameterized form,

$$\begin{aligned}\Psi_s(x, y) &= \sin(\nu \frac{\pi x}{L_x}) \sin(\nu \frac{\pi y}{L_y}), \\ \Phi_s(x, y) &= 2\pi \sin(\nu \frac{\pi x}{L_x}) \sin(\nu \frac{\pi y}{L_y}),\end{aligned}\tag{3.84}$$

where ν is a frequency parameter. We restrict this parameter to positive integers ($\nu = 1, 2, 3, \dots$) 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,

$$\begin{aligned}\Psi_g(x, y) &= C_M x(L_x - x)y(L_y - y) \exp\left(-\frac{(x - x_0)^2 + (y - y_0)^2}{\gamma}\right), \\ \Phi_g(x, y) &= 2\pi C_M x(L_x - x)y(L_y - y) \exp\left(-\frac{(x - x_0)^2 + (y - y_0)^2}{\gamma}\right),\end{aligned}\tag{3.85}$$

where the constants in the equations are:

$$C_M = \frac{100}{L_x^2 L_y^2} \quad \gamma = \frac{L_x L_y}{100}. \quad (3.86)$$

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 $\frac{225}{32}\pi \approx 22.1$.

For these two examples, we will use a different meshing strategy between them. The convergence rate analysis for the sinusoid problem will be performed on three different mesh types: 1) Cartesian meshes, 2) ordered-triangular meshes, and 3) polygonal meshes. The Cartesian and triangular meshes are the same as those used for the exactly-linear and exactly-quadratic problems. The polygonal meshes are the same as those used for the diffusion limit problem. We give an example of how the mesh will be refined for these different types in Figure 3.34. For the Gaussian problem, we will utilize AMR to form our polygonal meshes. The numbers of polygons and their generation history will depend on the basis functions used and the mesh irregularity that is imposed.

The convergence rates of the sinusoid solution are given in Figures 3.37, 3.38, and 3.39 for the Cartesian, triangular, and polygonal meshes, respectively. The total cross section was set to 1.0, and S_8 level-symmetric quadrature was used. First, we provide some examples of the sinusoid solutions. Figures 3.35 and 3.36 give the contour plots for the sinusoid solutions using the linear and quadratic PWL basis functions, respectively. The meshes used are identical (in the same order) as those

given in Figure 3.34. Next, we plot the L_2 -norm of the error against the number of spatial degrees of freedom in the problem. From the theoretical analysis of Section 2.6.1, we expected and obtained slope values of the convergence rate of -1 and $-3/2$ for the linear and quadratic basis functions, respectively.

Because we are using AMR for the Gaussian problem, we will only analyze the PWL, MV, and ME basis functions since the Wachspress functions cannot handle degenerate polygons. For both the linear and quadratic AMR runsets, the refinement criterion, α , was set to 0.1. For each linear and quadratic basis function, we performed mesh refinement where the maximum irregularity was set to either 1, 2, or 3. Figures 3.40, 3.41, and 3.42 give the convergence history of the 2D Gaussian error in the L_2 -norm using the PWL, mean value, and maximum entropy basis functions, respectively. We plot the different AMR cases with the corresponding uniform refinement case ($\alpha = 0$) for comparison. We can clearly see that we obtain the proper convergence rates for all cases, and the AMR sets provide better solution accuracy with significantly less spatial degrees of freedom. Finally, we provide an example of a pair of AMR meshes and solutions in Figure 3.43 using the ME basis functions. We can see a clear difference in how the linear and quadratic basis functions will refine, with the quadratic functions causing much smoother refinement to occur.

3.5.4 Convergence Rate Analysis in a Purely-Absorbing Medium

Our next numerical examples involve studying the convergence rates of transport solutions in a purely-absorbing medium. Specifically, we seek to analyze the effects of mesh alignment along the solution discontinuities. From the theory presented in Section 2.6.1, the convergence rate of the discretized transport solution, Φ_h , to the exact solution, Φ , is given by

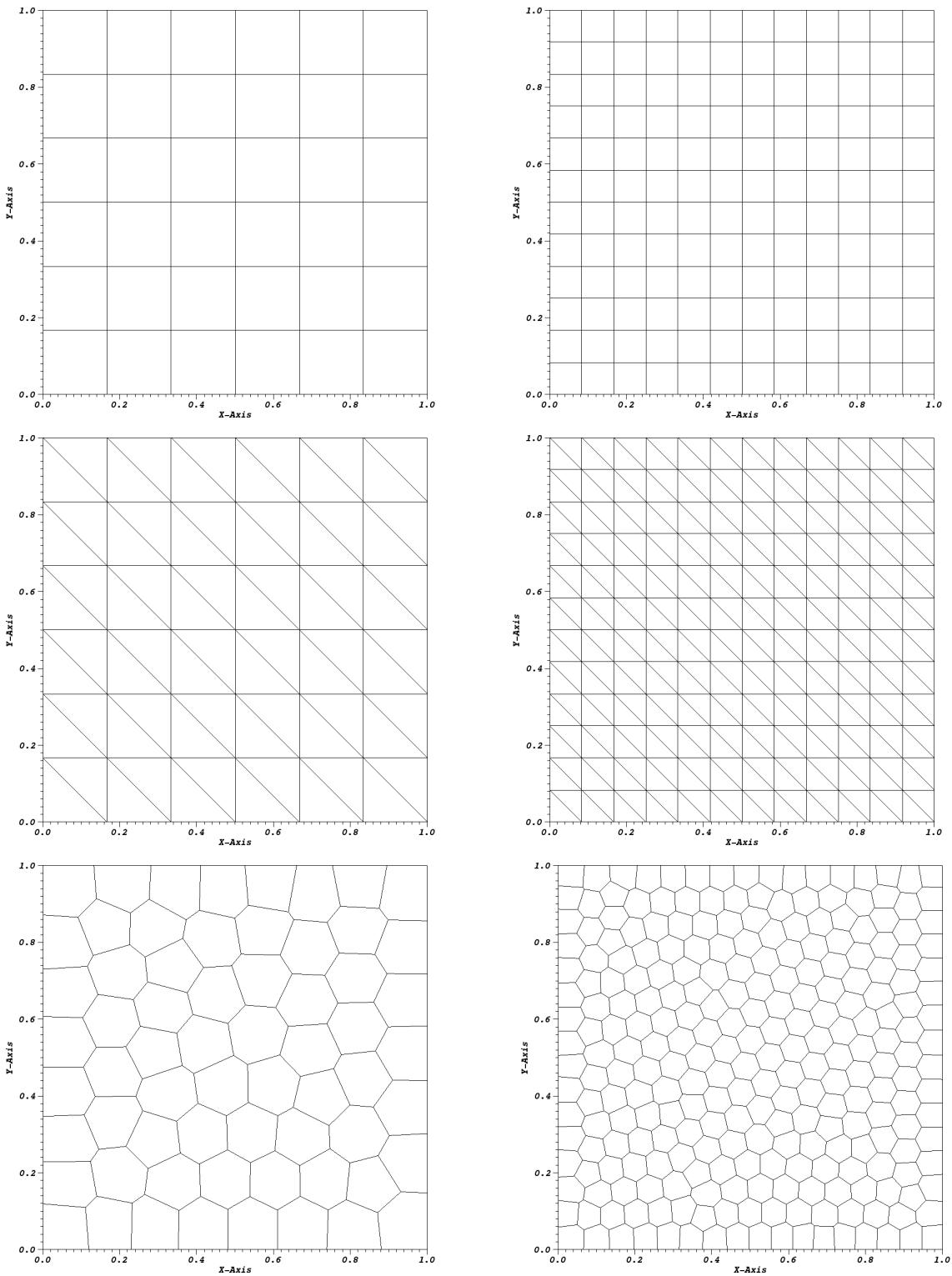


Figure 3.34: Examples of the mesh refinement for the sinusoid MMS transport problem for Cartesian (top), triangular (middle), and polygonal (bottom) meshes.

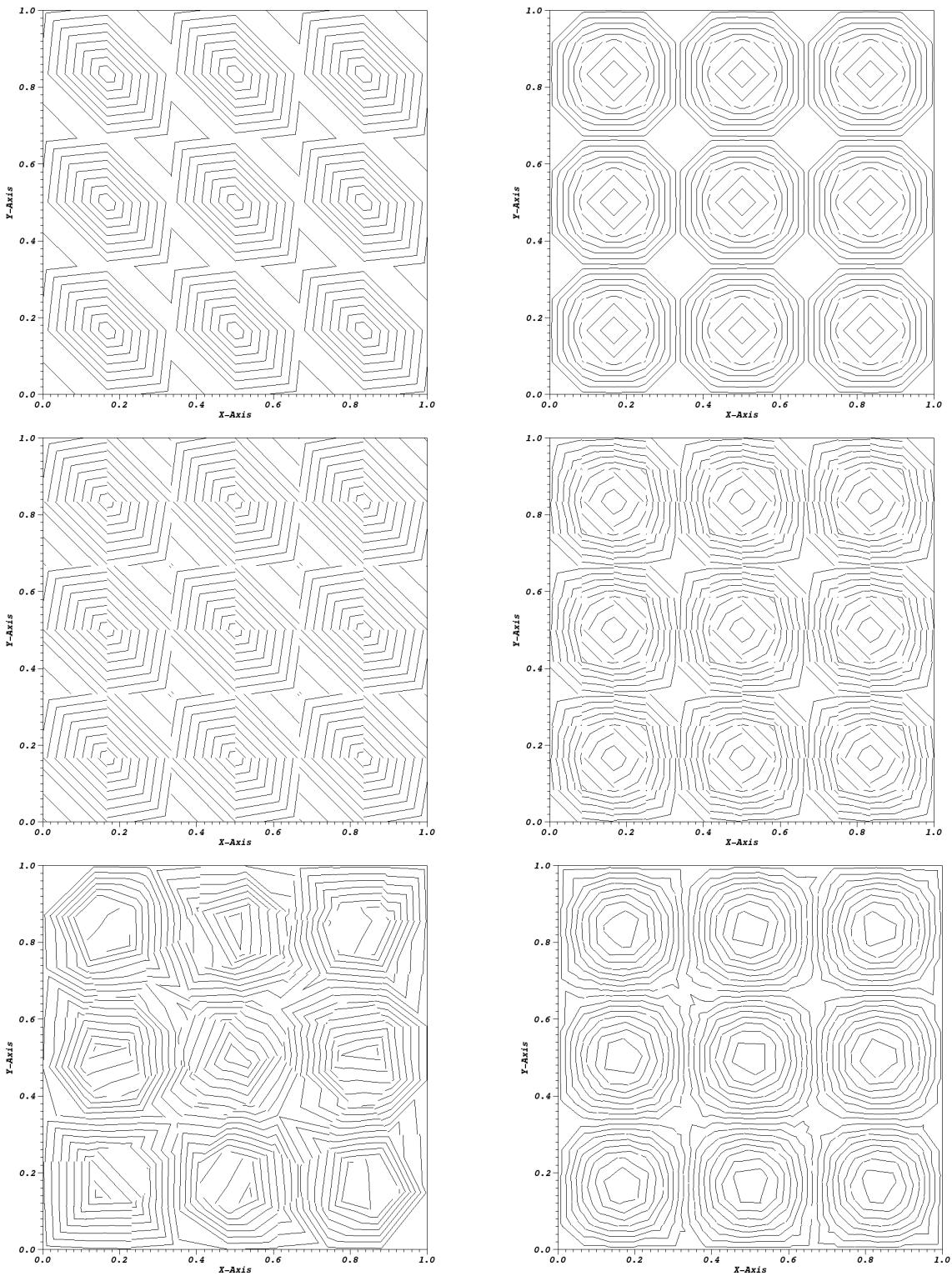


Figure 3.35: Example contour plots for the sinusoid problem using the linear PWL basis functions. The meshes used are those from Figure 3.34.

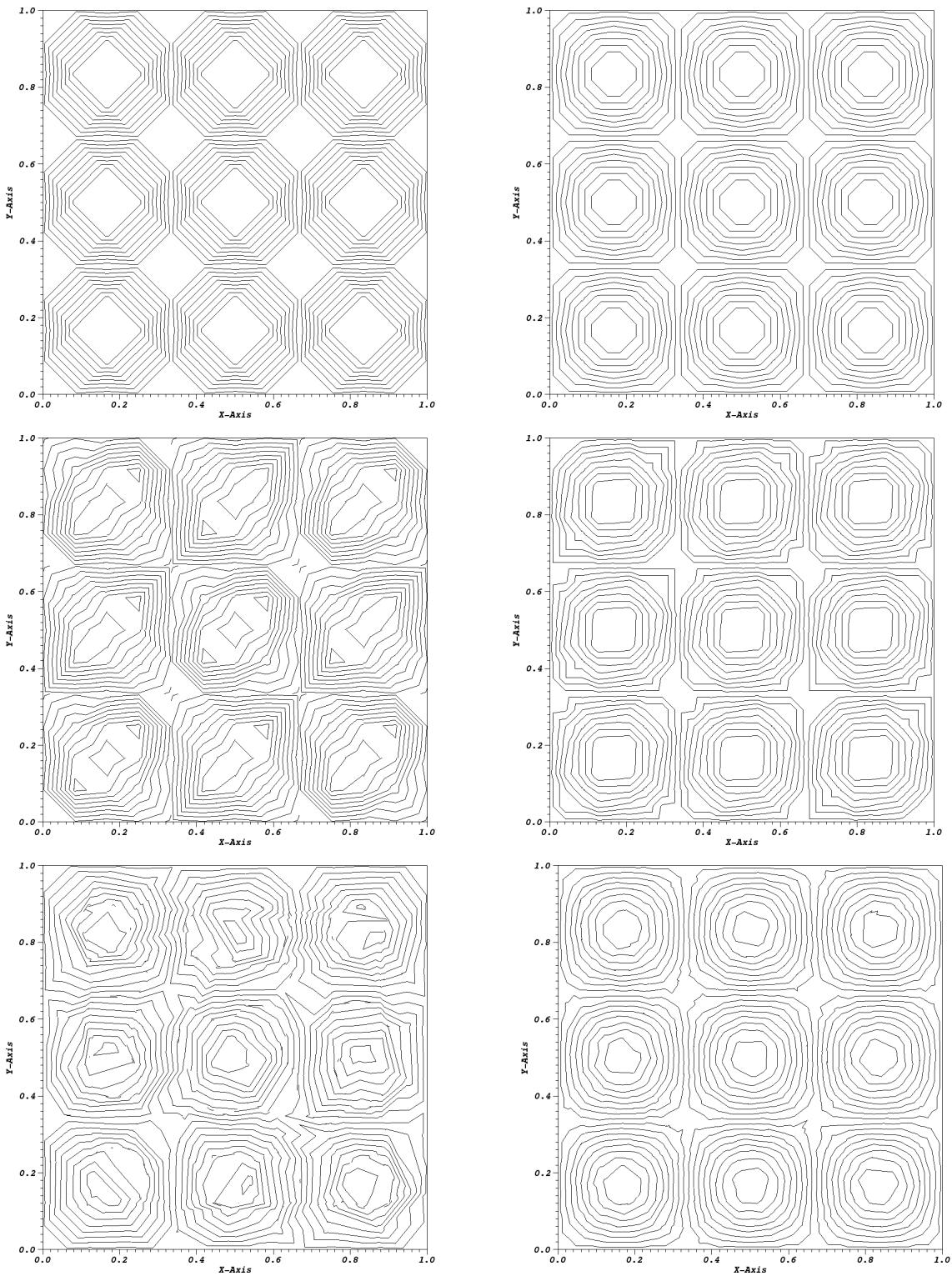


Figure 3.36: Example contour plots for the sinusoid problem using the quadratic PWL basis functions. The meshes used are those from Figure 3.34.

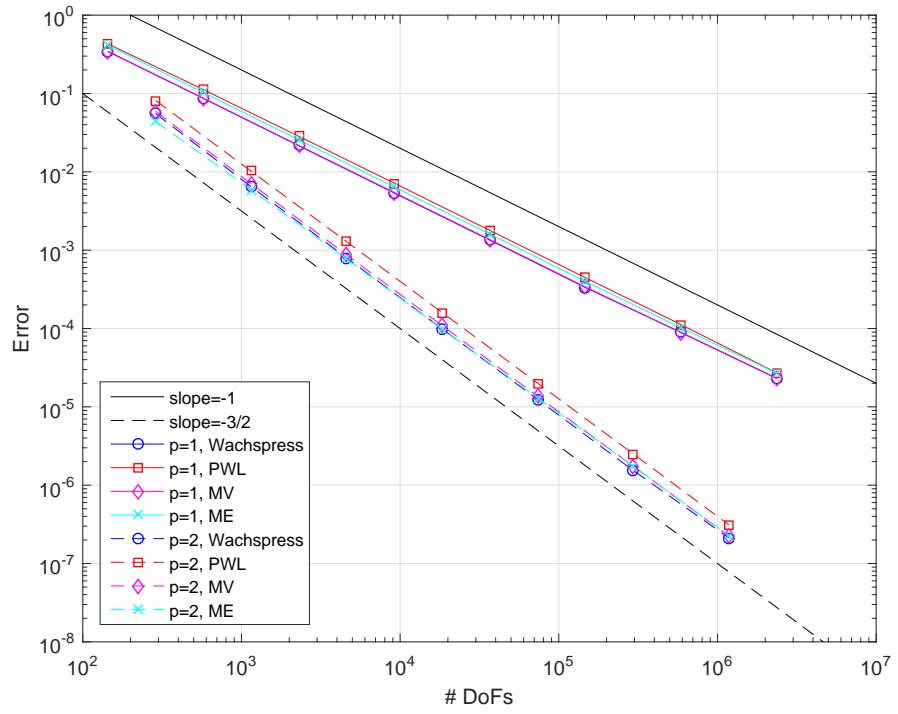


Figure 3.37: Convergence rates for the sinusoid MMS problem on a Cartesian mesh.

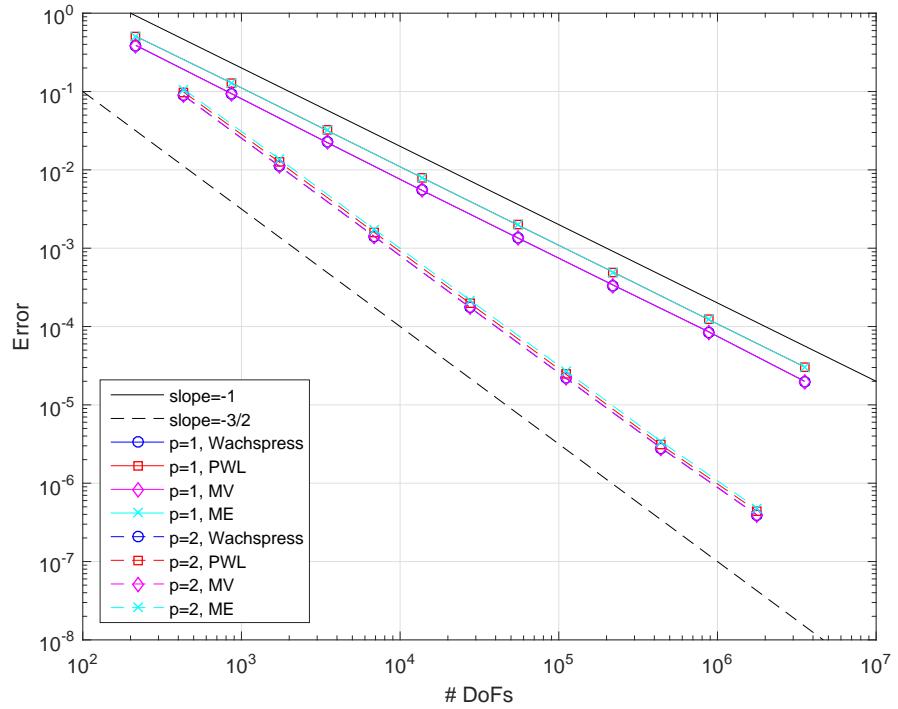


Figure 3.38: Convergence rates for the sinusoid MMS problem on an ordered-triangular mesh.

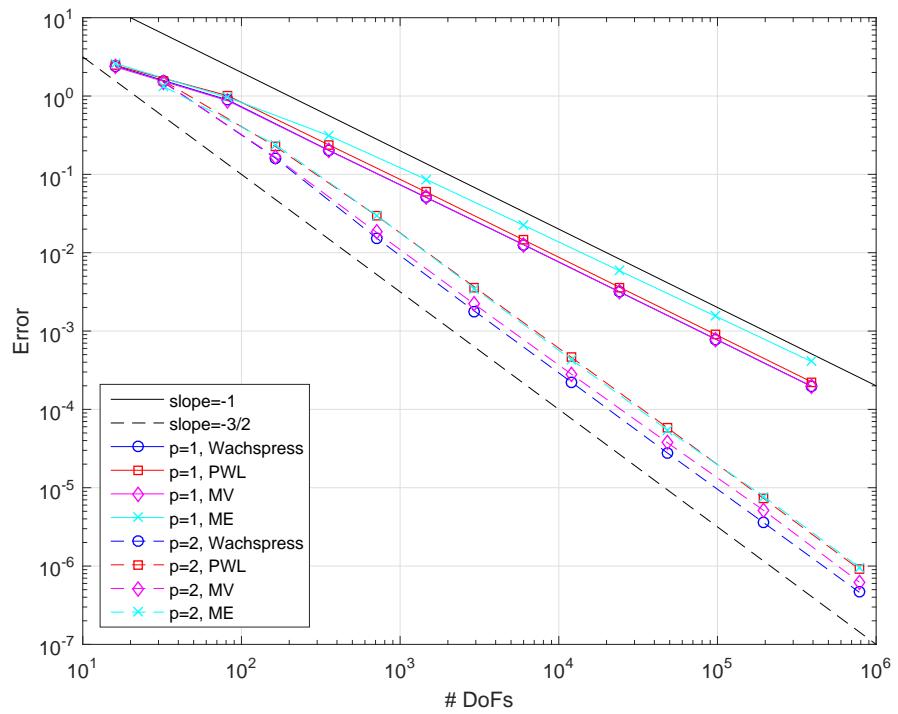


Figure 3.39: Convergence rates for the sinusoid MMS problem on a regular polygonal mesh.

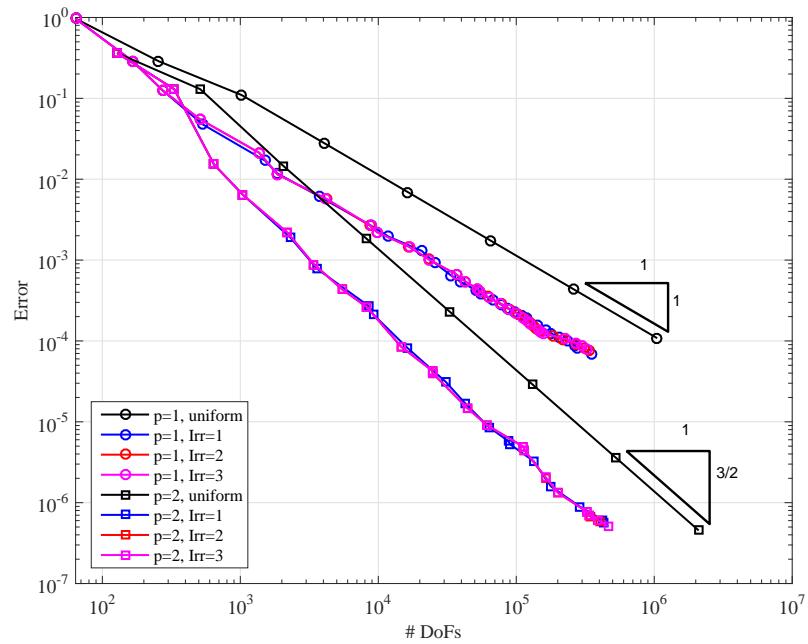


Figure 3.40: Convergence rates for the 2D Gaussian MMS problem using the PWL basis functions.

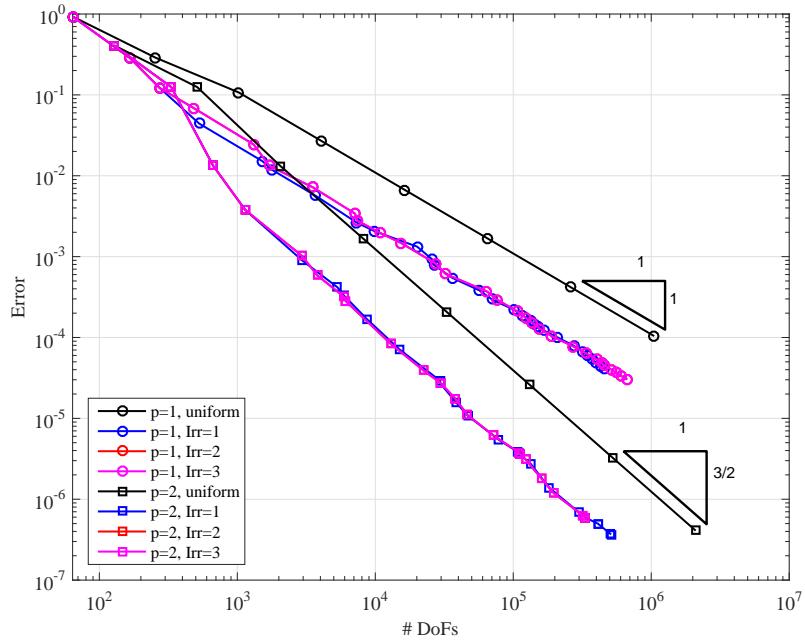


Figure 3.41: Convergence rates for the 2D Gaussian MMS problem using the mean value basis functions.

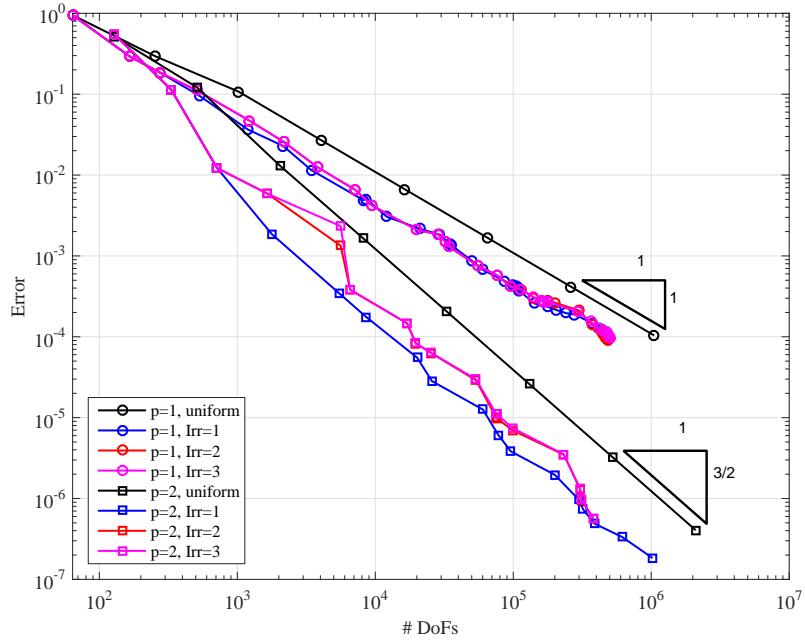


Figure 3.42: Convergence rates for the 2D Gaussian MMS problem using the maximum entropy basis functions.

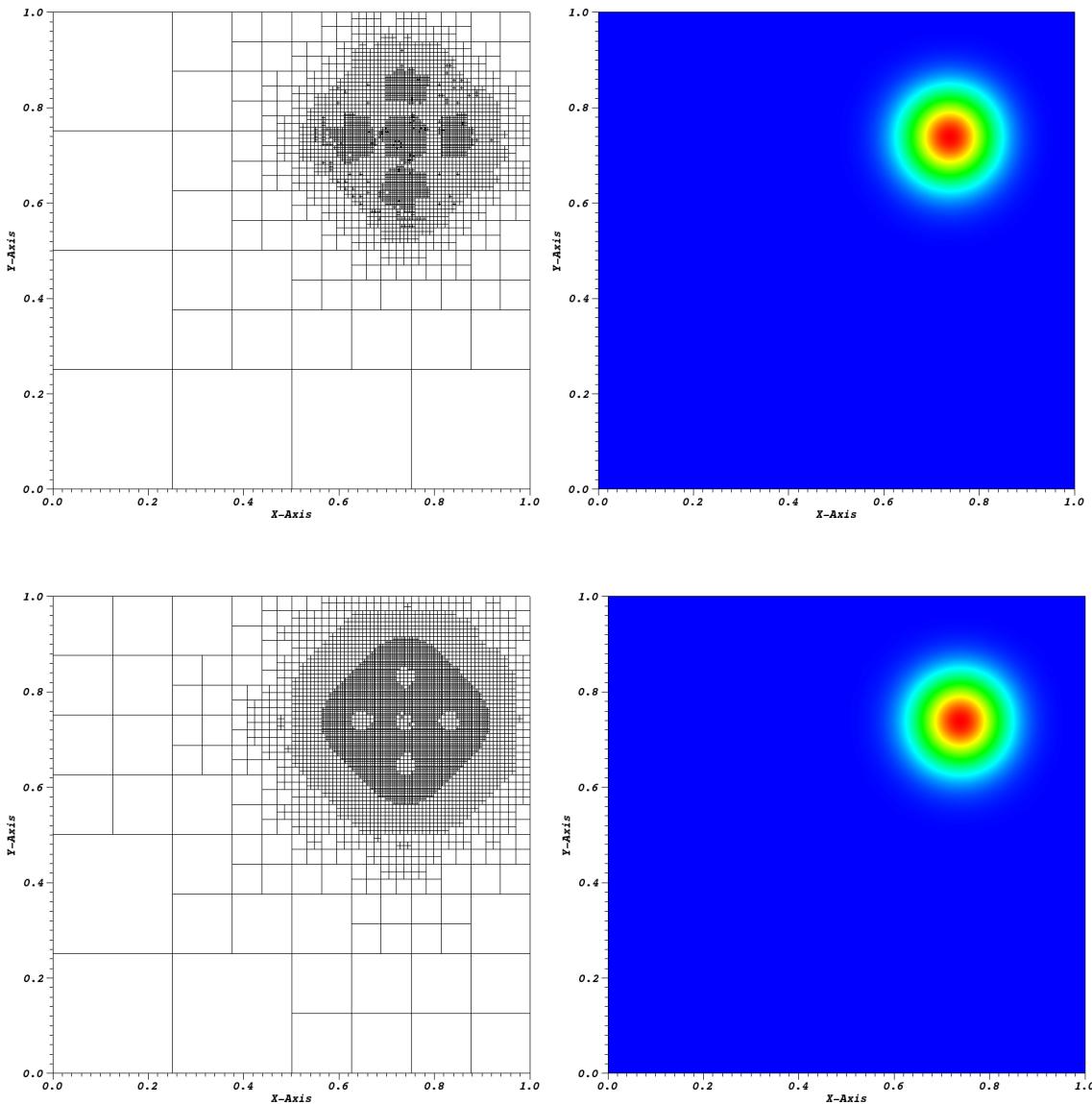


Figure 3.43: AMR meshes and solutions for the Gaussian MMS problem using the maximum entropy coordinates: (top) linear basis functions at cycle 15 and (bottom) quadratic serendipity basis functions at cycle 08.

$$\|\Phi - \Phi_h\|_{L_2} \leq C \frac{h^{q+1/2}}{(p+1)^q}, \quad (3.87)$$

where $q = \min(p + 1/2, r - 1/2)$ and r is the regularity of the transport solution. If the mesh does not align with the discontinuities of the transport solution, then the solution convergence is restricted by r . However, if the mesh aligns with all of the solution discontinuities, then the maximum $p + 1$ converge rates can be observed. To study this behavior, we will analyze cases with aligned and non-aligned meshes.

For this example, two different transport problems will be evaluated. Both problems will consist of a purely-absorbing medium ($\sigma_s = 0$), and there will be no distributed source within the domain. All meshes for both problems will be contained in the unit square: $[0, 1]^2$ and S_4 level-symmetric quadrature is used. The first problem has incident angular flux on the left face at a downward 45° angle. The second problem has the same incident flux on the left face, but also has additional incident angular flux on the top face at a 45° angle aligned with that of the left face. This means that there is a solution discontinuity for both problems along the line from the vertex $(0, 1)$ to $(1, 0)$. Furthermore, the first problem has a void in the upper-right portion of the domain. Therefore, the problems have solution regularities of $1/2$ and $3/2$ for the first and second problem, respectively.

We will analyze four different mesh types. Two of them will align with the solution discontinuity, and the other two will not. The meshes used are an ordered triangular mesh (aligns with the discontinuity), a regular Cartesian mesh (does not align with the discontinuity), a polygonal mesh (does not align with the discontinuity), and a split-polygonal mesh (aligns with the discontinuity). The split-polygonal meshes are formed by bisecting the polygonal meshes along the solution discontinuity. These meshes are shown in Figure 3.44. Next, in Figures 3.45 and 3.46, we provide example

solutions for the problems with left-face incidence and then left-face and top-face incidence, respectively. It is easy to see in Figure 3.45 how the alignment of the triangular and split-polygonal meshes allows us to capture the void in the upper-right portion of the meshes. The numerical dispersion of the beam due to the nature of the upwind scheme forces bleeding of the flux into the void region for the Cartesian and polygonal meshes.

Next, we provide the convergence plots of the solution histories for both problems, for all meshes, for all basis functions, and for several values of σ_t . For each combination of problem and mesh type, convergence rate analysis was performed using σ_t values of 1, 10, 50, and 100. First we analyze the problem with only left-face incidence. For the non-aligned meshes, we expect convergence rates of $1/2$ since the transport solutions only live in the $H^{1/2-\epsilon}$ space. The convergence rates are measured in the L_2 -norm and are plotted in Figures 3.47, 3.48, 3.49, and 3.50 for the triangular, Cartesian, polygonal, and split-polygonal meshes, respectively. With the aligned meshes, we can clearly see that convergence rates of $p + 1$ are obtained. The DGFEM solution “does not see” the lack of regularity in the transport solution. For the Cartesian and polygonal meshes, we see that convergence rates of about $1/2$ are obtained for optically thin meshes. However, for thicker domains in the pre-asymptotic range, higher convergence rates (without quite obtaining $p + 1$ values) are observed.

Finally, we analyze the problem with both left-face and top-face incidence of the angular flux. For the non-aligned meshes, we expect convergence rates of $3/2$ since the transport solutions only live in the $H^{3/2-\epsilon}$ space. Again, the convergence rates are measured in the L_2 -norm and are plotted in Figures 3.51, 3.52, 3.53, and 3.54 for the triangular, Cartesian, polygonal, and split-polygonal meshes, respectively. For the triangular and split-polygonal meshes, we again see convergence rates of order

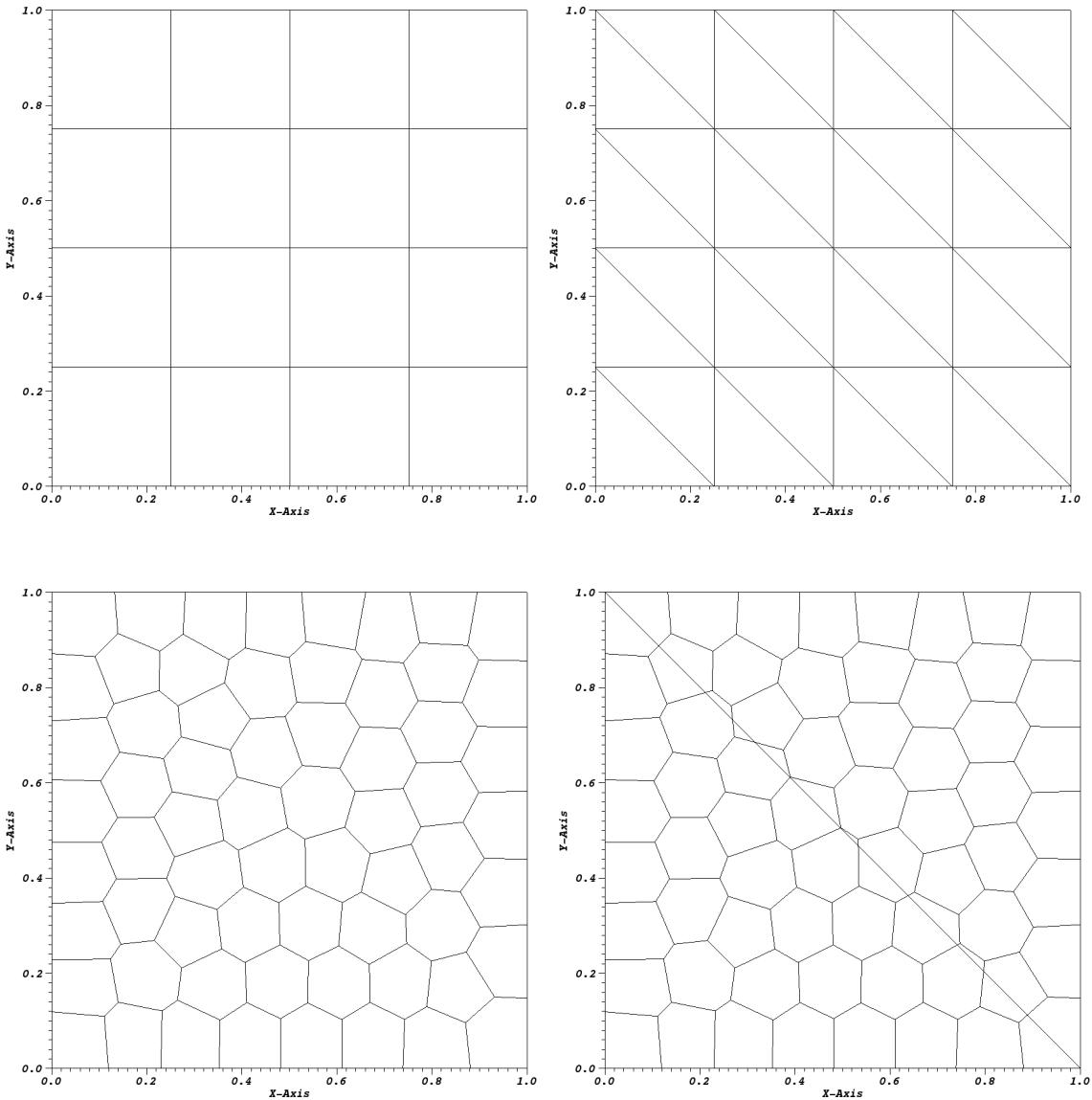


Figure 3.44: Mesh types used for the purely-absorbing medium problem.

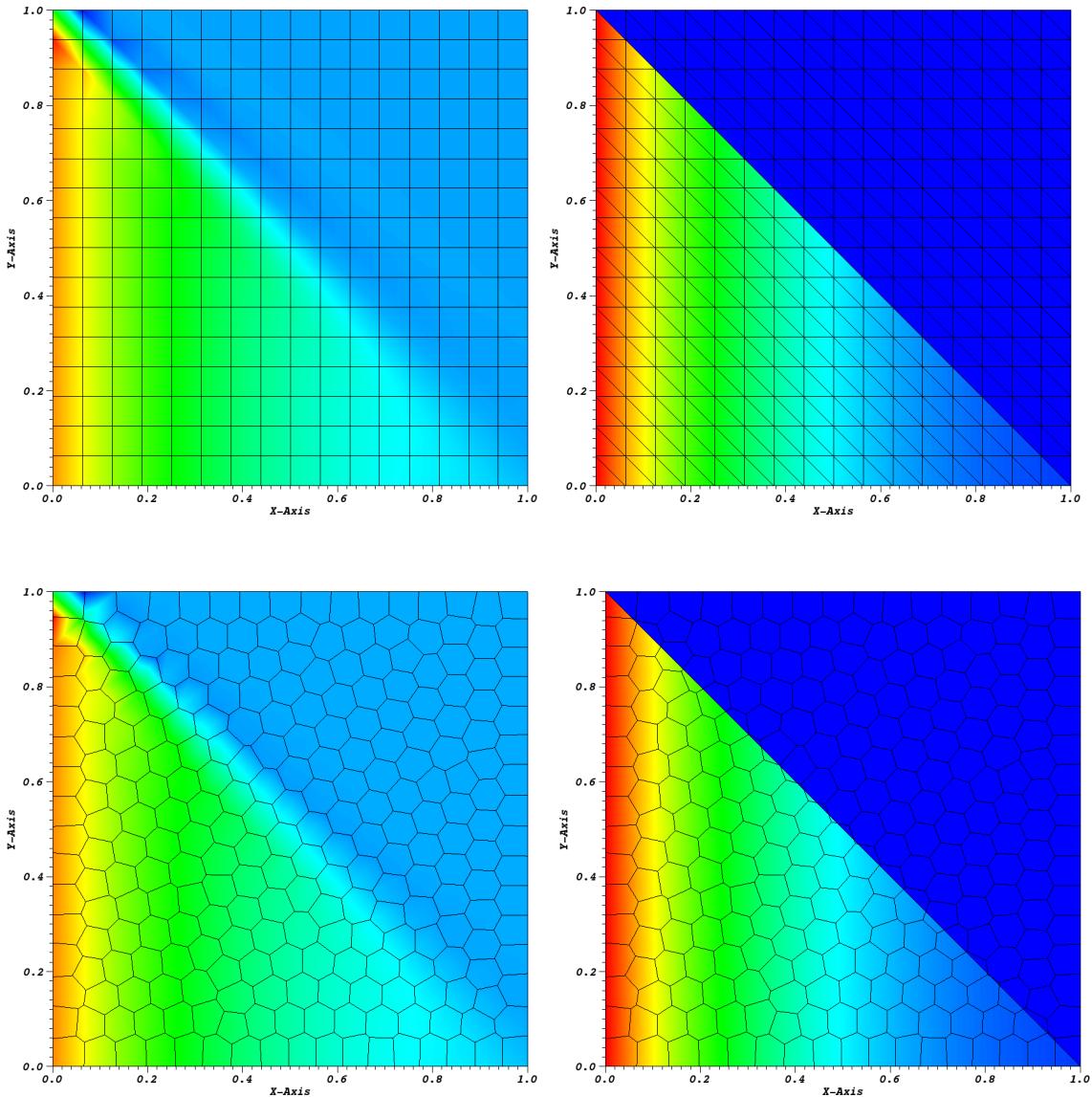


Figure 3.45: Example solution of the purely-absorbing medium case with left-face incidence and $\sigma_t = 1$ using the linear PWL basis functions.

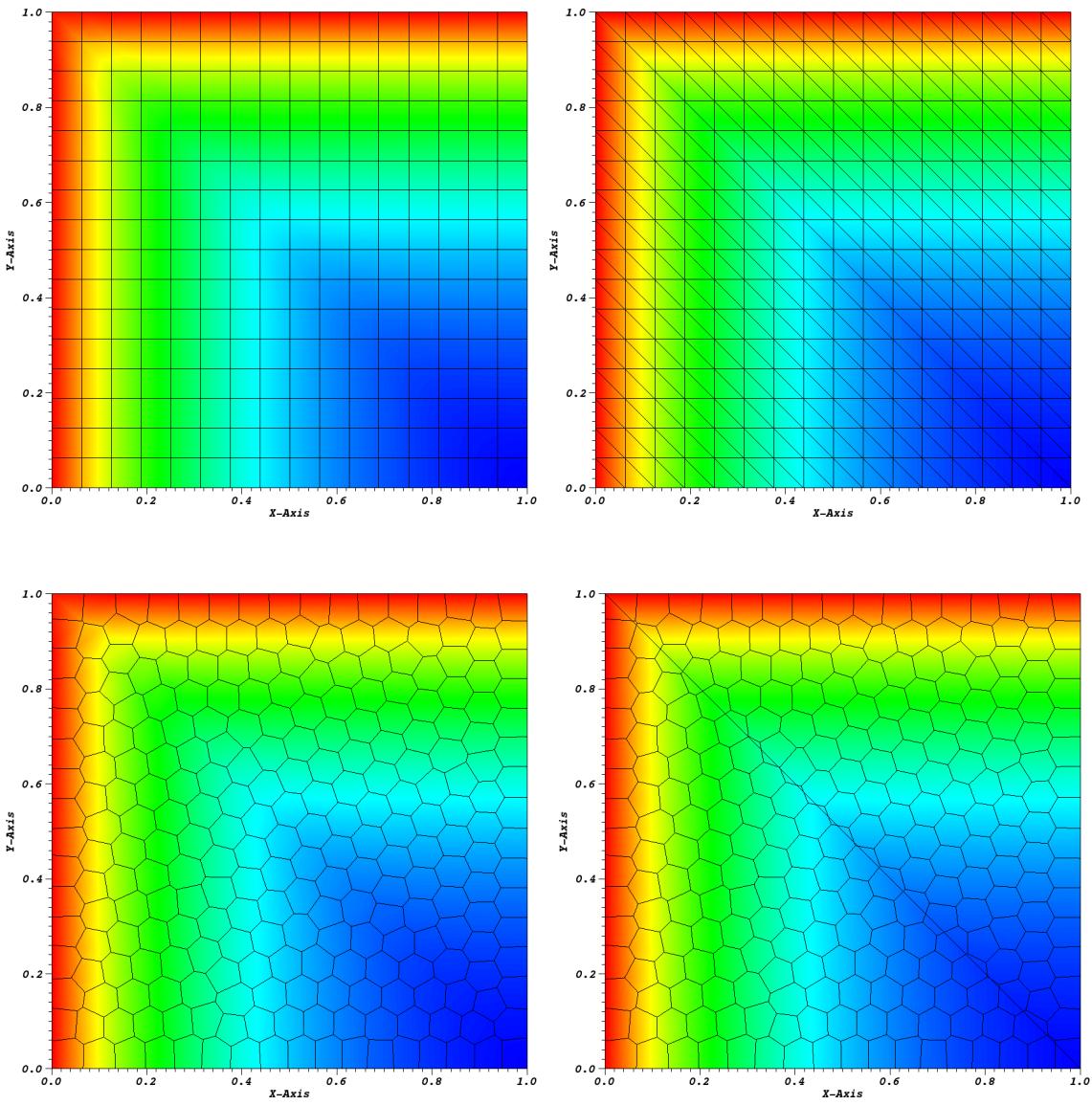


Figure 3.46: Example solution of the purely-absorbing medium case with left-face and top-face incidence and $\sigma_t = 1$ using the linear PWL basis functions.

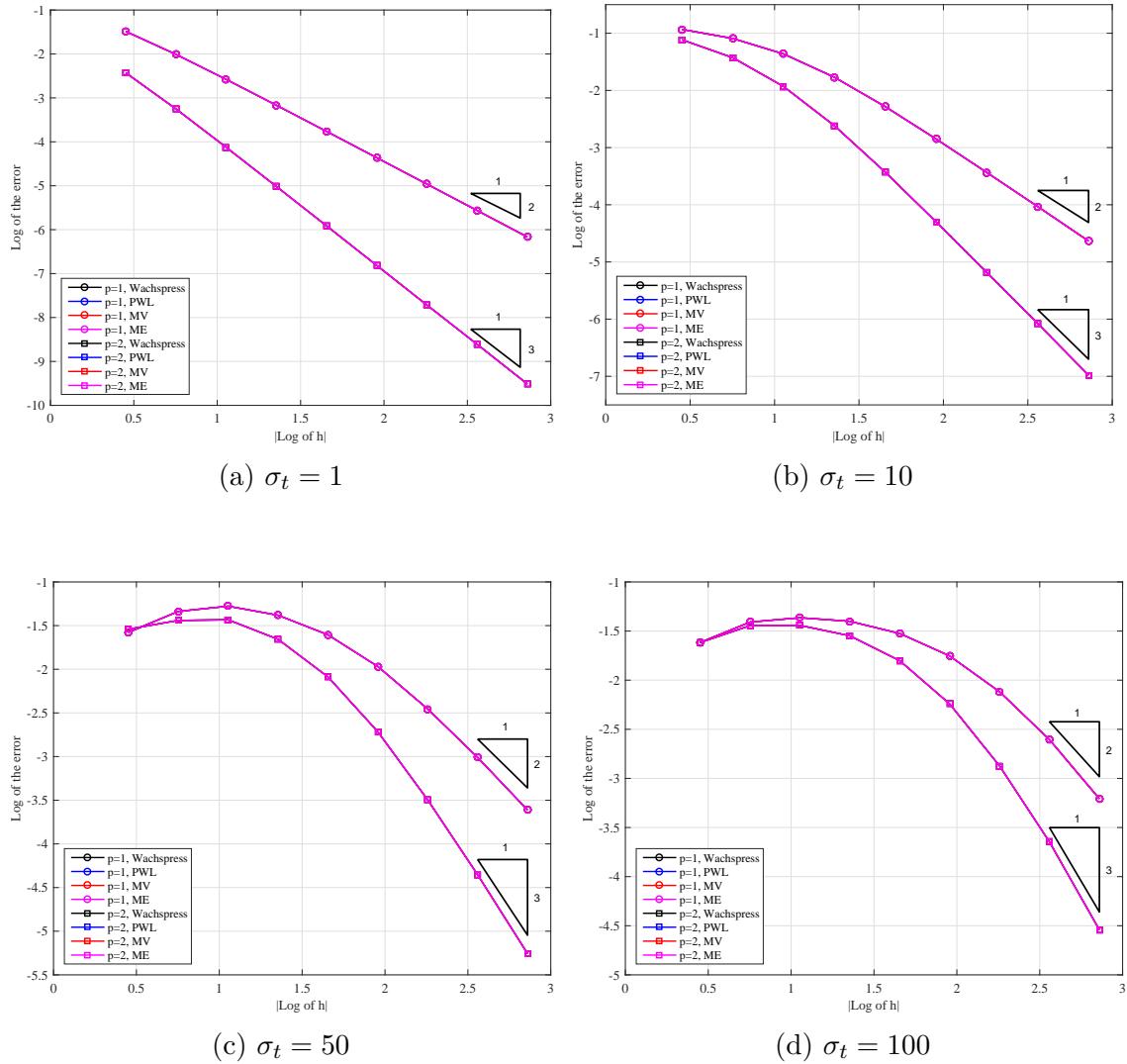


Figure 3.47: Convergence rates for the pure absorber problem with left-face incidence on triangular meshes with different values of σ_t .

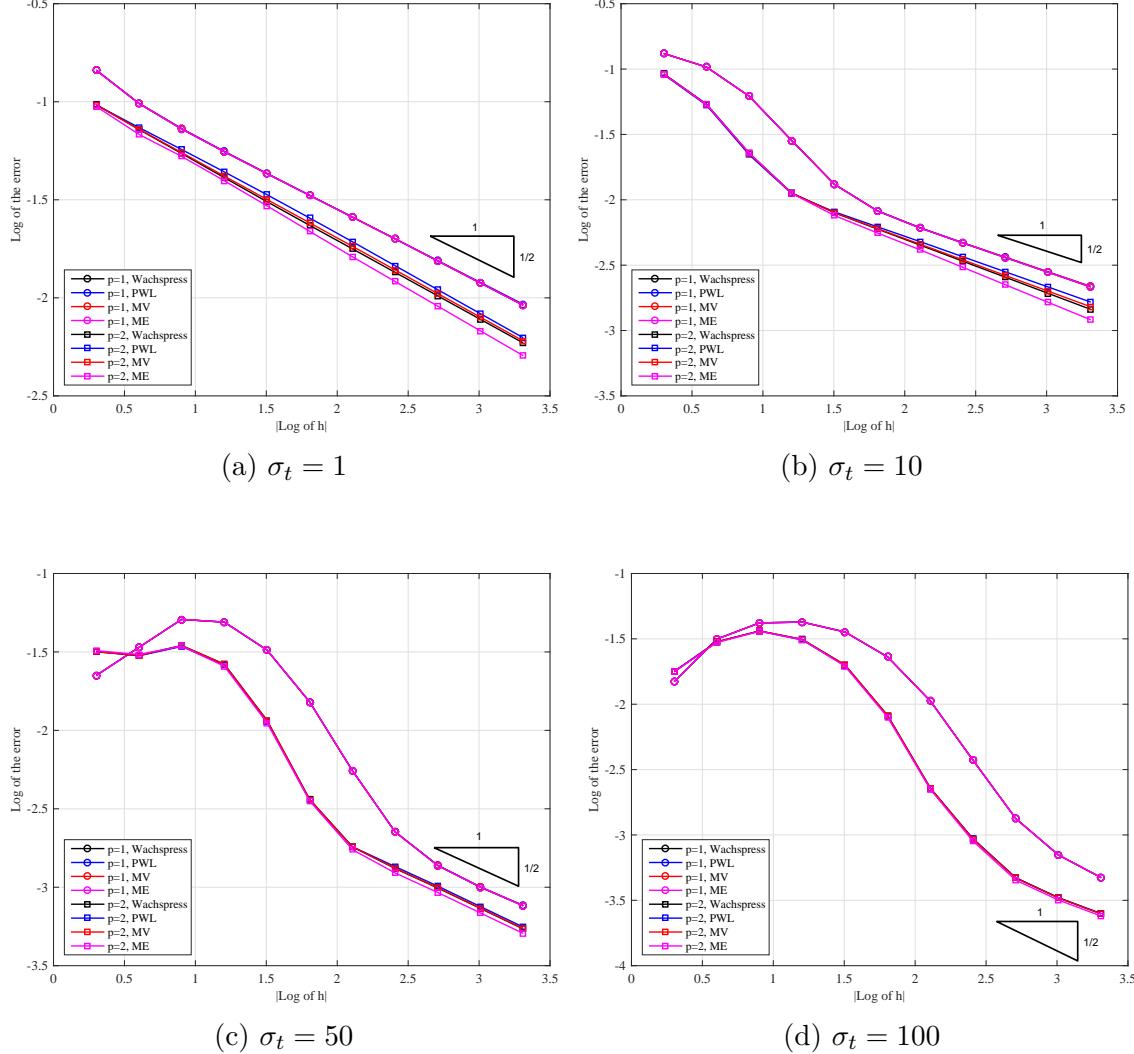


Figure 3.48: Convergence rates for the pure absorber problem with left-face incidence on Cartesian meshes with different values of σ_t .

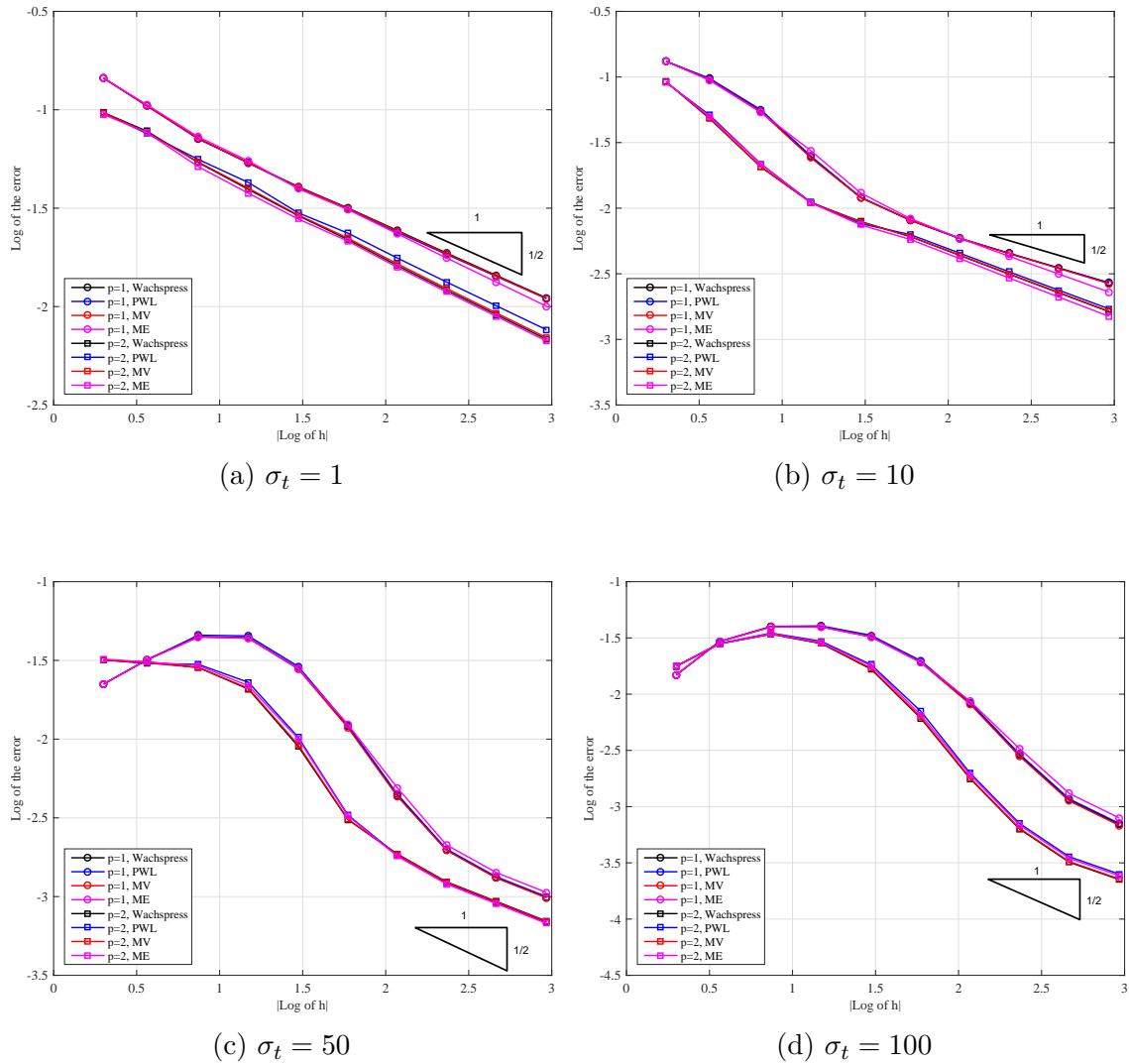


Figure 3.49: Convergence rates for the pure absorber problem with left-face incidence on polygonal meshes with different values of σ_t .

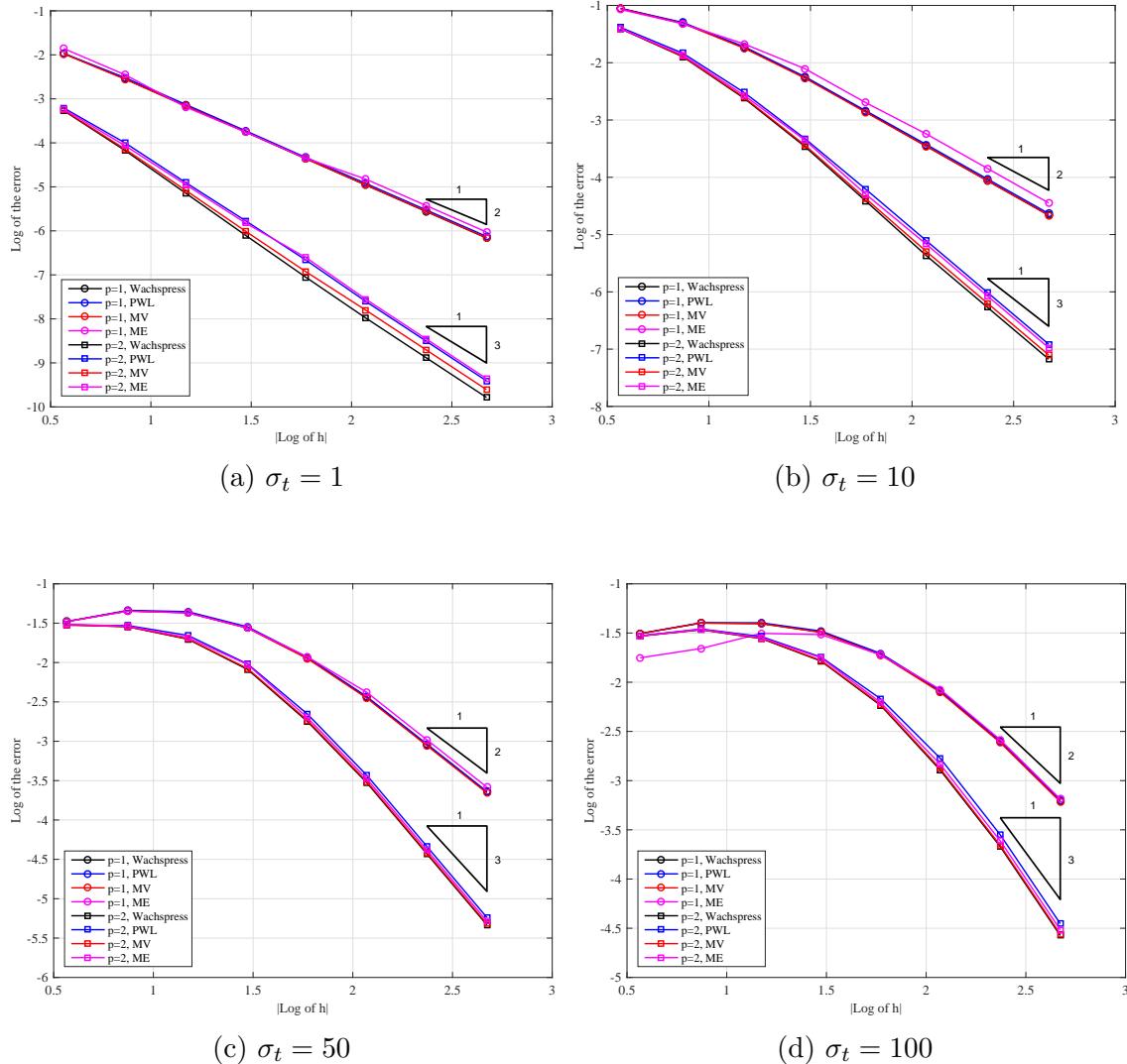


Figure 3.50: Convergence rates for the pure absorber problem with left-face incidence on split-polygonal meshes with different values of σ_t .

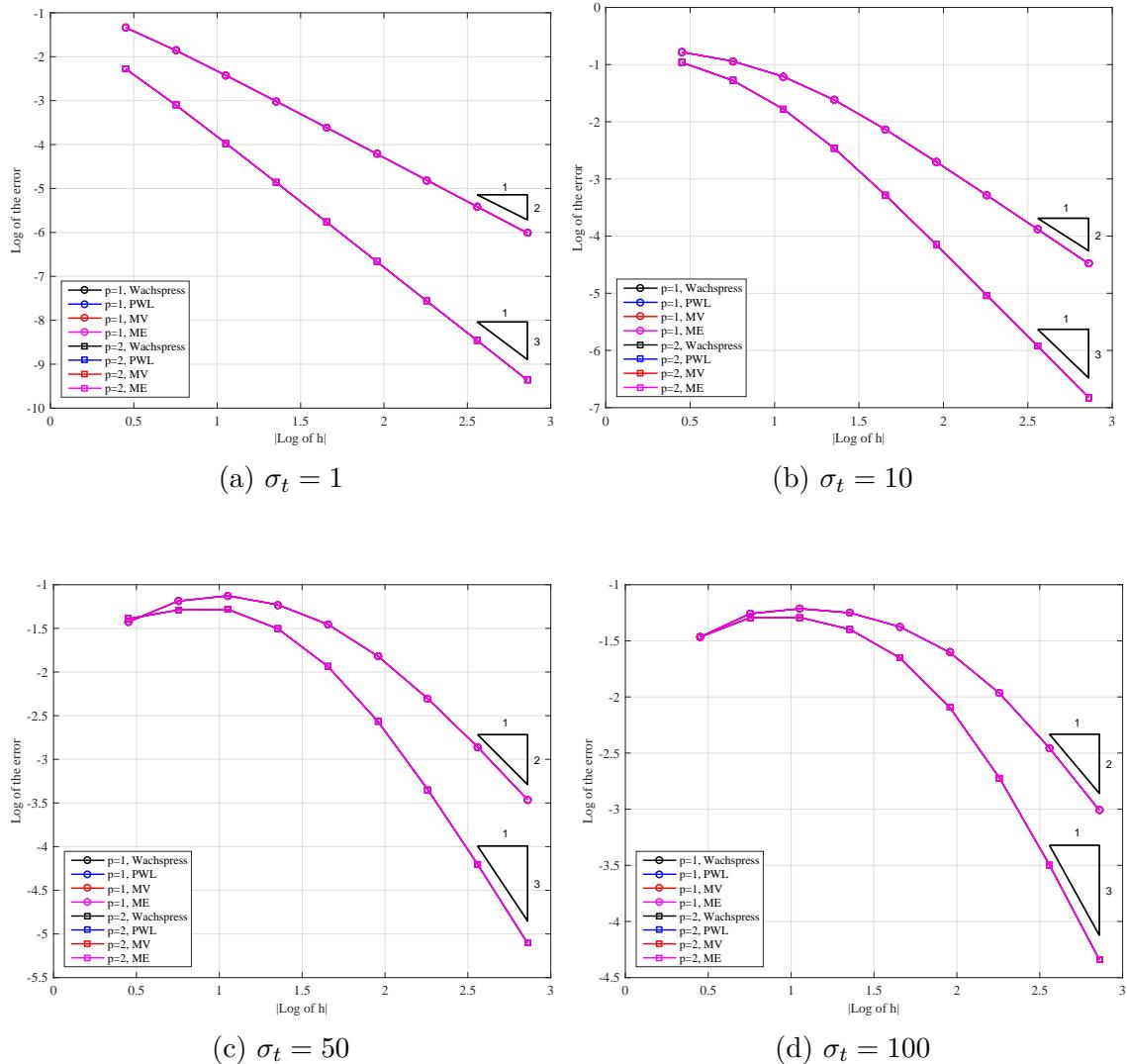


Figure 3.51: Convergence rates for the pure absorber problem with left-face and top-face incidence on triangular meshes with different values of σ_t .

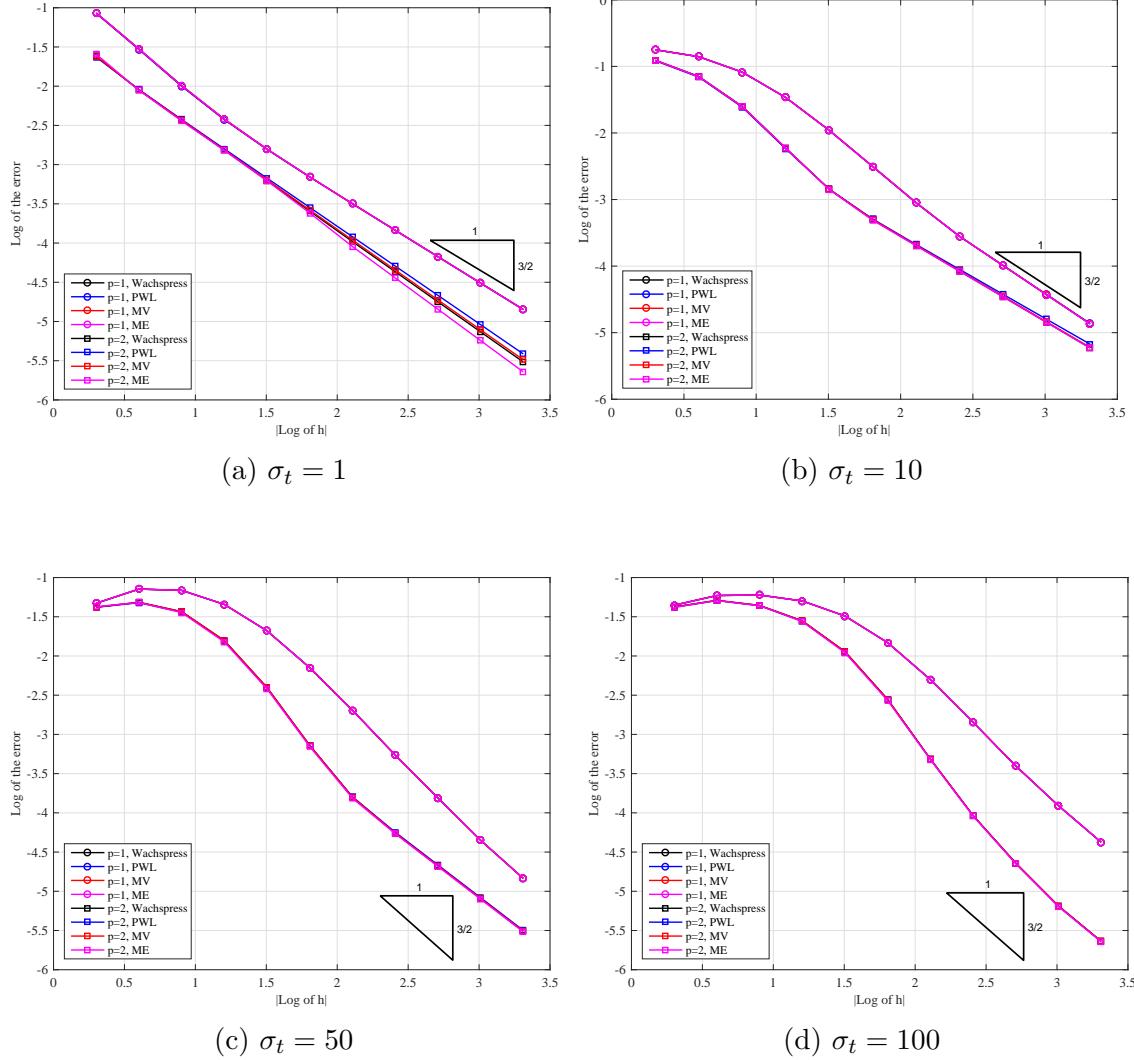


Figure 3.52: Convergence rates for the pure absorber problem with left-face and top-face incidence on Cartesian meshes with different values of σ_t .

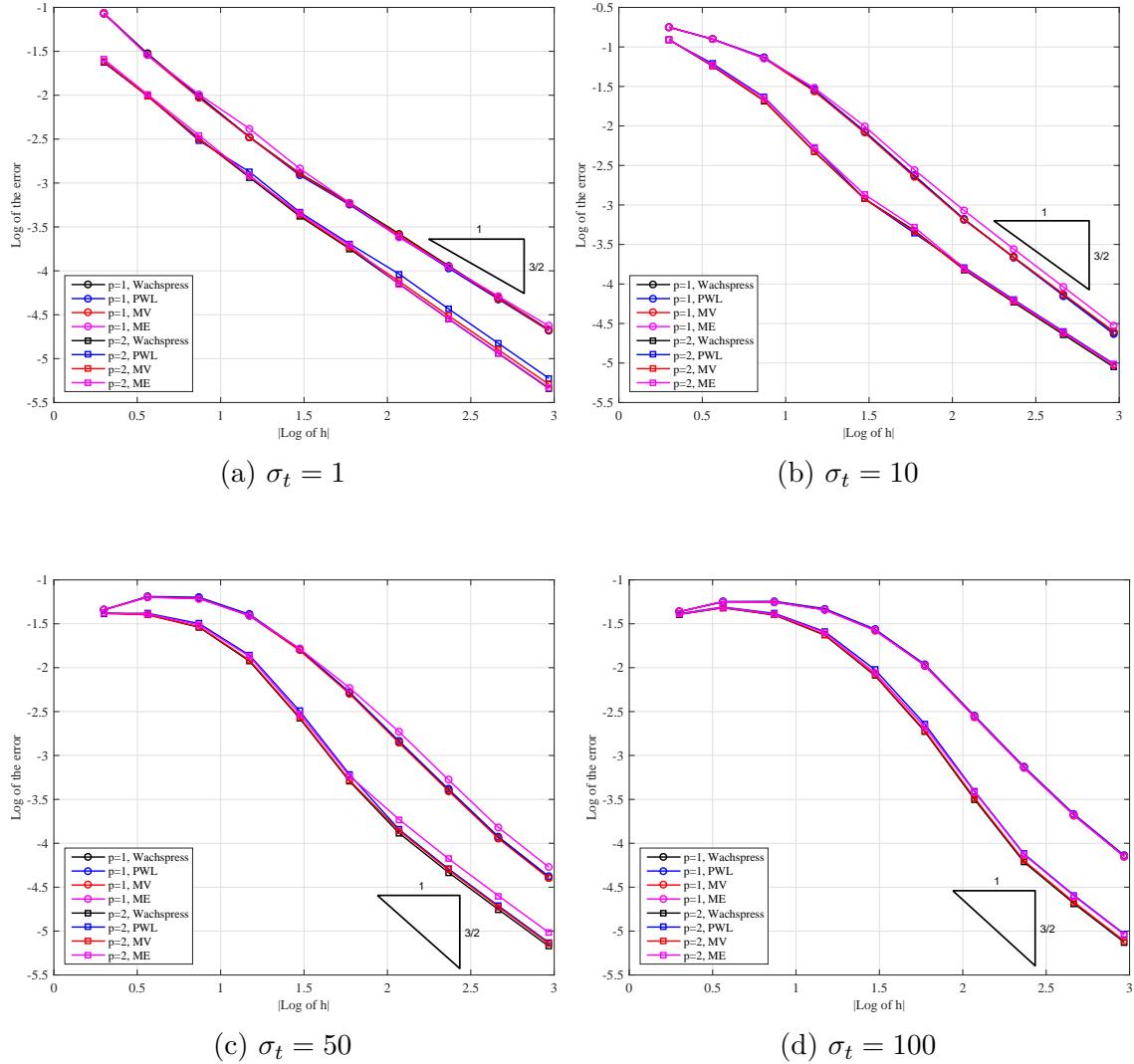


Figure 3.53: Convergence rates for the pure absorber problem with left-face and top-face incidence on polygonal meshes with different values of σ_t .

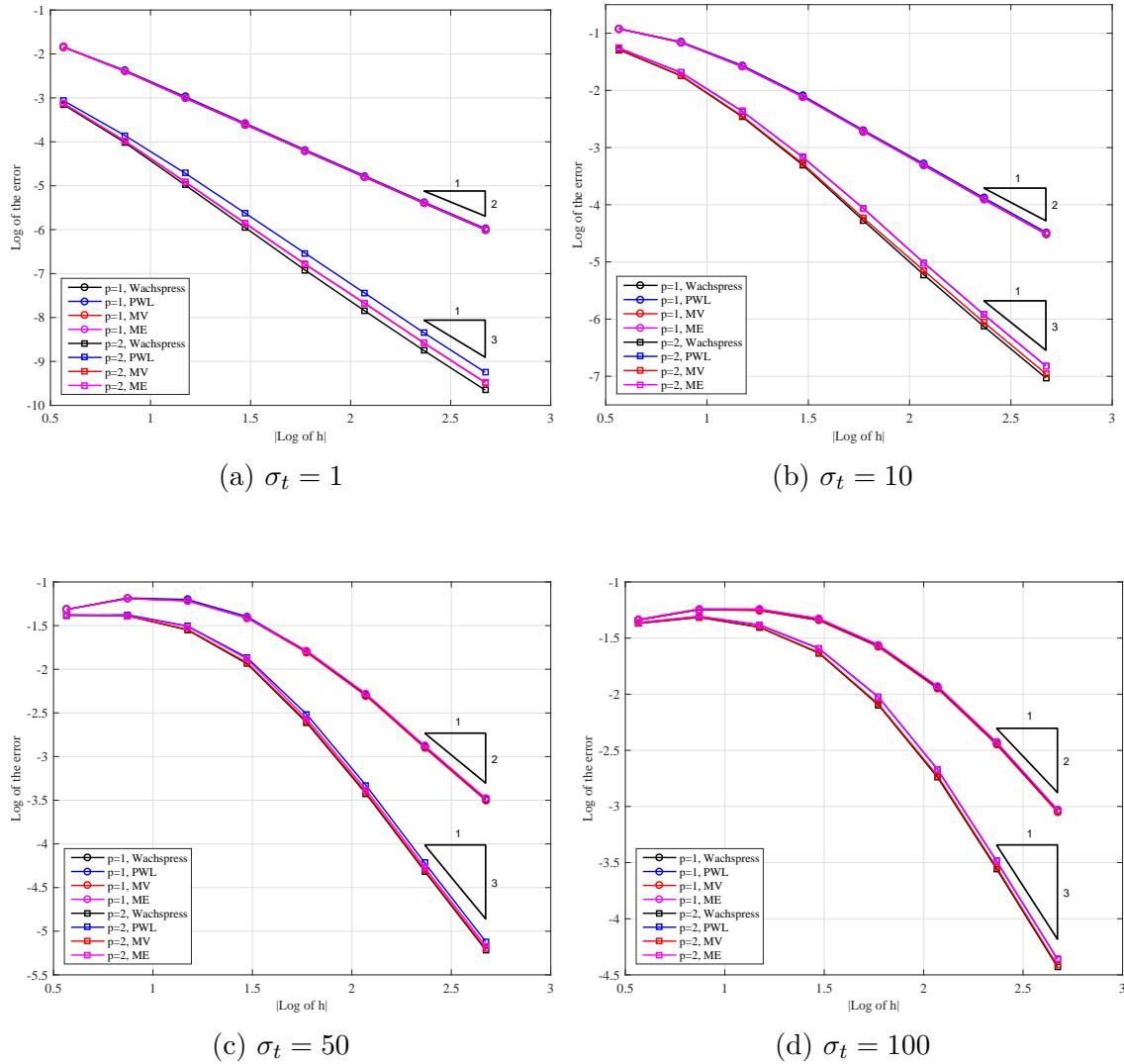


Figure 3.54: Convergence rates for the pure absorber problem with left-face and top-face incidence on split-polygonal meshes with different values of σ_t .

$p + 1$ in an identical manner to the left-face incidence problem. The non-aligned Cartesian and polygonal meshes give convergence rates of $3/2$ for the optically thin meshes. This higher convergence rate is achieved because the transport solution is now C^0 continuous, which yields a higher solution regularity. Again, we also see for the thicker domains in the pre-asymptotic range higher convergence rates that tend to $p + 1$.

3.5.5 Transport Solutions in the Thick Diffusive Limit

Next, we present numerical verification that the various 2D polygonal finite element basis functions provided in Chapter 3 satisfy the thick diffusion limit. From the work of Adams [137], it was shown that the spatial basis functions used for the DGFEM S_N transport equation were required to satisfy two properties to sufficiently show full resolution in the thick diffusion limit. First, the basis function are required to have “locality” so that only tightly clustered faces of an element produce non-zero surface integrals. Second, the basis function are required to have the surface-matching property.

We investigate the transport problem with isotropic scattering and an isotropic distributed source given by the following:

$$\vec{\Omega} \cdot \vec{\nabla} \Psi + \sigma_t \Psi = \frac{\sigma_s}{4\pi} \Phi + \frac{Q_0}{4\pi}. \quad (3.88)$$

As the transport problem becomes more optically thick, the total mean free paths of the neutrons 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,

$$\begin{aligned}\sigma_t &\rightarrow \frac{\sigma_t}{\epsilon} \\ \sigma_a &\rightarrow \epsilon\sigma_t \\ \sigma_s &\rightarrow \left(\frac{1}{\epsilon} - \epsilon\right)\sigma_t \\ \frac{Q_0}{4\pi} &\rightarrow \epsilon\frac{Q_0}{4\pi}\end{aligned}\tag{3.89}$$

Inserting these scaled cross sections and source term into Eq. (3.88) leads to the following scaled transport equation:

$$\vec{\Omega} \cdot \vec{\nabla} \Psi + \frac{\sigma_t}{\epsilon} \Psi = \sigma_t \left(\frac{1}{\epsilon} - \epsilon \right) \frac{\Phi}{4\pi} + \epsilon \frac{Q_0}{4\pi}.\tag{3.90}$$

We can also use the scaled terms of Eq. (3.89) to give the corresponding scaled diffusion equation. If we expand the angular flux as powers of ϵ in Eq. (3.90) and assume a Fick's Law, then the scaled diffusion equation is

$$-\epsilon \vec{\nabla} \cdot \frac{1}{3\sigma_t} \vec{\nabla} \Phi + \epsilon \sigma_t \Phi = \epsilon Q_0.\tag{3.91}$$

One can immediately see that Eq. (3.91) 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 that is independent of ϵ .

For the sake of analysis, we seek to simplify Eqs. (3.90) and (3.91) by normalization. We choose to set σ_t and Q_0 to unity, which gives the final transport and diffusion equations as

$$\vec{\Omega} \cdot \vec{\nabla} \Psi + \frac{1}{\epsilon} \Psi = \left(\frac{1}{\epsilon} - \epsilon \right) \frac{\Phi}{4\pi} + \frac{\epsilon}{4\pi}, \quad (3.92)$$

and

$$-\frac{\epsilon}{3} \nabla^2 \Phi + \epsilon \Phi = \epsilon, \quad (3.93)$$

respectively.

From previous work [137], it is already known that linear interpolants with properties corresponding to barycentric coordinates satisfy the thick diffusion limit. However, it needs to be demonstrated that the quadratic serendipity extensions will also satisfy the limit. We will demonstrate this both qualitatively and quantitatively. The transport and diffusion equations to be solved are Eqs. (3.92) and (3.93), respectively. The diffusion equation is discretized using a Continuous Finite Element Method (CFEM) form. Vacuum boundary conditions are applied for the transport equations, and homogeneous Dirichlet conditions are applied for the diffusion equations. With this choice of boundary conditions, the transport and diffusion solutions will converge only as ϵ gets small. Specifically, they converge at a rate of $O(\epsilon)$ with an L_2 -norm if the gradients of the basis functions span the same space as the functions. This is true for the Wachspress, mean value, and maximum entropy functions but not the PWL functions. For some range of ϵ , PWL will converge as $O(\epsilon)$, but will eventually plateau. This is because PWL converges to a slightly different diffusion solution than CFEM diffusion.

We first provide an example of the diffusion solution in Figure 3.55 on a polygonal grid for both the linear and quadratic mean value basis functions. Next, Figures 3.56, 3.57, 3.58, and 3.59 provide the transport solutions with varying ϵ values using the Wachspress, PWL, mean value, and maximum entropy coordinates, respectively. We

see that as we reduce ϵ from 10^{-1} to 10^{-5} , our transport solutions converge to our diffusion solutions. Finally, we show the convergence rates of the discretized transport, Φ_T , and diffusion, Φ_D , solutions under the L_2 -norm. The L_2 -norm of the difference between the transport and diffusion solutions, $\|\Phi_T - \Phi_D\|_{L_2}$, should decrease as $O(\epsilon)$ for all the basis functions except the linear and quadratic PWL functions. We demonstrate this as true in Figures 3.60 and 3.61 for a Cartesian and polygonal mesh, respectively. For both meshes, the linear and quadratic Wachspress, MV, and ME basis functions show convergence rates of $O(\epsilon)$, but PWL plateaus. However, we notice that the quadratic PWL functions converge as $O(\epsilon)$ longer because of their increased interpolation accuracy.

3.5.6 Searchlight Problem

Our final numerical example models a beam or searchlight through a vacuum. Similar problems were investigated in Dedner and Vollm  ller [73] and Wang and Ragusa [76]. In this problem, an incident beam of neutrons is shined onto a small portion of a boundary, propagates 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 the 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. The initial mesh is a uniform 5x5 Cartesian grid as shown in Figure 3.62. We note that with this choice of initial grid, the analytical path of the beam is

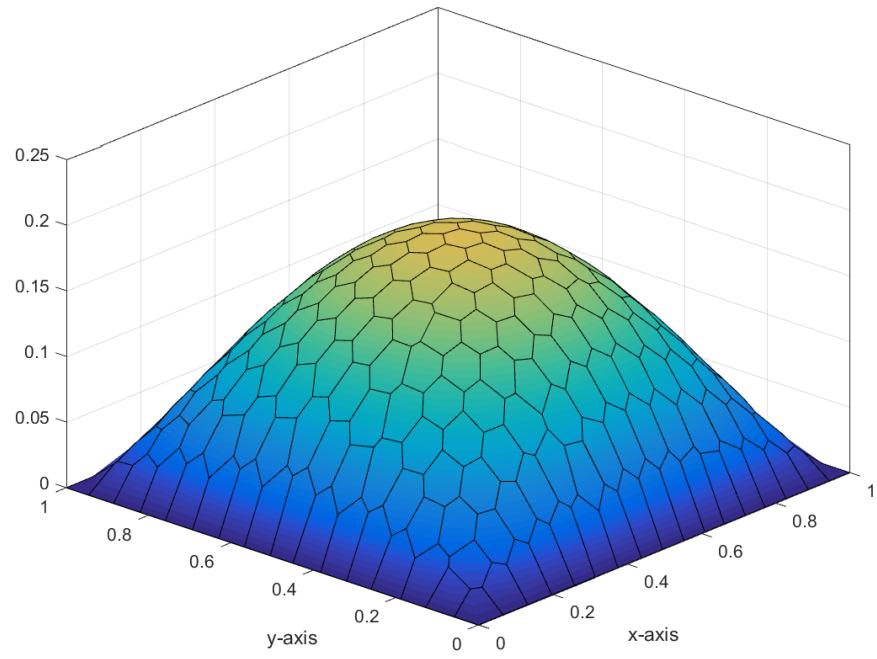
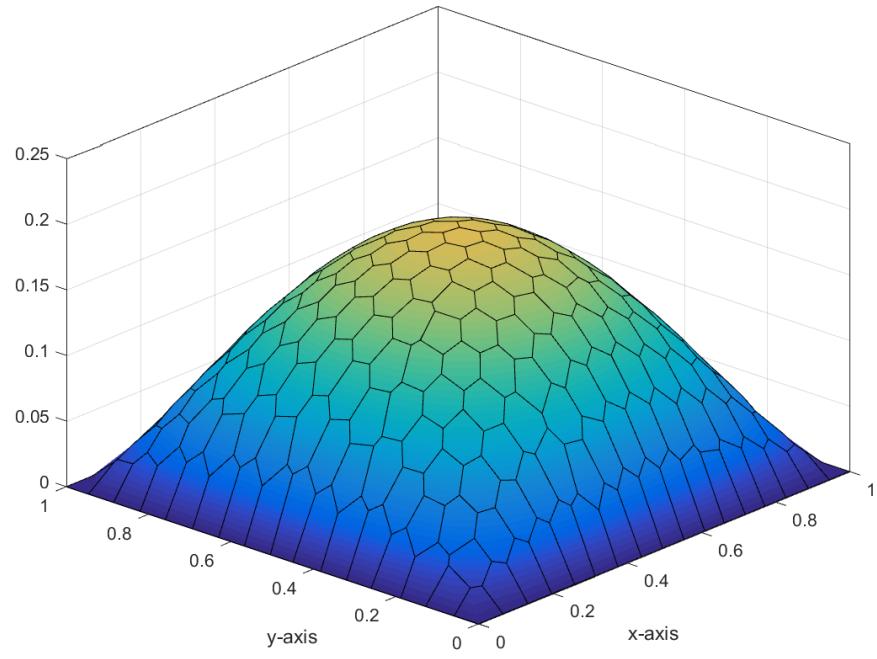


Figure 3.55: Diffusion solution representing the thick diffusion limit problem with linear (top) and quadratic (bottom) mean value basis functions.

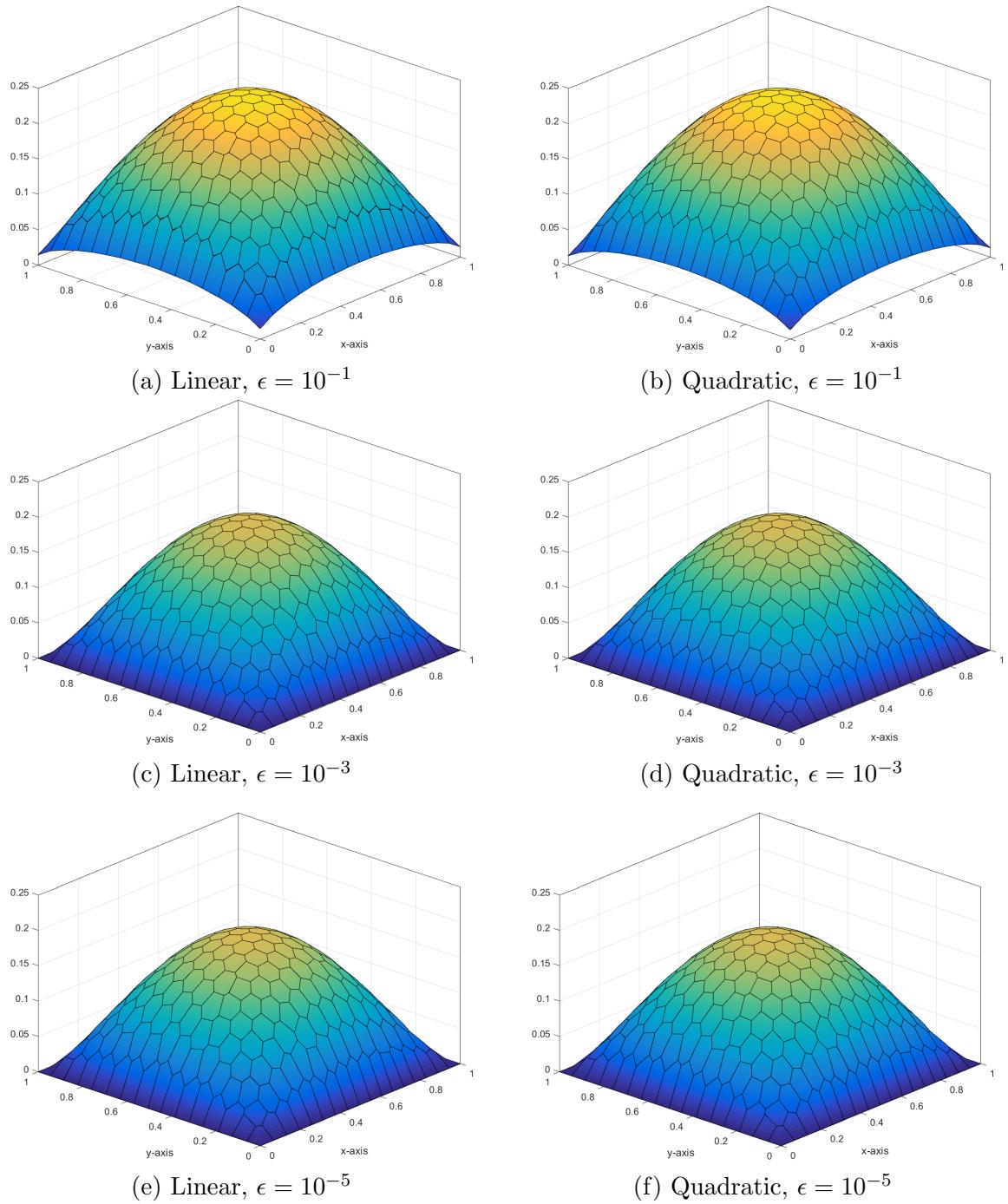


Figure 3.56: Transport solutions of the thick diffusion limit problem using the Wachspress basis functions for varying values of the scaling parameter, ϵ .

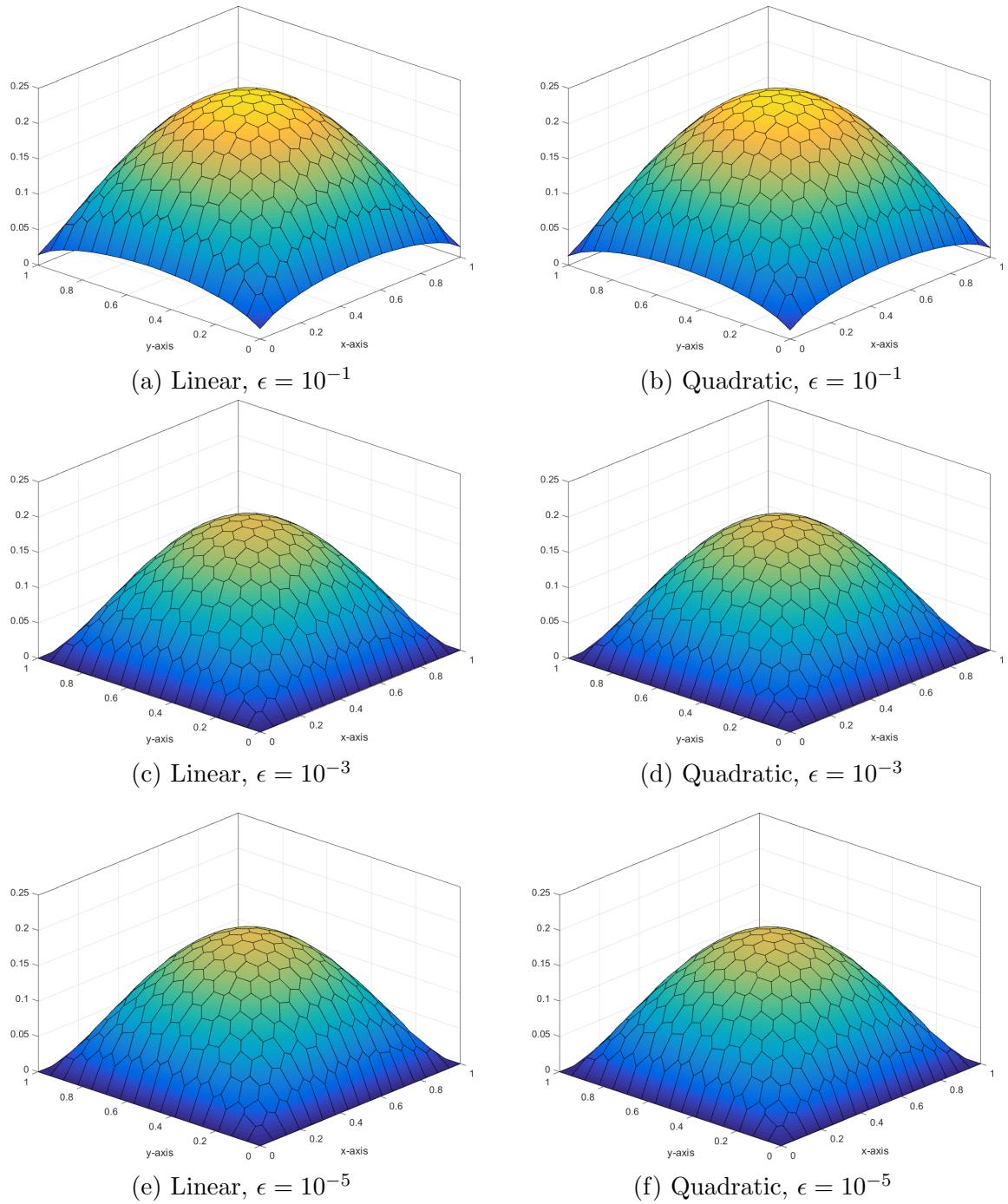


Figure 3.57: Transport solutions of the thick diffusion limit problem using the PWL basis functions for varying values of the scaling parameter, ϵ .

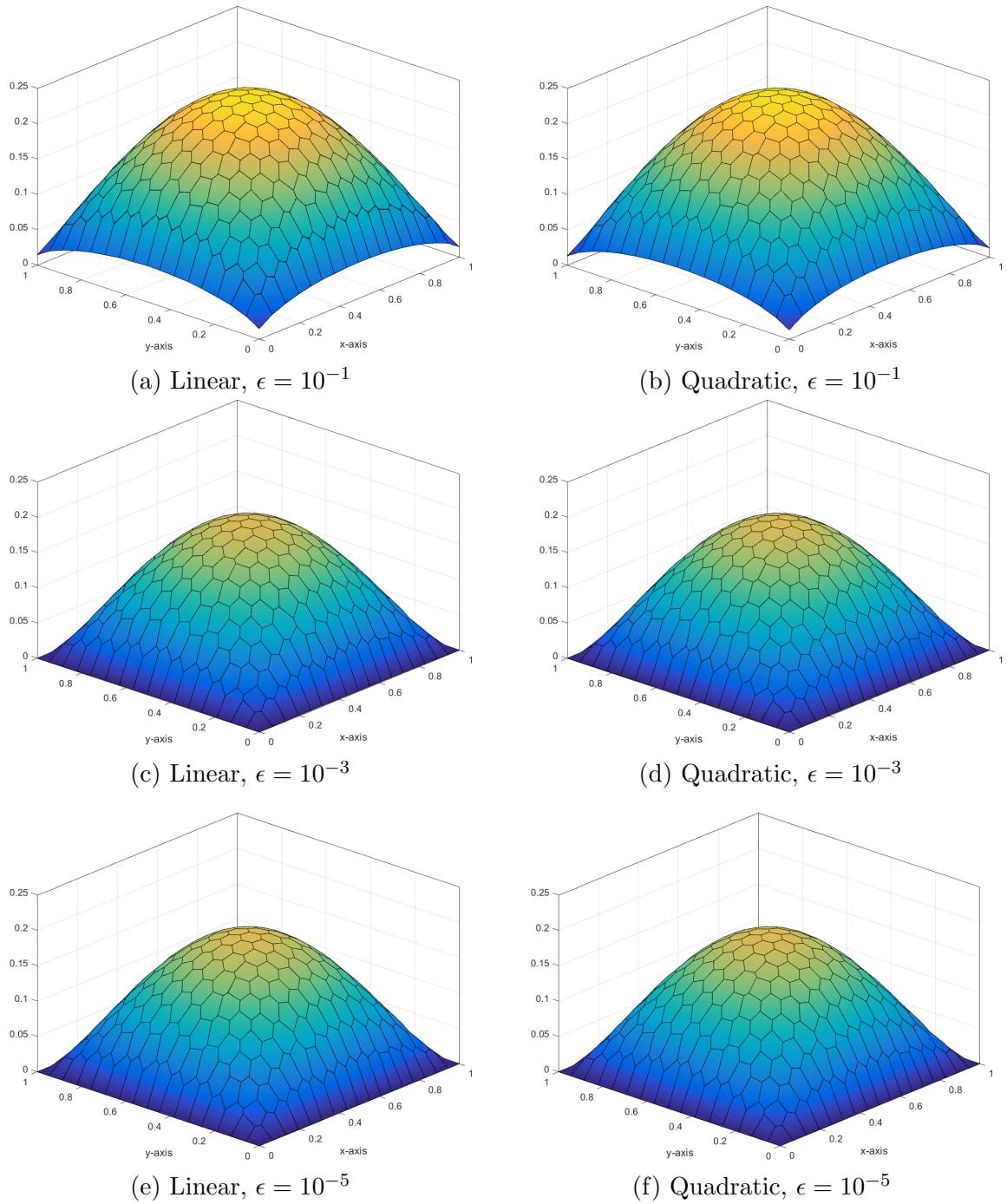


Figure 3.58: Transport solutions of the thick diffusion limit problem using the mean value basis functions for varying values of the scaling parameter, ϵ .

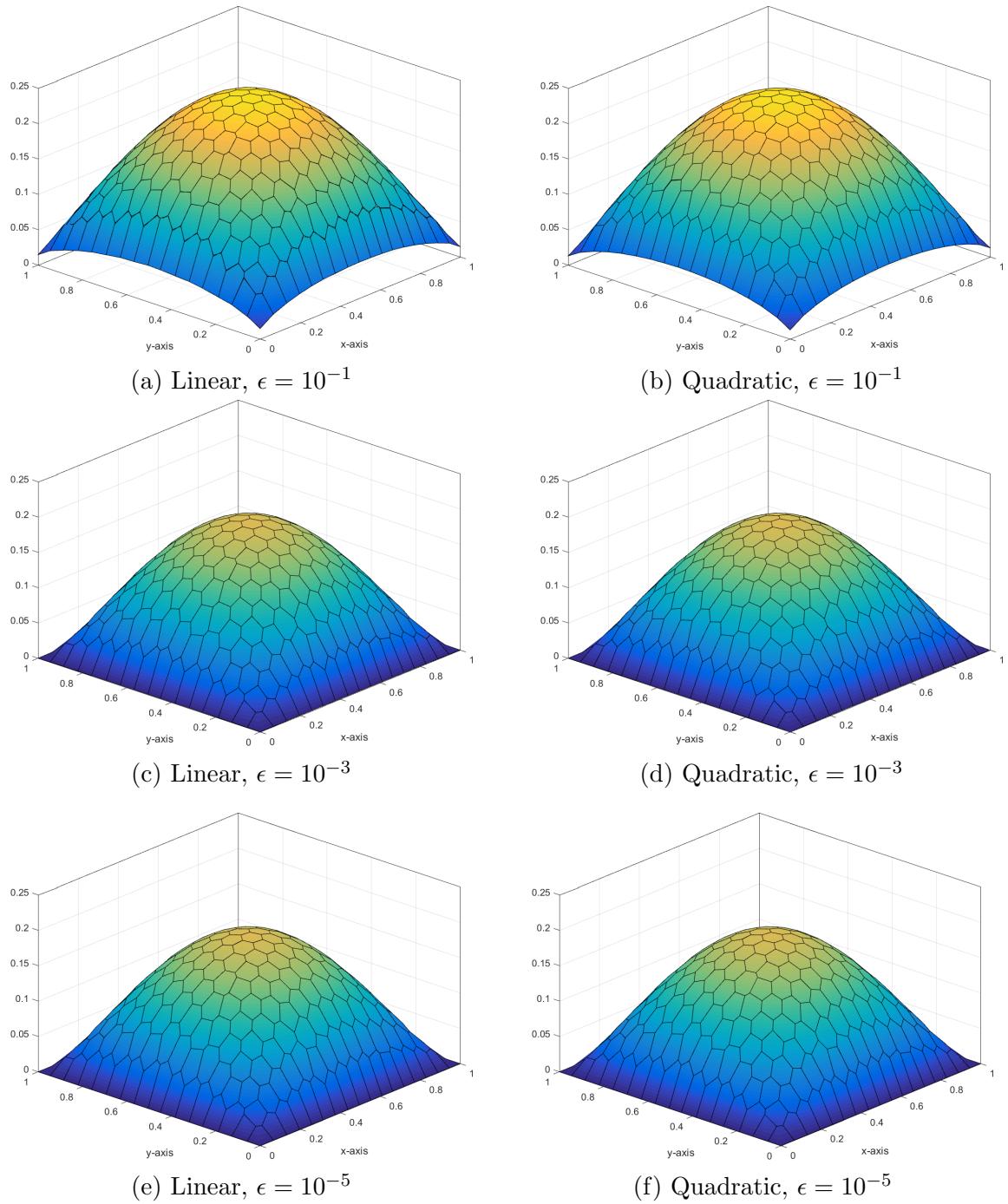


Figure 3.59: Transport solutions of the thick diffusion limit problem using the maximum entropy basis functions for varying values of the scaling parameter, ϵ .

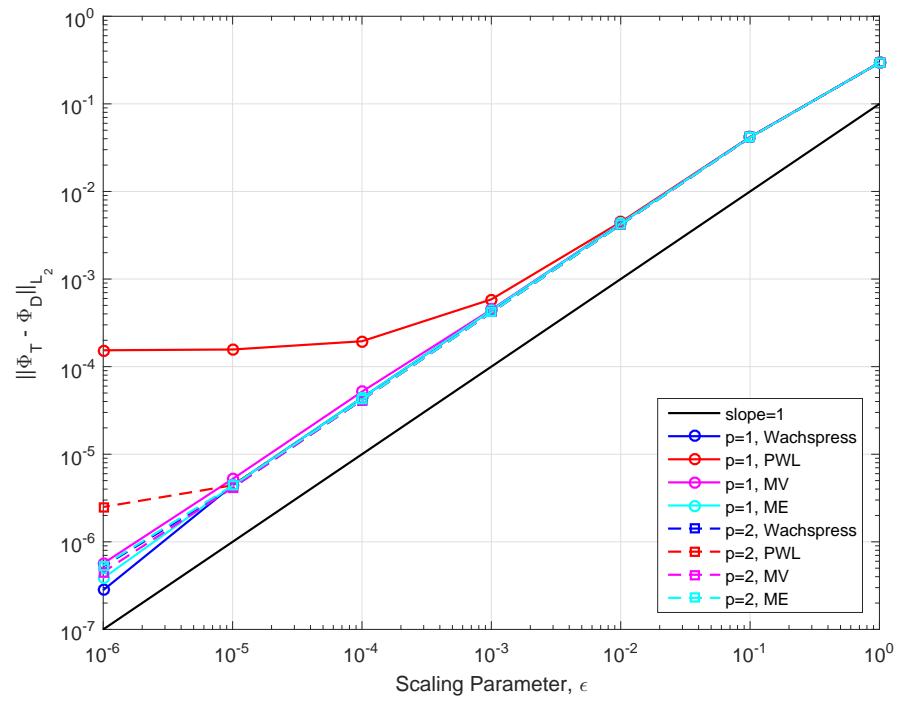


Figure 3.60: Convergence rates for the diffusion limit problem on a Cartesian mesh.

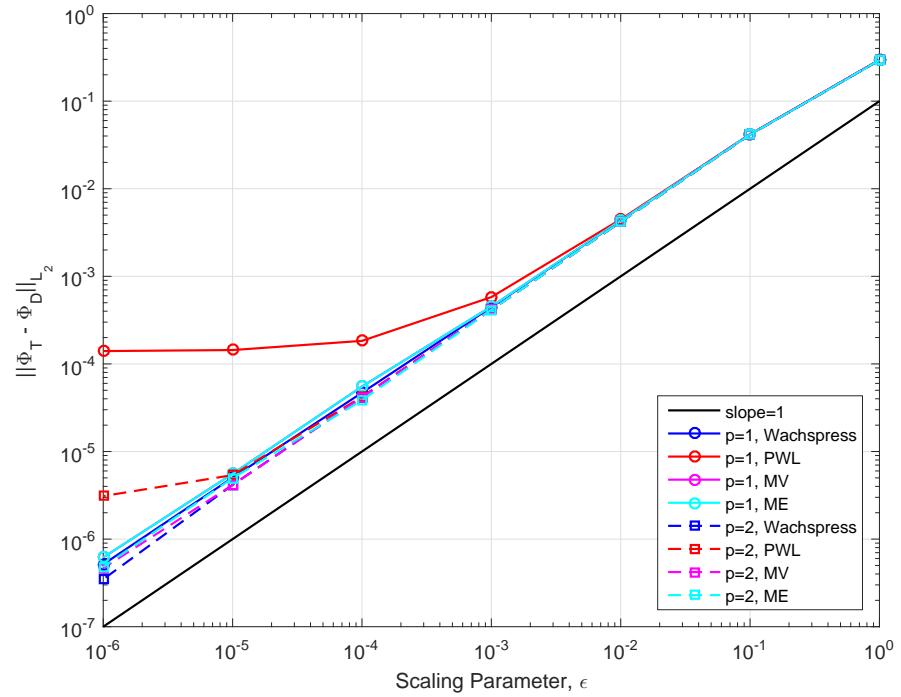


Figure 3.61: Convergence rates for the diffusion limit problem on a polygonal mesh.

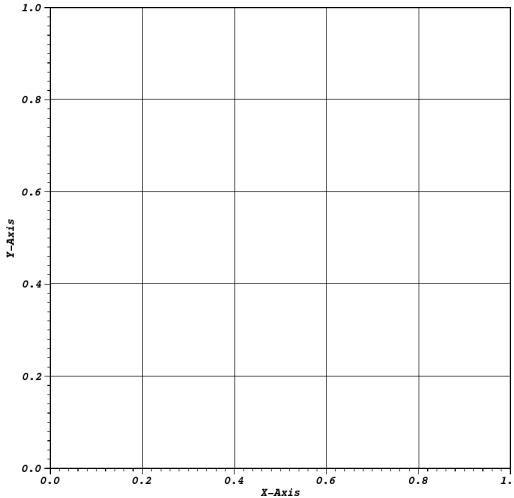


Figure 3.62: Initial mesh configuration for the searchlight problem before any refinement cycles.

fully contained within a spatial cell (a mesh cell does not bisect the analytical beam at the incoming and outgoing faces).

Since we are again using AMR for this problem, we will analyze the linear and quadratic PWL, mean value, and maximum entropy basis functions (not the Wachspress basis functions). The refinement criterion, α , was set to 0.2 for all the linear elements and 0.1 for all the quadratic elements. The relative error of the leakage (in %) along the right face from $0.6 \leq y \leq 0.8$ is given in Figures 3.63, 3.64, and 3.65 for the PWL, mean value, and maximum entropy coordinates, respectively. For each of the figures we plotted the convergences histories for uniform refinement ($\alpha = 0.0$) as well as AMR runs where the maximum mesh irregularities were set to 1, 2, or 3. Like the purely-absorbing media cases without meshes aligned to the discontinuities, the convergence of the transport solution is limited by its regularity. This is apparent in that the AMR solutions give significantly better results than those obtained with uniform refinement.

Next, we plot the outgoing angular flux along the right face. We wish to see how

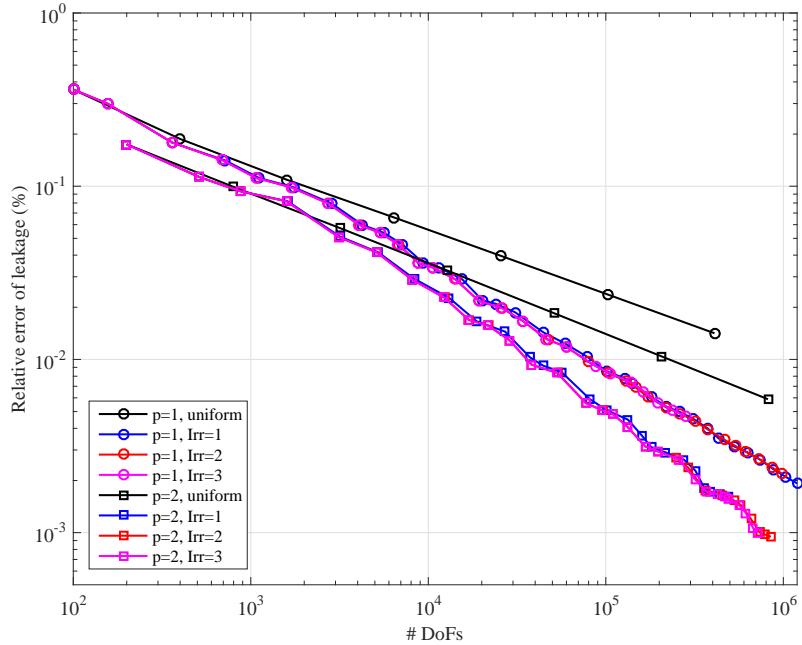


Figure 3.63: Convergence history for the searchlight problem using the PWL basis functions.

the numerical dispersion is reduced with refinement. Figure 3.66 shows the outgoing fluxes for all of the linear and quadratic basis functions under uniform refinement. For all of the linear and quadratic basis functions we see the solution converging to the proper exit flux solution (looks like a step function along the face). However, there is still some over-shooting and under-shooting occurring, even for the most refined case. We then plot the outgoing angular fluxes for the AMR cases in Figures 3.67, 3.68, and 3.69 using the PWL, MV, and ME basis functions, respectively. For all of these cases, we can see better convergence to the analytical outgoing flux for the later AMR cycles. However, the over-shooting and under-shooting still remain.

3.6 Conclusions

In this chapter, we presented four different linearly-complete, barycentric, 2D polygonal basis functions to be used with the DGFEM transport equation: the

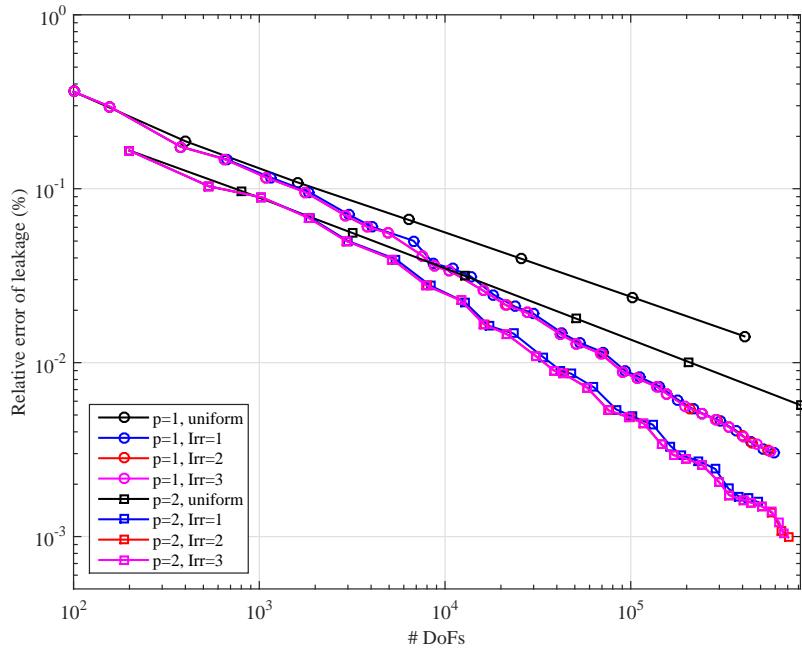


Figure 3.64: Convergence history for the searchlight problem using the mean value basis functions.

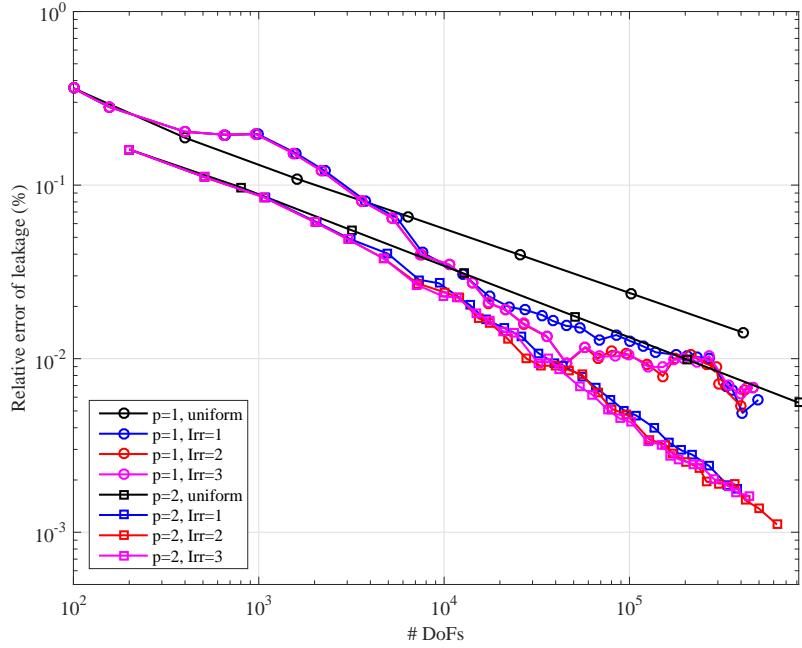


Figure 3.65: Convergence history for the searchlight problem using the maximum entropy basis functions.

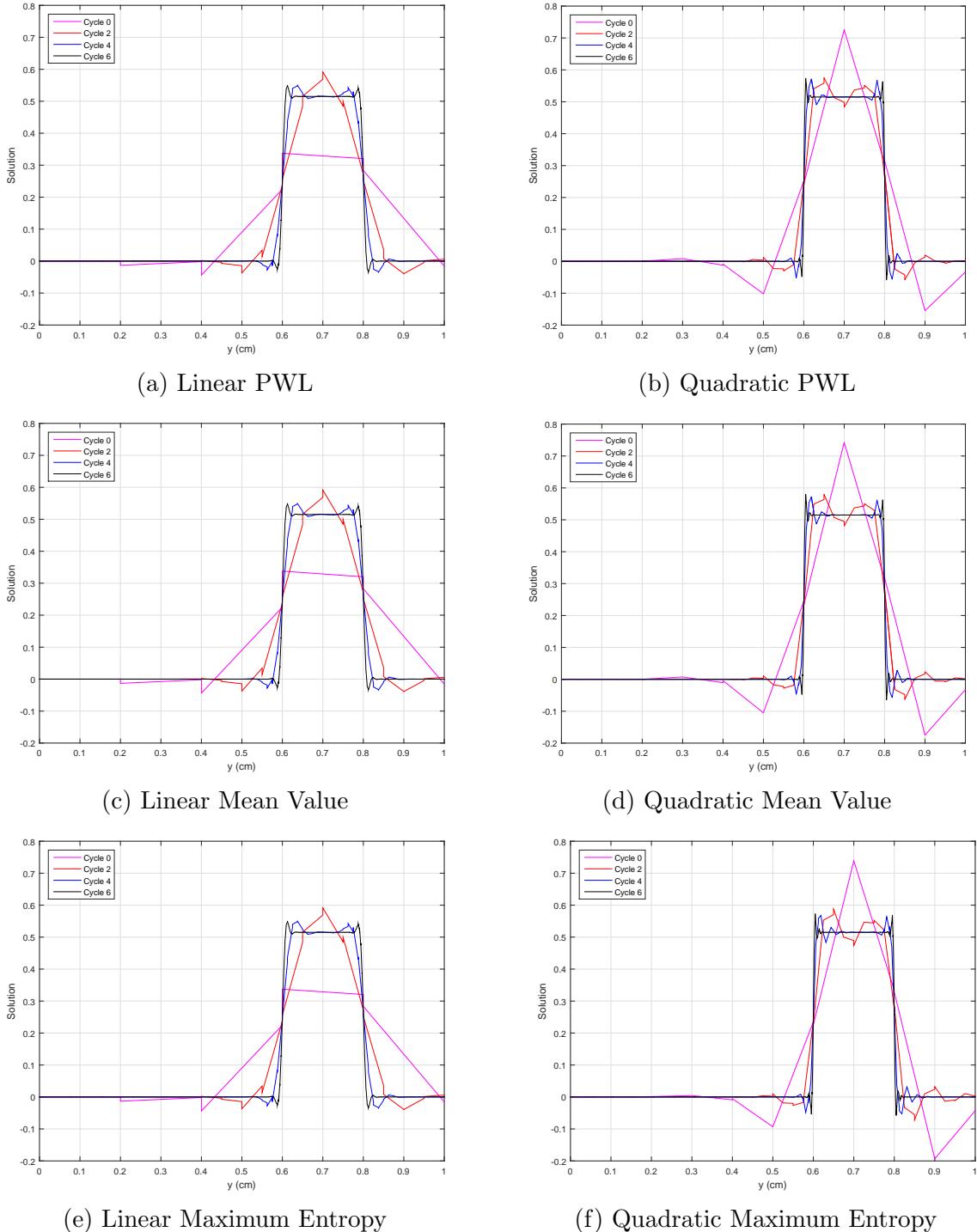


Figure 3.66: Exiting angular flux on the right boundary with uniform refinement.

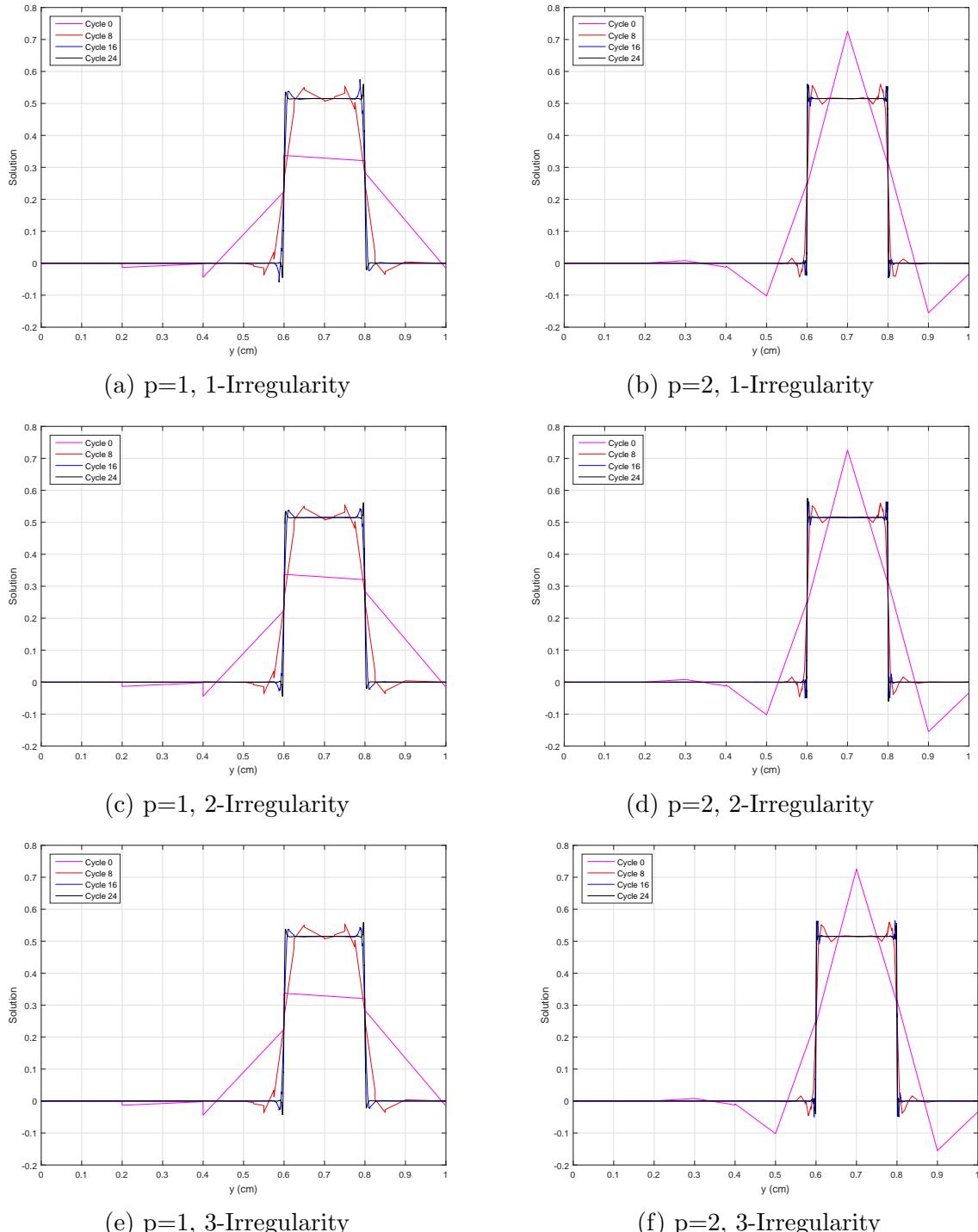


Figure 3.67: Exiting angular flux on the right boundary with AMR and the PWL basis functions with different mesh irregularities.

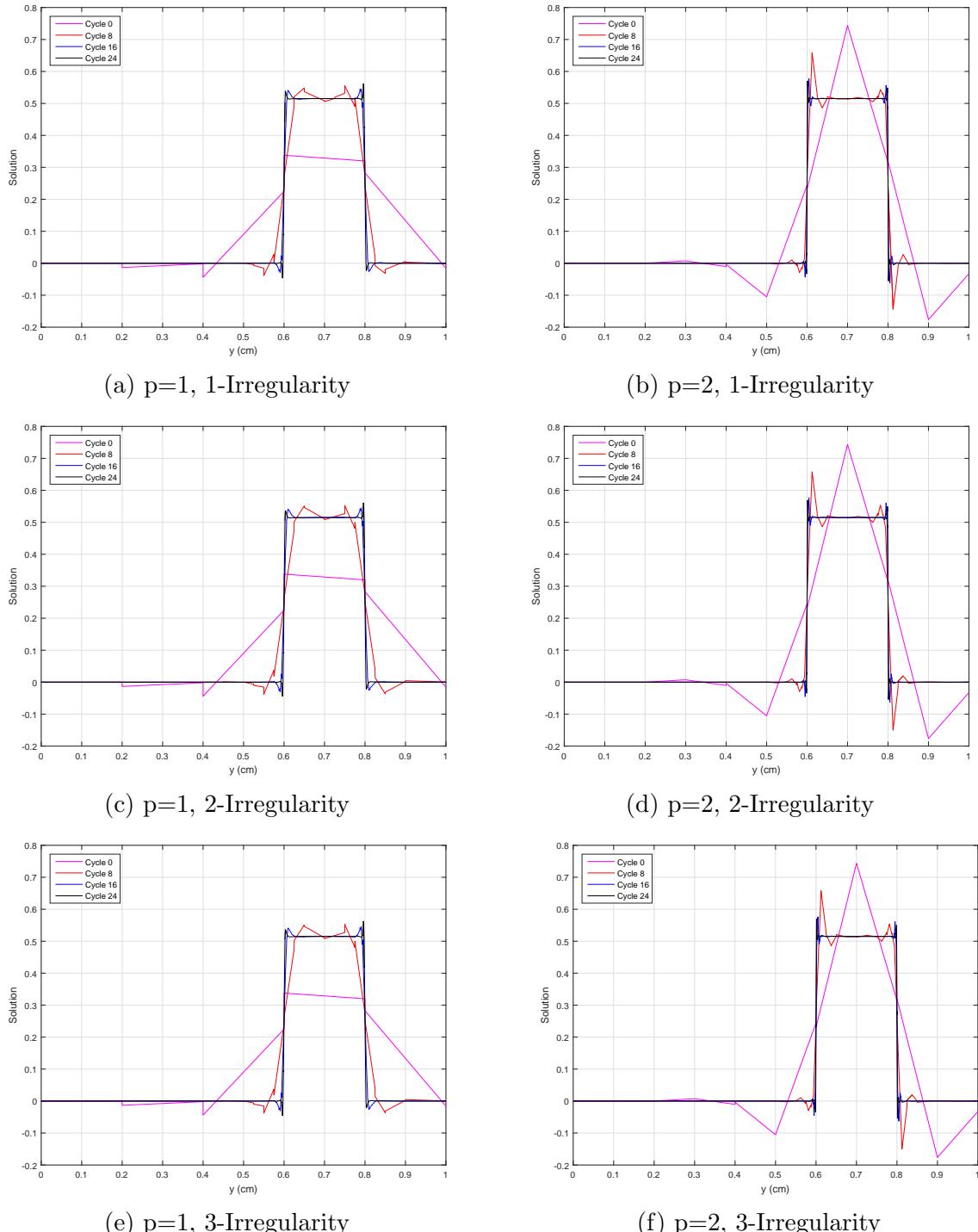


Figure 3.68: Exiting angular flux on the right boundary with AMR and the mean value basis functions with different mesh irregularities.

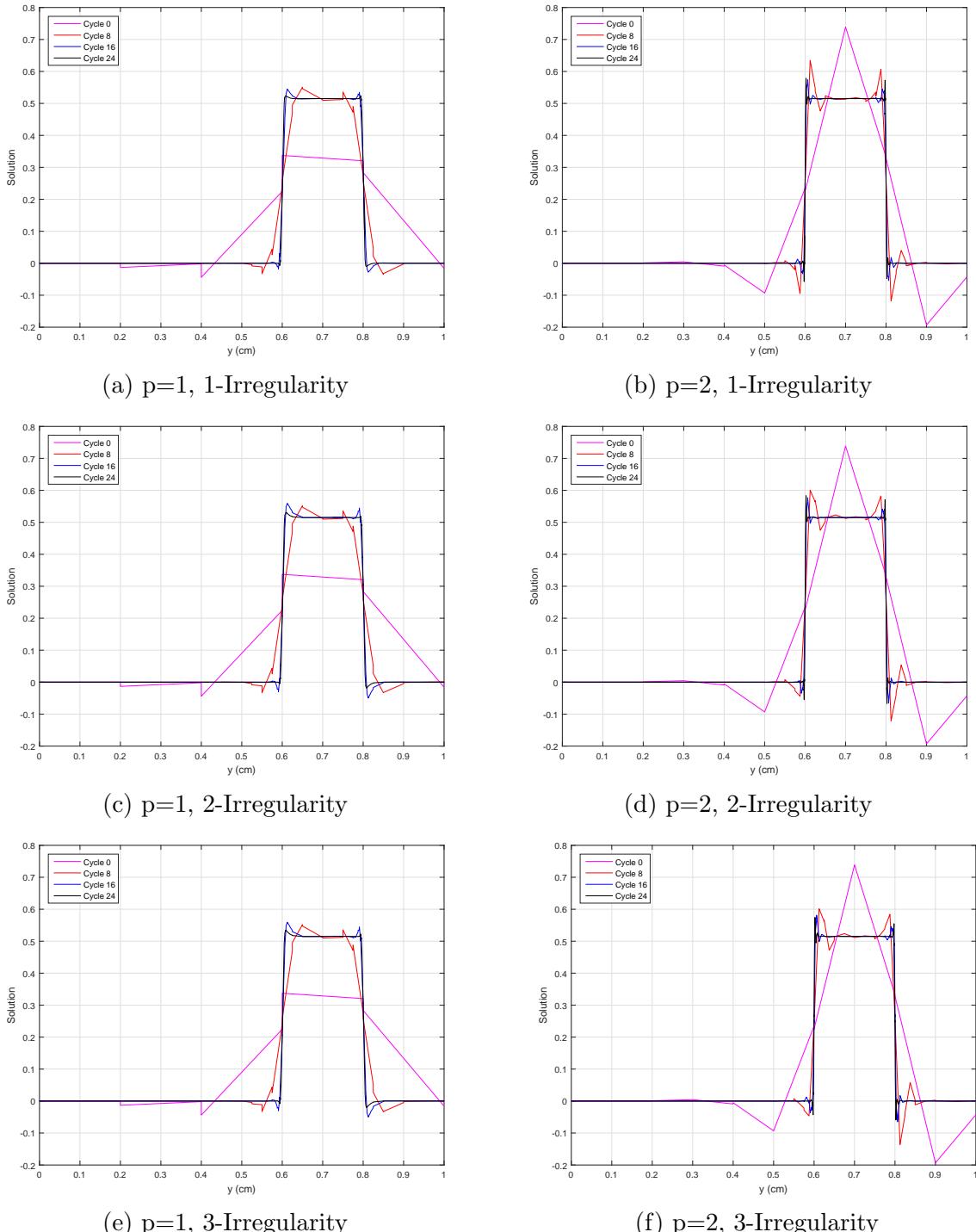


Figure 3.69: Exiting angular flux on the right boundary with AMR and the maximum entropy basis functions with different mesh irregularities.

Wachspress rational functions, the PWL coordinates, the mean value coordinates, and the maximum entropy coordinates. The Wachspress and the PWL coordinates had been previously utilized for DGFEM transport calculations, and we have extended the analysis to include the other two. Next, the procedure for converting these barycentric coordinates into the quadratic serendipity space of functions was given. For both the linear and quadratic coordinates, a simple quadrature rule for arbitrary polygons based on triangulation was used. We also provided some details of the 3D PWL coordinates that will be used in Chapter 4 for completeness.

Numerical results were obtained to demonstrate the completeness and convergence properties of these coordinates. The 2D linear and quadratic basis functions capture the thick diffusion limit, which is a necessary property for thermal radiative transport calculations. The linear and quadratic basis functions can capture exactly-linear and exactly-quadratic solution spaces, respectively. Next, a series of convergence studies were performed to test the convergence behavior of the basis functions under different conditions. Using MMS, the transport solutions converged at a rate of $p + 1$ under uniform mesh refinement, which is perfectly in alignment with FEM theory. AMR was also used on MMS problems to achieve more accurate solutions with the same $p+1$ convergence rates using less degrees of freedom. Finally, we concluded with some numerical problems involving purely absorbing media containing a solution discontinuity. For these problems, the solutions converge at a rate of $\min(p + 1, r)$ depending on the regularity of the alignment of the transport solution. If the spatial mesh is aligned with the discontinuity, then the $p + 1$ convergence are still observed. If the meshes are not aligned, then convergence rates imposed by the regularity ($r = 1/2$ or $3/2$) are observed for optically thin meshes. However, for meshes that are still optically thick convergence rates of $p + 1$ are observed in the pre-asymptotic region before being restricted by r as the mesh gets optically thin.

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

4.1 Introduction

In this chapter, we present the theory, implementation, and analysis of Diffusion Synthetic Acceleration (DSA) schemes for the DGFEM S_N transport equation compatible with arbitrary polytope meshes. The Modified Interior Penalty (MIP) form of the diffusion equation is used as the DSA discretization scheme [50, 51]. There are three specific topics that we investigate in this chapter. First, we analyze the effectiveness of DSA schemes with the linear and quadratic basis functions presented in Chapter 3. Second, we seek to analyze the efficacy of MIP DSA for within-group acceleration for massively-parallel transport calculations. We do this by extending the previous analysis of MIP DSA to 3D configurations and analyzing the scalability of the method's implementation out to $O(10^5)$ processors. Finally, we present a pair of novel schemes to accelerate the thermal neutron upscattering iterations that is compatible with massively-parallel transport sweeps. The method is fully-parallelizable and MIP is again used as the diffusion operator.

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 diffusion 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 are given in Section 4.5. The theoretical Fourier analysis tool is given in Section

4.6. Theoretical and numerical results are provided in Section 4.7 (including scaling results out to $O(10^5)$ processors), and we finish the chapter with some concluding remarks in Section 4.8

4.1.1 *Review of Diffusion Synthetic Acceleration 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 most 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 [42]. 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 such regimes in which convergence is prohibitively slow, additional steps should be taken to speed up, or accelerate, solution convergence [42]. The most-used methods to assist in solution convergence are often called synthetic acceleration techniques. These techniques were first introduced by Kopp [138] and Lebedev [139, 140, 141, 142, 143, 144, 145] 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 [146]. 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

[147]. 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) [43, 44, 148]. 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 [45, 46]. However, Adams and Martin then showed that partially-consistent diffusion discretizations could effectively accelerate DFEM discretizations of the neutron transport equation [48]. 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 [47]. 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 [149], the fully consistent DSA (FCDSA) [47], and the partially consistent MIP DSA [50, 150, 51].

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 [50, 150]. Turcksin and Ragusa then extended the work to arbitrary polygonal meshes [51]. The MIP diffusion operator is symmetric positive definite (SPD), and was shown to be efficiently solved with preconditioned conjugate gradient (PCG) and advanced preconditioners such as algebraic multi-grid (AMG) [51].

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,

$$(\mathbf{L} - \mathbf{S}) \Psi = \mathbf{Q}, \quad (4.1)$$

where \mathbf{L} and \mathbf{S} are the loss and scattering operators, respectively, 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 $(\mathbf{L} - \mathbf{S})$ directly, then Ψ could be directly computed:

$$\Psi = (\mathbf{L} - \mathbf{S})^{-1} \mathbf{Q}. \quad (4.2)$$

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

To compute Ψ , Source Iteration is employed to yield

$$\mathbf{L}\Psi^{(k+1)} = \mathbf{S}\Psi^{(k)} + \mathbf{Q}, \quad (4.3)$$

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

$$\Psi^{(k+1)} = \mathbf{L}^{-1}\mathbf{S}\Psi^{(k)} + \mathbf{L}^{-1}\mathbf{Q}. \quad (4.4)$$

The inversion of \mathbf{L}^{-1} is a transport sweep. For brevity, we define a new operator

$\mathbf{C} = \mathbf{L}^{-1}\mathbf{S}$ 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,

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

which can also be written as the following:

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

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:

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

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

$$\Psi = \Psi^{(k)} + (\mathbf{I} + \mathbf{C} + \mathbf{C}^2 + \dots) \mathbf{C}r^{(k)}. \quad (4.8)$$

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:

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

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

$$\Psi = \Psi^{(k)} + (\mathbf{L} - \mathbf{S})^{-1} \mathbf{S} r^{(k)}. \quad (4.10)$$

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 $(\mathbf{L} - \mathbf{S})$ 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 $(\mathbf{L} - \mathbf{S})$ operator. Instead, suppose that we could define a low-order operator, \mathbf{W} , that closely approximates $(\mathbf{L} - \mathbf{S})$ but it is easily invertible. If \mathbf{W} efficiently approximates the slowest converging error modes of $(\mathbf{L} - \mathbf{S})$, 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

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

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,

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

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:

$$\Psi^{(k+1)} = [\mathbf{I} - \mathbf{W}^{-1}(\mathbf{L} - \mathbf{S})] \mathbf{C}\Psi^{(k)} + (\mathbf{I} + \mathbf{W}^{-1}\mathbf{S}) \mathbf{L}^{-1}\mathbf{Q}. \quad (4.13)$$

Observe in Eq. (4.13) that as \mathbf{W} more closely approximates $(\mathbf{L} - \mathbf{S})$, the operator $\mathbf{W}^{-1}(\mathbf{L} - \mathbf{S})$ 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 $(\mathbf{L} - \mathbf{S})$ 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 as 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,

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

where we do not include the spatial parameter for clarity. If D is the diameter of the

problem domain, then Eq. (4.14) can slowly converge if the problem is optically thick ($\sigma_t D \gg 1$) 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, $(k + 1/2)$,

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

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. (4.14). This forms the following error equation,

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

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

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

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

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

and

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

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

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

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

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

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 [42]. 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

$$\vec{\nabla} \cdot \vec{J} + \sigma_a \phi = Q, \quad (4.22)$$

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

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

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

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

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

$$\vec{\Omega} \vec{\Omega} = \begin{bmatrix} \Omega_x \Omega_x & \Omega_x \Omega_y & \Omega_x \Omega_z \\ \Omega_y \Omega_x & \Omega_y \Omega_y & \Omega_y \Omega_z \\ \Omega_z \Omega_x & \Omega_z \Omega_y & \Omega_z \Omega_z \end{bmatrix}. \quad (4.25)$$

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

$$\psi = \frac{1}{4\pi} [\phi + 3\vec{\Omega} \cdot \vec{J}]. \quad (4.26)$$

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

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

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:

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

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,

$$-\vec{\nabla} \cdot D \vec{\nabla} \phi + \sigma_a \phi = Q, \quad (4.29)$$

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:

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

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:

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

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-discretized, 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,

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

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, Σ 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

$$M_{m,p,n} \equiv \frac{2p+1}{4\pi} Y_{p,n}(\vec{\Omega}_m), \quad (4.33)$$

and

$$D_{m,p,n} \equiv w_m Y_{p,n}(\vec{\Omega}_m), \quad (4.34)$$

respectively. These operators perform mappings between the discrete angular fluxes and the angular moments. This means that the operation \mathbf{DL}^{-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:

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

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

$$\Phi^{(k+1/2)} = \mathbf{DL}^{-1}\mathbf{M}\Sigma\Phi^{(k)} + \mathbf{DL}^{-1}\mathbf{Q}. \quad (4.36)$$

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

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

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,

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

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

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

where \mathbf{P} and \mathbf{X} are the prolongation and restriction operators, respectively. The restriction operator, \mathbf{X} , acts by limiting the contributions to the low-order residual to the 0th and maybe 1st moments (filtering out the higher moments). Then once the diffusion correction, $\delta\Phi^{(k+1/2)}$, is calculated, the prolongation operator, \mathbf{P} , 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:

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

Equation (4.40) provides the accelerated iteration scheme where the matrix operator $[\mathbf{T} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\Sigma(\mathbf{T} - \mathbf{I})]$ is the corresponding iteration matrix for a preconditioned transport sweep. 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 knowledge 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 $\Phi^{(k)}$ to the final line of Eq. (4.40) to form the following equivalent iteration scheme,

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

Next, we replace the iteration terms of $\Phi^{(k)}$ and $\Phi^{(k+1)}$ with Φ 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,

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

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

$$(\mathbf{I} - \mathbf{T}) \Phi = \mathbf{b}. \quad (4.43)$$

This means that the DSA scheme is exactly a preconditioned Richardson iterative scheme where the operator $(\mathbf{I} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\Sigma)$ 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 be 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:

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

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 $\mathbf{b} = \mathbf{DL}^{-1}\mathbf{Q}$, which is computed by performing a single transport sweep on \mathbf{Q} . This means that the accelerated Krylov right-hand-side vector, $\mathbf{b} + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\Sigma\mathbf{b}$, 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 $(\Phi^{(k+1/2)} - \Phi^{(k)})$ 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 single energy group. 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 that can be employed. In particular, it is difficult to converge the scattering source for transport problems that are dominated by thermal neutron upscattering.

We next present four different DSA strategies to accelerate the neutron upscattering that we will analyze for this work. The Two-Grid (TG) acceleration method, originally proposed in 1993, is presented in Section 4.2.3.1 [151]. Next, we provide a Modified Two-Grid (MTG) acceleration scheme in Section 4.2.3.2 that is a variant of the work presented in [152]. Following the presentation of those two schemes, we then present our novel scheme to perform parallel thermal neutron upscattering acceleration. Section 4.2.3.3 gives the Multigroup Jacobi Acceleration (MJA) method which is a two-grid-like method similar to TG and MTG but executed in parallel. Finally in Section 4.2.3.4, we propose a variant to the MJA method by performing an additional acceleration step to give a better estimate of the within-group scattering error. This additional step remains fully-parallelizable.

4.2.3.1 Standard Two-Grid Acceleration

The first thermal neutron upscatter acceleration methodology that we will investigate is the standard Two-Grid acceleration scheme devised by Adams and Morel [151]. 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 all of the thermal groups and converge the inner iterations;
2. Factorize the multigroup iteration error into a spatial component and a spectral component;
3. Perform a spectral collapse of the transport iteration error into a 1-group diffusion equation;

4. Solve the 1-group diffusion equation for the spatial variation of the transport iteration error;
5. Interpolate the diffusion multigroup error back onto the transport solution.

We now provide full 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,

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

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

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

and

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

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:

$$[\mathbf{I} - \mathbf{DL}^{-1}\mathbf{M}(\mathbf{S}_L + \mathbf{S}_D)] \Phi^{(k+1/2)} = \mathbf{DL}^{-1}\mathbf{MS}_U\Phi^{(k)} + \mathbf{DL}^{-1}\mathbf{Q}. \quad (4.48)$$

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

$$\Phi^{(k+1/2)} = [\mathbf{I} - \mathbf{DL}^{-1}\mathbf{M}(\mathbf{S}_L + \mathbf{S}_D)]^{-1} \mathbf{DL}^{-1}\mathbf{MS}_U\Phi^{(k)} + \mathbf{b}, \quad (4.49)$$

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

$$\mathbf{b} = [\mathbf{I} - \mathbf{DL}^{-1}\mathbf{M}(\mathbf{S}_L + \mathbf{S}_D)]^{-1} \mathbf{DL}^{-1}\mathbf{Q}. \quad (4.50)$$

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 $(k + 1/2)$,

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

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

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

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

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

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

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

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 multigroup 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,

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

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

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

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 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,

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

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. It is material dependent and can be computed once all the problem materials are known. This eigenvalue problem can be succinctly written as,

$$(\mathbf{T} - \mathbf{S}_{L,0} - \mathbf{S}_{D,0})^{-1} \mathbf{S}_{U,0} \xi = \rho \xi, \quad (4.58)$$

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. (4.57) into Eq. (4.55) and summing over energy groups gives

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

where the energy-averaged terms are

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

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,

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

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

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

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:

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

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 additive interpolation of the diffusive error. We now go through these steps using compact operator notation to arrive at a single matrix form for the TG accelerated transport iterations. First, we express the full phase-space update equation as

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

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

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

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

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

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

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

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 $[\mathbf{FMS_U} + \mathbf{PA}^{-1} \mathbf{XS_U} (\mathbf{FMS_U} - \mathbf{I})]$. 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 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 [152], 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 keep using 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,

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

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:

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

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,

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

and

$$\mathbf{b} = [\mathbf{I} - \mathbf{DL}^{-1}\mathbf{MS_L}]^{-1} \mathbf{DL}^{-1}\mathbf{Q}. \quad (4.71)$$

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,

$$(\mathbf{T} - \mathbf{S}_{L,0})^{-1} (\mathbf{S}_{D,0} + \mathbf{S}_{U,0}) \xi = \rho \xi, \quad (4.72)$$

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

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

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

$$\mathbf{A}\epsilon^{(k+1/2)} = \mathbf{X}(\mathbf{S}_D + \mathbf{S}_U) \left(\boldsymbol{\Phi}^{(k+1/2)} - \boldsymbol{\Phi}^{(k)} \right). \quad (4.74)$$

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

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

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

$$\begin{aligned} \boldsymbol{\Phi}^{(k+1)} &= \mathbf{FM}(\mathbf{S}_D + \mathbf{S}_U)\boldsymbol{\Phi}^{(k)} + \mathbf{PA}^{-1}\mathbf{X}(\mathbf{S}_D + \mathbf{S}_U) \left(\boldsymbol{\Phi}^{(k+1/2)} - \boldsymbol{\Phi}^{(k)} \right) + \mathbf{b} \\ &= \mathbf{FM}(\mathbf{S}_D + \mathbf{S}_U)\boldsymbol{\Phi}^{(k)} + \mathbf{PA}^{-1}\mathbf{X}(\mathbf{S}_D + \mathbf{S}_U) \left(\mathbf{FM}(\mathbf{S}_D + \mathbf{S}_U) - \mathbf{I} \right) \boldsymbol{\Phi}^{(k)} \\ &\quad + (\mathbf{PA}^{-1}\mathbf{X}(\mathbf{S}_D + \mathbf{S}_U) + \mathbf{I})\mathbf{b} \\ &= \left[\mathbf{FM}(\mathbf{S}_D + \mathbf{S}_U) + \mathbf{PA}^{-1}\mathbf{X}(\mathbf{S}_D + \mathbf{S}_U) \left(\mathbf{FM}(\mathbf{S}_D + \mathbf{S}_U) - \mathbf{I} \right) \right] \boldsymbol{\Phi}^{(k)} \\ &\quad + (\mathbf{PA}^{-1}\mathbf{X}(\mathbf{S}_D + \mathbf{S}_U) + \mathbf{I})\mathbf{b} \end{aligned}, \quad (4.76)$$

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 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 one transport sweep yields the following iteration scheme:

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

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

$$\mathbf{L}\Psi^{(k+1/2)} = \mathbf{M}\mathbf{S}\Phi^{(k)} + \mathbf{Q}, \quad (4.78)$$

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

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

and

$$\mathbf{b} = \mathbf{DL}^{-1}\mathbf{Q}, \quad (4.80)$$

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,

$$\mathbf{T}^{-1} (\mathbf{S}_{L,0} + \mathbf{S}_{D,0} + \mathbf{S}_{U,0}) \xi = \rho \xi, \quad (4.81)$$

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,

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

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

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

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

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

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 Multigroup Jacobi with Inner Acceleration

Up to this point, we have presented a two-grid-like methodology in the MJA method that is amenable to parallel computation. With modern transport codes that can solve energy groups simultaneously, the grind times for these multi-energy group set transport sweeps are significantly lower than single group sweeps (*i.e.*, TG and MTG methods). Therefore, it is possible to have “worse” theoretical convergence properties for MJA (compared to TG) but still converge to a solution faster based on wallclock time (we demonstrate this in Sections 4.7.4.2 and 4.7.4.3). However, like the

MTG method, MJA does not converge the inner iterations. For graphite with $O(10^1)$ - $O(10^2)$ thermal groups, there can still be several within-group scattering ratios above 0.9. Therefore, there will be a large contribution to the iteration error from the non-convergence of the within-group iterations. We quickly note that as the number of thermal groups approaches infinity, this behavior will disappear. Because of this behavior of the scattering ratios, we then have recourse to provide a better estimate of the within-group scattering error before performing the spectrally-collapsed, 1-group diffusion correction. However, we still seek to continue the parallel nature of the MJA algorithm. Therefore, we propose an acceleration step before the spectrally-collapsed correction where we perform G 1-group DSA calculations for the within-group error based on the methodology proposed in Section 4.2.2. These G diffusion calculations are independent and thus parallelizable. We choose to call this new scheme the Multigroup Jacobi with Inner Acceleration (MJIA) method and summarize it as follows:

1. Solve for the fast energy groups.
2. Build scattering source for the thermal groups.
3. Perform one transport sweep of all the thermal groups simultaneously in an identical manner to the MJA method.
4. Perform G distinct and parallelizable 1-group diffusion calculations for the within-group iteration error. The error residuals for these calculations only contain the $g \rightarrow g$ scattering term.
5. Perform a single, energy-collapsed diffusion calculation for the group-to-group iteration error (this step is almost identical to MJA).
6. Check for convergence. If we are not converged, then return to step 2.

With this iterative scheme detailed, there only remains a discussion on the differences in the diffusion steps compared to the MJA method. Because there are now two diffusion steps in the MJIA method (we can view it as a V-cycle with 2 levels), we will use a modified form for the update equation, where the acceleration steps are at the $(k + 1/3)$ and $(k + 2/3)$ iterate steps. For each thermal group, g , we express this two-level update equation for the scalar flux as

$$\begin{aligned}\phi_{g,0}^{(k+2/3)} &= \phi_{g,0}^{(k+1/3)} + \Delta\phi_{g,0}^{(k+1/3)} \\ \phi_{g,0}^{(k+1)} &= \phi_{g,0}^{(k+2/3)} + \delta\phi_{g,0}^{(k+2/3)},\end{aligned}\tag{4.85}$$

where $\phi_{g,0}^{(k+1/3)}$ corresponds to the solution after the transport sweep, $\Delta\phi_{g,0}^{(k+1/3)}$ corresponds to the error from the G separate within-group diffusion solves, and $\delta\phi_{g,0}^{(k+2/3)}$ corresponds to the error from the single, energy-collapsed diffusion solve. In an operator format, this iteration scheme can be combined and given by

$$\Phi^{(k+1)} = \Phi^{(k+1/3)} + \Delta\Phi^{(k+1/3)} + \delta\Phi^{(k+2/3)}.\tag{4.86}$$

We now give the form for the within-group diffusion solves. For each of the G thermal energy groups, we can write

$$-\vec{\nabla} \cdot D_g \vec{\nabla} \Delta\phi_g^{(k+1/3)} + \sigma_{a,g} \Delta\phi_g^{(k+1/3)} = \sigma_{s,0}^{gg} \left(\phi_g^{(k+1/3)} - \phi_g^{(k)} \right), \quad g = 1, \dots, G\tag{4.87}$$

From Eq. (4.87), we can clearly see the independent nature of these G diffusion solves. Therefore, each of these within-group error correction terms can be calculated independently. Equation (4.87) can then be written into operator form:

$$\mathbf{B}\Delta\Phi^{(k+1/3)} = \mathbf{X}_\Delta \mathbf{S}_D \left(\Phi^{(k+1/3)} - \Phi^{(k)} \right), \quad (4.88)$$

where \mathbf{B} is a block diagonal operator containing the G left-hand-side diffusion operators of Eq. (4.87) and \mathbf{X}_Δ is the restriction operator which restricts the flux contribution into the residual to only the 0th-order moment. We then solve for the $(k + 1/3)$ correction:

$$\Delta\Phi^{(k+1/3)} = \mathbf{P}_\Delta \mathbf{B}^{-1} \mathbf{X}_\Delta \mathbf{S}_D \left(\Phi^{(k+1/3)} - \Phi^{(k)} \right), \quad (4.89)$$

where the prolongation operator, \mathbf{P}_Δ , maps the error contribution back onto just the 0th-order flux moments for each of the G thermal groups.

After the G independent within-group error diffusion solves and their corresponding updates, MJIA calls for a spectrally-collapsed, 1-group diffusion solve for the across-group thermal iteration error. This error equation is similar to MJA except it does not include the within-group scattering. It also operates on the residual between the $(k + 2/3)$ and (k) iterates. We can write this 1-group diffusion equation for the spatial error, $\epsilon^{(k+2/3)}$, as

$$-\vec{\nabla} \cdot \langle D \rangle \vec{\nabla} \epsilon^{(k+2/3)} + \langle \sigma \rangle \epsilon^{(k+2/3)} = \langle R^{(k+2/3)} \rangle, \quad (4.90)$$

where $\langle D \rangle$ and $\langle \sigma \rangle$ can still be calculated from Eq. (4.60). The only differences in this equation compared to MJA's are the spectral shape, ξ , and the right-hand-side residual, $\langle R^{(k+2/3)} \rangle$. The spectral shape and the right-hand-side residual can be calculated from

$$(\mathbf{T} - \mathbf{S}_{D,0})^{-1} (\mathbf{S}_{L,0} + \mathbf{S}_{U,0}) \xi = \rho \xi, \quad (4.91)$$

and

$$\langle R^{(k+2/3)} \rangle = \sum_{g=1}^G \sum_{g' \neq g} \sigma_{s,0}^{gg'} \left(\phi_{g',0}^{(k+2/3)} - \phi_{g',0}^{(k)} \right), \quad (4.92)$$

respectively. We can see from Eqs. (4.91) and (4.92) that we do not include the within-group scattering in the residual. This is reflected in the calculation of the spectral shape in Eq. (4.91) where we have assumed that the G within-group diffusion error corrections have sufficiently accounted for the within-group error. We can then give the operator form of Eq. (4.90) as

$$\mathbf{A}\epsilon^{(k+2/3)} = \mathbf{X}_\delta (\mathbf{S}_L + \mathbf{S}_U) \left(\Phi^{(k+2/3)} - \Phi^{(k)} \right), \quad (4.93)$$

where \mathbf{X}_δ is the restriction operator that sums up only the 0th-order flux moments. We can then express the correction at the $(k + 2/3)$ iterate as

$$\delta\Phi^{(k+2/3)} = \mathbf{P}_\delta \mathbf{A}^{-1} \mathbf{X}_\delta (\mathbf{S}_L + \mathbf{S}_U) \left(\Phi^{(k+2/3)} - \Phi^{(k)} \right), \quad (4.94)$$

where \mathbf{P}_δ is the prolongation operator that maps the spatial diffusion error back onto the 0th-order moment of the transport solution. The contribution of the error correction into each energy group is given by: $\delta\phi_{g,0}^{(k+2/3)} = \xi_g \epsilon^{(k+2/3)}$. If we define $\mathbf{F} \equiv \mathbf{D}\mathbf{L}^{-1}$ and $\mathbf{b} = \mathbf{D}\mathbf{L}^{-1}\mathbf{Q}$ and insert the terms of Eqs. (4.89) and (4.94) into Eq. (4.86), then we obtain the complete operator form of the MJIA scheme:

$$\begin{aligned}
\Phi^{(k+1)} &= \Phi^{(k+1/3)} + \Delta\Phi^{(k+1/3)} + \delta\Phi^{(k+2/3)} \\
&= \mathbf{FMS}\Phi^{(k)} + \mathbf{b} + \mathbf{P}_\Delta\mathbf{B}^{-1}\mathbf{X}_\Delta\mathbf{S}_D \left(\Phi^{(k+1/3)} - \Phi^{(k)} \right) + \delta\Phi^{(k+2/3)} \\
&= \mathbf{FMS}\Phi^{(k)} + \mathbf{b} + \mathbf{P}_\Delta\mathbf{B}^{-1}\mathbf{X}_\Delta\mathbf{S}_D \left(\mathbf{FMS}\Phi^{(k)} + \mathbf{b} - \Phi^{(k)} \right) + \delta\Phi^{(k+2/3)} \\
&= [\mathbf{FMS} + \mathbf{P}_\Delta\mathbf{B}^{-1}\mathbf{X}_\Delta\mathbf{S}_D (\mathbf{FMS} - \mathbf{I})] \Phi^{(k)} + (\mathbf{I} + \mathbf{P}_\Delta\mathbf{B}^{-1}\mathbf{X}_\Delta\mathbf{S}_D) \mathbf{b} \\
&\quad + \delta\Phi^{(k+2/3)} \\
&= [\mathbf{FMS} + \mathbf{P}_\Delta\mathbf{B}^{-1}\mathbf{X}_\Delta\mathbf{S}_D (\mathbf{FMS} - \mathbf{I})] \Phi^{(k)} + (\mathbf{I} + \mathbf{P}_\Delta\mathbf{B}^{-1}\mathbf{X}_\Delta\mathbf{S}_D) \mathbf{b} \\
&\quad + \mathbf{P}_\delta\mathbf{A}^{-1}\mathbf{X}_\delta (\mathbf{S}_L + \mathbf{S}_U) \left(\Phi^{(k+2/3)} - \Phi^{(k)} \right) \\
&= [\mathbf{FMS} + \mathbf{P}_\Delta\mathbf{B}^{-1}\mathbf{X}_\Delta\mathbf{S}_D (\mathbf{FMS} - \mathbf{I})] \Phi^{(k)} + (\mathbf{I} + \mathbf{P}_\Delta\mathbf{B}^{-1}\mathbf{X}_\Delta\mathbf{S}_D) \mathbf{b} \\
&\quad + \mathbf{P}_\delta\mathbf{A}^{-1}\mathbf{X}_\delta (\mathbf{S}_L + \mathbf{S}_U) \left(\Phi^{(k+1/3)} + \Delta\Phi^{(k+1/3)} - \Phi^{(k)} \right) \\
&= [\mathbf{FMS} + \mathbf{P}_\Delta\mathbf{B}^{-1}\mathbf{X}_\Delta\mathbf{S}_D (\mathbf{FMS} - \mathbf{I})] \Phi^{(k)} + (\mathbf{I} + \mathbf{P}_\Delta\mathbf{B}^{-1}\mathbf{X}_\Delta\mathbf{S}_D) \mathbf{b} \\
&\quad + \mathbf{P}_\delta\mathbf{A}^{-1}\mathbf{X}_\delta (\mathbf{S}_L + \mathbf{S}_U) \left[(\mathbf{I} + \mathbf{P}_\Delta\mathbf{B}^{-1}\mathbf{X}_\Delta\mathbf{S}_D) (\mathbf{FMS} - \mathbf{I}) \right] \Phi^{(k)} \\
&\quad + \mathbf{P}_\delta\mathbf{A}^{-1}\mathbf{X}_\delta (\mathbf{S}_L + \mathbf{S}_U) (\mathbf{I} + \mathbf{P}_\Delta\mathbf{B}^{-1}\mathbf{X}_\Delta\mathbf{S}_D) \mathbf{b} \\
&= \left[\mathbf{I} + \left[\mathbf{I} + \mathbf{P}_\delta\mathbf{A}^{-1}\mathbf{X}_\delta (\mathbf{S}_L + \mathbf{S}_U) \right] (\mathbf{I} + \mathbf{P}_\Delta\mathbf{B}^{-1}\mathbf{X}_\Delta\mathbf{S}_D) (\mathbf{FMS} - \mathbf{I}) \right] \Phi^{(k)} \\
&\quad + \left[\mathbf{I} + \mathbf{P}_\delta\mathbf{A}^{-1}\mathbf{X}_\delta (\mathbf{S}_L + \mathbf{S}_U) \right] (\mathbf{I} + \mathbf{P}_\Delta\mathbf{B}^{-1}\mathbf{X}_\Delta\mathbf{S}_D) \mathbf{b}
\end{aligned} \tag{4.95}$$

The term $\left[\mathbf{I} + \left[\mathbf{I} + \mathbf{P}_\delta\mathbf{A}^{-1}\mathbf{X}_\delta (\mathbf{S}_L + \mathbf{S}_U) \right] (\mathbf{I} + \mathbf{P}_\Delta\mathbf{B}^{-1}\mathbf{X}_\Delta\mathbf{S}_D) (\mathbf{FMS} - \mathbf{I}) \right]$ is the iteration matrix for the MJIA method.

4.2.3.5 Summary of the Thermal Neutron Upscattering Acceleration Methods

In this work, we have defined four different methodologies to accelerate thermal neutron upscattering: the Two-Grid (TG) method, the Modified Two-Grid (MTG) method, the Multigroup Jacobi Acceleration (MJA) method, and the Multigroup Jacobi with Inner Acceleration (MJIA) 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 four 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 (we just perform a single sweep in this work). The MJA and MJIA methods are different from the other two in that the thermal groups are iterated upon in parallel. All thermal groups are solved simultaneously in the transport sweep. This means that neither method converges the within-group scattering for any of the thermal groups. To correct for this, we perform an additional acceleration step in the MJIA method to provide a better estimate of the within-group scattering error. Before the spectrally-collapsed low-order calculation, we perform independent (and parallelizable) 1-group DSA calculations for each thermal group. In these 1-group calculations, we only accelerate the within-group scattering.

As it was stated, each of these acceleration methods contains a spectrally-collapsed, 1-group diffusion operation. For materials with significant across-group scattering and minimal absorption, this diffusion equation eliminates the error mode corresponding to the thermal Maxwellian. The spectral shape to perform the energy collapse is calculated from the eigenproblem corresponding to the infinite medium iteration equations of the different acceleration methods. We give the appropriate eigenproblems for each of the acceleration methods in Table 4.2.

We conclude by again summarizing the iteration matrices of each of these acceleration methods. For the TG, MTG, and MJA methods, we can define a general expression for the full phase-space solution at each iteration by the following,

Table 4.1: Eigenproblems to compute the spectral shape of each upscatter acceleration method.

Method	Eigenproblem
Two-Grid	$(\mathbf{T} - \mathbf{S}_{L,0} - \mathbf{S}_{D,0})^{-1} \mathbf{S}_{U,0} \xi = \rho \xi$
Modified Two-Grid	$(\mathbf{T} - \mathbf{S}_{L,0})^{-1} (\mathbf{S}_{D,0} + \mathbf{S}_{U,0}) \xi = \rho \xi$
Multigroup Jacobi	$\mathbf{T}^{-1} (\mathbf{S}_{L,0} + \mathbf{S}_{D,0} + \mathbf{S}_{U,0}) \xi = \rho \xi$
Multigroup Jacobi with Inner	$(\mathbf{T} - \mathbf{S}_{D,0})^{-1} (\mathbf{S}_{L,0} + \mathbf{S}_{U,0}) \xi = \rho \xi$

$$\begin{aligned} \Phi^{(k+1)} &= [\mathbf{F}\mathbf{M}\Sigma + \mathbf{P}\mathbf{A}^{-1}\mathbf{X}\Sigma(\mathbf{F}\mathbf{M}\Sigma - \mathbf{I})] \Phi^{(k)} \\ &\quad + (\mathbf{P}\mathbf{A}^{-1}\mathbf{X}\Sigma + \mathbf{I}) \mathbf{F}\mathbf{D}\mathbf{L}^{-1}\mathbf{Q} \end{aligned}, \quad (4.96)$$

where the differences lie in the \mathbf{F} and Σ terms. These terms are given in Table 4.2, and we can clearly see that the differences in the schemes arise from the ordering of the scattering operators. The scattering operators are split based on which portion of the scattering matrix is converged at each iteration. The MJIA method has a different form than the other three methods because of its two-level nature. Through the careful algebraic manipulation provided in Section 4.2.3.4, we can write the iteration matrix of the MJIA method as

$$\begin{aligned} \Phi^{(k+1)} &= \left[\mathbf{I} + [\mathbf{I} + \mathbf{P}_\delta \mathbf{A}^{-1} \mathbf{X}_\delta (\mathbf{S}_L + \mathbf{S}_U)] (\mathbf{I} + \mathbf{P}_\Delta \mathbf{B}^{-1} \mathbf{X}_\Delta \mathbf{S}_D) (\mathbf{F}\mathbf{M}\mathbf{S} - \mathbf{I}) \right] \Phi^{(k)} \\ &\quad + [\mathbf{I} + \mathbf{P}_\delta \mathbf{A}^{-1} \mathbf{X}_\delta (\mathbf{S}_L + \mathbf{S}_U)] (\mathbf{I} + \mathbf{P}_\Delta \mathbf{B}^{-1} \mathbf{X}_\Delta \mathbf{S}_D) \mathbf{b} \end{aligned}. \quad (4.97)$$

In Eq. (4.97), we note that there are two sets of restriction (\mathbf{X}_δ and \mathbf{X}_Δ) and prolongation (\mathbf{P}_δ and \mathbf{P}_Δ) operators. These operators act to have the diffusion error equations only provide contributions to the 0th-order flux moment.

Table 4.2: Iteration terms for the Two-Grid, Modified Two-Grid, and Multigroup Jacobi Acceleration methods.

Method	\mathbf{F}	Σ
Two-Grid	$[\mathbf{I} - \mathbf{DL}^{-1}\mathbf{M}(\mathbf{S}_L + \mathbf{S}_D)]^{-1}\mathbf{DL}^{-1}$	\mathbf{S}_U
Modified Two-Grid	$[\mathbf{I} - \mathbf{DL}^{-1}\mathbf{MS}_L]^{-1}\mathbf{DL}^{-1}$	$\mathbf{S}_D + \mathbf{S}_U$
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 [153, 154, 155].

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

$$-\vec{\nabla} \cdot D(\vec{r})\vec{\nabla}\Phi(\vec{r}) + \sigma\Phi(\vec{r}) = Q(\vec{r}), \quad \vec{r} \in \mathcal{D}, \quad (4.98)$$

with Dirichlet boundary conditions

$$\Phi(\vec{r}) = \Phi_0(\vec{r}), \quad \vec{r} \in \partial\mathcal{D}^d, \quad (4.99)$$

Neumann boundary conditions

$$-D\partial_n\Phi(\vec{r}) = J_0(\vec{r}), \quad \vec{r} \in \partial\mathcal{D}^n, \quad (4.100)$$

and Robin boundary conditions

$$\frac{1}{4}\Phi(\vec{r}) + \frac{D}{2}\partial_n\Phi(\vec{r}) = J^{inc}(\vec{r}), \quad \vec{r} \in \partial\mathcal{D}^r. \quad (4.101)$$

We then convert Eq. (4.98) into its weak formulation by left-multiplying it with the test function, b , and apply Gauss' theorem to the Laplacian term,

$$(D\vec{\nabla}b, \vec{\nabla}\Phi)_\mathcal{D} - \langle b, D\partial_n\Phi \rangle_{\partial\mathcal{D}} + (\sigma b, \Phi)_\mathcal{D} = (b, Q)_\mathcal{D} \quad (4.102)$$

If we were to use the CFEM form of Eq. (4.102), then there would be no further formulations required except on how to properly apply the boundary term: $\langle b, D\partial_n\Phi \rangle_{\partial\mathcal{D}}$. 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. (4.100) and (4.101) into Eq. (4.102). 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 [92]. However, a CFEM diffusion discretization is not consistent enough for use in DSA.

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. (4.102) to only contain the appropriate inner products for element K,

$$(D\vec{\nabla}b, \vec{\nabla}\Phi)_K - \langle b, D\partial_n\Phi \rangle_{\partial K} + (\sigma b, \Phi)_K = (b, Q)_K, \quad (4.103)$$

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 ($\partial K \setminus \partial \mathcal{D}$), 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:

$$\begin{aligned} & \left(D\vec{\nabla}b, \vec{\nabla}\Phi \right)_K + \left(\sigma b, \Phi \right)_K - \left\langle b, D\partial_n\Phi \right\rangle_{\partial K \setminus \partial \mathcal{D}} \\ & - \left\langle b, D\partial_n\Phi \right\rangle_{\partial K \cap \partial \mathcal{D}^d} - \left\langle b, D\partial_n\Phi \right\rangle_{\partial K \cap \partial \mathcal{D}^n} - \left\langle b, D\partial_n\Phi \right\rangle_{\partial K \cap \partial \mathcal{D}^r} \\ & = \left(b, Q \right)_K \end{aligned} \quad (4.104)$$

We can then immediately utilize the definitions of the outgoing currents from Eqs. (4.100) and (4.101), and add the Neumann and Robin boundary contributions from element K :

$$\begin{aligned} & \left(D\vec{\nabla}b, \vec{\nabla}\Phi \right)_K + \left(\sigma b, \Phi \right)_K \\ & - \left\langle b, D\partial_n\Phi \right\rangle_{\partial K \setminus \partial \mathcal{D}} - \left\langle b, D\partial_n\Phi \right\rangle_{\partial K \cap \partial \mathcal{D}^d} + \frac{1}{2} \left\langle b, \Phi \right\rangle_{\partial K \cap \partial \mathcal{D}^r} \\ & = \left(b, Q \right)_K - \left\langle b, J_0 \right\rangle_{\partial K \cap \partial \mathcal{D}^n} + 2 \left\langle b, J^{inc} \right\rangle_{\partial K \cap \partial \mathcal{D}^r}. \end{aligned} \quad (4.105)$$

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 [156], where the weakly-enforced Dirichlet conditions

now have the form,

$$\Phi(\vec{r}) + \frac{1}{\kappa} D\partial_n \Phi(\vec{r}) = \Phi_0(\vec{r}), \quad \vec{r} \in \partial\mathcal{D}^d, \quad (4.106)$$

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 [157], 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,

$$\begin{aligned} & \left(D\vec{\nabla} b, \vec{\nabla} \Phi \right)_{\mathcal{D}} - \left\langle D\partial_n b, \Phi \right\rangle_{\partial\mathcal{D}^d} - \left\langle b, D\partial_n \Phi \right\rangle_{\partial\mathcal{D}^d} \\ & + \left\langle \kappa b, (\Phi - \Phi_0) \right\rangle_{\partial\mathcal{D}^d} = - \left\langle D\partial_n b, \Phi_0 \right\rangle_{\partial\mathcal{D}^d} \end{aligned} \quad (4.107)$$

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 [158]. 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,

$$\left\langle \kappa [\![b]\!], [\![\Phi]\!] \right\rangle_{E_h^i} + \left\langle [\![b]\!], \{ \{ D\partial_n \Phi \} \} \right\rangle_{E_h^i} + \left\langle \{ \{ D\partial_n b \} \}, [\![\Phi]\!] \right\rangle_{E_h^i} = 0, \quad (4.108)$$

where κ is again a penalizing coefficient to ensure stability. The mean value and the jump of the terms on a face from Eq. (4.108) are defined as

$$\{ \{ \Phi \} \} \equiv \frac{\Phi^+ + \Phi^-}{2} \quad \text{and} \quad [\![\Phi]\!] \equiv \Phi^+ - \Phi^-, \quad (4.109)$$

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

$$\Phi^\pm \equiv \lim_{s \rightarrow 0^\pm} \Phi(\vec{r} + s\vec{n}), \quad (4.110)$$

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. (4.105). 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

$$a^{SIP}(b, \Phi) = b^{SIP}(b), \quad (4.111)$$

with the following bilinear matrix:

$$\begin{aligned} a^{SIP}(b, \Phi) &= \left(D\vec{\nabla}b, \vec{\nabla}\Phi \right)_D + \left(\sigma b, \Phi \right)_D + \frac{1}{2} \left\langle b, \Phi \right\rangle_{\partial\mathcal{D}^r} \\ &+ \left\langle \kappa_f^{SIP} [\![b]\!], [\![\Phi]\!] \right\rangle_{E_h^i} + \left\langle [\![b]\!], \{ \{ D\partial_n \Phi \} \} \right\rangle_{E_h^i} + \left\langle \{ \{ D\partial_n b \} \}, [\![\Phi]\!] \right\rangle_{E_h^i}, \\ &+ \left\langle \kappa_f^{SIP} b, \Phi \right\rangle_{\partial\mathcal{D}^d} - \left\langle b, D\partial_n \Phi \right\rangle_{\partial\mathcal{D}^d} - \left\langle D\partial_n b, \Phi \right\rangle_{\partial\mathcal{D}^d} \end{aligned} \quad (4.112)$$

and with the following linear right-hand-side:

$$\begin{aligned} b^{SIP}(b) &= \left(b, Q \right)_D - \left\langle b, J_0 \right\rangle_{\partial\mathcal{D}^n} + 2 \left\langle b, J^{inc} \right\rangle_{\partial\mathcal{D}^r} \\ &+ \left\langle \kappa_f^{SIP} b, \Phi_0 \right\rangle_{\partial\mathcal{D}^d} - \left\langle D\partial_n b, \Phi_0 \right\rangle_{\partial\mathcal{D}^d} \end{aligned} \quad (4.113)$$

As previously stated, the general penalty term, κ needs to have sufficient positive measure to ensure stability. From previous investigations [50, 150, 51], we choose the SIP penalty coefficient to be face dependent with the following form,

$$\kappa_f^{SIP} = \begin{cases} \frac{c}{2} \left(\frac{D^+}{h^+} + \frac{D^-}{h^-} \right) & f \in E_h^i \\ c \frac{D^-}{h^-} & f \in \partial\mathcal{D} \end{cases}, \quad (4.114)$$

for interior, E_h^i , and boundary, $\partial\mathcal{D}$, faces respectively. In Eq. (4.114), h^\pm is the orthogonal projection of the face, f , into the cells defined by the trace in Eq. (4.110). The orthogonal projection, h^\pm , is the length of the cell in the direction orthogonal to the face f . Turcksin and Ragusa, [51], defined h^\pm for 2D polygons, whose definitions can be seen in Table 4.3. 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 4.3, we define our choice for the extension of the orthogonal projections to 3D in Table 4.4. 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 4.3: 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 4.4: 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. (4.112) and (4.113), there are two additional elementary matrix types required other than those presented in Chapter 2. The first has the form of $(D\vec{\nabla}b, \vec{\nabla}\Phi)_K$ and is referred to as the stiffness matrix [92]. For a cell K with N_K basis functions, we define the stiffness matrix \mathbf{S} as

$$\mathbf{S}_K = \int_K \vec{\nabla} \mathbf{b}_K \cdot \vec{\nabla} \mathbf{b}_K^T dr, \quad (4.115)$$

which has dimensionality ($N_K \times N_K$). The \mathbf{b}_K basis functions are a column vector of length N_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

$$\mathbf{S}_K = \sum_{q=1}^{N_q} w_q^K \vec{\nabla} \mathbf{b}_K(\vec{x}_q) \cdot \vec{\nabla} \mathbf{b}_K^T(\vec{x}_q). \quad (4.116)$$

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

$$\mathbf{S}_K = \begin{bmatrix} \int_K \vec{\nabla} b_1 \cdot \vec{\nabla} b_1 & \dots & \int_K \vec{\nabla} b_1 \cdot \vec{\nabla} b_j & \dots & \int_K \vec{\nabla} b_1 \cdot \vec{\nabla} b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_K \vec{\nabla} b_i \cdot \vec{\nabla} b_1 & \dots & \int_K \vec{\nabla} b_i \cdot \vec{\nabla} b_j & \dots & \int_K \vec{\nabla} b_i \cdot \vec{\nabla} b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_K \vec{\nabla} b_{N_K} \cdot \vec{\nabla} b_1 & \dots & \int_K \vec{\nabla} b_{N_K} \cdot \vec{\nabla} b_j & \dots & \int_K \vec{\nabla} b_{N_K} \cdot \vec{\nabla} b_{N_K} \end{bmatrix}, \quad (4.117)$$

where an individual matrix entry is of the form:

$$S_{i,j,K} = \int_K \vec{\nabla} b_i \cdot \vec{\nabla} b_j. \quad (4.118)$$

4.3.2 Elementary Surface Gradient Matrices

The second new elementary matrix that we present for the SIP discretization 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: $\langle D\partial_n b, \Phi \rangle_f$. These terms are analogous 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 following form:

$$\mathbf{N}_{f,K} = \int_f \vec{n}(s) \cdot \vec{\nabla} \mathbf{b}_K \mathbf{b}_K^T ds. \quad (4.119)$$

where the direction of the surface normal, \vec{n} , is arbitrarily chosen to either go from cell K to cell K' or from cell K' to cell K . From the analytical integral, we can see that these matrices have dimensionality, $(N_K \times N_K)$. 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:

$$\begin{aligned} \langle [\![u]\!], \{\{\partial_n v\}\} \rangle_f &= -\langle \{\{\vec{n}u\}\}, \{\{\vec{n}\partial_n v\}\} \rangle_f, \\ \langle \{\{\partial_n u\}\}, [\![v]\!] \rangle_f &= -\langle \{\{\vec{n}\partial_n u\}\}, \{\{\vec{n}v\}\} \rangle_f \end{aligned} \quad (4.120)$$

If the direction of the normal is changed from \vec{n} to $-\vec{n}$ in Eq. (4.120), 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 :

$$\mathbf{N}_{f,K} = \sum_{q=1}^{N_q^f} w_q^f \vec{n}(\vec{x}_q) \vec{\nabla} \mathbf{b}_K(\vec{x}_q) \mathbf{b}_K^T(\vec{x}_q). \quad (4.121)$$

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

$$\mathbf{N}_{f,K} = \begin{bmatrix} \int_f \vec{n} \cdot \vec{\nabla} b_1 b_1 & \dots & \int_f \vec{n} \cdot \vec{\nabla} b_1 b_j & \dots & \int_f \vec{n} \cdot \vec{\nabla} b_1 b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_f \vec{n} \cdot \vec{\nabla} b_i b_1 & \dots & \int_f \vec{n} \cdot \vec{\nabla} b_i b_j & \dots & \int_f \vec{n} \cdot \vec{\nabla} b_i b_{N_K} \\ \vdots & & \vdots & & \vdots \\ \int_f \vec{n} \cdot \vec{\nabla} b_{N_K} b_1 & \dots & \int_f \vec{n} \cdot \vec{\nabla} b_{N_K} b_j & \dots & \int_f \vec{n} \cdot \vec{\nabla} b_{N_K} b_{N_K} \end{bmatrix}, \quad (4.122)$$

where an individual matrix entry is of the form:

$$N_{i,j,f,K} = \int_f \vec{n} \cdot \vec{\nabla} b_i b_j. \quad (4.123)$$

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

$$\delta J^{inc} = \sum_{\vec{\Omega}_m \cdot \vec{n} > 0} w_m (\vec{\Omega}_m \cdot \vec{n}) \delta\Psi_m^{(k)}. \quad (4.124)$$

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

$$a^{MIP}(b, \delta\Phi) = b^{MIP}(b), \quad (4.125)$$

with the following bilinear matrix,

$$\begin{aligned} a^{MIP}(b, \delta\Phi) &= \left(D\vec{\nabla}b, \vec{\nabla}\delta\Phi \right)_{\mathcal{D}} + \left(\sigma b, \delta\Phi \right)_{\mathcal{D}} \\ &+ \left\langle \kappa_f^{MIP}[\![b]\!], [\![\delta\Phi]\!] \right\rangle_{E_h^i} + \left\langle [\![b]\!], \{ \{ D\partial_n \delta\Phi \} \} \right\rangle_{E_h^i} + \left\langle \{ \{ D\partial_n b \} \}, [\![\delta\Phi]\!] \right\rangle_{E_h^i}, \\ &+ \left\langle \kappa_f^{MIP}b, \delta\Phi \right\rangle_{\partial\mathcal{D}^d} - \frac{1}{2} \left\langle b, D\partial_n \delta\Phi \right\rangle_{\partial\mathcal{D}^d} - \frac{1}{2} \left\langle D\partial_n b, \delta\Phi \right\rangle_{\partial\mathcal{D}^d} \end{aligned} \quad (4.126)$$

and with the following linear right-hand-side,

$$b^{MIP}(b) = \left(b, Q \right)_{\mathcal{D}} + \left\langle b, \delta J^{inc} \right\rangle_{\partial\mathcal{D}^{ref}}. \quad (4.127)$$

The MIP penalty coefficient also needs to be of a different form than the one used for SIP. From Eq. (4.114), 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 [75] that if the penalty coefficient becomes too small, then MIP used as a DSA acceleration 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 [50]. This new definition of κ_f^{MIP} can be succinctly written as

$$\kappa_f^{MIP} = \max \left(\kappa_f^{SIP}, \frac{1}{4} \right). \quad (4.128)$$

Just like the SIP penalty coefficient, this definition of κ_f^{MIP} ensures that the MIP bilinear form of Eq. (4.126) is SPD. We next describe the procedures used to efficiently solve the MIP system matrix for our work.

4.5 Solving the MIP Diffusion Problem

Equations (4.126) and (4.127) 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 [97]. If the system of equations you seek to solve 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 for well-conditioned matrices. 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 $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$. 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 $\mathbf{Mz}_k = \mathbf{r}_k$ 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 [97]. 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: $\mathbf{M}(\omega) = \frac{\omega}{2-\omega} (\frac{1}{\omega}\mathbf{D} + \mathbf{L}) \mathbf{D}^{-1} (\frac{1}{\omega}\mathbf{D} + \mathbf{L})^T$. The PCG algorithm can be simplified with this preconditioner choice by the Eisenstat trick [159]. The final simple preconditioner that we will consider is Incomplete LU Factorization (ILU). Instead of simply decomposing the system matrix, we approximately 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 [51] that multigrid preconditioning methods can efficiently invert the MIP operator. Both Algebraic MultiGrid (AMG) [160, 161] and AGgregation-based algebraic MultiGrid (AGMG) [162, 163, 164, 165] were shown to be effective. We leave the general details of multigrid methods to the previously defined references.

Algorithm 2 PCG Algorithm

```

1: Compute  $\mathbf{r}_0 = \mathbf{b} - \mathbf{Ax}_0$ 
2: for  $k=1,2,\dots$  do
3:   Solve:  $\mathbf{Mz}_{i-1} = \mathbf{r}_{i-1}$ 
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:   $\mathbf{q}_i = \mathbf{Ap}_i$ 
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  $\frac{\|\mathbf{r}_i\|}{\|\mathbf{b}\|} < \text{tol}$  then
15:    Exit
16:  end if
17: end for

```

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 decompose the

iteration error into Fourier modes. 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 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 are provided in Appendix C. 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 $(\Delta x_1, \Delta x_2, \dots, \Delta x_{N_x})$ in the x -dimension, its N_y cell widths $(\Delta y_1, \Delta y_2, \dots, \Delta y_{N_y})$ in the y -dimension, and its N_z cell widths $(\Delta z_1, \Delta z_2, \dots, \Delta z_{N_z})$ 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 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

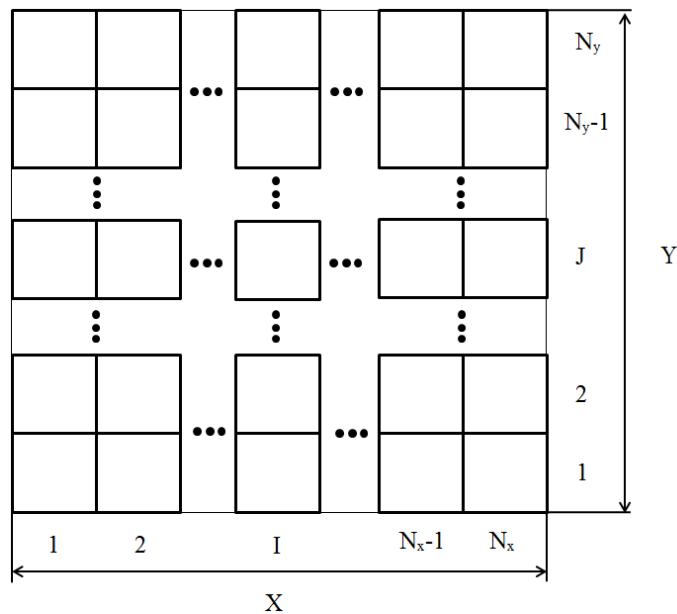


Figure 4.1: Fourier domain for 2D quadrilateral cells or an axial slice of 3D hexahedral cells in a regular grid.

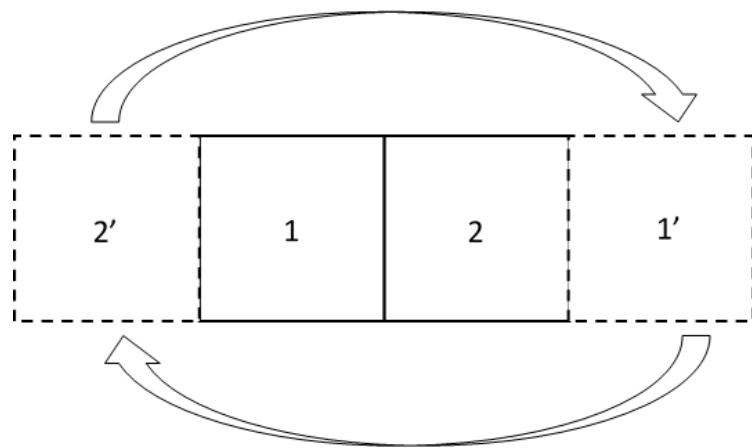


Figure 4.2: Representation of the periodic boundary conditions used for Fourier analysis for a 2-cell geometric layout.

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

$$\Psi^{(k)}(\vec{r}, \vec{\Omega}) = \hat{\Psi}^{(k)}(\vec{\Omega}) e^{i\vec{\lambda}\cdot\vec{r}}, \quad (4.129)$$

where $i = \sqrt{-1}$ and $\hat{\Psi}^{(k)}(\vec{\Omega})$ is the Fourier expansion coefficient for the angular flux. Also, the error modes for the discretized angular flux along direction m and the flux moments can be given by

$$\Psi_m^{(k)}(\vec{r}) = \hat{\Psi}_m^{(k)} e^{i\vec{\lambda}\cdot\vec{r}}, \quad (4.130)$$

and

$$\Phi^{(k)}(\vec{r}) = \hat{\Phi}^{(k)} e^{i\vec{\lambda}\cdot\vec{r}}, \quad (4.131)$$

respectively. In Eqs. (4.129) - (4.131), 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. (4.131) by

$$\Phi^{(k)}(\vec{r}) = \mathbb{P}(\vec{\lambda}) \hat{\Phi}^{(k)}, \quad (4.132)$$

where the diagonal phase matrix has the form:

$$\mathbb{P}(\vec{\lambda}) = \begin{bmatrix} e^{i\vec{\lambda} \cdot \vec{r}_1} & 0 & 0 & 0 \\ 0 & e^{i\vec{\lambda} \cdot \vec{r}_2} & 0 & 0 \\ 0 & 0 & e^{i\vec{\lambda} \cdot \vec{r}_3} & 0 \\ 0 & 0 & 0 & e^{i\vec{\lambda} \cdot \vec{r}_4} \end{bmatrix}. \quad (4.133)$$

One can also utilize an alternative version of Eq. (4.133) 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

$$\mathbf{D}\tilde{\mathbf{L}}^{-1}\mathbf{M}\tilde{\boldsymbol{\Sigma}}, \quad (4.134)$$

and

$$\mathbf{D}\tilde{\mathbf{L}}^{-1}\mathbf{M}\tilde{\boldsymbol{\Sigma}} + \mathbf{P}\tilde{\mathbf{A}}^{-1}\mathbf{X}\tilde{\boldsymbol{\Sigma}} (\mathbf{D}\tilde{\mathbf{L}}^{-1}\mathbf{M}\tilde{\boldsymbol{\Sigma}} - \mathbf{I}), \quad (4.135)$$

respectively. In Eqs. (4.134) and (4.135), the $\tilde{\mathbf{L}}$, $\tilde{\boldsymbol{\Sigma}}$, 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 [166]. 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.

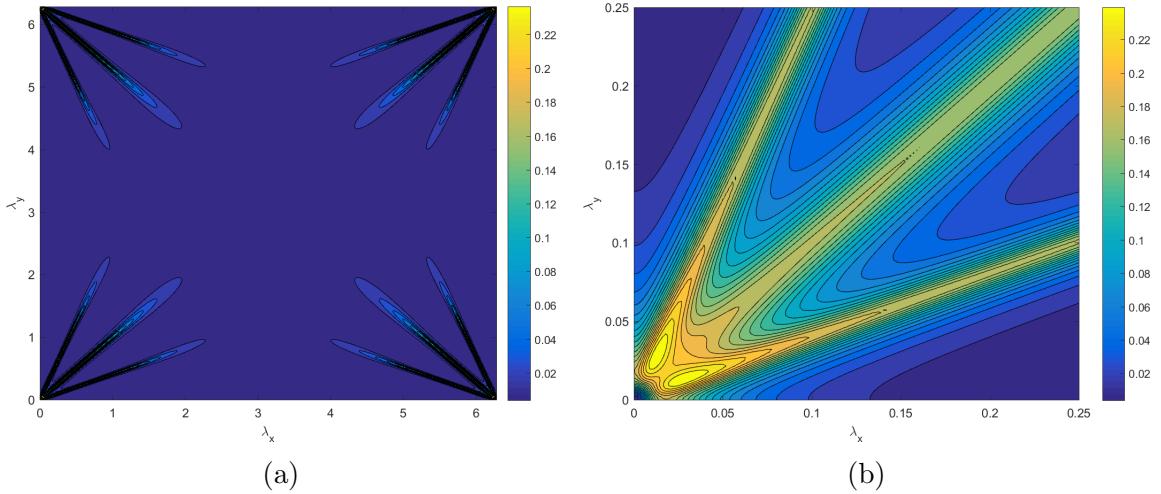


Figure 4.3: 2D Fourier wave form for MIP, with 1 square cell with $1e-2$ mfp, with the PWL coordinates, with the LS4 quadrature, and where the wave numbers range from: (a) $[\lambda_x, \lambda_y] = [0, 2\pi]^2$ and (b) $[\lambda_x, \lambda_y] = [0, 1/4]^2$.

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 [167].

4.7 Numerical Results

We now present all theoretical and applied results pertaining to the described DSA schemes. First, we provide results in Section 4.7.1 on the efficacy of the SIP form as a diffusion solver. Section 4.7.2 provides all theoretical and numerical results pertaining to the 1-group DSA scheme. Then, Section 4.7.3 demonstrates the scalability of the MIP diffusion form on massively-parallel computer architectures. Finally, Section 4.7.4 gives the results pertaining to the analysis performed for the thermal neutron upscattering acceleration methods.

4.7.1 *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 [154]. The MIP DSA form has also been successfully utilized for unstructured 2D polygonal grids [51]. Here, we first seek to extend the efficacy of the SIP form as a diffusion solver on polyhedral mesh cells. 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 4.7.1.1. Next, the exactly-linear solution analysis is performed in Section 4.7.1.2. Finally, the MMS analysis to confirm the second order convergence rates of the 3D PWL basis functions is presented in Section 4.7.1.3.

4.7.1.1 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 reuse the Cartesian, ordered-triangular, polygonal sinusoidal, and polygonal-z meshes. We will also use purely-randomized polygonal grids formed from Voronoi mesh generation. To form the needed 3D polyhedral grids, we will take these 2D grids and simply extrude the meshes in the z-dimension.

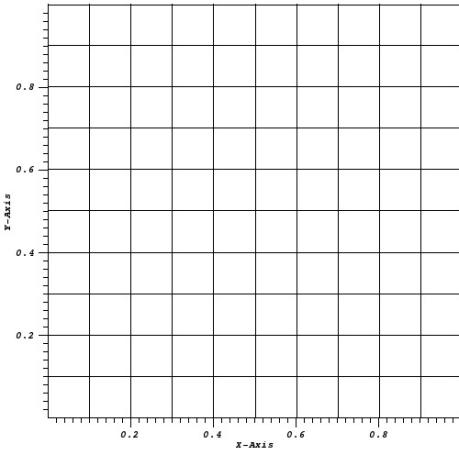
Figure 4.4 provides the 2D mesh types that will be utilized in this analysis. Figure 4.5 then provides the same meshes after they have been extruded in the z-dimension. We note that it will not simply be these exact grids that are employed. For the MMS analysis in Section 4.7.1.3, various refinements of these meshes will be utilized.

4.7.1.2 Exactly-Linear Solution

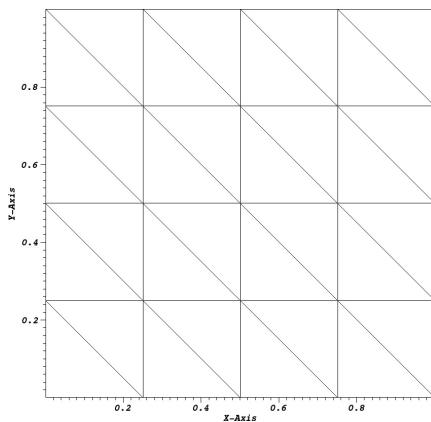
We first test SIP by enforcing a system that yields an exactly-linear diffusion solution. Linear finite elements should then theoretically fully span the solution space. We can achieve this mathematically by setting the cross-section and right-hand-source terms to zero, $\sigma = Q = 0$. Robin boundary 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

$$\Phi(x, y, z) = \frac{4J^{inc}}{L + 4D} (L + 2D - y), \quad (4.136)$$

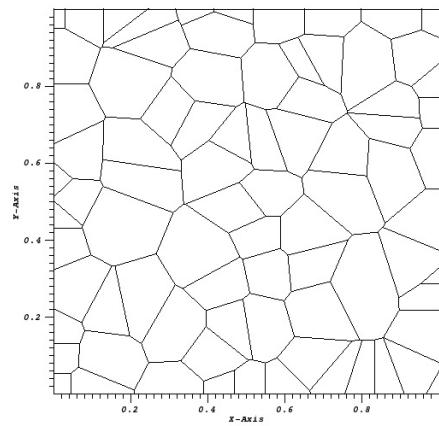
with the following boundary conditions in the y-direction:



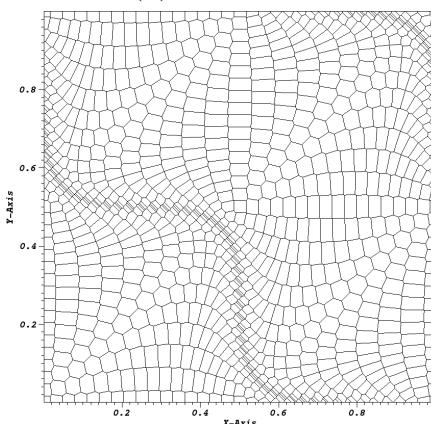
(a) Cartesian



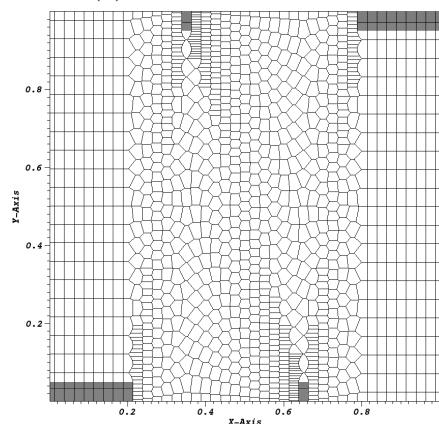
(b) Triangular



(c) Random Polygons

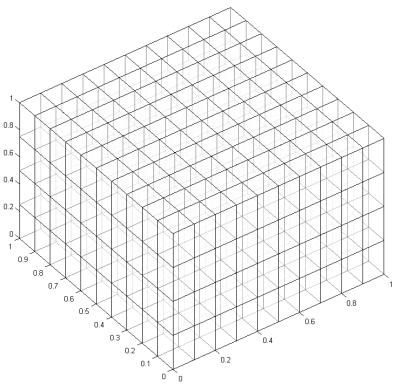


(d) Sinusoid Polygons

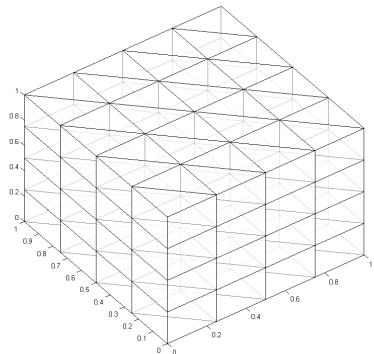


(e) Polygonal Z-Mesh

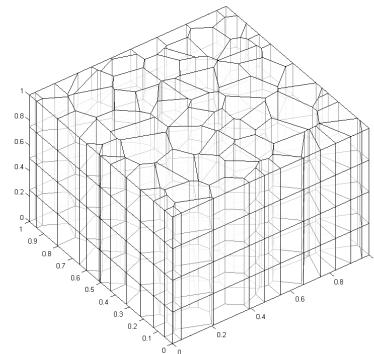
Figure 4.4: 2D polygonal grids to be extruded for the 3D SIP calculations.



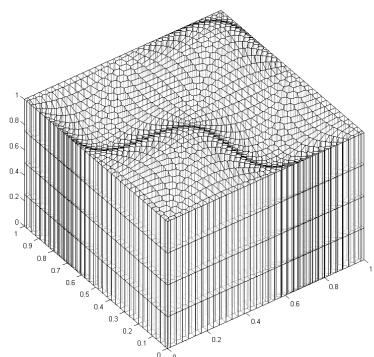
(a) Cartesian



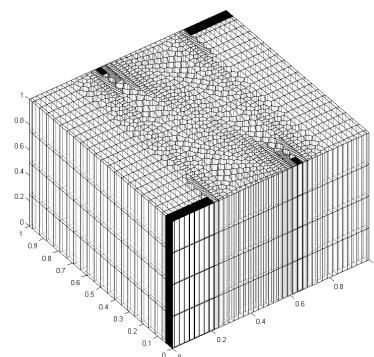
(b) Triangular



(c) Random Polygons



(d) Sinusoid Polygons



(e) Polygonal Z-Mesh

Figure 4.5: Extrusion of the different polygonal meshes.

$$\begin{aligned} \Phi - 2D\partial_y\Phi &= 4J^{inc}, & \forall(x, z), y = 0 \\ \Phi + 2D\partial_y\Phi &= 0, & \forall(x, z), y = L \end{aligned} . \quad (4.137)$$

In Eqs. (4.136) and (4.137), 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. (4.136), our solution has a value of 20 at $y = 0$ and linearly-decreases to 16 at $y = 1$.

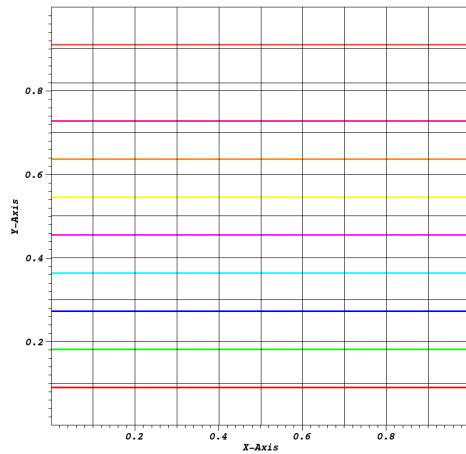
Using the 3D PWL basis functions, which were previously used for SIP on 2D polygons [154], the linear solutions are presented in Figure 4.6 at the midplane axial slice of $z = L/2$. We can see from the exact contour lines in the plots that SIP can capture an exactly-linear solution even on some highly distorted polyhedral grids.

4.7.1.3 Method of Manufactured Solutions

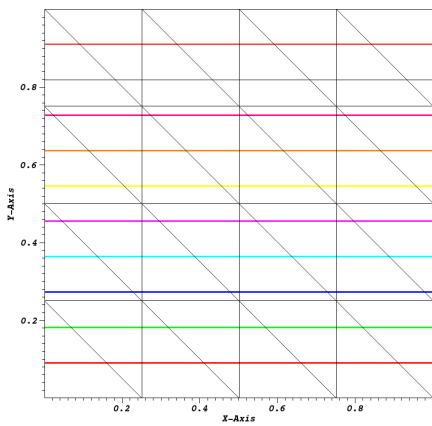
We next test to see if the SIP diffusion form can capture the appropriate second order convergence rates with the 3D PWL basis functions. These basis functions were previously shown to capture the appropriate convergence rates for the DGFEM transport equation on 3D hexahedral grids [39]. The convergence for the SIP diffusion form is similar to the DGFEM transport equation. Since there are no impositions on the regularity of the diffusion solution, then the SIP convergence rate in the L_2 -norm can be given by

$$||\Phi - \Phi_h||_{L_2} \leq Ch^{p+1}. \quad (4.138)$$

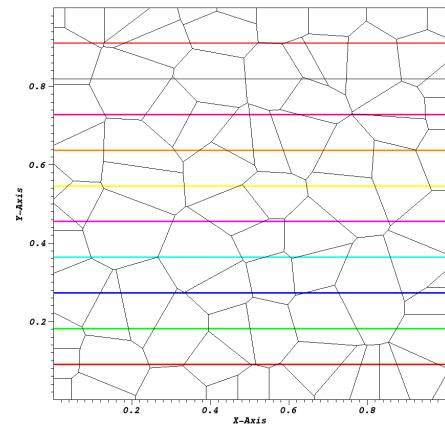
In a similar manner to the DGFEM transport equation, we can express a proportional relation between the mesh cell diameters, h , and the number of degrees of freedom, N_{dof} : $N_{dof} \propto h^{-d}$. Therefore, the convergence rate of the SIP diffusion form can be expressed with the degrees of freedom as



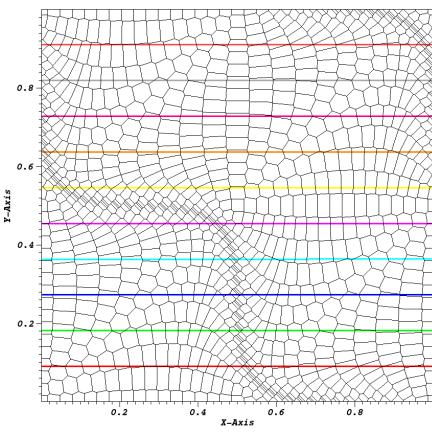
(a) Cartesian



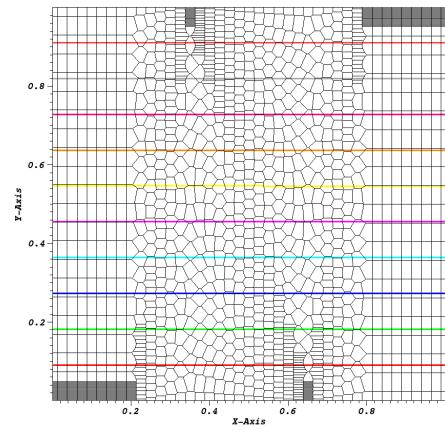
(b) Triangular



(c) Random Polygons



(d) Sinusoid Polygons



(e) Polygonal Z-Mesh

Figure 4.6: Axial slice showing the contours for the linear solution of the SIP diffusion form.

$$\|\Phi - \Phi_h\|_{L_2} \leq CN_{dof}^{-\frac{p+1}{d}}. \quad (4.139)$$

Equation (4.139) states that for this 3D problem using linear basis functions, we expect the slope of our convergence rates to be $-2/3$.

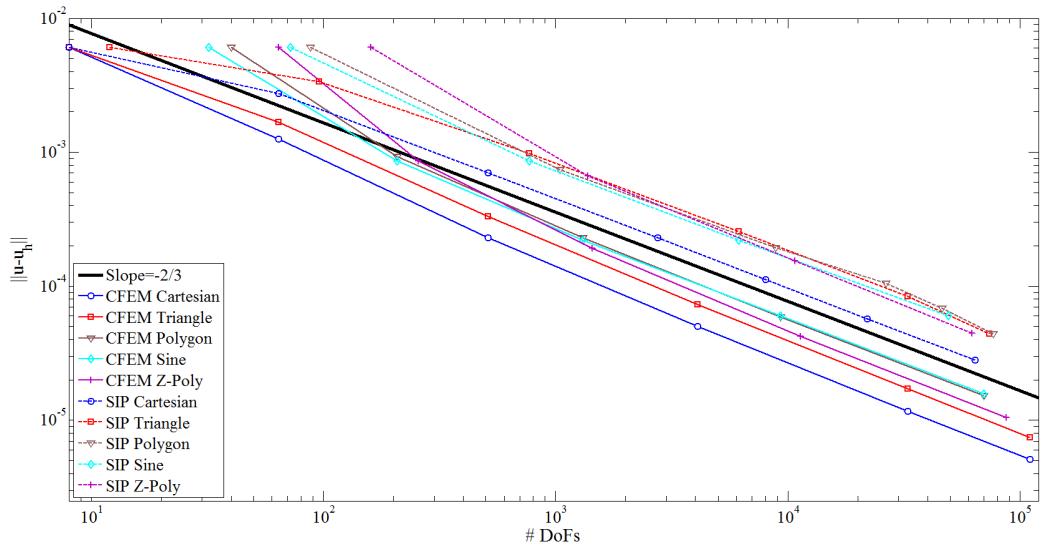
We test two different MMS solutions: 1) a pairwise quadratic function and 2) a product of the pairwise quadratic function and a localized Gaussian function. The quadratic and Gaussian functions are given by

$$\Phi^{quad}(x, y, z) = x(1-x)y(1-y)z(1-z), \quad (4.140)$$

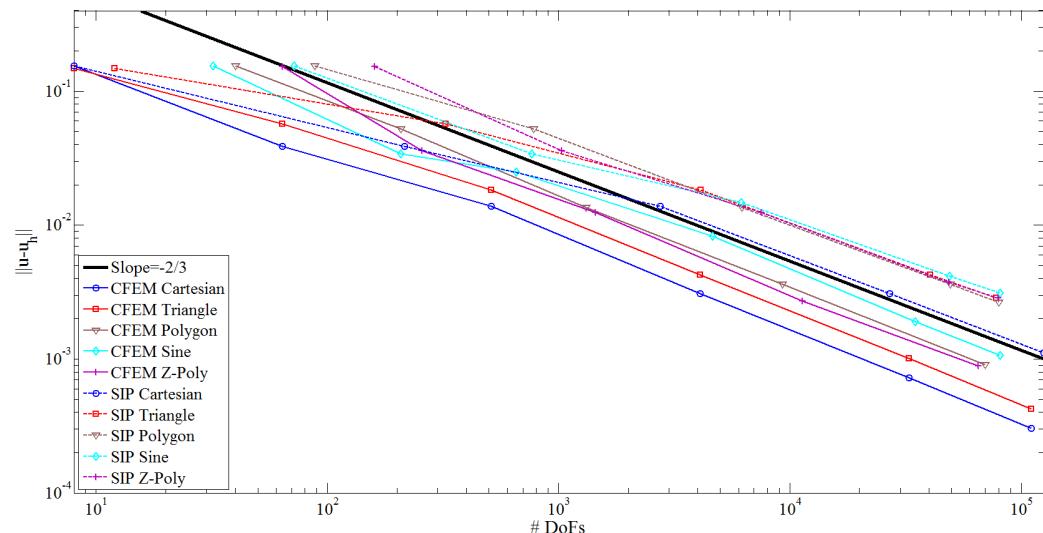
and

$$\Phi^{gauss}(x, y, z) = \Phi^{quad}(x, y, z) \exp(-(\vec{r} - \vec{r}_0) \cdot (\vec{r} - \vec{r}_0)^T), \quad (4.141)$$

respectively, where $\vec{r} = (x, y, z)$ and $\vec{r}_0 = (x_0, y_0, z_0)$. We plot the MMS error of the SIP diffusion form in Figure 4.7 for both the quadratic and Gaussian solutions. We also provide solutions using a CFEM diffusion form for comparison. In Figure 4.7, we clearly see that SIP captures the proper convergence rate. It is also easy to see that the CFEM diffusion solutions provide reduced error based on the number of spatial degrees of freedom. This is an obvious result since SIP has multiple degrees of freedom for every vertex on the mesh whereas CFEM only has 1. However, we will be using the MIP DSA form as the low-order acceleration operator and not as a stand-alone diffusion solver. Therefore, this observed non-optimal convergence rate (as well as capturing the exactly-linear solution) is required (and satisfied).



(a) Quadratic Solution



(b) Gaussian Solution

Figure 4.7: Error in the SIP diffusion form. Additional results using a CFEM diffusion form are provided for comparison.

4.7.2 1 Group DSA Analysis

We now present our analysis of the 1-group MIP DSA scheme for simple 2D and 3D problems. First in Section 4.7.2.1, we provide the analysis on 2D homogeneous configurations using all of the linear and quadratic basis functions presented in Chapter 3. Then in Section 4.7.2.2, we provide the analysis on 3D homogeneous configurations using the 3D PWL basis functions. Finally, we provide analysis on 2D heterogeneous configurations in Section 4.7.2.3.

4.7.2.1 2D Homogeneous Medium Case

The first set of analysis involving the 1-group DSA scheme is over 2D homogeneous configurations. Specifically, we will provide the Fourier Analysis of the different linear and quadratic polygonal basis functions for different homogeneous configurations. For all the results obtained, a single quadrilateral mesh cell comprised of the unit square ($X = Y = 1$) was used. We plot the FA results against the mesh cell size in terms of mean free paths by varying the total cross section. The scattering ratio (σ_s/σ_t) is always set to 0.9999 and scattering is also always isotropic. We also performed all of the analysis with the S_2 , S_4 , S_8 , and S_{16} level-symmetric quadrature sets. The constant in the MIP penalty coefficient, c , was always set to 4.

The FA spectral radii for the linear and quadratic basis functions for the Wachspress, PWL, mean value, and maximum entropy coordinates are given in Figures 4.8, 4.9, 4.10, and 4.11, respectively. We can see from these figures that the MIP scheme is unconditionally stable for all of the linear and quadratic basis functions. Also, the spectral radii are all similar in their distributions except for the linear PWL basis functions. We conclude this analysis of the 2D homogeneous problems by providing some examples of the Fourier wave number distributions. In these examples, we use a fine grid of Fourier wave numbers ($\vec{\lambda} = [\lambda_x, \lambda_y]$) and calculate their associated spec-

tral radii. Then, contour plots can be generated of these spectral radii distributions. Figures 4.12, 4.13, 4.14, and 4.15 give these spectral radii distributions for the 2D PWL coordinates on the unit square for the LS_2 , LS_4 , LS_8 , and LS_{16} quadratures. In each figure, we provide six wave number distributions corresponding to different cell mean free path values of 10^{-2} , 10^{-1} , 10^0 , 10^1 , 10^2 , and 10^3 . In the fine mesh limit examples (mean free path values of 10^{-2} and 10^{-1}), we can clearly see the effects of the individual quadrature angles.

4.7.2.2 3D Homogeneous Medium Case

In this Section, we perform similar analysis to that presented in Section 4.7.2.1. We perform both Fourier and numerical analysis on the 3D MIP DSA scheme on homogeneous domains. The 3D PWL basis functions are used for all of the analysis presented here. First, we show results of Fourier Analysis performed on the unit cube ($X = Y = Z = 1$) in Figure 4.16. A single mesh cell is used, and the spectral radius of the FA is plotted against the cell size in terms of mean free paths. Since we only use the unit cube in this analysis, the cell sizes change by varying the total cross section. In Figure 4.16, scattering is isotropic with a scattering ratio (σ_s/σ_t) of 0.9999. Level-symmetric quadrature of orders 2, 4, 8, and 16 was used and values of 1 and 4 were used for the constant in the MIP penalty coefficient, c . We can see from both the plots that the 3D MIP form is unconditionally stable between the two penalty coefficient constants. All spectral radii values are below 0.6 except for the case of S_2 quadrature with the penalty coefficient constant value of 1. This means that $c = 1$ offers insufficient penalization, and a higher value should be used.

Next, Figure 4.17 provides the Fourier Analysis for cells having different aspect ratios. We change the aspect ratio by fixing $X = 1$ and varying the values of Y and Z . For these problems $Y = Z$, and we again analyze both $c = 1$ and $c = 4$. The

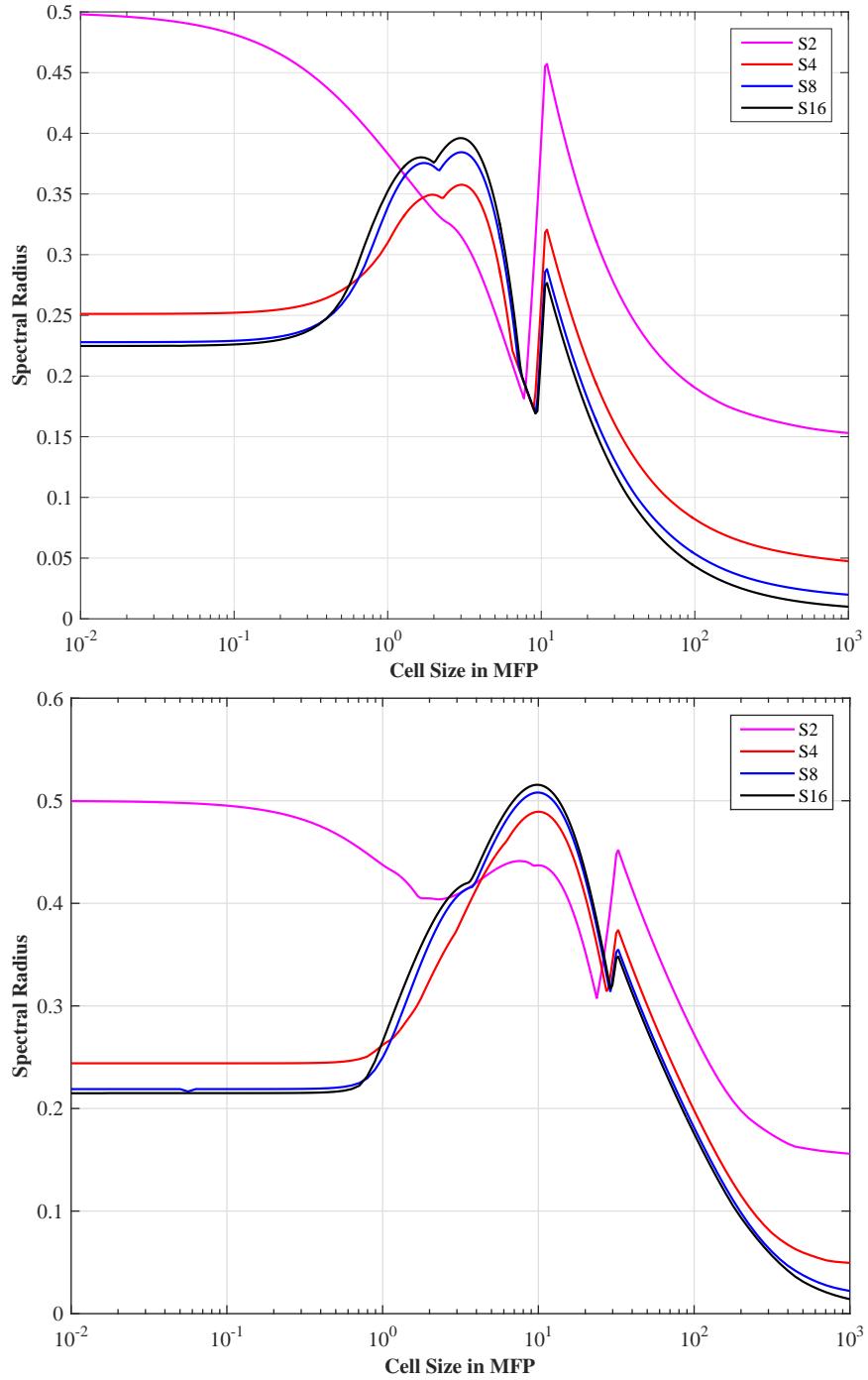


Figure 4.8: Fourier analysis of the 2D MIP form with $c = 4$ and using the linear (top) and quadratic (bottom) Wachspress basis functions.

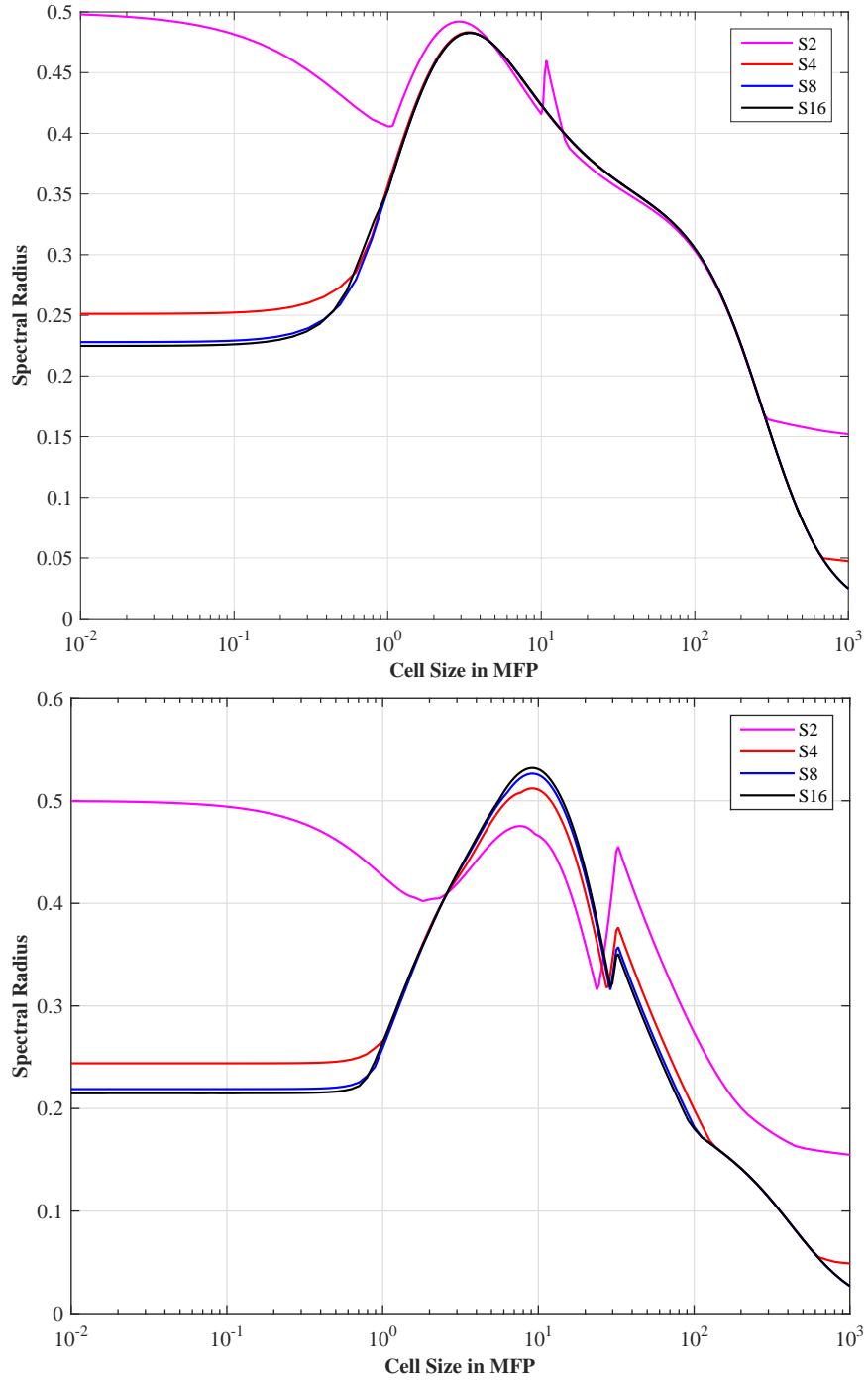


Figure 4.9: Fourier analysis of the 2D MIP form with $c = 4$ and using the linear (top) and quadratic (bottom) PWL basis functions.

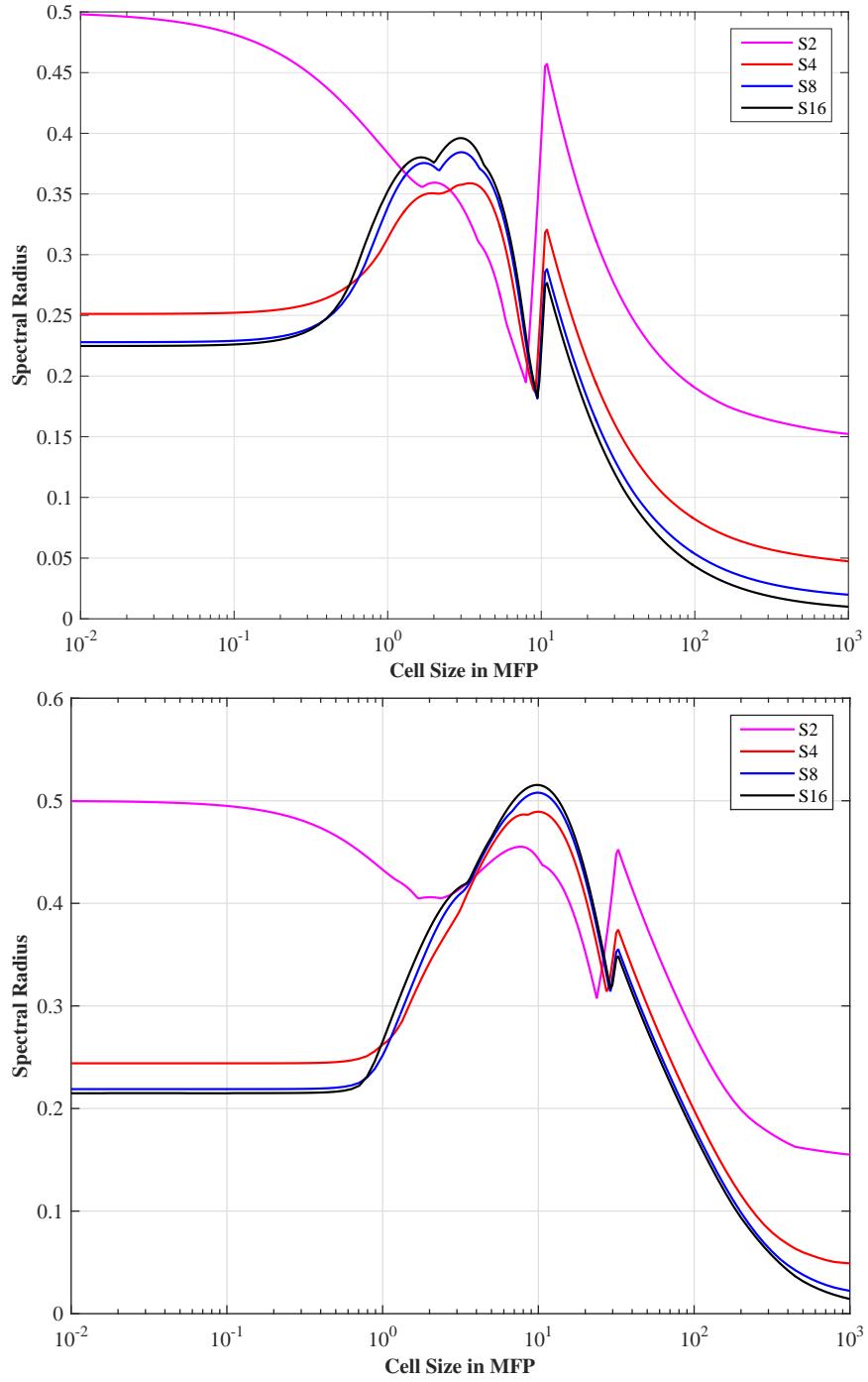


Figure 4.10: Fourier analysis of the 2D MIP form with $c = 4$ and using the linear (top) and quadratic (bottom) mean value basis functions.

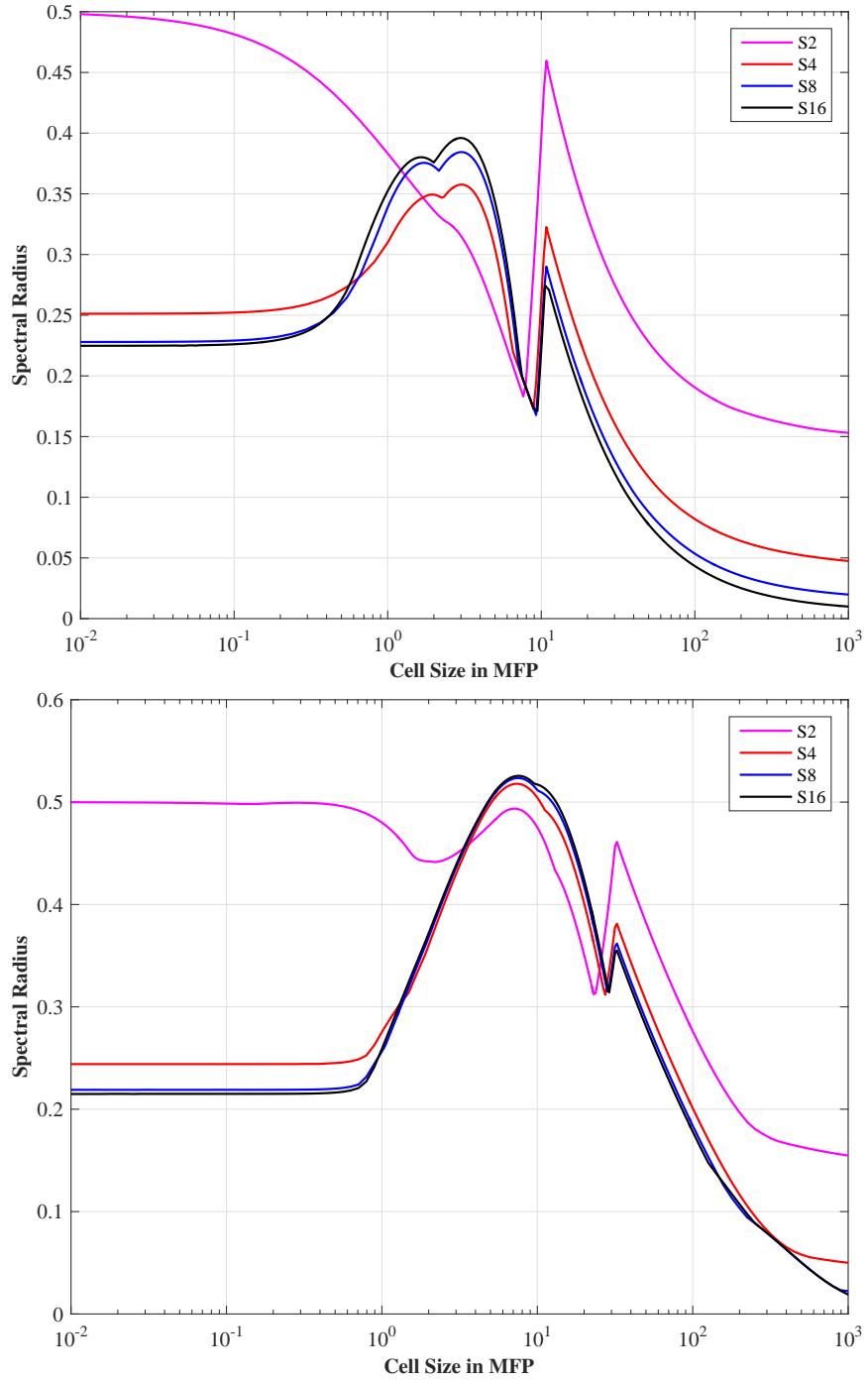


Figure 4.11: Fourier analysis of the 2D MIP form with $c = 4$ and using the linear (top) and quadratic (bottom) maximum entropy basis functions.

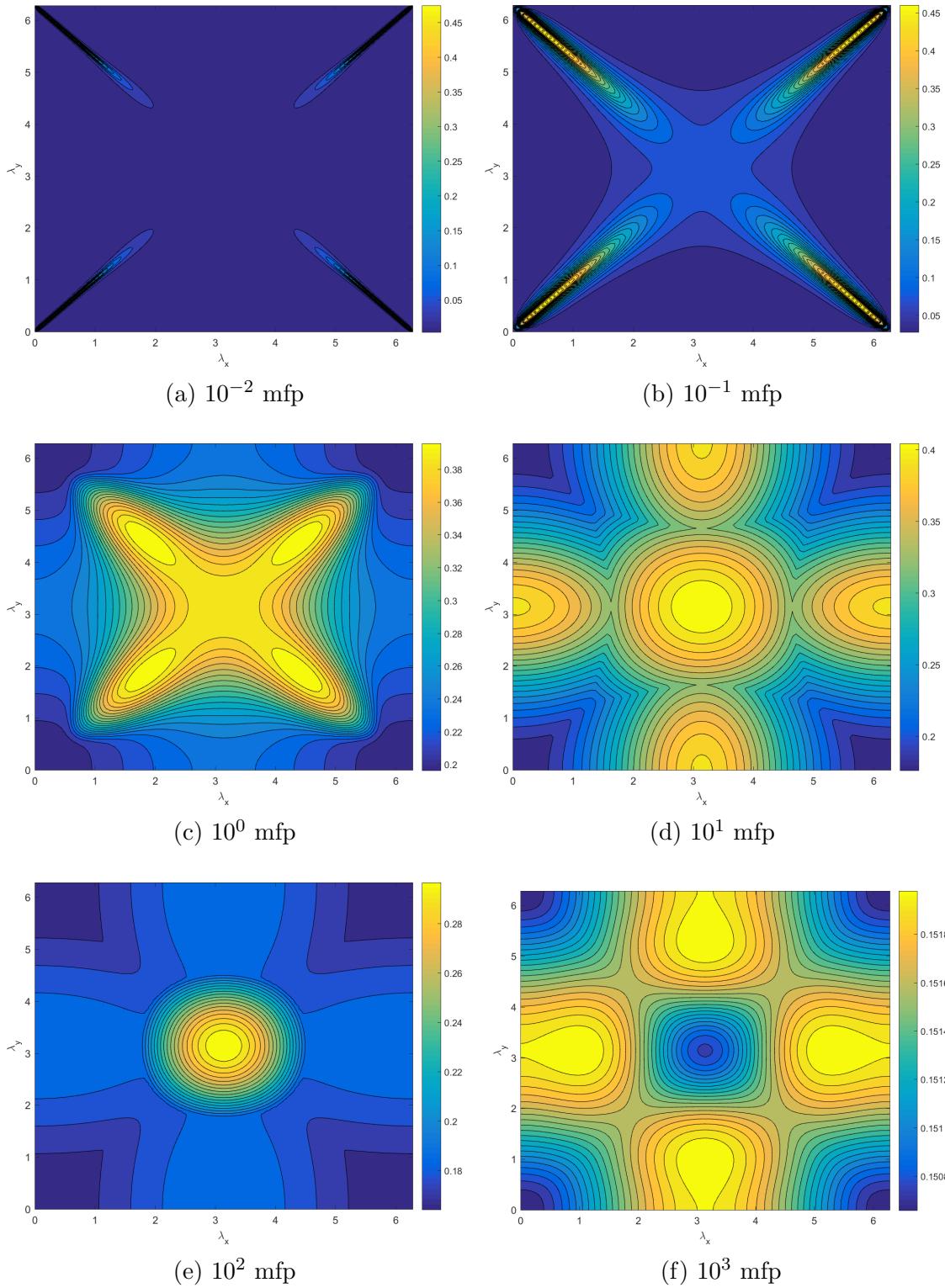


Figure 4.12: Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL basis functions and LS_2 quadrature.

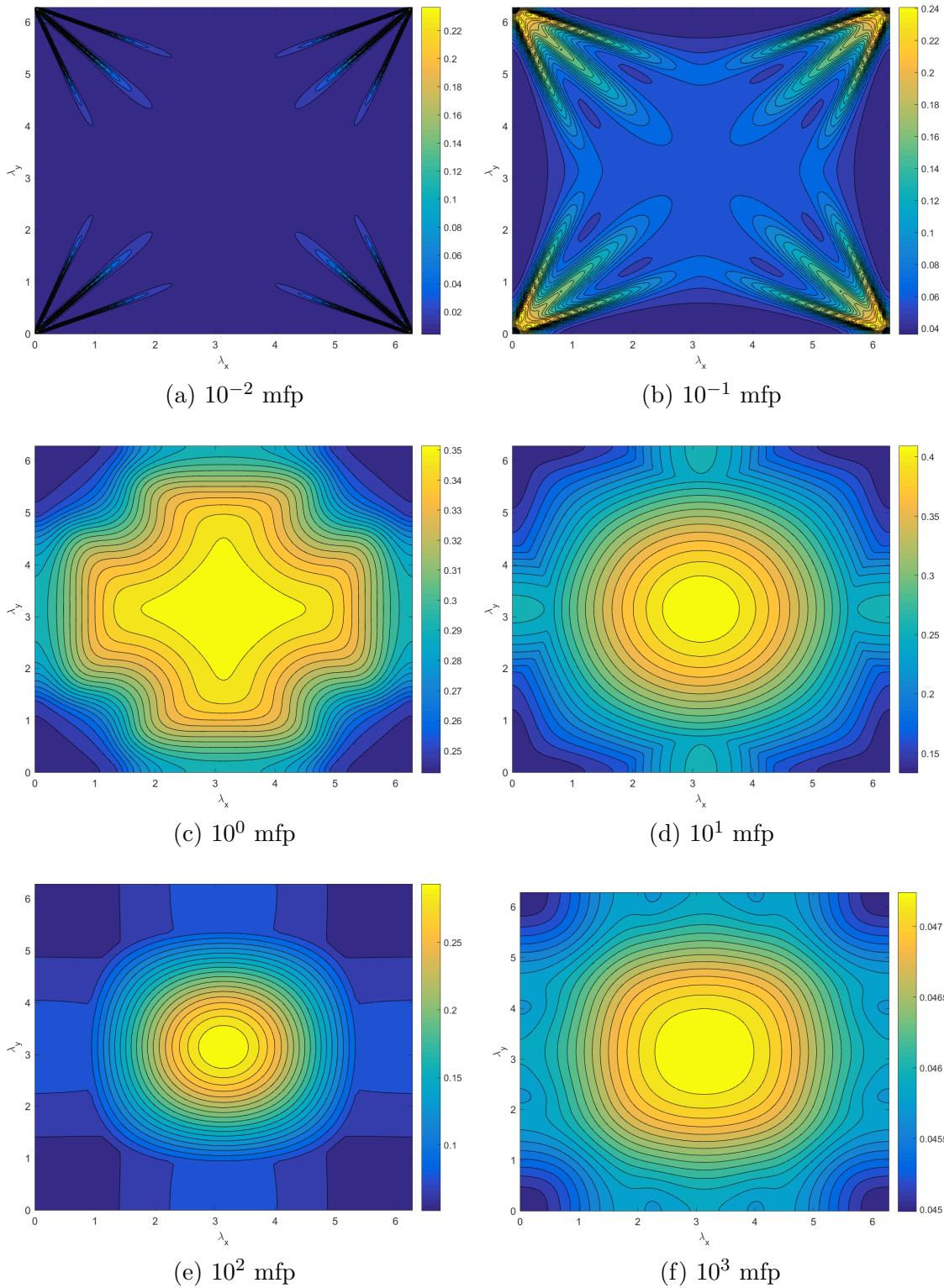


Figure 4.13: Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL basis functions and LS_4 quadrature.

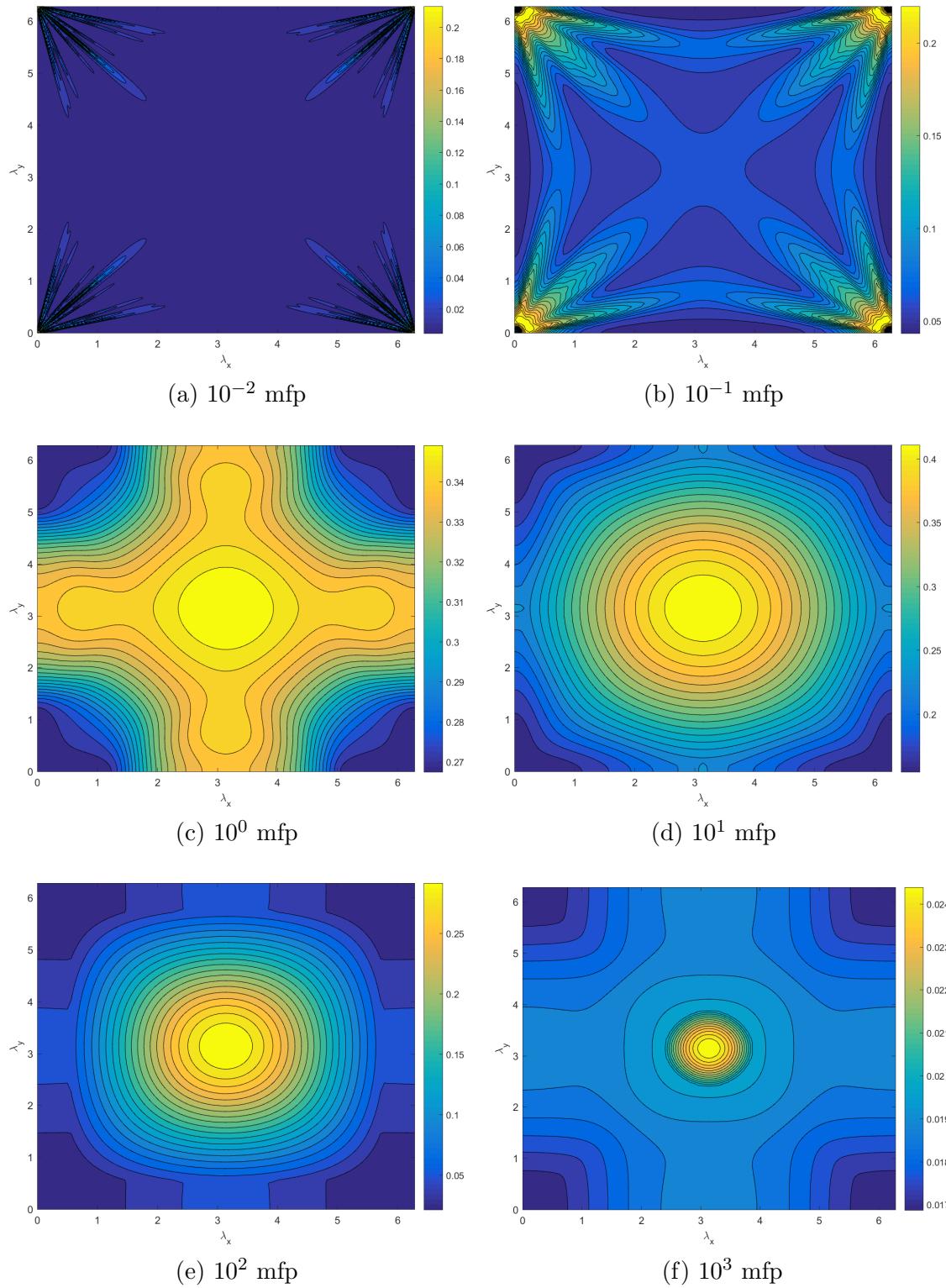


Figure 4.14: Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL basis functions and LS_8 quadrature.

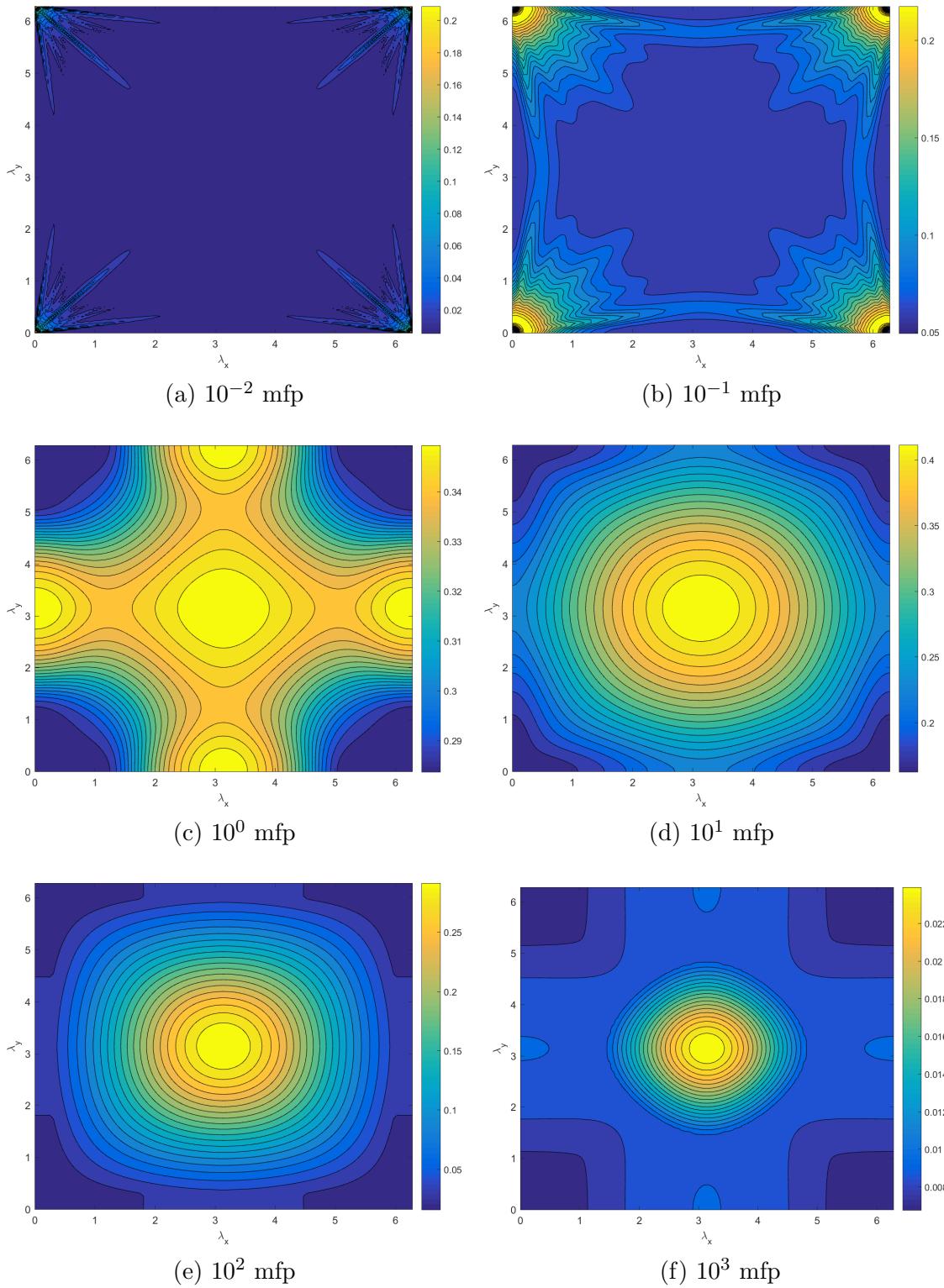


Figure 4.15: Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL basis functions and LS_{16} quadrature.

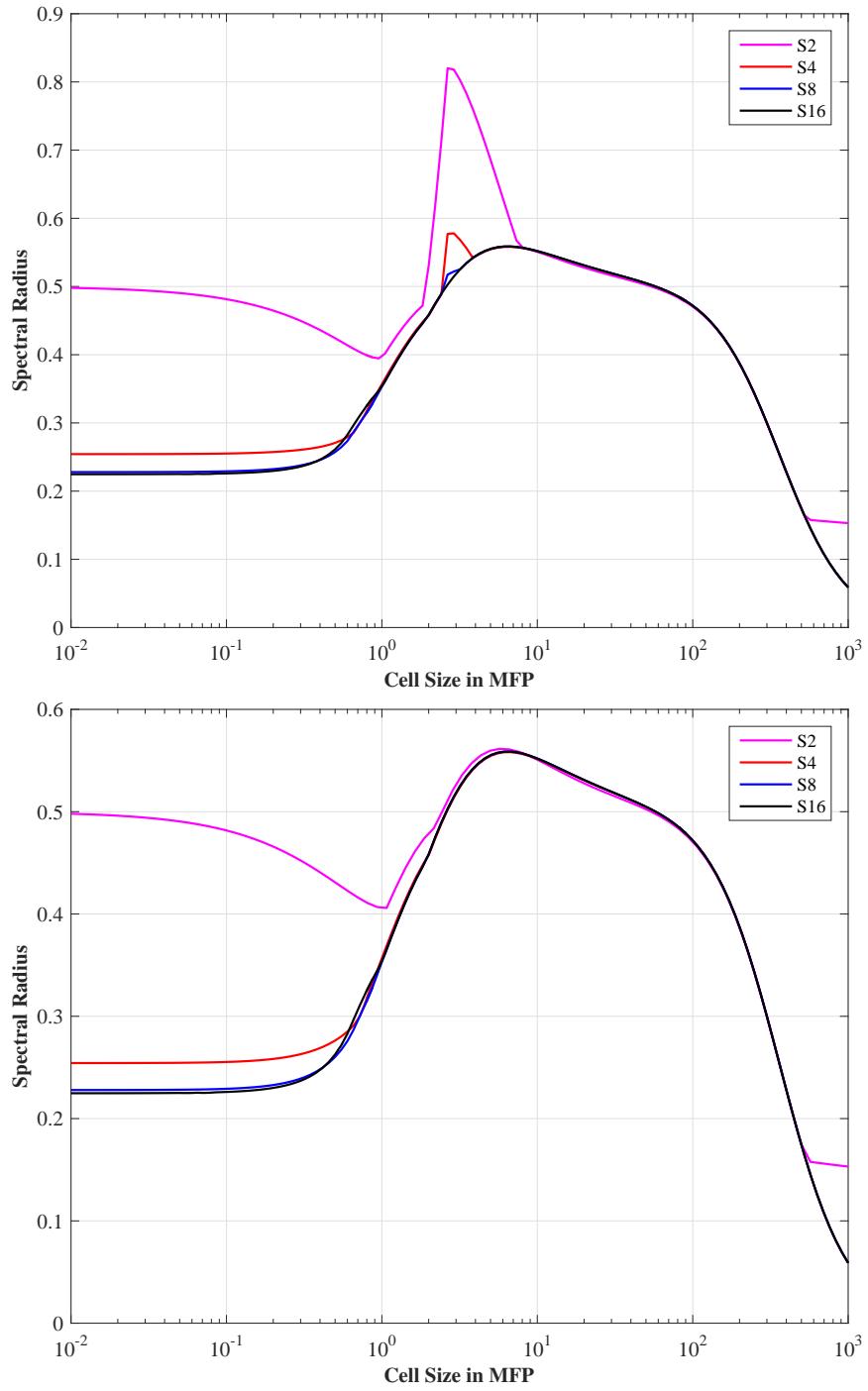


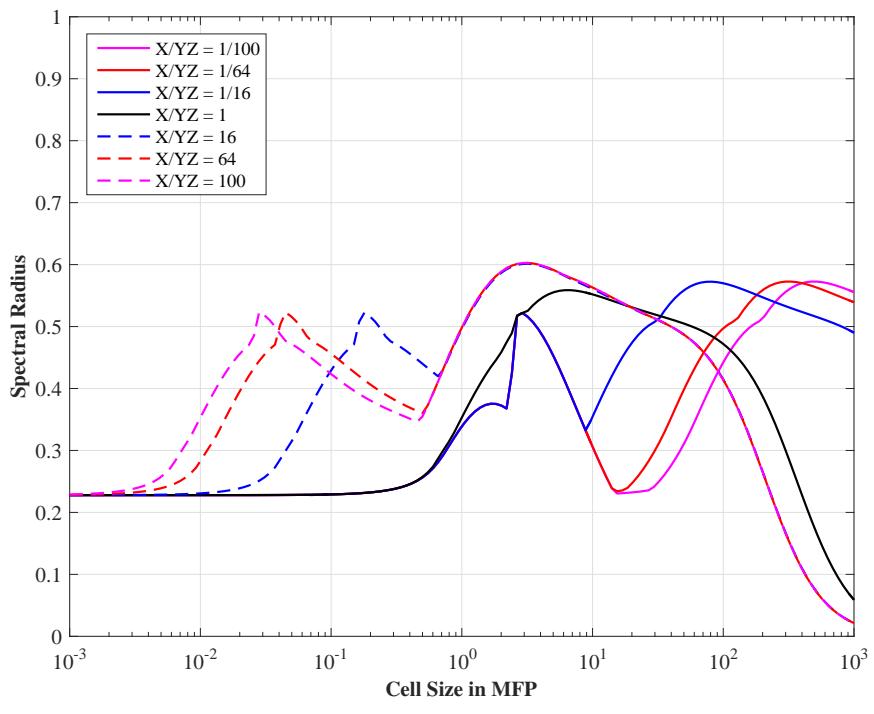
Figure 4.16: Fourier spectral radius of the 3D MIP form with $c = 1$ (top) and $c = 4$ (bottom).

aspect ratios had values of $(1/100, 1/64, 1/16, 1, 16, 64, 100)$. Only LS_8 quadrature is used. We can see from the two figures that the MIP acceleration continues to be unconditionally stable on 3D hexahedral cells with high aspect ratios.

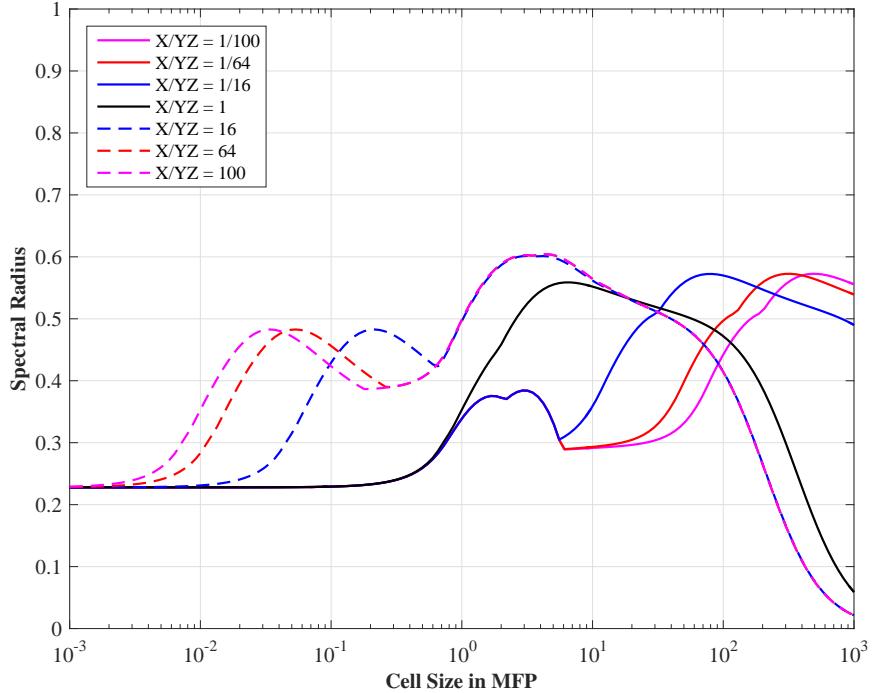
Finally, we present results that compare Fourier and numerical analyses. Figure 4.18 plots the S_2 , S_4 , and S_8 FA results from Figure 4.16 along with accompanying numerical results. The numerical problem analyzed is still the unit cube ($X = Y = Z = 1$), but we now have 80 mesh cells in each dimension (as compared to Fourier Analysis which only had 1 mesh cell). We stress that a large number of mesh cells are required. Otherwise, the flux gradient at the boundary is poorly represented and additional leakage occurs. This additional leakage, which is solely due to the spatial discretization, would artificially lower the spectral radius of the numerical results. We can see from Figure 4.18 that our numerical analysis follows very closely with the results from Fourier Analysis. Discrepancies between the Fourier and numerical spectral radii arise for cell sizes in the fine mesh limit, but these have been reported previously [50].

4.7.2.3 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 [168]. 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 analyze a slightly modified form of the standard PHI problem and fix $\sigma_1 = 1$. We then define a tuning parameter, σ , so that $\sigma_2 = 1/\sigma$. Therefore, increasing the value of σ will increase



(a) $c = 1$



(b) $c = 4$

Figure 4.17: Fourier spectral radii for MIP form with LS_8 quadrature on cells with different aspect ratios.

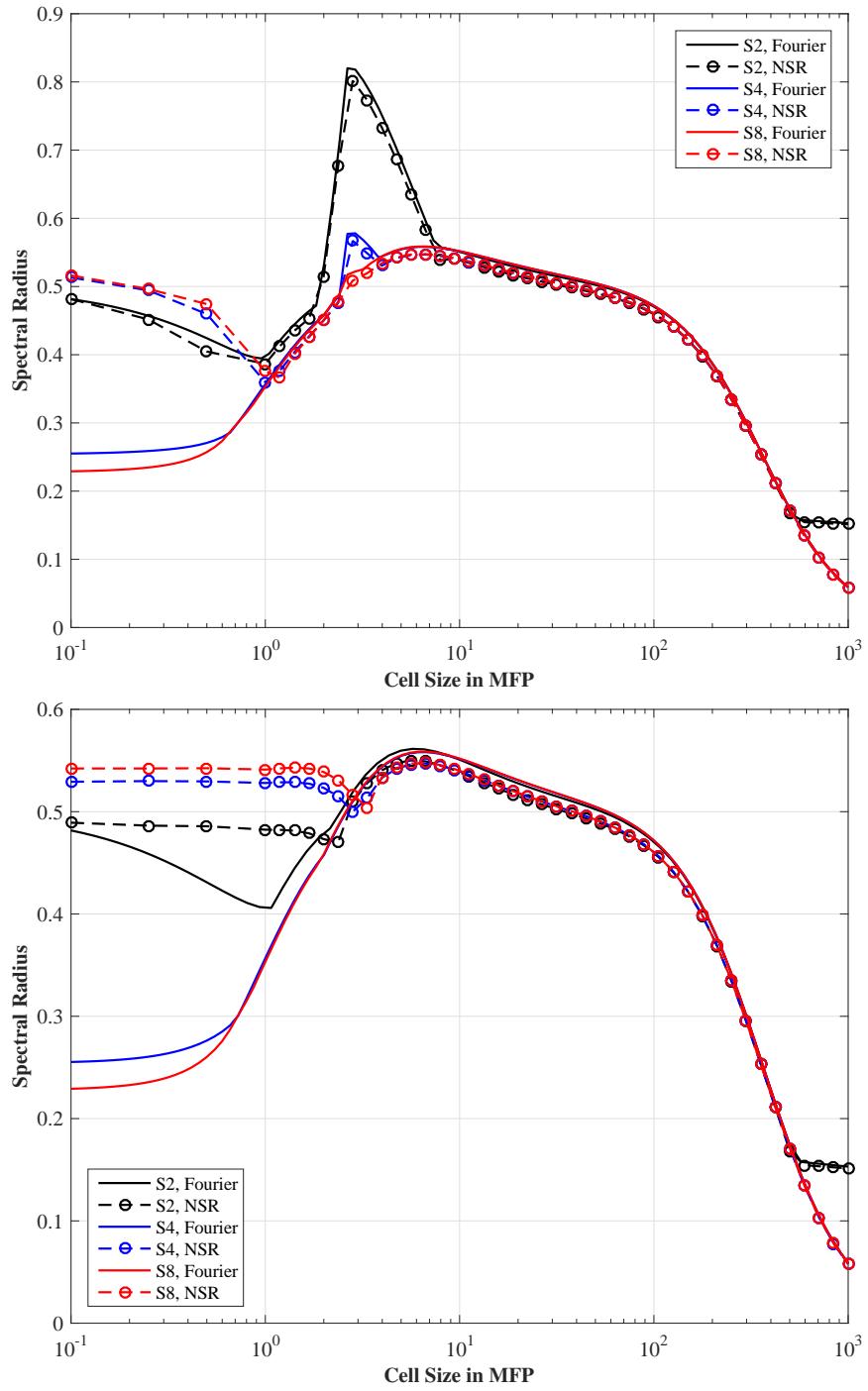


Figure 4.18: Comparison between the numerical spectral radii and the theoretical Fourier Analysis spectral radii on the unit cube with $c = 1$ (top) and $c = 4$ (bottom).

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 4.7.2.1, 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, $\sigma = [10^1, 10^2, 10^3, 10^4, 10^5, 10^6]$, 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 4.5 - 4.8 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 cases. 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 4.19 provides the eigenvalue distribution based off the Fourier wave numbers for the different quadrature sets for $\sigma = 10$ and $c = 0.9$. Figure 4.20 then provides the same information for $\sigma = 10^4$ and $c = 0.9999$.

Table 4.5: Spectral radius for the 2D PHI problem with the PWL basis functions and LS2 quadrature.

σ	Scattering ratios					
	0.9	0.99	0.999	0.9999	0.99999	0.99999
10^1	0.60142	0.66600	0.67251	0.67316	0.67322	0.67323
10^2	0.75689	0.90833	0.94287	0.95227	0.95508	0.95596
10^3	0.85933	0.97258	0.99038	0.99413	0.99512	0.99542
10^4	0.88808	0.98554	0.99729	0.99907	0.99944	0.99954
10^5	0.89629	0.98868	0.99855	0.99973	0.99991	0.99994
10^6	0.89883	0.98959	0.99887	0.99985	0.99997	0.99999

Table 4.6: Spectral radius for the 2D PHI problem with the PWL basis functions and LS4 quadrature.

σ	Scattering ratios					
	0.9	0.99	0.999	0.9999	0.99999	0.99999
10^1	0.43185	0.56089	0.60118	0.61387	0.61788	0.61915
10^2	0.75639	0.90612	0.94056	0.94998	0.95281	0.95369
10^3	0.85852	0.97206	0.99004	0.99386	0.99488	0.99518
10^4	0.88779	0.98542	0.99723	0.99904	0.99942	0.99952
10^5	0.89619	0.98864	0.99854	0.99972	0.99990	0.99994
10^6	0.89880	0.98958	0.99886	0.99985	0.99997	0.99999

Table 4.7: Spectral radius for the 2D PHI problem with the PWL basis functions and LS8 quadrature.

σ	Scattering ratios					
	0.9	0.99	0.999	0.9999	0.99999	0.99999
10^1	0.44705	0.54910	0.58578	0.59782	0.60166	0.60288
10^2	0.75183	0.90261	0.93769	0.94734	0.95024	0.95114
10^3	0.85724	0.97143	0.98966	0.99356	0.99460	0.99491
10^4	0.88744	0.98528	0.99717	0.99899	0.99939	0.99949
10^5	0.89608	0.98860	0.99852	0.99972	0.99990	0.99994
10^6	0.89877	0.98957	0.99886	0.99985	0.99997	0.99999

Table 4.8: Spectral radius for the 2D PHI problem with the PWL basis functions and LS16 quadrature.

σ	Scattering ratios					
	0.9	0.99	0.999	0.9999	0.99999	0.99999
10^1	0.46168	0.55373	0.57370	0.58286	0.58604	0.58707
10^2	0.74842	0.89889	0.93453	0.94440	0.94738	0.94830
10^3	0.85588	0.97073	0.98923	0.99322	0.99429	0.99461
10^4	0.88705	0.98513	0.99710	0.99896	0.99935	0.99946
10^5	0.89597	0.98856	0.99851	0.99971	0.99990	0.99994
10^6	0.89873	0.98955	0.99885	0.99985	0.99997	0.99999

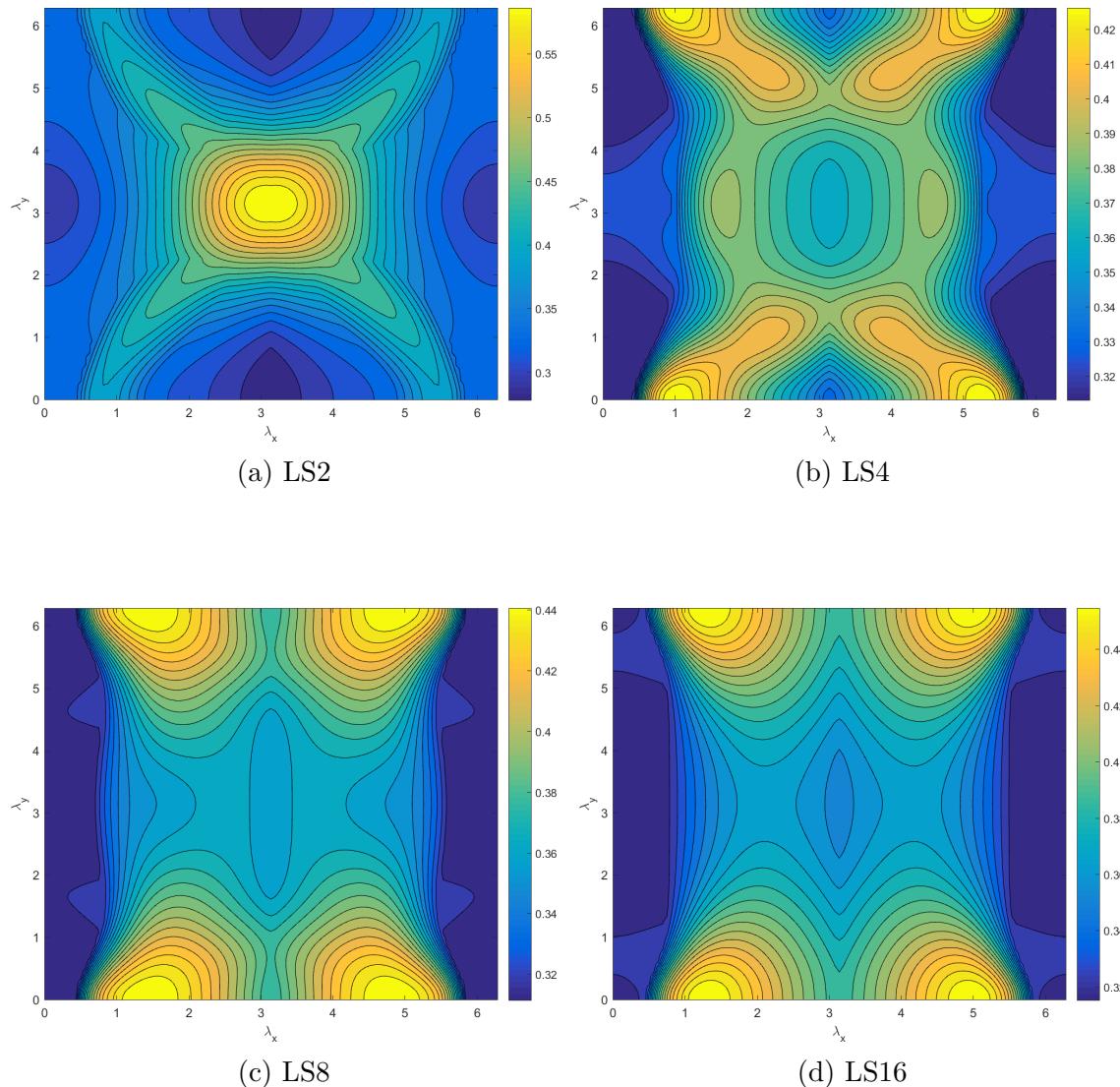


Figure 4.19: Fourier wave number distribution for the 2D PHI problem with $\sigma = 10$ and $c = 0.9$ and different level-symmetric quadratures.

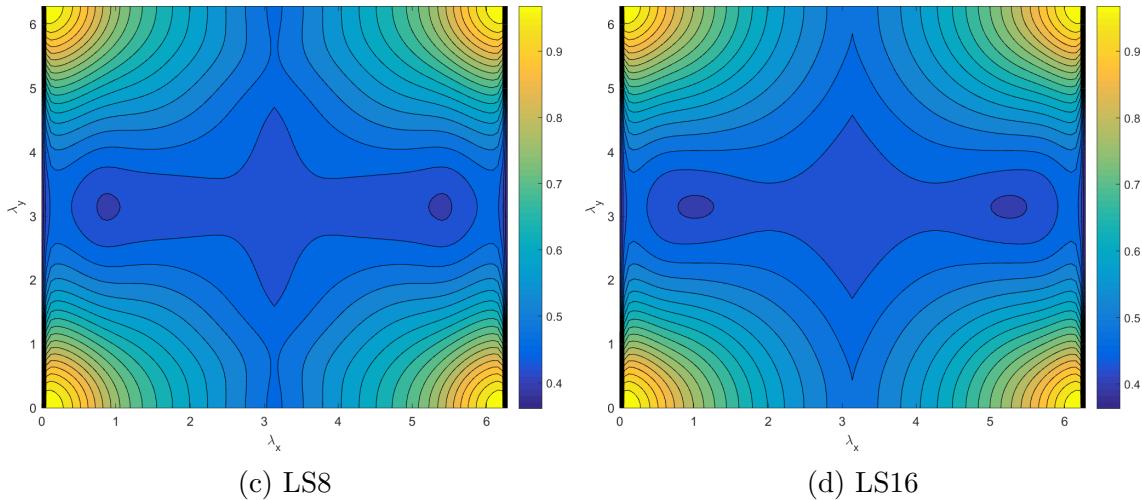
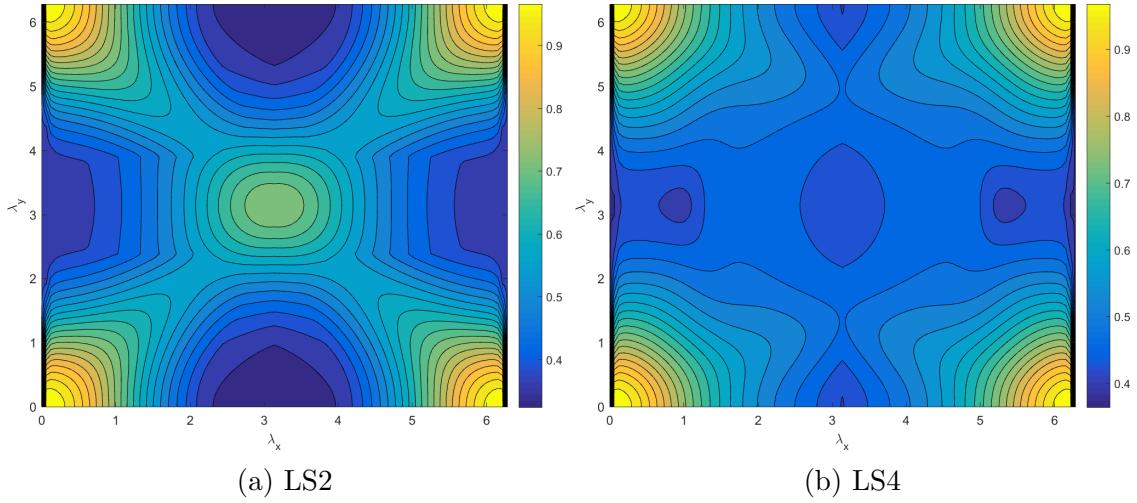


Figure 4.20: Fourier wave number distribution for the 2D PHI problem with $\sigma = 10^4$ and $c = 0.9999$ and different level-symmetric quadratures.

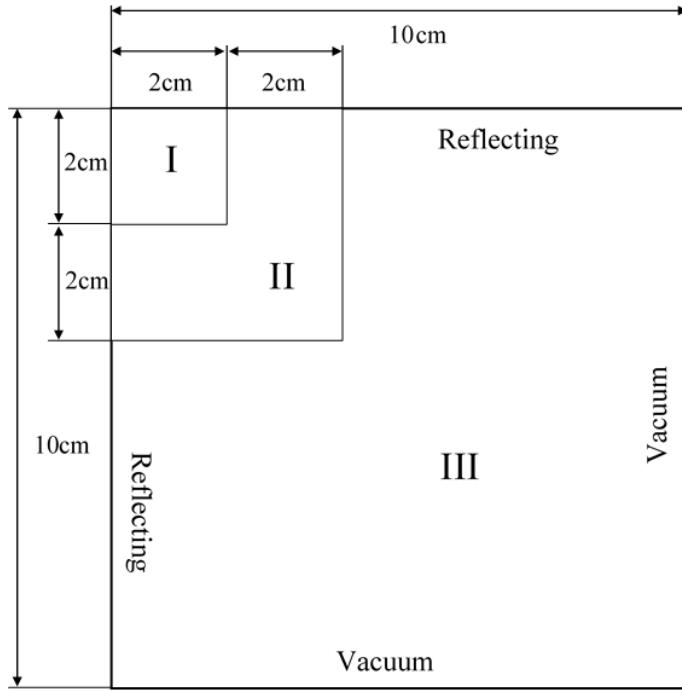


Figure 4.21: Geometry description for the Iron-Water problem.

4.7.2.4 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 [169]. It was modified by Wang and Ragusa for use with higher-order basis functions on triangular meshes with hanging nodes [50]. We will reexamine their work on degenerate polygonal grids and not use hanging nodes. The complete geometric description of our problem including boundary conditions and material distributions is given in Figure 4.21. The material properties for each region which include the total cross section, scattering ratio ($c = \sigma_s/\sigma_t$), and source strength are given in Table 4.9. Scattering is isotropic.

For this analysis, we will test multiple problem configurations. All AMR histories

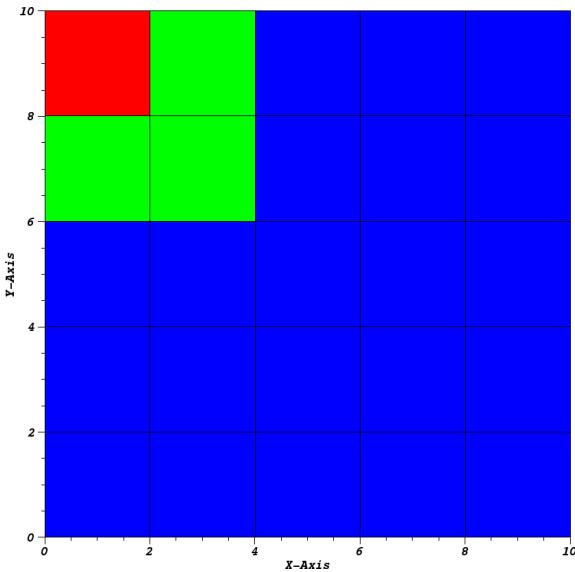


Figure 4.22: Initial mesh for the Iron-Water problem.

Table 4.9: 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

will be calculated from an initial starting mesh provided in Figure 4.22 where the material regions are given by the different mesh cell colors. Both linear and quadratic basis functions will be analyzed, but we only use the 2D PWL coordinates. Two different angular quadratures will be used: a S_4 level-symmetric (LS) quadrature and a S_{24}^2 PGLC quadrature. We expect additional solution discontinuities (besides those corresponding to the material discontinuities) to arise along the S_N rays. Therefore, we are also performing runs using the PCLG quadrature with a high number of azimuthal angles to mitigate these S_N discontinuities. We also investigate the effects

of solution bootstrapping as well as the effectiveness of the five preconditioner types presented in Section 4.5.

We can summarize all the different problem permutations that will be analyzed:

1. Linear and quadratic PWL basis functions;
2. Use of either LS_4 or S_{24}^2 PGLC angular quadrature;
3. Bootstrapping or reinitialization of the solution at each AMR cycle;
4. Use of five different preconditioner types: no preconditioning, Jacobi preconditioning, SSOR preconditioning, ILU preconditioning, and AGMG preconditioning.

This means that there are four physical problem configurations (basis function order and angular quadrature). For each of these problem configurations, AMR refinement cycles either utilize solution bootstrapping or reinitialization (starting flux is all zero) and 1 of the 5 preconditioner types. Therefore, there are 40 total AMR runs required for this analysis. For each run, we carried out 24 refinement cycles with a maximum mesh irregularity of 1. The refinement criterion, α , was set to 1/4 for the linear basis functions and 1/10 for the quadratic basis functions.

We first present the tabulated data pertaining to the number of SI iterations and the number of preconditioner iterations required for the different problem configurations. Tables 4.10 and 4.11 give the iteration counts for solution reinitialization and bootstrapping, respectively, for all of the preconditioner types, using the linear basis functions and LS_4 quadrature. Likewise, Tables 4.12 and 4.13 give the iteration counts for solution reinitialization and bootstrapping, respectively, for all of the preconditioner types, using the quadratic basis functions and LS_4 quadrature. Then, Tables 4.14 and 4.15 give the iteration counts for solution reinitialization and

bootstrapping, respectively, for the S_{24}^2 PGLC quadrature and linear basis functions. Finally, Tables 4.16 and 4.17 give the iteration counts for solution reinitialization and bootstrapping, respectively, for the S_{24}^2 PGLC quadrature and quadratic basis functions. We can see clear trends from all of these tabulated results. Solution bootstrapping leads to less (up to half) SI iterations and fewer preconditioner iterations than solution reinitialization. This arises since the bootstrapped solution provides a better initial guess at each refinement level than simply a solution that is characteristically zero. From the preconditioner iteration counts, we can clearly see that the AGMG preconditioner provides the highest efficiency compared to the other preconditioners investigated. This result conforms to those presented by Turcksin and Ragusa [51].

Next, we provide some example images of the refined meshes for the different problem configurations at different refinement cycles. All of these meshes shown come from the runs using the AGMG preconditioner with bootstrapping at refinement cycles 06, 12, 18, and 24. Figures 4.23 and 4.24 show the meshes for the LS_4 quadrature using the linear and quadratic basis functions, respectively. Then, Figures 4.25 and 4.26 show the meshes for the S_{24}^2 PGLC quadrature using the linear and quadratic basis functions, respectively. For the LS_4 quadrature runs, we can clearly see the effects of the S_N discretization in the refinement history. This effect is significantly more noticeable for the quadratic basis functions. However, for the S_{24}^2 PGLC runs we see an elimination of these S_N discretization effects with the larger-angle quadrature set. Instead, the mesh refinements are dominated by the solution discontinuity at the material interfaces.

Table 4.10: DSA counts based on preconditioner type for the Iron-Water problem with LS_4 quadrature, linear basis functions, and solution reinitialization.

Cycle	SI Iter.	Preconditioner Type				
		None	Jacobi	SSOR	ILU	AGMG
0	19	357	282	58	94	19
1	21	635	506	75	170	21
2	17	831	700	91	233	17
3	16	917	791	100	269	127
4	13	1164	1062	158	351	114
5	12	1787	1547	218	501	106
6	12	2187	1883	266	594	116
7	12	2407	2123	330	670	122
8	12	3116	2698	395	843	111
9	11	3176	2835	449	903	113
10	11	4058	3515	563	1109	112
11	11	4442	3821	657	1221	110
12	11	5378	4659	788	1495	119
13	11	5989	5209	860	1680	123
14	11	6648	5824	957	1893	123
15	11	7655	6640	1098	2153	130
16	11	7973	6934	1149	2250	134
17	11	9456	8200	1378	2686	139
18	11	10181	8890	1373	2906	145
19	11	11111	9716	1550	3139	143
20	11	11934	10494	1780	3406	143
21	11	12456	10940	1779	3567	135
22	11	13896	12308	2043	3981	144
23	11	15216	13531	2253	4401	146
24	11	15932	14188	2436	4646	144

Table 4.11: DSA counts based on preconditioner type for the Iron-Water problem with LS_4 quadrature, linear basis functions, and solution bootstrapping.

Cycle	SI Iters.	Preconditioner Type				
		None	Jacobi	SSOR	ILU	AGMG
0	19	357	282	58	94	19
1	22	675	538	73	179	22
2	17	787	708	78	230	17
3	14	788	692	74	228	118
4	13	1028	953	116	328	123
5	11	1452	1317	149	470	110
6	10	1592	1449	165	497	106
7	10	1762	1581	143	501	106
8	9	2335	1976	217	633	97
9	9	2434	1989	269	606	110
10	8	2775	2392	309	762	92
11	8	2951	2531	310	775	95
12	8	3570	3200	384	1029	95
13	8	4197	3676	363	1172	103
14	7	4159	3604	377	1183	92
15	6	3961	3529	391	1087	84
16	6	4301	3612	333	1198	88
17	7	6035	5182	498	1711	104
18	6	5352	4790	578	1550	93
19	6	5897	5156	626	1655	89
20	6	6343	5478	518	1758	89
21	6	6578	5692	661	1875	89
22	6	6709	6212	685	1925	92
23	6	8376	7213	583	2346	98
24	5	7051	6148	659	1937	80

Table 4.12: DSA counts based on preconditioner type for the Iron-Water problem with LS_4 quadrature, quadratic basis functions, and solution reinitialization.

Cycle	SI Iter.	Preconditioner Type				
		None	Jacobi	SSOR	ILU	AGMG
0	18	681	553	93	108	18
1	14	902	750	120	202	69
2	14	1358	1196	163	245	78
3	12	1464	1277	216	265	65
4	12	1977	1704	283	349	60
5	11	2858	2379	536	494	56
6	11	3813	3210	714	653	66
7	11	4450	3784	902	779	112
8	11	6465	5403	1259	1105	167
9	11	7541	6370	1691	1328	155
10	11	9107	7477	1948	1557	146
11	11	10535	8648	2554	1793	174
12	11	11843	9662	2660	2016	166
13	11	12810	10481	2960	2167	158
14	11	14227	11790	3179	2451	178
15	11	14888	12411	3422	2598	177
16	11	15652	12902	3408	2690	185
17	11	16327	13723	4120	2820	189
18	11	17611	14647	4105	3006	177
19	11	18922	15392	4319	3192	179
20	11	19457	16009	4527	3304	179
21	11	20099	16798	5032	3467	178
22	11	21359	17446	5198	3599	183
23	11	22000	17880	5397	3702	188
24	11	22363	18465	5539	3815	167

Table 4.13: DSA counts based on preconditioner type for the Iron-Water problem with LS_4 quadrature, quadratic basis functions, and solution bootstrapping.

Cycle	SI Iter.	Preconditioner Type				
		None	Jacobi	SSOR	ILU	AGMG
0	18	681	553	93	108	18
1	14	885	722	84	195	70
2	13	1315	1132	112	246	78
3	11	1383	1195	170	242	66
4	10	1722	1451	218	292	60
5	10	2663	2201	326	461	61
6	10	3509	2971	448	611	67
7	9	3918	3072	386	651	102
8	8	4919	4070	621	837	129
9	8	5643	4658	776	969	117
10	8	6770	5639	848	1160	123
11	9	8430	6727	1028	1430	146
12	8	8611	7236	1015	1492	132
13	8	9382	7725	1112	1606	125
14	7	8915	7495	1065	1559	120
15	7	9646	7776	1295	1623	120
16	7	10235	8594	1336	1788	124
17	7	10577	8442	1127	1747	131
18	7	11190	9431	1540	1949	116
19	7	11867	9672	1553	2069	125
20	7	12494	10271	1540	2090	131
21	7	12724	10963	1668	2259	118
22	7	13367	11471	1375	2370	137
23	7	12816	10481	1398	2207	127
24	7	14850	12149	1706	2560	125

Table 4.14: DSA counts based on preconditioner type for the Iron-Water problem with S_{24}^2 PGLC quadrature, linear basis functions, and solution reinitialization.

Cycle	SI Iter.	Preconditioner Type				
		None	Jacobi	SSOR	ILU	AGMG
0	19	360	282	55	95	19
1	20	592	478	70	161	20
2	17	794	664	90	225	122
3	14	958	850	123	310	117
4	13	1571	1386	163	443	137
5	12	2170	1926	259	597	124
6	12	2308	2075	298	668	131
7	11	3269	2837	416	901	131
8	11	3836	3376	491	1093	130
9	10	3853	3290	570	1072	118
10	10	4563	4062	694	1264	115
11	10	4625	4288	694	1331	119
12	10	4932	4458	724	1386	119
13	10	5028	4557	744	1422	122
14	10	6507	5831	968	1822	124
15	10	6431	5936	1021	1873	123
16	10	8128	6994	1212	2186	125
17	10	8510	7421	1276	2353	128
18	10	8974	8043	1198	2571	131
19	10	8999	8110	1226	2615	127
20	10	9242	8236	1447	2654	132
21	10	9304	8327	1367	2710	132
22	10	11464	10143	1807	3249	135
23	10	11629	10272	1815	3286	135
24	10	11956	10565	1974	3386	134

Table 4.15: DSA counts based on preconditioner type for the Iron-Water problem with S_{24}^2 PGLC quadrature, linear basis functions, and solution bootstrapping.

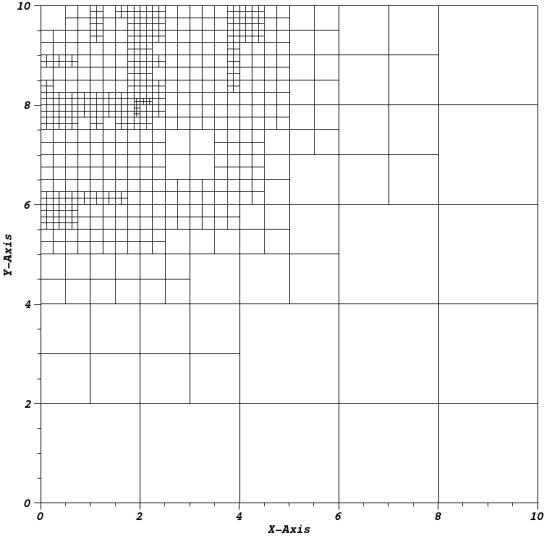
Cycle	SI Iters.	Preconditioner Type				
		None	Jacobi	SSOR	ILU	AGMG
0	19	360	282	55	95	19
1	21	629	512	70	171	21
2	16	787	681	75	227	118
3	13	991	875	93	298	112
4	11	1271	1064	121	357	128
5	10	1879	1682	168	508	119
6	10	1883	1605	177	526	115
7	9	2419	2168	217	675	99
8	8	2427	2100	271	712	88
9	8	2624	2325	282	720	88
10	7	3170	2723	327	846	82
11	7	3132	2700	325	874	81
12	7	3069	2470	332	841	79
13	7	3310	2850	327	933	74
14	7	3547	3340	418	1037	77
15	7	4307	3889	360	1214	76
16	6	4505	3992	604	1301	72
17	6	5036	4460	501	1425	73
18	6	5162	4580	544	1467	75
19	6	5236	4572	552	1459	69
20	6	5525	4849	643	1523	72
21	6	5597	4876	613	1579	74
22	6	6249	5652	770	1880	72
23	6	6202	5537	666	1805	72
24	6	5918	5375	741	1717	72

Table 4.16: DSA counts based on preconditioner type for the Iron-Water problem with S_{24}^2 PGLC quadrature, quadratic basis functions, and solution reinitialization.

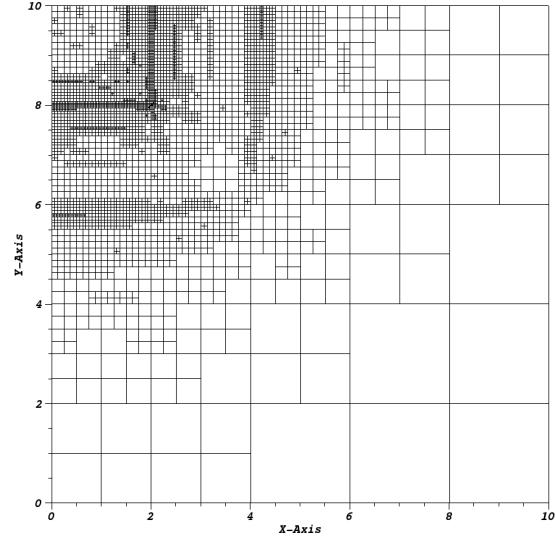
Cycle	SI Iter.	Preconditioner Type				
		None	Jacobi	SSOR	ILU	AGMG
0	17	432	518	47	109	19
1	16	710	592	76	192	79
2	15	953	794	144	219	84
3	13	1150	958	197	239	70
4	14	1885	1571	261	339	70
5	13	2604	2170	414	487	66
6	12	2770	2308	477	594	72
7	12	3923	3269	666	708	122
8	11	4603	3836	786	921	167
9	10	4624	3853	912	1006	141
10	10	5476	4563	1110	1180	133
11	10	5550	4625	1110	1358	158
12	10	5918	4932	1158	1527	151
13	10	6034	5028	1190	1642	144
14	10	7808	6507	1549	1857	162
15	10	7717	6431	1634	1968	161
16	10	9754	8128	1939	2038	168
17	10	10212	8510	2042	2136	172
18	10	10769	8974	1917	2277	161
19	10	10799	8999	1962	2418	163
20	10	11090	9242	2315	2503	163
21	10	11165	9304	2187	2627	162
22	10	13757	11464	2891	2727	166
23	10	13955	11629	2904	2805	171
24	10	14347	11956	3158	2890	152

Table 4.17: DSA counts based on preconditioner type for the Iron-Water problem with S_{24}^2 PGLC quadrature, quadratic basis functions, and solution bootstrapping.

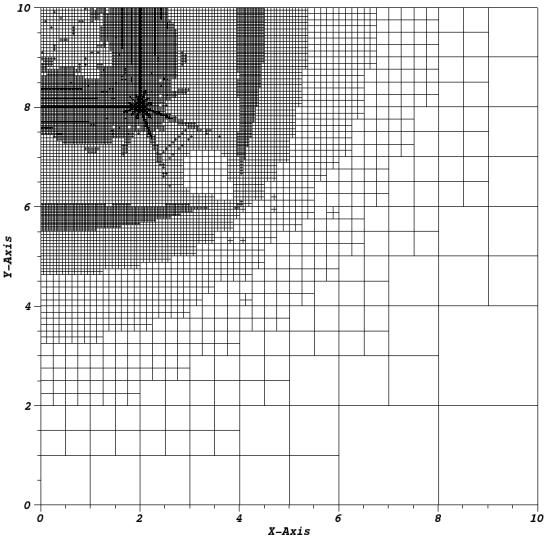
Cycle	SI Iters.	Preconditioner Type				
		None	Jacobi	SSOR	ILU	AGMG
0	17	432	518	47	109	19
1	14	652	737	57	176	69
2	12	801	1134	105	184	67
3	10	929	1449	163	193	54
4	10	1414	1987	205	254	50
5	10	2103	2909	277	393	51
6	8	1939	2784	350	416	48
7	9	3089	3955	551	558	92
8	8	3515	4215	544	703	121
9	8	3884	5981	674	845	113
10	7	4025	5614	843	867	93
11	8	4662	6274	865	1141	126
12	8	4971	7819	862	1283	121
13	7	4435	7467	959	1207	101
14	7	5739	7428	1037	1365	113
15	7	5672	8906	928	1446	113
16	7	7169	10259	1308	1498	118
17	7	7506	10420	1381	1570	120
18	7	7915	10927	1391	1674	113
19	6	6803	10102	1349	1523	98
20	6	6987	10368	1311	1577	98
21	6	7034	10574	1619	1655	97
22	6	8667	11796	1819	1718	100
23	6	8792	10624	1290	1767	103
24	5	7532	10376	756	1517	76



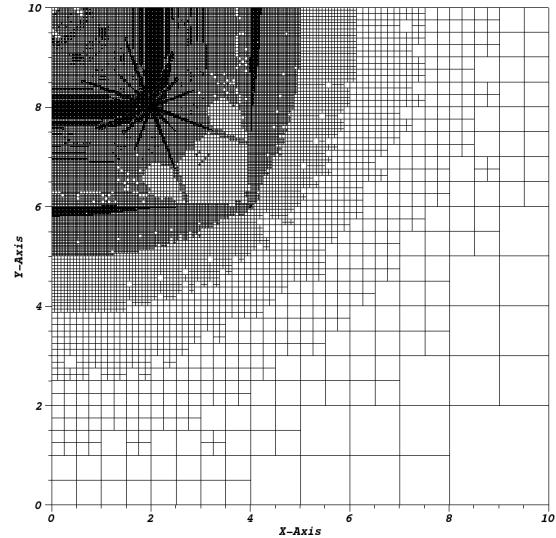
(a) Cycle #6



(b) Cycle #12

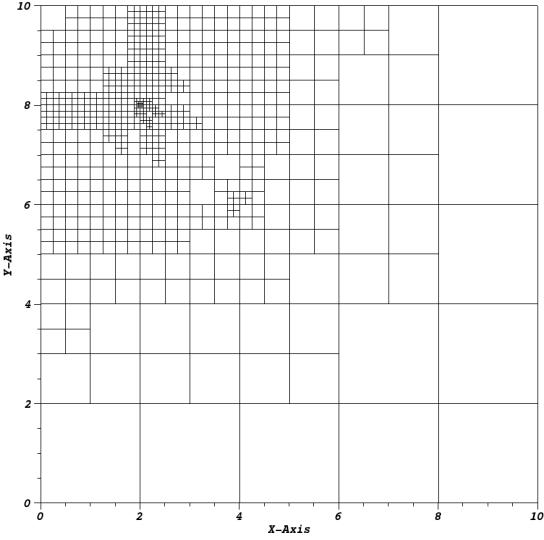


(c) Cycle #18

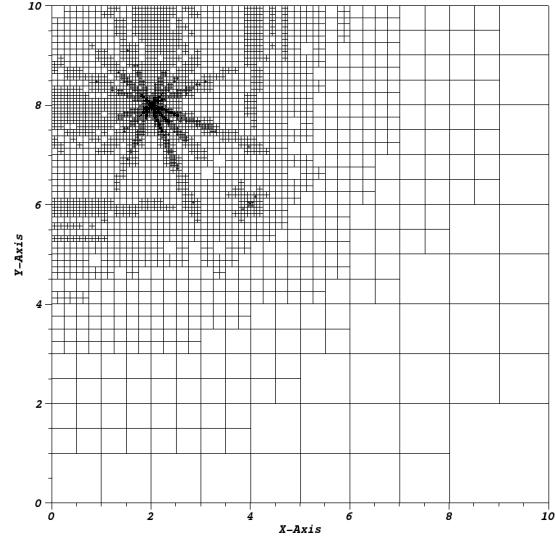


(d) Cycle #24

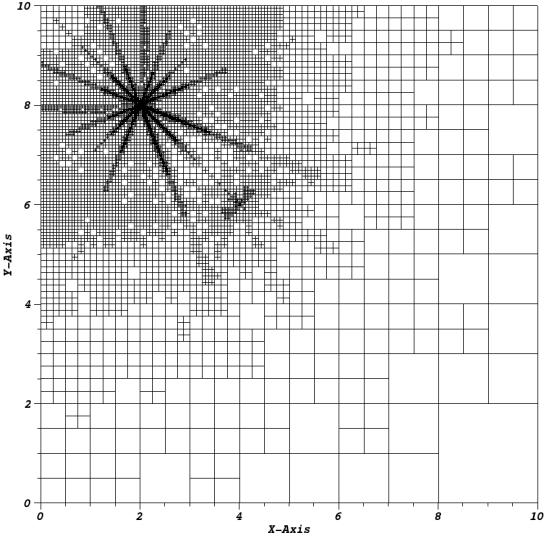
Figure 4.23: Meshes for the Iron-Water problem using the linear PWL coordinates and LS4 quadrature.



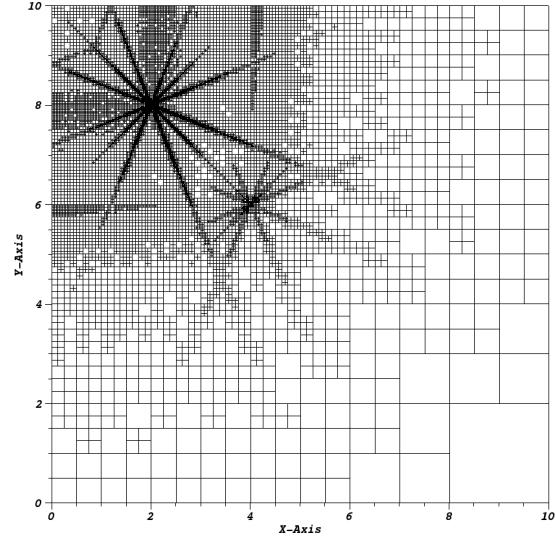
(a) Cycle #6



(b) Cycle #12

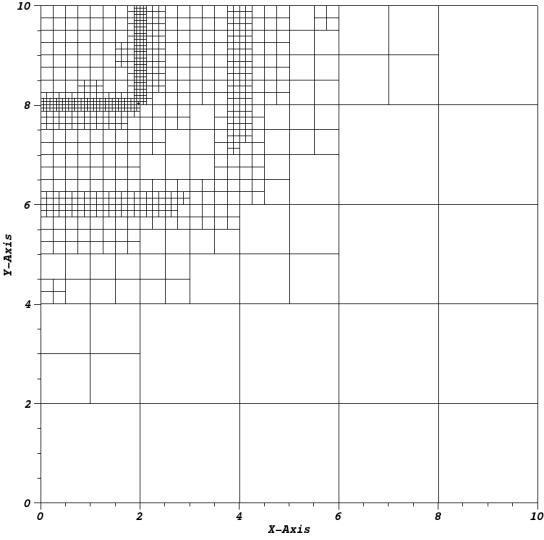


(c) Cycle #18

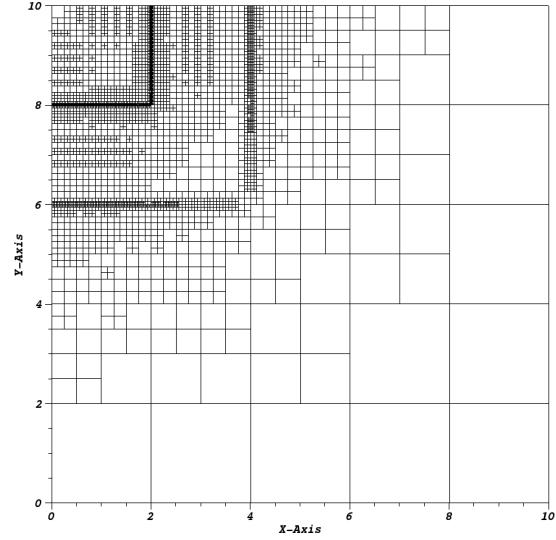


(d) Cycle #24

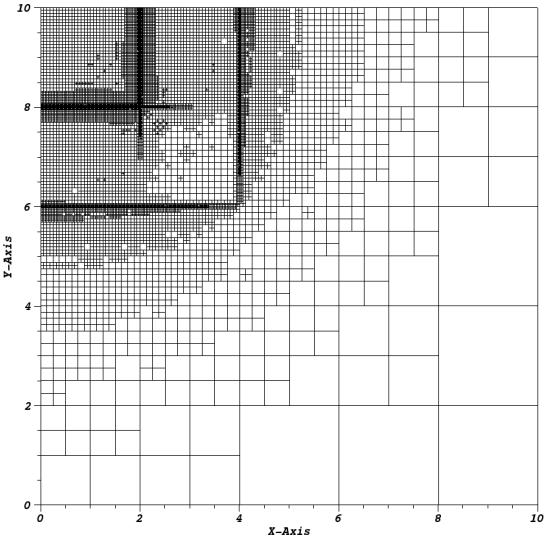
Figure 4.24: Meshes for the Iron-Water problem using the quadratic PWL coordinates and LS4 quadrature.



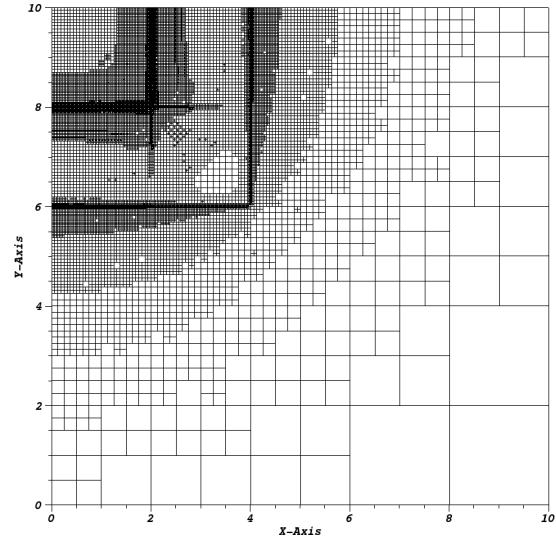
(a) Cycle #6



(b) Cycle #12

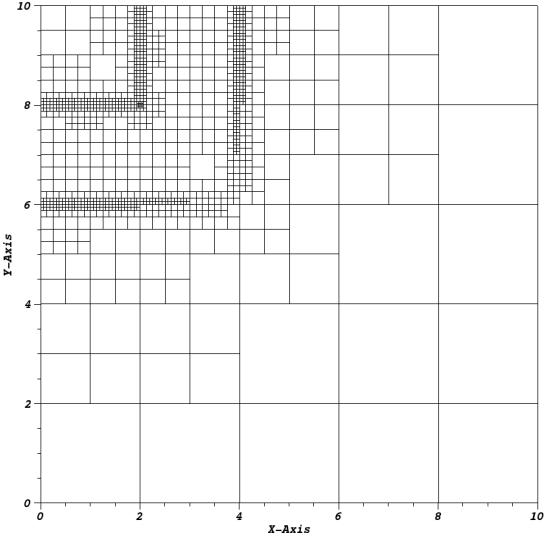


(c) Cycle #18

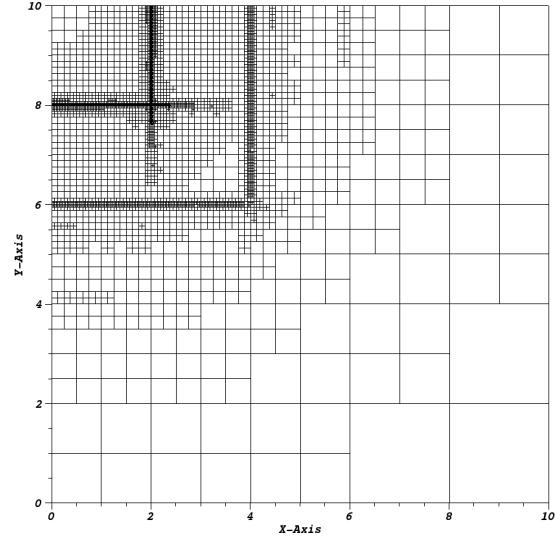


(d) Cycle #24

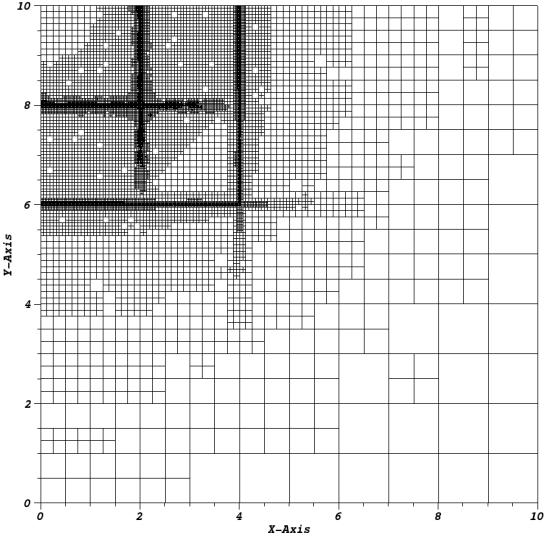
Figure 4.25: Meshes for the Iron-Water problem using the linear PWL coordinates and S_{24}^2 PGLC quadrature.



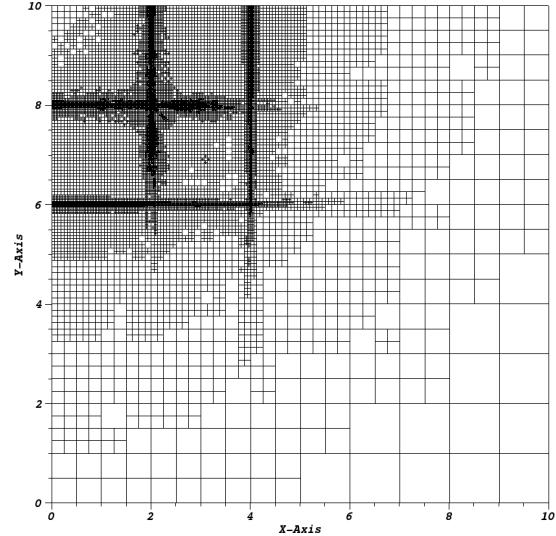
(a) Cycle #6



(b) Cycle #12



(c) Cycle #18



(d) Cycle #24

Figure 4.26: Meshes for the Iron-Water problem using the quadratic PWL coordinates and S_{24}^2 PGLC quadrature.

4.7.3 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 configurations can diminish the effectiveness of MIP DSA.

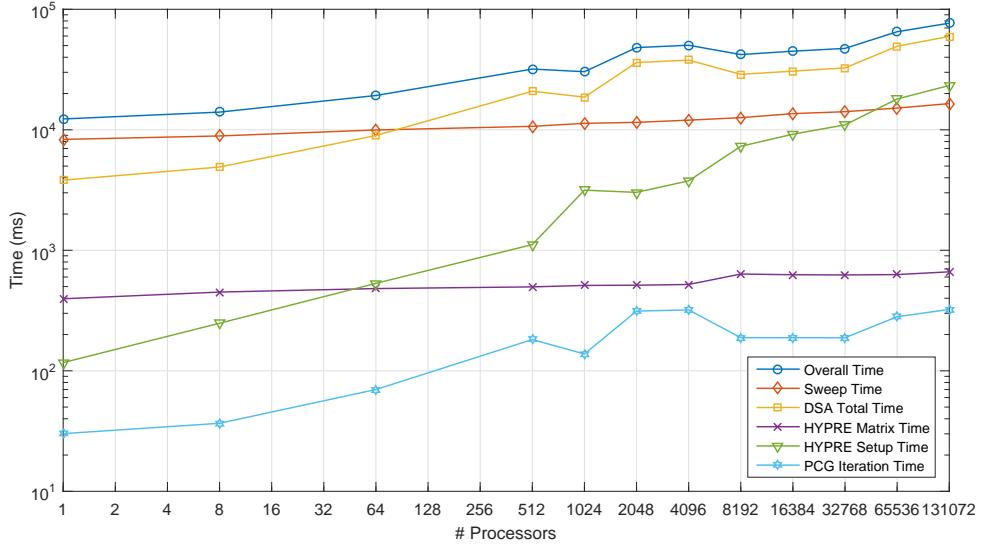
Now, we need to demonstrate the scalability of MIP DSA preconditioning onto large massively-parallel computer architectures. Our ability to utilize this transport acceleration form would be greatly diminished if the DSA solve times scaled at a much worse rate compared to the transport sweep times. We 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 4.7.2.4, 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.

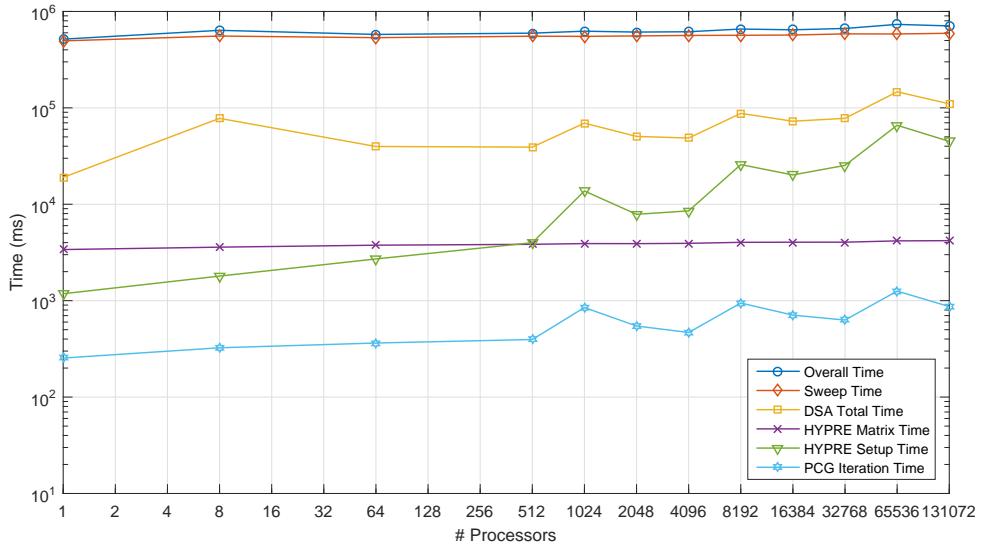
We have implemented the 1-group and thermal upscattering DSA methodologies of Section 4.2 with the MIP form into Texas A&M University’s PDT code. It is a massively-parallel DGFEM S_N transport code that has had good sweep scaling efficiency out to $O(10^5) - O(10^6)$ processes [102, 103]. BoomerAMG of the HYPRE library is used as the AMG diffusion preconditioner [170, 171]. We next analyze the scalability of HYPRE’s PCG solver with BoomerAMG preconditioning and how this

performs in comparison to PDT’s transport sweeping.

We test MIP’s scaling with HYPRE in PDT by analyzing a simple homogenized version of the Zerr weak-scaling problem [98]. 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 $1 \frac{n}{cm^3 s}$. Vacuum boundary conditions are used for all boundaries, and Richardson Iteration is performed. We perform two different weak-scaling runsets. The first contains 512 spatial cells per processor, and the other contains 4096 spatial cells per processor. For each of these mesh cell numbers, we vary the number of angular directions performed in a sweep. For the 512 cell run, we analyze angular quadratures containing 128, 512, 2048, and 8192 directions. For the 4096 cell run, we analyze angular quadratures containing 128, 512, and 2048 directions. First, we present a pair of results from the extremes of these runsets. Figure 4.27 provides the timing data for two cases. The first case is a 512 cell run with 128 angular directions. The second case is a 4096 cell run with 2048 angular directions. In each figure, we provide the timing results for the overall solve time, the sweep time, the overall DSA time, the time required to build the HYPRE matrix, the time to perform a HYPRE setup call, and the PCG iteration time. From both figures, we can clearly see that the DSA routines lose scaling efficiency at a faster rate than the transport sweep routines at high processor counts. However, we emphasize the results of the 4096 cell and 2048 direction run. We can see that as we give more work to the sweep algorithm, the fraction of time spent in the DSA routines is significantly reduced. This behavior is captured in Figure 4.28 for the fraction of time spent in DSA for all the runs. For a high-fidelity simulation that must mitigate ray effects (*i.e.*, requiring $O(10^3)$ - $O(10^5)$ angular directions), the time spent performing DSA routines will stay to a minimum even at high processor counts.

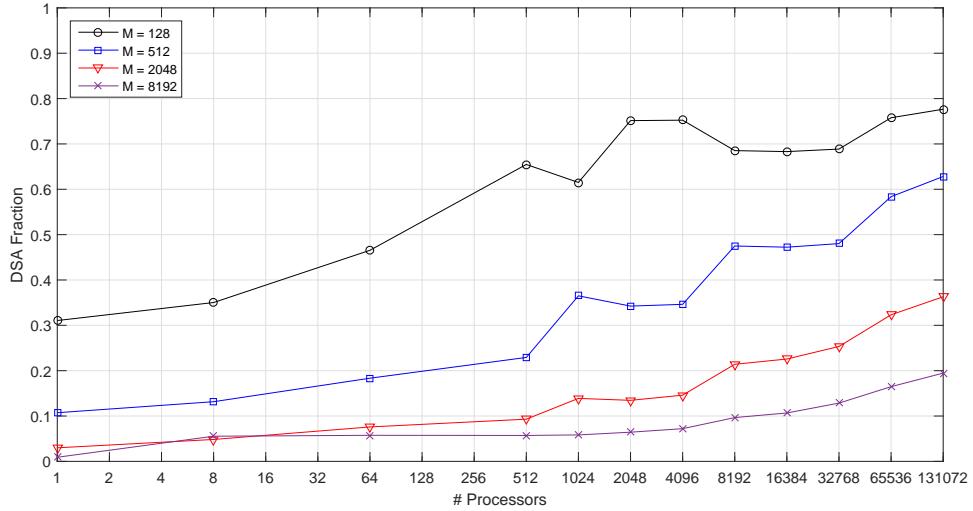


(a) 128 angles and 512 cells per processor

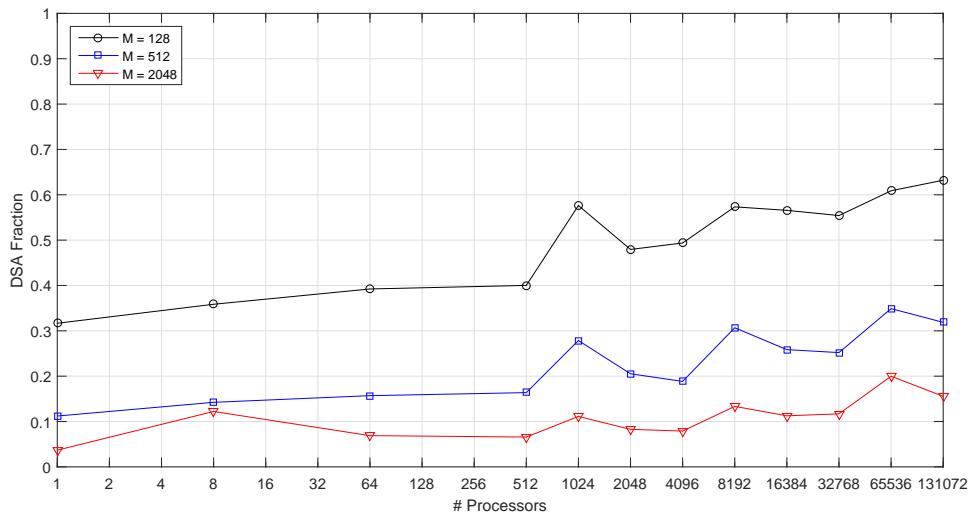


(b) 2048 angles and 4096 cells per processor

Figure 4.27: Timing data for the MIP DSA implementation in PDT using HYPRE on VULCAN.



(a) 512 cells per processor



(b) 4096 cells per processor

Figure 4.28: Fraction of time spent performing DSA based on number of total angles.

4.7.4 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. All the results presented in this section center around the Impurity Model 1 (IM1) experiments performed by the Center for Exascale Radiation Transport (CERT) group at Texas A&M University. The IM1 experiment seeks to quantify the amount of impurities present in a set of graphite bars by treating them as $1/E$ absorbers. The materials present in the experiment consist of graphite, air, wood, high-density polyethylene (HDPE), borated high-density polyethylene (B-HDPE), Steel, Boral, an AmBe source, and a BF₃ detector. The experimental configuration can be seen in Figure 4.29. The AmBe source sits inside an HDPE cylinder surrounded by a B-HDPE shell. A graphite bar is placed above the HDPE cylinder and sits between the AmBe source and the BF₃ detector. A 99 energy group structure with 42 fast and 57 thermal groups is used. For these results, we first present the theoretical Fourier Analysis of the different upscattering acceleration methods in Section 4.7.4.1. Then, numerical analysis of a simple 2D variant of the experimental setup is performed in Section 4.7.4.2 to yield a complete set of runs for the different methods. We analyze the iteration counts and timing statistics for the different methods. Finally, we conclude in Section 4.7.4.3 with a numerical analysis of the 3D configuration for an approximate hexahedral geometry and one formed by triangular prisms (extrusion of a 2D triangular mesh).

4.7.4.1 Fourier Analysis for the Upscatter Acceleration Methods

Before we perform any numerical analysis, we first analyze the theoretical limits of the different upscatter acceleration methods. For each of the methods, a spectral-collapsed DSA step is performed. The spectral shape of the energy collapse is gen-

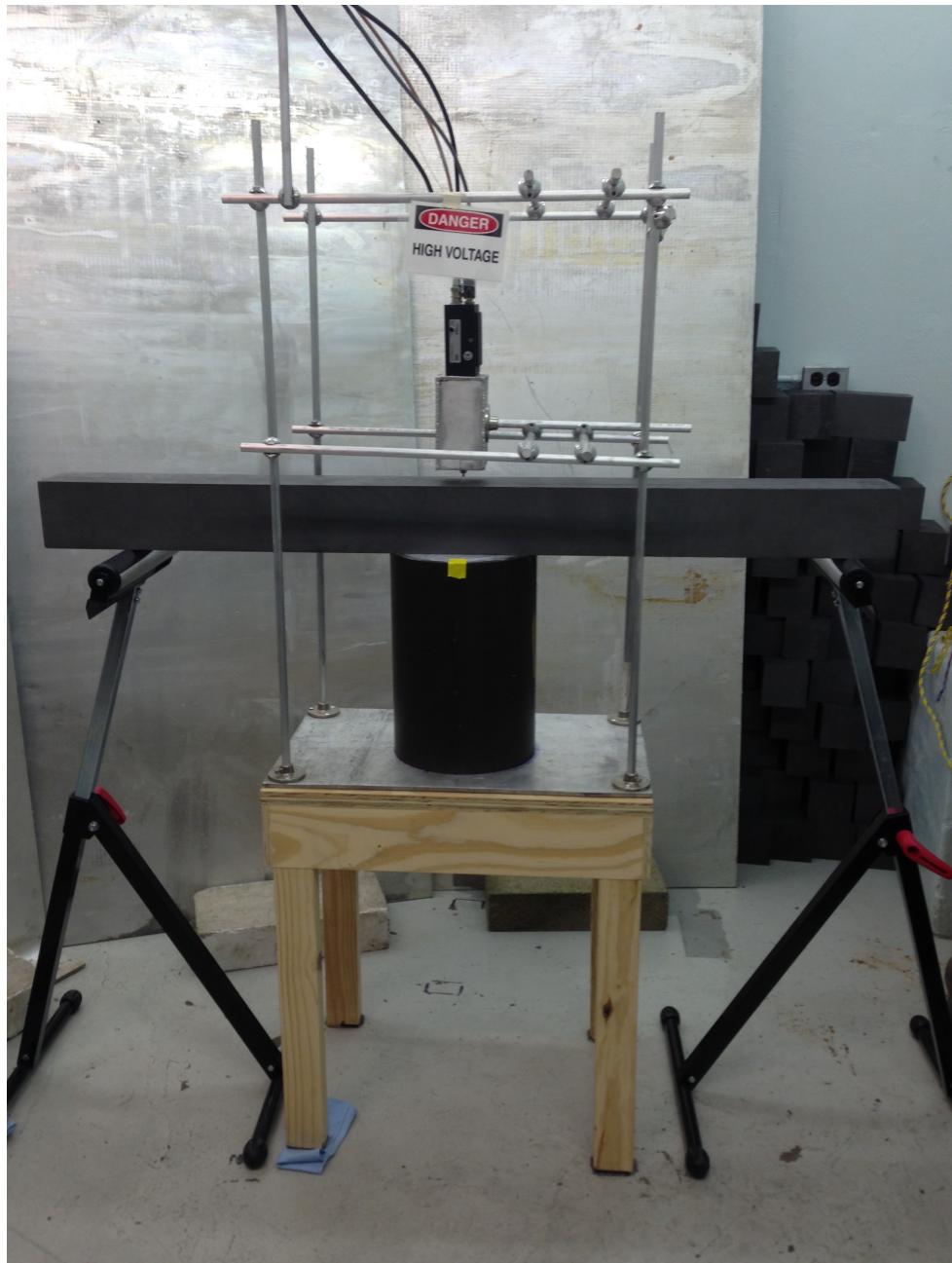
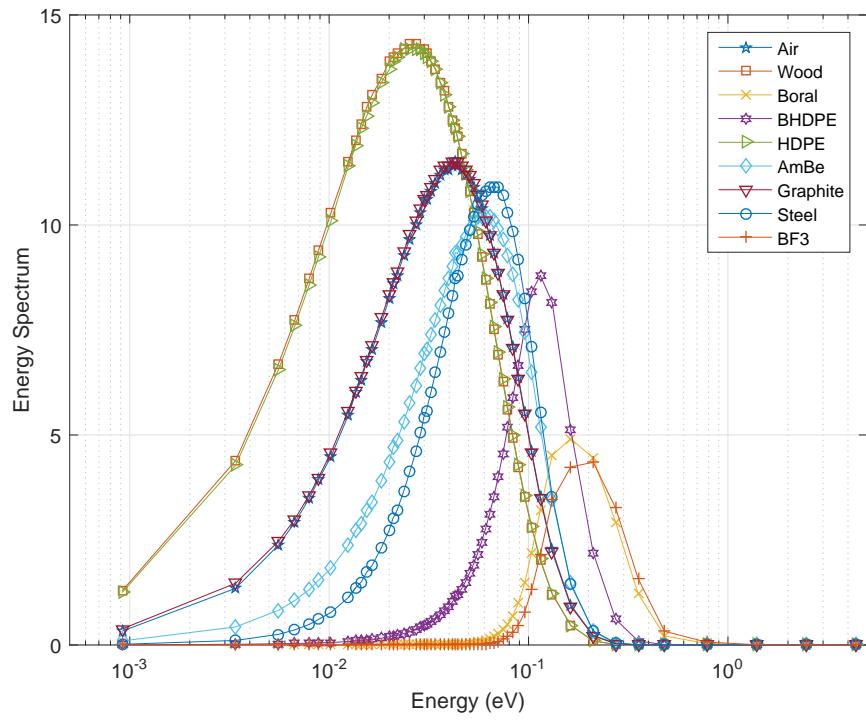


Figure 4.29: Experimental setup of the IM1 problem.

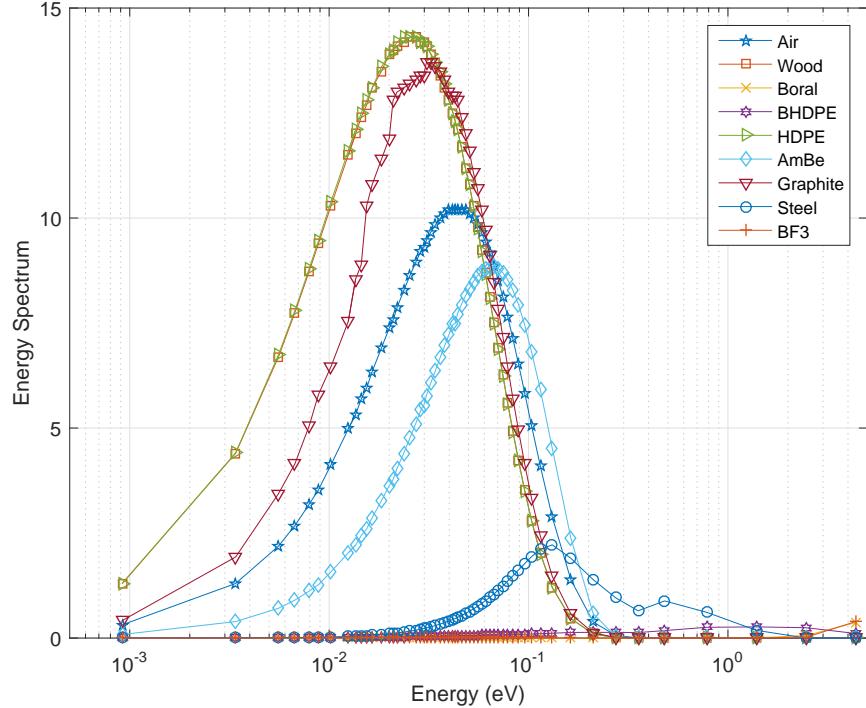
erated by the infinite medium iteration matrices. Figure 4.30 gives the spectral shapes for each of the IM1 experiment materials for the TG and MTG methods. Then, Figure 4.31 gives the spectral shapes for each of the IM1 experiment materials for the MJA and MJIA methods. These spectral shapes were only computed over the thermal energy groups. For the TG method, we can clearly see that each of the spectral shapes is a shifted Maxwellian spectrum. The materials with higher absorption (Boral and BF₃) harden the spectral shape to higher thermal energies.

Next, we provide a set of homogeneous Fourier Analysis results in Table 4.18. For the TG, MTG, MJA, and MJIA methods, the table gives the infinite medium FA results for each IM1 material for both unaccelerated and accelerated cases. These infinite medium analyses correspond to the Fourier flat mode. From this table, we can determine the theoretical convergence rates for each of the methods. We clearly see that the TG and MJIA methods have the best theoretical convergence properties because of their low spectral radii. However, even though the MTG and MJA methods yield less optimal spectral radii, they still decrease the graphite spectral radius from about 0.999 to about 0.96.

For each of these methods, we can then gain additional knowledge by analyzing the spectrum of the flat mode eigenvalues. Figure 4.32 provides the eigenvalue spectra for the TG and MTG methods for graphite with 99 energy groups. For TG, we can see in the unaccelerated case that all the eigenvalues except one are well away from 1.0. The single eigenvalue close to 1.0 corresponds to the thermal Maxwellian error mode. Performing the energy-collapsed diffusion solve (accelerated case) attenuates this mode and leaves the largest eigenvalue with a value of 0.4084 remaining. For MTG, we can see a very different eigenvalue distribution. Because the inner iterations are not converged for MTG, the eigenvalues are clustered closer to 1.0 compared to TG. However, there is still the thermal Maxwellian error mode with a value of 0.9993



(a) Two-Grid



(b) Modified Two-Grid

Figure 4.30: Spectral shape of the infinite medium iteration matrices of the IM1 problem materials for the TG and MTG schemes.

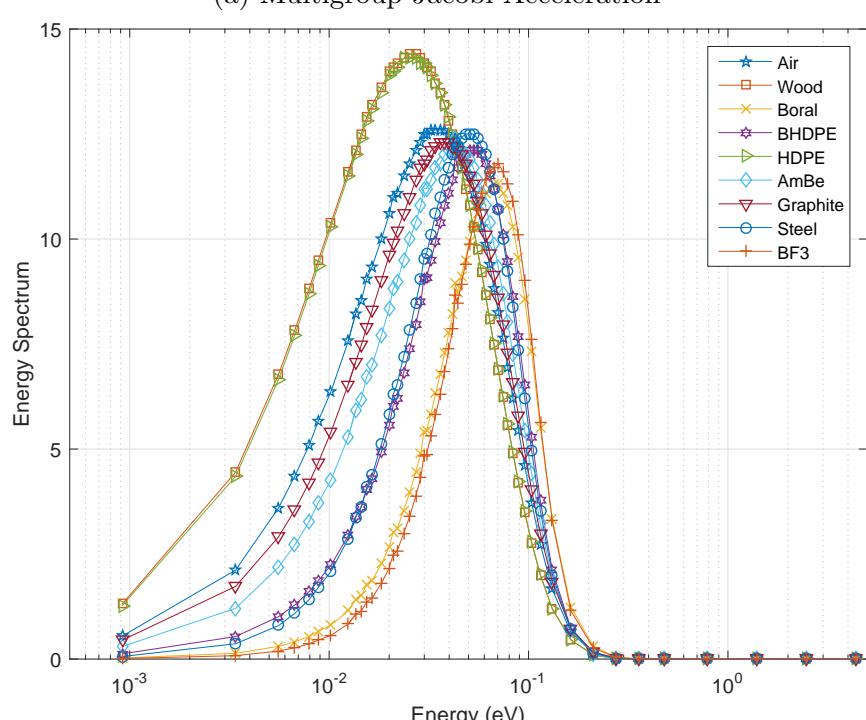
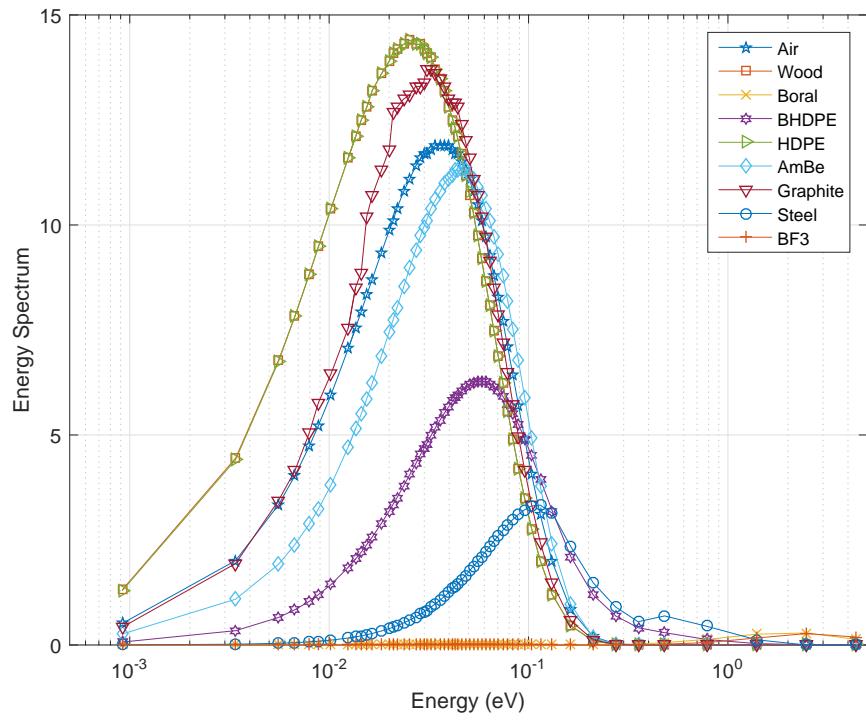


Figure 4.31: Spectral shape of the infinite medium iteration matrices of the IM1 problem materials for the MJA and MJIA schemes.

Table 4.18: Infinite medium spectral radii of the IM1 materials for the three thermal neutron upscattering methods. We include both the unaccelerated (U) and accelerated (A) cases.

Material	U. TG	A. TG	U. MTG	A. MTG	U. MJA	A. MJA	A. MJIA
Graphite	0.9883	0.4084	0.9993	0.9604	0.9993	0.9613	0.6462
HDPE	0.8916	0.4343	0.9918	0.7527	0.9943	0.8015	0.6631
B-HDPE	0.0258	0.0177	0.1331	0.1221	0.1336	0.1223	0.0639
Wood	0.9820	0.2101	0.9840	0.3836	0.9915	0.5326	0.4684
AmBe	0.4835	0.2724	0.5646	0.5554	0.7068	0.5596	0.4947
Steel	0.6989	0.5809	0.9243	0.9215	0.9255	0.9215	0.7547
Boral	0.0023	0.0016	0.0782	0.0602	0.0782	0.0602	0.0039
BF3	0.0008	0.0006	0.0351	0.0266	0.0351	0.0266	0.0086
Air	0.7580	0.5282	0.8121	0.7828	0.8845	0.7896	0.7166

which the acceleration step attenuates. Therefore, the MTG method reduces the dominating eigenmode from a value of 0.9993 to 0.9604.

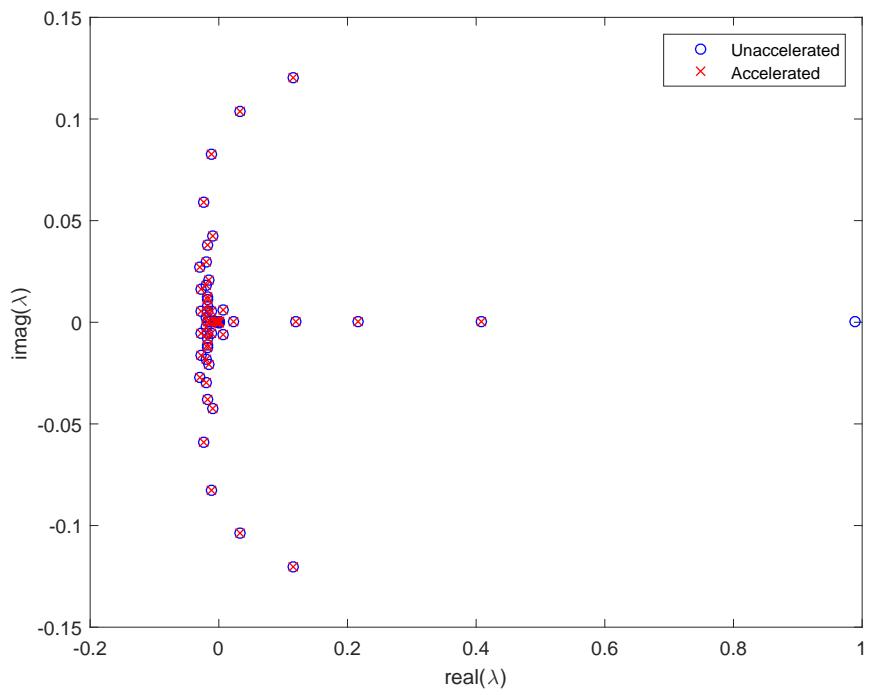
Next, we analyze the flat mode eigenvalues of the two Jacobi methods. Figure 4.33 provides the eigenvalue spectra for the MJA and MJIA methods for graphite with 99 energy groups. The distribution of the eigenvalues for both the unaccelerated and accelerated MJA method is similar to MTG. The eigenmodes are clustered close to 1.0 with a single dominant eigenmode corresponding to the thermal Maxwellian at 0.9993. The energy-collapsed acceleration step attenuates the 0.9993 mode and leaves a dominant eigenmode of 0.9613. Finally, we analyze the flat eigenmodes of the MJIA method and notice similarities to both TG and MJA. The unaccelerated eigenmodes (blue circles) are identical to MJA (the transport sweep is identical). However, performing the G within-group DSA calculations (red x) moves most of the eigenvalues close to 0. There is still a single dominant eigenmode left around 1 corresponding to the thermal Maxwellian. Performing the energy-collapsed acceler-

ation step (black squares) then attenuates this error mode and leaves the remaining largest eigenvalue of 0.6462.

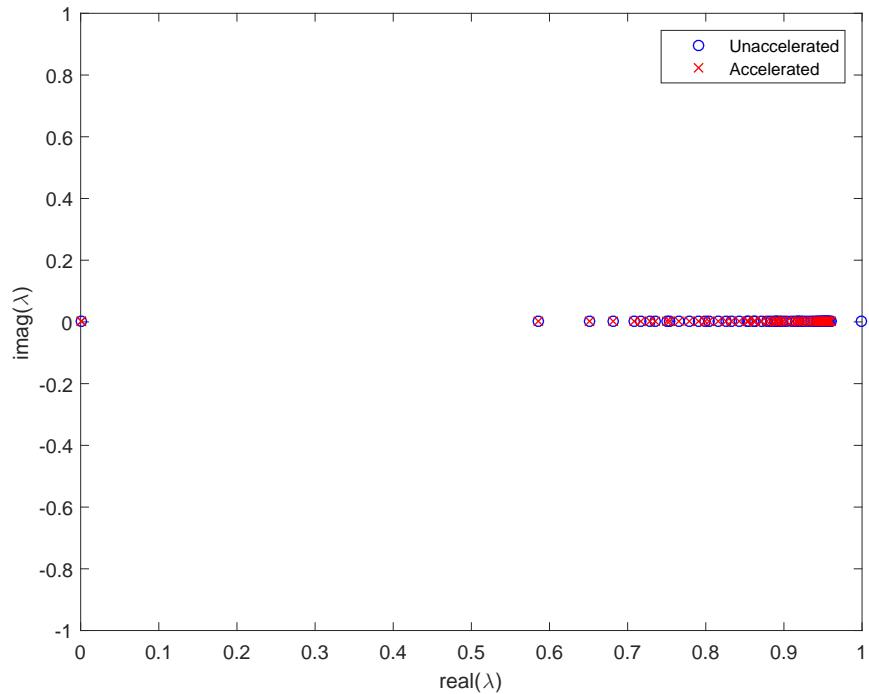
We now briefly discuss an issue that may arise for some heterogeneous configurations. Section 4.7.2.3 detailed the PHI problem and demonstrated that the MIP DSA scheme can lose effectiveness for problems with large material discontinuities. From the materials in the IM1 experiment, we have several materials with thermal group total cross sections of $O(10^0)$ - $O(10^1)$, and air has thermal group total cross sections of $O(10^{-4})$. These cross section values are in the range of those presented in the PHI problem results. Therefore, we posit that this behavior could arise for these upscatter acceleration methods. We numerically verify this a little later in Sections 4.7.4.2 and 4.7.4.3. However, we first provide a simple Fourier Analysis example to show that this behavior could arise. We ran a Phi-like TG problem with a 2×2 cell layout and a domain of $[0, 1]^2$. The top two cells are air and the bottom two cells are HDPE. The spectral radius of this problem is about 0.9, which is about the same as the unaccelerated TG results. Figure 4.34 provides the Fourier wave number distribution for this problem. We can see that this wave number distribution closely matches the results obtained in the PHI analysis.

4.7.4.2 2D IM1 Results

We first wish to perform numerical testing of the upscatter acceleration methods on a simple 2D configuration. This simple problem seeks to emulate some of the geometric configuration of the experimental model. Figure 4.35 provides the layout of this problem along with the mesh used. The left face is a reflecting boundary. We restrict the IM1 materials to only include graphite, HDPE, air, and the AmBe source. From Figure 4.35, we can see that the configuration mimics the radial nature of the IM1 experiment. Going outwards, we have the inner air gap with the AmBe

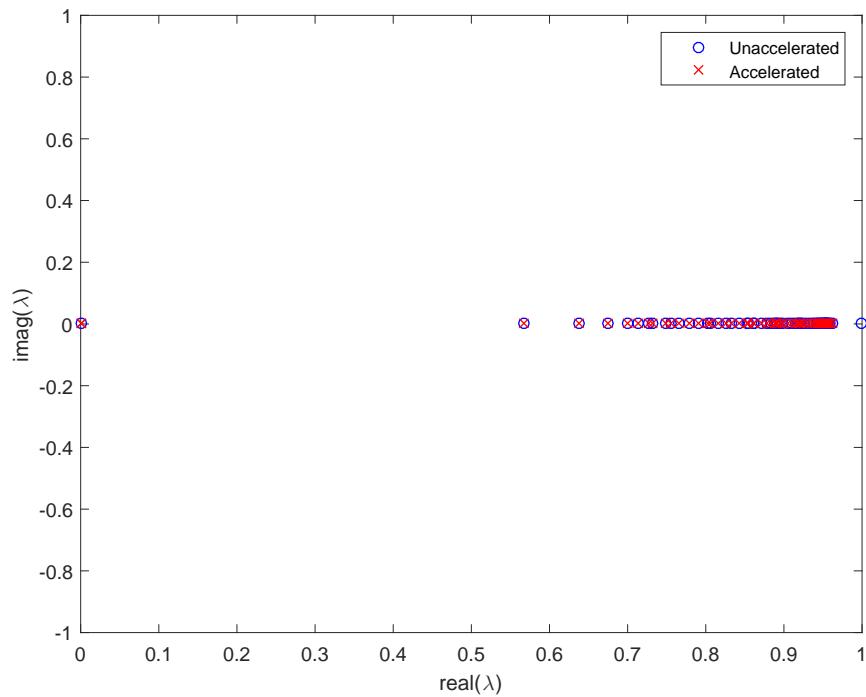


(a) Two-Grid

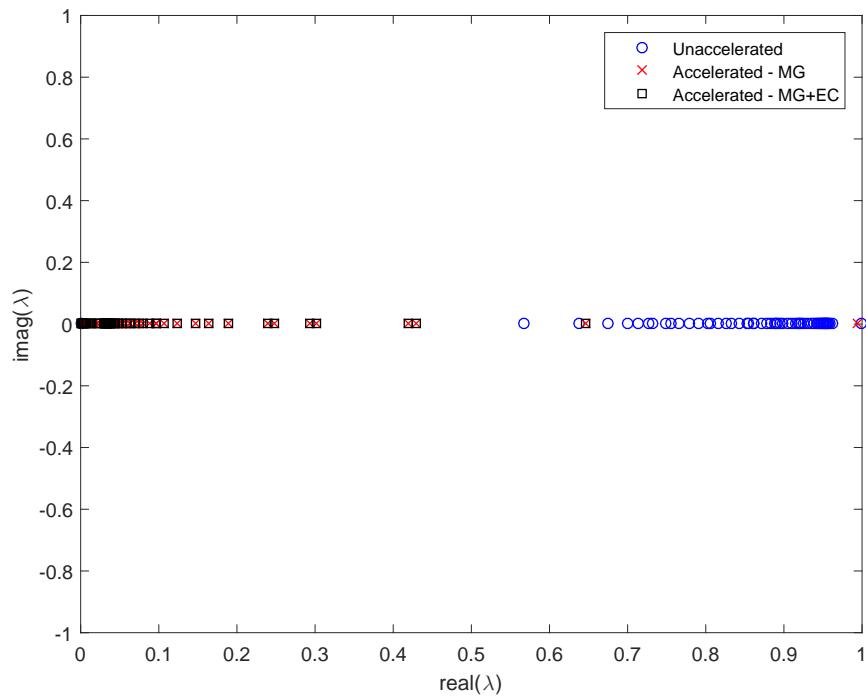


(b) Modified Two-Grid

Figure 4.32: Flat mode eigenvalues for the TG and MTG schemes.

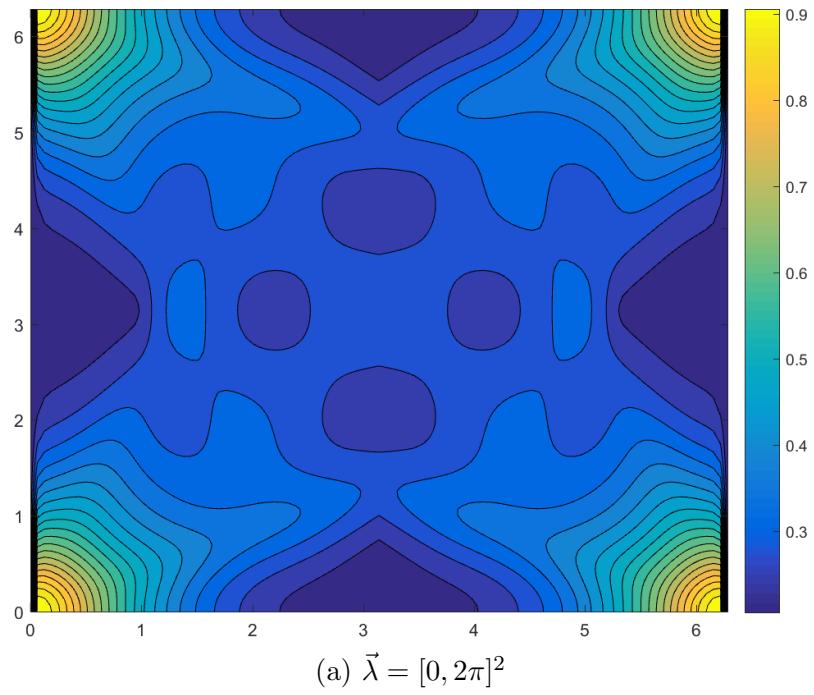


(a) Multigroup Jacobi Acceleration

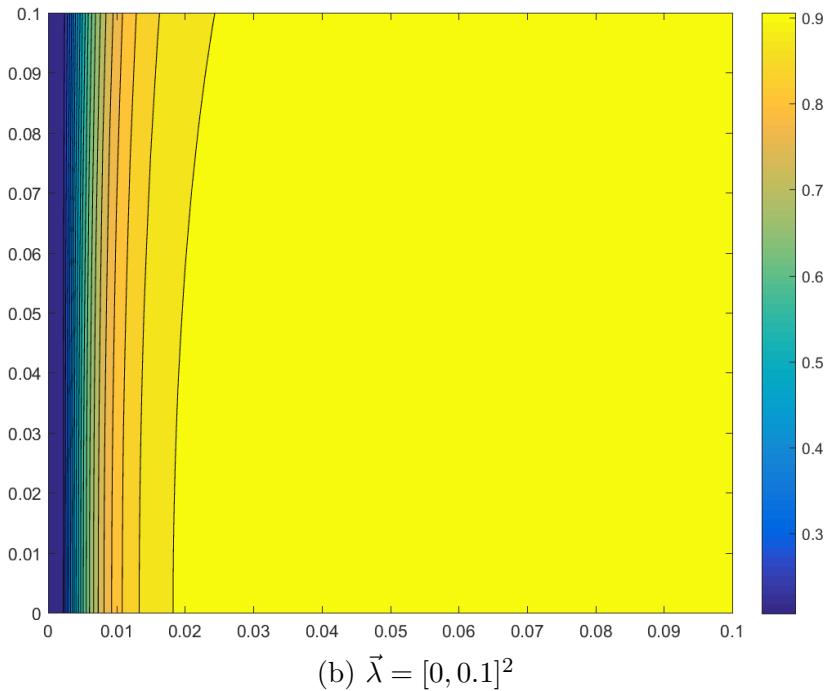


(b) Multigroup Jacobi with Inner Acceleration

Figure 4.33: Flat mode eigenvalues for the MJA and MJIA schemes.



(a) $\vec{\lambda} = [0, 2\pi]^2$



(b) $\vec{\lambda} = [0, 0.1]^2$

Figure 4.34: Fourier wave form distribution for a PHI-like problem of the TG method with Air and HDPE.

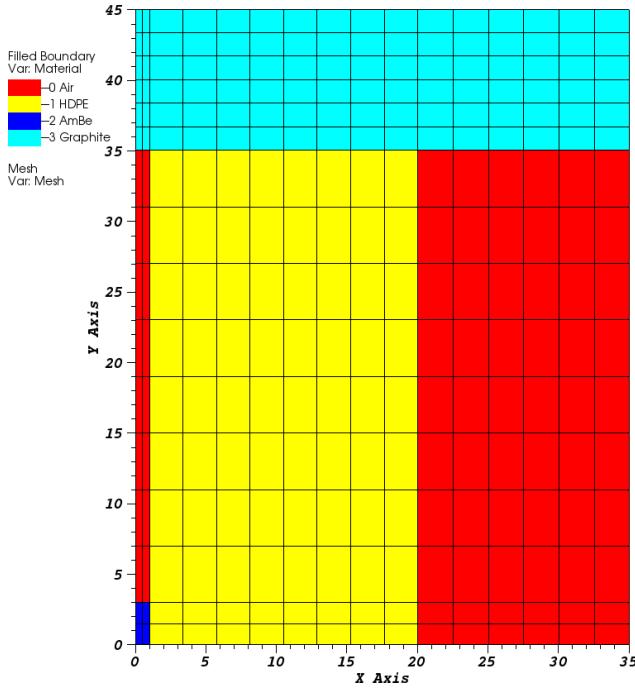


Figure 4.35: Configuration of the 2D variant of the IM1 problem. The materials are restricted to only air, HDPE, graphite, and an AmBe source.

source, a “ring” of HDPE, and an outer air “ring”. A graphite brick sits on top of the configuration.

For this problem, we analyzed the iteration and timing results for the TG, MTG, and MJA methodologies with both Richardson Iteration and GMRES. The outer iteration tolerance was set to 10^{-6} , and the inner tolerance for the TG method was set to 10^{-7} . 64 processors were used for all runs. Tables 4.19, 4.20, and 4.21 provide the full set of iteration and timing results for the TG, MTG, and MJA methods, respectively. For each of the upscatter acceleration methods, we performed accelerated and unaccelerated Richardson (SI) and GMRES runs. We can clearly see several results from the tables which we summarize as the following:

1. The MJA method provides the best performance in regards to solution times.

Table 4.19: Sweep count and timing data for the 2D IM1 variant problem using Two-Grid Acceleration.

Problem	Outer Iter.	1-Group Sweeps	Solve Time (min)
SI	361	185,422	486.50
SI+DSA	55	35,699	96.48
GMRES	38	41,575	128.63
GMRES+DSA	14	19,053	57.92

Table 4.20: Sweep count and timing data for the 2D IM1 variant problem using Modified Two-Grid Acceleration.

Problem	Outer Iter.	1-Group Sweeps	Solve Time (min)
SI	536	77,632	275.83
SI+DSA	73	10,846	40.60
GMRES	78	4,845	25.82
GMRES+DSA	26	1,881	11.09

Table 4.21: Sweep count and timing data for the 2D IM1 variant problem using Multigroup Jacobi Acceleration.

Problem	Outer Iter.	1-Group Sweeps	Solve Time (min)
SI	1,734	98,838	111.07
SI+DSA	157	8,949	15.49
GMRES	118	6,726	8.65
GMRES+DSA	35	1,995	4.16

The parallel nature of the transport sweeps with all thermal groups in a group set provides significant savings compared to the TG and MTG methods.

2. There is a significant penalty for the serial-in-energy nature of the TG and MTG methods. Even when less transport sweeps are performed than MJA, it yields significantly longer wallclock times.
3. A Krylov (GMRES) iteration scheme is needed to account for the PHI problem behavior and yield effective DSA preconditioning. This is easily visible because the SI runs require significantly more iterations (more than what infinite medium analysis predicts). Instead, the accelerated GMRES runs recover the theoretical convergence behaviors.

4.7.4.3 3D IM1 Results

We conclude the investigation of the thermal neutron upscattering methodologies by presenting the numerical results pertaining to analysis of the 3D configuration of the IM1 experiment. The PDT model of the problem is given in Figure 4.36 where we have removed the outer layers of air for visualization purposes. The domain is filled with Cartesian parallelepipeds, and the cylinders and sphere are modeled as mass-preserving parallelepipeds and a cube, respectively. Reflecting boundaries are employed so that the runs are quarter geometry to minimize memory footprints.

The runset for this analysis is similar to that performed for the 2D variant problem. We give the iteration and timing results for the brick-geometry IM1 problem in Tables 4.22, 4.23, and 4.24 for the TG, MTG, and MJA, respectively. All runs were performed on 3496 processors. We note that not all of the data points are populated due to prohibitive run times. This is why the 2D variant was essential to provide a complete set of results for all algorithm and method types. However, from the

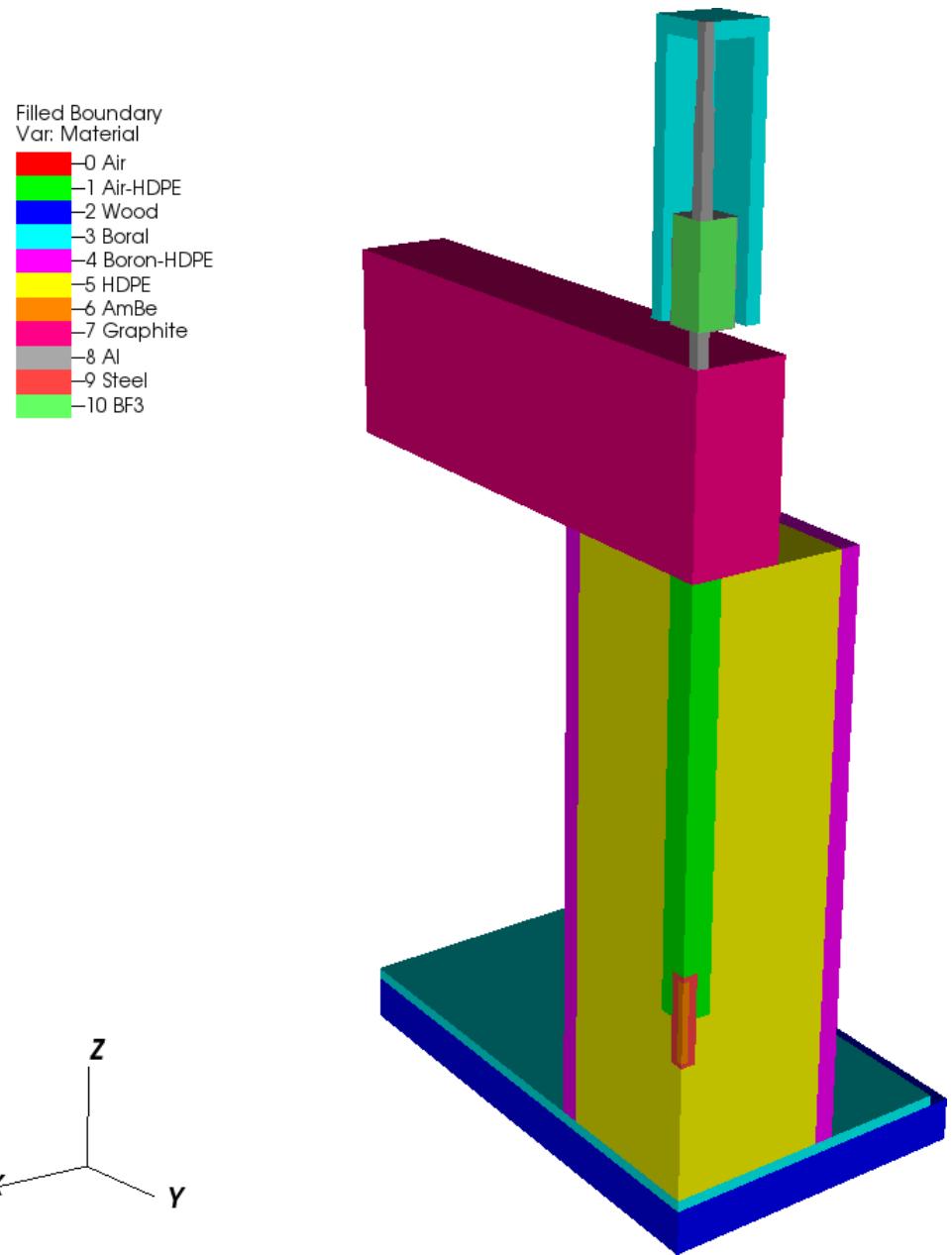


Figure 4.36: Configuration of the IM1 problem with the outer layers of air removed.

Table 4.22: Sweep count and timing data for the 3D brick IM1 problem using Two-Grid Acceleration.

Problem	Outer Iter.	1-Group Sweeps	Solve Time (hr)
SI	-	-	-
SI+DSA	-	-	-
GMRES	-	-	-
GMRES+DSA	14	11,104	32.5

Table 4.23: Sweep count and timing data for the 3D brick IM1 problem using Modified Two-Grid Acceleration.

Problem	Outer Iter.	1-Group Sweeps	Solve Time (hr)
SI	-	-	-
SI+DSA	-	-	-
GMRES	81	4,617	14.2
GMRES+DSA	34	1,938	4.82

Table 4.24: Sweep count and timing data for the 3D brick IM1 problem using Multi-group Jacobi Acceleration.

Problem	Outer Iter.	1-Group Sweeps	Solve Time (hr)
SI	-	-	-
SI+DSA	256	14,592	11.2
GMRES	120	6,840	4.93
GMRES+DSA	31	1,767	1.76

data points that were obtainable, we can determine that the results for the different algorithms and methodologies matches the 2D variant problem closely. The serial natures of the TG and MTG methods in their Gauss-Seidel process in energy are prohibitively slow for problems at these scales. Only the accelerated GMRES TG case could be completed as well as the unaccelerated and accelerated GMRES MTG cases. We can see similar convergence properties (similar iteration counts) to the 2D variant problem for all the methods and both algorithms. However, it is once again apparent that the parallel concurrence of the transport sweeps with the MJA method leads to the best performance.

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. The MIP scheme is an extension of the Symmetric Interior Penalty diffusion form compatible with DSA schemes. We analyzed both a 1-group DSA scheme and a suite of two-grid methodologies to accelerate problems dominated by thermal neutron upscattering: the Two-Grid Acceleration scheme, the Modified Two-Grid Acceleration scheme, the Multigroup Jacobi Acceleration, and the Multigroup Jacobi with Inner Acceleration scheme. Extensive Fourier Analysis was performed for the 1-group scheme in 2D and 3D configurations. All of the 2D linear and quadratic basis functions analyzed showed effective and robust behavior from the fine-mesh to the coarse-mesh limits. The scheme was also analyzed for heterogeneous configurations with the 2D PWL basis functions. The PHI problem was analyzed and the degradation of the DSA scheme was observed for large material discontinuities. DSA preconditioning was also shown to be effective for diffusive problems in conjunction with spatial Adaptive Mesh Refinement.

The efficacy of the low-order diffusion operator for the use in massively-parallel transport calculations was also analyzed. The scalability of the MIP DSA form was studied on the VULCAN supercomputer. The methodology was implemented in the Texas A&M University PDT code, and the HYPRE software was linked to solve the MIP equations. Scaling results demonstrated that the MIP diffusion form scales well out to $O(10^5)$ processors. When sufficient work is performed with the transport sweeps (*i.e.*, a large number of transport unknowns per processor), the fraction of time spent performing DSA operations can be as low as 10% - 20%. At high processor counts, the scaling of the MIP solves begins to degrade. Specifically, the HYPRE setup call time begins to grow at a high rate compared to the transport sweeps. The time per PCG solve grows as well but at a slower rate. We note that the time to build the MIP matrix in HYPRE remains flat in a weak scaling sense.

Finally, we analyzed the theoretical and numerical performances of the different thermal neutron upscattering acceleration methodologies. Fourier analysis states that the TG and MJIA methods provide the optimal convergence properties. However, the TG method contains the serial treatment of the energy groups, whereas MJIA does not. The MJIA method is completely parallelizable, and we predict its parallel concurrence will yield the maximum possible parallel efficiency. Numerical timing results of the TG, MTG, and MJA methods determined that MJA provides the best performance in terms of reduction in wall clock times (at the time of this work, the MJIA method had not been implemented in the PDT code). The TG and MTG methods pay a steep penalty with their serialization in energy with the Gauss-Seidel iterations. Even though fewer outer iterations are performed with TG and MTG compared to MJA, the MJA experienced up to factor of 3 speedups. Thus, the suite of MJA and MJIA methodologies constitutes the most efficient parallel upscatter acceleration schemes in a massively-parallel setting.

5. CONCLUSIONS

5.1 Conclusions

In this dissertation, we have performed work to advance the state-of-the-art in solving the DGFEM S_N transport equation on polytope meshes with massively-parallel computer architectures. We have done this by investigating two different topical areas:

1. The use of higher-order basis functions for transport calculations on arbitrary polygonal grids.
2. Analysis of DSA schemes to accelerate the within-group and thermal neutron upscattering iterations.

For the work involving basis functions for transport calculations on arbitrary polygonal grids, we first investigated four different linearly-complete polygonal coordinate systems. Then, we investigated the methodology to convert these linearly-complete functions into the quadratic serendipity space of functions. Higher-order FEM basis functions have two-fold benefits. One, they achieve enhanced convergence rates with mesh refinement due to their higher-dimensional interpolation space. Two, they can enhance parallel computing efficiency by decreasing memory-access bottlenecks by giving the CPUs more data to compute per solve. Currently, memory-access calls are the limiting bottleneck on massively-parallel computer architectures.

Following the work on the basis functions on polygonal meshes, we analyzed the performance of DSA schemes to accelerate the within-group and thermal neutron upscattering iterations. We sought to analyze the performance and scalability of the MIP diffusion discretization out to high processor counts. We then presented

a novel, parallelizable methodology to accelerate the thermal neutron upscattering iterations.

We tested both of these topical areas through theoretical and numerical analyses. Our conclusions involving the work on the polygonal basis functions are listed below:

1. All of the linearly-complete and quadratically-complete basis functions capture the thick diffusion limit on structured and polygonal meshes. The difference between the discretized transport and diffusion solutions in the L_2 -norm converge at a rate of $O(\epsilon)$.
2. All of the linearly-complete polygonal basis functions can capture an exactly-linear transport solution, even on highly-distorted polygonal meshes.
3. All of the quadratically-complete serendipity polygonal basis functions can capture an exactly-quadratic transport solution that lives in the $\{1, x, y, x^2, xy, y^2\}$ space of functions. This corresponds to the first three levels of Pascal's triangle. The method cannot capture the $\{x^2y, xy^2, x^2y^2\}$ space of functions, unlike the Q9 elements, since we form the quadratic functions by taking pairwise products of the $\{1, x, y\}$ space of functions.
4. We also tested various numerical problems to determine the convergence rates of the proposed linear and quadratic basis functions. We specifically wanted to analyze the convergence rates that would be constrained by the regularity of the transport solutions. This was done using both MMS and a purely-absorbing media problem where the exact analytical solutions are known. For the MMS problems, we observed the theoretically maximum convergence rates of $p + 1$ since the transport solutions had infinite regularity. For the purely-absorbing media case, the solution of the S_N transport equation lived in either the $H^{1/2}$ or

$H^{3/2}$ space. For meshes that are not aligned with the singularity, convergence rates of $1/2$ and $3/2$ were observed. However, if the meshes were aligned with the singularity, then convergence rates of $p + 1$ were recovered.

Next, our conclusions for the DSA work on massively-parallel computer architectures are listed below:

1. The Symmetric Interior Penalty discretization of the diffusion equation is an efficient solver for the 3D diffusion problem using DFEM. It is Symmetric Positive Definite, and its system matrix is easily solvable with a Preconditioned Conjugate Gradient Method. We demonstrated that linearly-complete polyhedral basis functions can capture exactly-linear diffusion solutions on arbitrary grids. We also showed that the proper second-order convergence is obtained in the L_2 -norm.
2. The Modified Interior Penalty method is a modification to the SIP scheme that can be used as a DFEM diffusion form for DSA calculations. It is also SPD and easily solvable the same as SIP. Fourier and numerical analysis showed that it is efficient and robust on parallelepipeds, even those with high aspect ratios. However, this efficiency degrades with large discontinuities across material cross sections.
3. MIP DSA was successfully implemented in the Texas A&M University massively-parallel S_N sweeping code: PDT. The HYPRE library was linked, and their PCG solver and BoomerAMG preconditioner were used to solve the MIP system matrix. Scaling studies were performed on the VULCAN supercomputer out to $O(10^5)$ processes. It was shown that, as the angular quadrature was refined to $O(10^3)$ number of angles, the DSA solve times became minimal com-

pared to the sweep times. This means that for high-fidelity transport calculations involving $O(10^3) - O(10^4)$ and $O(10^2)$ energy groups, the DSA computational times become insignificant. Therefore, the use of DSA as the low-order acceleration operator for massively-parallel transport problems is realizable.

4. We tested four different thermal neutron upscattering acceleration schemes: the Two-Grid scheme, the Modified Two-Grid scheme, the Multigroup Jacobi Acceleration scheme, and the Multigroup Jacobi with Inner Acceleration scheme. The MJA and MJIA methods are fully-parallelizable in energy, whereas TG and MTG are serialized in energy. Theoretical Fourier and numerical analysis was performed to understand the convergence properties of these methods for infinite medium and periodic domains. It was shown that TG provides the best spectral radius of all of the methods tested for several materials. However, because of its serialization in energy using Gauss-Seidel along with the necessity to converge the inner iterations, it had the worst performance in terms of wall clock time. The MTG scheme had better run-time performance than TG because only one sweep was performed for each energy group in the Gauss-Seidel iteration. However, MJA had the best run-time performance because the inversion of the thermal group loss operators could be done simultaneously with a single transport sweep. No numerical results were presented for the MJIA method since its implementation in PDT had not been completed at the time of this work. However, because of its Fourier results, it shows great promise as a methodology to accelerate thermal neutron upscattering.

5.2 Open Items

While the work in this dissertation answered several open questions related to the calculation of the DGFEM S_N transport equation on massively-parallel architectures,

several items remain for ongoing research. We now list the open items that we have identified:

1. *Quadratic serendipity basis functions on 3D polyhedra:*

The direct extension of the work involving the 2D quadratic serendipity basis functions would be to form quadratically-complete, analogous serendipity coordinates for arbitrary 3D polyhedra. To maintain quadratic completeness in 3D, the coordinates would be beholden to the ten quadratic 3D constraints which would require exact interpolation of the $\{1, x, y, z, xy, xz, yz, x^2, y^2, z^2\}$ span of functions. Along a polyhedral face, the methodology presented in Chapter 3 has a direct 3D analogue. However, careful consideration is required to remove all of the diameter nodes within the polyhedron and is an open area of research in the applied mathematics community.

2. *Higher-order 2D serendipity polygonal basis functions:*

The quadratic serendipity basis functions were formed by taking pairwise products of the linear barycentric basis functions, followed by removal of the interior nodes. For a given polynomial order p , the monomials that the basis functions need to exactly interpolate are $x^\sigma y^\tau$, where $\sigma + \tau = p$. We can see that all of the higher-order functional spaces can be generated by taking pairwise products of terms from lower-order functions. Mukherjee and Webb have just recently developed a means to generate these higher-order polygonal finite elements through a hierarchical approach [172]. Their methodology can be immediately implemented and analyzed for the DGFEM S_N transport equation.

3. *Alternate integration schemes on polygons:*

For this work, our quadrature integration scheme on arbitrary polygons consisted of a simple triangulation scheme where each sub-triangle had points

mapped onto it from the reference triangle. We did not focus on efficiency for this work, but instead simply used a high-order reference quadrature set. However, by performing our integration this way, the basis function values and gradients must be computed for each polygon in the mesh. This becomes computationally expensive for meshes with many cells containing polygons with large vertex counts. An alternative approach could consist of the use of Schwarz-Christoffel Conforming Maps (SCCM) [173, 174]. Generation of the polygonal basis functions and gradients could be computed on reference (regular) polygons and then conformally mapped to any arbitrary polygon for integration [175].

4. *Implementing MJIA method and performing tests at scale:*

In this work, only the TG, MTG, and MJA methods were numerically implemented and verified. The next step is to implement the MJIA method in the PDT code where the only addition is the implementation of the G within-group diffusion solves.

5. *Mixed-mode parallelism with DSA preconditioning:*

In this work, spatial parallelism was done with domain decomposition using the Message Passing Interface (MPI). No shared memory parallelism was utilized. A more advanced parallelization methodology involves the use of a mixed-mode parallel methodology. This would consist of a shared-memory location (an MPI rank) where several processes could operate concurrently in parallel. This methodology is currently being implemented in the sweep portion of the PDT code. To maximize the efficiency of modern computer architectures, the DSA calculations would also need to utilize this mixed-mode scheme.

REFERENCES

- [1] K. BERGMAN, S. BORKAR, D. CAMPBELL, W. CARLSON, W. DALLY, M. DENNEAU, P. FRANZON, W. HARROD, K. HILL, J. HILLER, ET AL., “Exascale computing study: Technology challenges in achieving exascale systems,” Tech. rep. (2008).
- [2] C. JOHNSON, U. NÄVERT, and J. PITKÄRANTA, “Finite element methods for linear hyperbolic problems,” *Computer Methods in Applied Mechanics and Engineering*, **45**, 1–3, 285–312 (1984).
- [3] J. E. MOREL and J. S. WARSA, “An S_n spatial discretization scheme for tetrahedral meshes,” *Nuclear Science and Engineering*, **151**, 2, 157–166 (2005).
- [4] STAR-CCM+, “<http://www.cd-adapco.com>,” (2015).
- [5] M. YIP, J. MOHLE, and J. BOLANDER, “Automated Modeling of Three-Dimensional Structural Components Using Irregular Lattices,” *Computer-Aided Civil and Infrastructure Engineering*, **20**, 6, 393–407 (2005).
- [6] A. ERN and J.-L. GUERMOND, *Theory and practice of finite elements*, vol. 159, Springer Science & Business Media (2013).
- [7] J. J. DUDESTADT and L. J. HAMILTON, *Nuclear reactor analysis*, Wiley (1976).
- [8] G. I. BELL and S. GLASSTONE, *Nuclear reactor theory*, RE Krieger Publishing Company (1979).
- [9] M. CHADWICK, P. OBLOŽINSKÝ, M. HERMAN, N. GREENE, R. MCKNIGHT, D. SMITH, P. YOUNG, R. MACFARLANE, G. HALE, S. FRANKLE, ET AL., “ENDF/B-VII. 0: next generation evaluated nuclear data li-

brary for nuclear science and technology,” *Nuclear Data Sheets*, **107**, 12, 2931–3060 (2006).

- [10] M. CHADWICK, M. HERMAN, P. OBLOŽINSKÝ, M. E. DUNN, Y. DANON, A. KAHLER, D. L. SMITH, B. PRITYCHENKO, G. ARBANAS, R. ARCILLA, ET AL., “ENDF/B-VII. 1 nuclear data for science and technology: cross sections, covariances, fission product yields and decay data,” *Nuclear Data Sheets*, **112**, 12, 2887–2996 (2011).
- [11] K. SHIBATA, T. KAWANO, T. NAKAGAWA, O. IWAMOTO, J.-I. KATAKURA, T. FUKAHORI, S. CHIBA, A. HASEGAWA, T. MURATA, H. MATSUNOBU, ET AL., “Japanese evaluated nuclear data library version 3 revision-3: JENDL-3.3,” *Journal of Nuclear Science and Technology*, **39**, 11, 1125–1136 (2002).
- [12] A. KONING, R. FORREST, M. KELLETT, R. MILLS, H. HENRIKSSON, Y. RUGAMA, ET AL., *The JEFF-3.1 nuclear data library*, OECD (2006).
- [13] R. MACFARLANE, D. MUIR, and R. BOISCURT, “NJOY-99 nuclear data processing system,” *Los Alamos National Laboratory, Theoretical Division* (2002).
- [14] A. KAHLER, D. MUIR, R. BOISCURT, and R. MACFARLANE, “The NJOY Nuclear Data Processing System, Version 2012,” *Los Alamos National Laboratory, Theoretical Division* (2012).
- [15] M. DUNN and N. GREENE, “AMPX-2000: A cross-section processing system for generating nuclear data for criticality safety applications,” *Transactions of the American Nuclear Society*, **86**, 118–119 (2002).

- [16] G. ALIBERTI, G. PALMIOTTI, M. SALVATORES, T. KIM, T. TAIWO, M. ANITESCU, I. KODELI, E. SARTORI, J. BOSQ, and J. TOMMASI, “Nuclear data sensitivity, uncertainty and target accuracy assessment for future nuclear systems,” *Annals of Nuclear Energy*, **33**, 8, 700–733 (2006).
- [17] M. A. JESSEE, *Cross-section adjustment techniques for BWR adaptive simulation*, Ph.D. thesis, North Carolina State University (2008).
- [18] B. CARLSON and K. LATHROP, *Computing methods in reactor physics: Transport Theory - The Method of Discrete Ordinates*, Gordon and Breach Science Publishers, Inc. (1968).
- [19] E. E. LEWIS and W. F. MILLER, *Computational methods of neutron transport*, John Wiley and Sons, Inc., New York, NY (1984).
- [20] E. GELBARD, “Application of spherical harmonics method to reactor problems,” Tech. rep., Bettis Atomic Power Laboratory, WAPD-BT-20 (1960).
- [21] K. D. LATHROP, “Ray effects in discrete ordinates equations,” *Nuclear Science and Engineering*, **32**, 3, 357–369 (1968).
- [22] K. LATHROP, “Remedies for ray effects,” *Nuclear Science and Engineering*, **45**, 3, 255–268 (1971).
- [23] R. L. J.E. MOREL, T.A. WAREING and D. PARSONS, “Analysis of ray-effect mitigation techniques,” *Nuclear Science and Engineering*, **144**, 1–22 (2003).
- [24] W. H. REED and T. HILL, “Triangular mesh methods for the neutron transport equation,” *Los Alamos Report LA-UR-73-479* (1973).
- [25] J. MOREL and J. MCGHEE, “A self-adjoint angular flux equation,” *Nuclear Science and Engineering*, **132**, 3, 312–325 (1999).

- [26] J. ASKEW, “A characteristics formulation of the neutron transport equation in complicated geometries,” Tech. rep., United Kingdom Atomic Energy Authority (1972).
- [27] R. E. ALCOUFFE and E. W. LARSEN, “Review of characteristic methods used to solve the linear transport equation,” Tech. rep., Los Alamos Scientific Lab., NM (USA) (1981).
- [28] R. SANCHEZ and N. J. MCCORMICK, “Review of neutron transport approximations,” *Nuclear Science and Engineering*, **80**, 4 (1982).
- [29] P. LESAINT and P. 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).
- [30] W. H. REED, T. HILL, F. BRINKLEY, and K. LATHROP, “TRIPLET: A two-dimensional, multigroup, triangular mesh, planar geometry, explicit transport code,” Tech. rep., Los Alamos Scientific Lab., N. Mex.(USA) (1973).
- [31] T. SEED, W. MILLER JR, and F. BRINKLEY JR, “TRIDENT: a two-dimensional, multigroup, triangular mesh discrete ordinates, explicit neutron transport code,” Tech. rep., Los Alamos Scientific Lab., NM (USA) (1977).
- [32] T. SEED, W. MILLER, and G. BOSLER, “TRIDENT: A new triangular mesh discrete ordinates code,” in “ANS Meeting,” (1978), pp. 157–167.
- [33] 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).
- [34] Y. WANG and J. C. RAGUSA, “On the convergence of DGSEM applied to the discrete ordinates transport equation for structured and unstructured tri-

- angular meshes,” *Nuclear Science and Engineering*, **163**, 1, 56–72 (2009).
- [35] Y. WANG and J. C. RAGUSA, “A high-order discontinuous Galerkin method for the S_N transport equations on 2D unstructured triangular meshes,” *Annals of Nuclear Energy*, **36**, 7, 931–939 (2009).
- [36] G. G. DAVIDSON and T. S. PALMER, “Finite element transport using Wachspress rational basis functions on quadrilaterals in diffusive regions,” *Nuclear Science and Engineering*, **159**, 3, 242–255 (2008).
- [37] 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).
- [38] 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).
- [39] 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).
- [40] N. SUKUMAR and E. MALSCH, “Recent advances in the construction of polygonal finite element interpolants,” *Archives of Computational Methods in Engineering*, **13**, 1, 129–163 (2006).
- [41] A. RAND, A. GILLETTE, and C. BAJAJ, “Interpolation error estimates for mean value coordinates over convex polygons,” *Advances in Computational Mathematics*, **39**, 2, 327–347 (2013).

- [42] M. ADAMS and E. LARSEN, “Fast iterative methods for discrete-ordinates particle transport calculations,” *Progress in Nuclear Energy*, **40**, 1, 3–159 (2002).
- [43] R. E. ALCOUFFE, “Stable diffusion synthetic acceleration method for neutron transport iterations,” Los Alamos Scientific Lab., NM (1976), vol. 23.
- [44] R. E. ALCOUFFE, “The Diffusion Synthetic Acceleration Method Applied to Two-Dimensional Neutron Transport Problems,” Los Alamos Scientific Lab., NM (1977), vol. 27.
- [45] E. W. LARSEN, “Unconditionally stable diffusion-synthetic acceleration methods for the slab geometry discrete ordinates equations. Part I: Theory,” *Nuclear Science and Engineering*, **82**, 47 (1982).
- [46] D. R. MCCOY and E. W. LARSEN, “Unconditionally stable diffusion-synthetic acceleration methods for the slab geometry discrete ordinates equations. Part II: Numerical results,” *Nuclear Science and Engineering*, **82**, 64 (1982).
- [47] 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).
- [48] M. L. ADAMS and W. R. MARTIN, “Diffusion synthetic acceleration of discontinuous finite element transport iterations,” *Nuclear Science and Engineering*, **111**, 145–167 (1992).
- [49] T. WAREING, E. LARSEN, and M. ADAMS, “Diffusion accelerated discontinuous finite element schemes for the S_N equations in slab and x, y geometries,”

in “Proc. ANS Topical Meeting, Advances in Mathematics, Computations, and Reactor Physics,” (1991), p. 245.

- [50] 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).
- [51] B. TURCKSIN and J. C. RAGUSA, “Discontinuous diffusion synthetic acceleration for S_N transport on 2D arbitrary polygonal meshes,” *Journal of Computational Physics*, **274**, 356–369 (2014).
- [52] 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).
- [53] J. R. SHEWCHUK, “Delaunay refinement algorithms for triangular mesh generation,” *Computational Geometry*, **22**, 1, 21–74 (2002).
- [54] H. SI, “TetGen, a Delaunay-based quality tetrahedral mesh generator,” *ACM Transactions on Mathematical Software (TOMS)*, **41**, 2, 11 (2015).
- [55] C. GEUZAIN 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).
- [56] D. SIEGER, P. ALLIEZ, and M. BOTSCHE, “Optimizing Voronoi diagrams for polygonal finite element computations,” in “Proceedings of the 19th International Meshing Roundtable,” Springer, pp. 335–350 (2010).
- [57] M. S. EBEIDA and S. A. MITCHELL, “Uniform random Voronoi meshes,” in “Proceedings of the 20th International Meshing Roundtable,” Springer, pp.

273–290 (2011).

- [58] S. P. LLOYD, “Least squares quantization in PCM,” *Information Theory, IEEE Transactions on*, **28**, 2, 129–137 (1982).
- [59] Y. LINDE, A. BUZO, and R. M. GRAY, “An algorithm for vector quantizer design,” *Communications, IEEE Transactions on*, **28**, 1, 84–95 (1980).
- [60] M. S. EBEIDA, A. A. DAVIDSON, A. PATNEY, P. M. KNUPP, S. A. MITCHELL, and J. D. OWENS, “Efficient maximal poisson-disk sampling,” in “ACM Transactions on Graphics (TOG),” ACM (2011), vol. 30, p. 49.
- [61] M. S. EBEIDA, S. A. MITCHELL, A. PATNEY, A. A. DAVIDSON, and J. D. OWENS, “A Simple Algorithm for Maximal Poisson-Disk Sampling in High Dimensions,” in “Computer Graphics Forum,” Wiley Online Library (2012), vol. 31, pp. 785–794.
- [62] Q. DU, V. FABER, and M. GUNZBURGER, “Centroidal Voronoi tessellations: applications and algorithms,” *SIAM Review*, **41**, 4, 637–676 (1999).
- [63] S. VALETTE and J.-M. CHASSERY, “Approximated centroidal Voronoi diagrams for uniform polygonal mesh coarsening,” in “Computer Graphics Forum,” Wiley Online Library (2004), vol. 23, pp. 381–389.
- [64] C. TALISCHI, G. H. PAULINO, A. PEREIRA, and I. F. MENEZES, “PolyMesher: a general-purpose mesh generator for polygonal elements written in Matlab,” *Structural and Multidisciplinary Optimization*, **45**, 3, 309–328 (2012).
- [65] G. F. CAREY, *Computational Grids: Generations, Adaptation & Solution Strategies*, CRC Press (1997).
- [66] T. PLEWA, T. LINDE, and V. G. WEIRS, *Adaptive Mesh Refinement - Theory and Applications*, Springer Science & Business Media (2005).

- [67] 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).
- [68] G. KARNIADAKIS and S. SHERWIN, *Spectral/hp element methods for computational fluid dynamics*, Oxford University Press (2013).
- [69] C. SCHWAB, *p-and hp-finite element methods: Theory and applications in solid and fluid mechanics*, Oxford University Press (1998).
- [70] P. SOLIN, K. SEGETH, and I. DOLEZEL, *Higher-order finite element methods*, CRC Press (2003).
- [71] C. FÜHRER and G. KANSCHAT, “A posteriori error control in radiative transfer,” *Computing*, **58**, 4, 317–334 (1997).
- [72] R. HARTMANN, *Adaptive finite element methods for the compressible euler equations*, Ph.D. thesis, University of Heidelberg (2002).
- [73] 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).
- [74] 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).
- [75] Y. WANG and J. C. RAGUSA, “A two-mesh adaptive mesh refinement technique for S_N neutral-particle transport using a higher-order DGFEM,” *Journal of Computational and Applied Mathematics*, **233**, 12, 3178–3188 (2010).

- [76] 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).
- [77] R. LERNER and G. TRIGG, *Encyclopaedia of Physics*, 2 ed. (1991).
- [78] C. PARKER, *McGraw Hill Encyclopaedia of Physics*, 2 ed. (1994).
- [79] J. J. DUDESTADT and W. R. MARTIN, *Transport theory*, John Wiley & Sons (1979).
- [80] K. OTT and W. BEZELLA, *Introductory Nuclear Reactor Statics*, American Nuclear Society (1989).
- [81] B. CARLSON, “On a more precise definition of discrete ordinates methods,” in “Proceedings of the Second Conference on Transport Theory,” U.S. Atomic Energy Commission (1971), pp. 348–390.
- [82] I. ABU-SHUMAYS, “Compatible product angular quadrature for neutron transport in xy geometry,” *Nuclear Science and Engineering*, **64**, 2, 299–316 (1977).
- [83] M. ABRAMOWITZ, I. A. STEGUN, ET AL., “Handbook of mathematical functions,” *Applied Mathematics Series*, **55**, 62 (1966).
- [84] M. ABRAMOWITZ and I. A. STEGUN, *Handbook of mathematical functions: with formulas, graphs, and mathematical tables*, 55, Courier Corporation (1964).
- [85] 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).

- [86] 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).
- [87] E. MAGENES and J.-L. LIONS, *Problèmes aux limites non homogènes et applications* (1968).
- [88] 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).
- [89] R. H. MACNEAL and R. L. HARDER, “Eight nodes or nine?” *International Journal for Numerical Methods in Engineering*, **33**, 5, 1049–1058 (1992).
- [90] D. N. ARNOLD and G. AWANOU, “The serendipity family of finite elements,” *Foundations of Computational Mathematics*, **11**, 3, 337–344 (2011).
- [91] O. ZEINKIEWICZ, R. TAYLOR, and J. ZHU, *The finite element method: its basis and fundamentals*, SElsevier Butterworth-Heinemann (2005).
- [92] J. AKIN, *Application and implementation of finite element methods*, Academic Press, Inc. (1982).
- [93] S. OLIVEIRA and Y. DENG, “Preconditioned Krylov subspace methods for transport equations,” *Progress in Nuclear Energy*, **33**, 1, 155–174 (1998).
- [94] 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).
- [95] 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).

- [96] 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).
- [97] Y. SAAD, *Iterative methods for sparse linear systems*, Siam (2003).
- [98] 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).
- [99] 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.
- [100] K. R. KOCH, R. S. BAKER, and R. E. ALCOUFFE, “Solution of the First-Order Form of Three-Dimensional Discrete Ordinates Equations on a Massively Parallel Machine,” in “Transactions of the American Nuclear Society,” (1992), vol. 65, pp. 198–199.
- [101] R. S. BAKER and K. R. KOCH, “An S_N algorithm for the massively parallel CM-200 computer,” *Nuclear Science and Engineering*, **128**, 3 (1998).
- [102] W. D. HAWKINS, T. SMITH, M. P. ADAMS, L. RAUCHWERGER, N. M. AMATO, and M. L. ADAMS, “Efficient Massively Parallel Transport Sweeps,” in “Transactions of the American Nuclear Society,” (2012), vol. 107, pp. 477–481.
- [103] 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).

- [104] R. VERFÜRTH, *A review of a posteriori error estimation and adaptive mesh-refinement techniques*, John Wiley & Sons Inc (1996).
- [105] M. AINSWORTH and J. T. ODEN, *A posteriori error estimation in finite element analysis*, vol. 37, John Wiley & Sons (2011).
- [106] L. DEMKOWICZ, *Computing with hp-Adaptive Finite Elements: Volume 1 One and Two Dimensional Elliptic and Maxwell problems*, CRC Press (2006).
- [107] A. RAND, A. GILLETTE, and C. BAJAJ, "Quadratic serendipity finite elements on polygons using generalized barycentric coordinates," *Mathematics of Computation*, **83**, 290, 2691–2716 (2014).
- [108] E. L. WACHSPRESS, "A Rational Finite Element Basis," *Mathematics in Science and Engineering* (1975).
- [109] J. WARREN, S. SCHAEFER, A. N. HIRANI, and M. DESBRUN, "Barycentric coordinates for convex sets," *Advances in Computational Mathematics*, **27**, 3, 319–338 (2007).
- [110] M. S. FLOATER, "Mean value coordinates," *Computer Aided Geometric Design*, **20**, 1, 19–27 (2003).
- [111] M. S. FLOATER, "Generalized barycentric coordinates and applications," *Acta Numerica*, **24**, 161–214 (2015).
- [112] N. SUKUMAR, "Construction of polygonal interpolants: a maximum entropy approach," *International Journal for Numerical Methods in Engineering*, **61**, 12, 2159–2181 (2004).

- [113] 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).
- [114] K. HORMANN and N. SUKUMAR, “Maximum entropy coordinates for arbitrary polytopes,” in “Computer Graphics Forum,” Wiley Online Library (2008), vol. 27, pp. 1513–1520.
- [115] C. E. SHANNON, “A mathematical theory of communication,” *Bell Systems Technical Journal*, **27**, 379–423 (1948).
- [116] E. T. JAYNES, “Information theory and statistical mechanics,” *Physical review*, **106**, 4, 620–630 (1957).
- [117] S. KULLBACK and R. A. LEIBLER, “On information and sufficiency,” *The Annals of Mathematical Statistics*, pp. 79–86 (1951).
- [118] E. T. JAYNES, “Information theory and statistical mechanics. II,” *Physical review*, **108**, 2, 171 (1957).
- [119] S. BOYD and L. VANDENBERGHE, *Convex optimization*, Cambridge University Press (2004).
- [120] R. L. BURDEN and J. D. FAIRES, *Numerical analysis*, Brooks/Cole, 8 ed. (2004).
- [121] M. S. FLOATER, G. KÓS, and M. REIMERS, “Mean value coordinates in 3D,” *Computer Aided Geometric Design*, **22**, 7, 623–631 (2005).
- [122] M. WICKE, M. BOTSCH, and M. GROSS, “A finite element method on convex polyhedra,” in “Computer Graphics Forum,” Wiley Online Library (2007), vol. 26, pp. 355–364.

- [123] D. GONZÁLEZ, E. CUETO, and M. DOBLARÉ, “A higher order method based on local maximum entropy approximation,” *International Journal for Numerical Methods in Engineering*, **83**, 6, 741–764 (2010).
- [124] N. SUKUMAR, “Quadratic maximum-entropy serendipity shape functions for arbitrary planar polygons,” *Computer Methods in Applied Mechanics and Engineering*, **263**, 27–41 (2013).
- [125] R. PENROSE, “A generalized inverse for matrices,” in “Mathematical proceedings of the Cambridge philosophical society,” Cambridge University Press (1955), vol. 51, pp. 406–413.
- [126] P. SILVESTER, “Symmetric quadrature formulae for simplexes,” *Mathematics of Computation*, **24**, 109, 95–100 (1970).
- [127] D. DUNAVANT, “High degree efficient symmetrical Gaussian quadrature rules for the triangle,” *International Journal for Numerical Methods in Engineering*, **21**, 6, 1129–1148 (1985).
- [128] S. WANDZURAT and H. XIAO, “Symmetric quadrature rules on a triangle,” *Computers & Mathematics with Applications*, **45**, 12, 1829–1840 (2003).
- [129] 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.
- [130] R. COOLS and A. HAEGEMANS, “Construction of minimal cubature formulae for the square and the triangle, using invariant theory,” Tech. rep., Department of Computer Science, KU Leuven (1987).
- [131] M. NOOIJEN, G. TE VELDE, and E. BAERENDS, “Symmetric numerical integration formulas for regular polygons,” *SIAM Journal on Numerical Analysis*

ysis, **27**, 1, 198–218 (1990).

- [132] G. DASGUPTA, “Integration within polygonal finite elements,” *Journal of Aerospace Engineering*, **16**, 1, 9–18 (2003).
- [133] 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).
- [134] 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).
- [135] A. SHESTAKOV, D. KERSHAW, and G. ZIMMERMAN, “Test problems in radiative transfer calculations,” *Nuclear Science and Engineering*, **105**, 1, 88–104 (1990).
- [136] D. S. KERSHAW, “Differencing of the diffusion equation in Lagrangian hydrodynamic codes,” *Journal of Computational Physics*, **39**, 2, 375–395 (1981).
- [137] M. L. ADAMS, “Discontinuous finite element transport solutions in thick diffusive problems,” *Nuclear Science and Engineering*, **137**, 3, 298–333 (2001).
- [138] H. KOPP, “Synthetic method solution of the transport equation,” *Nuclear Science and Engineering*, **17**, 65 (1963).
- [139] V. LEBEDEV, “The Iterative *KP* Method for the Kinetic Equation,” in “Proc. Conf. on Mathematical Methods for Solution of Nuclear Physics Problems,” (1964).
- [140] 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*, pp. 154–176 (1966).

- [141] V. LEBEDEV, “On Finding Solutions of Kinetic Problems,” *USSR Comp. Math. and Math. Phys.*, **6**, 895 (1966).
- [142] V. LEBEDEV, “An Iterative *KP* Method,” *USSR Comp. Math. and Math. Phys.*, **7**, 1250 (1967).
- [143] V. LEBEDEV, “Problem of the Convergence of a Method of Estimating Iteration Deviations,” *USSR Comp. Math. and Math. Phys.*, **8**, 1377 (1968).
- [144] V. LEBEDEV, “Convergence of the *KP* Method for Some Neutron Transfer Problems,” *USSR Comp. Math. and Math. Phys.*, **9**, 226 (1969).
- [145] V. LEBEDEV, “Construction of the *P* Operation in the *KP* Method,” *USSR Comp. Math. and Math. Phys.*, **9**, 762 (1969).
- [146] E. GELBARD and L. HAGEMAN, “The synthetic method as applied to the S_N equations,” *Nuclear Science and Engineering*, **37**, 2, 288 (1969).
- [147] W. H. REED, “The effectiveness of acceleration techniques for iterative methods in transport theory,” *Nuclear Science and Engineering*, **45**, 3, 245 (1971).
- [148] R. E. ALCOUFFE, “Diffusion synthetic acceleration methods for the diamond-differenced discrete-ordinates equations,” *Nuclear Science and Engineering*, **64**, 2, 344–355 (1977).
- [149] 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.
- [150] Y. WANG, *Adaptive mesh refinement solution techniques for the multigroup S_N transport equation using a higher-order discontinuous finite element method*, Ph.D. thesis, Texas A&M University (2009).

- [151] B. ADAMS and J. MOREL, “A two-grid acceleration scheme for the multi-group S_N equations with neutron upscattering,” *Nuclear Science and Engineering*, **115**, 3, 253–264 (1993).
- [152] 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).
- [153] 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).
- [154] 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).
- [155] M. HACKEMACK and J. RAGUSA, “A DFEM Formulation of the Diffusion Equation on Arbitrary Polyhedral Grids,” in “Transactions of the American Nuclear Society,” (2014).
- [156] J. LIONS, “Problemes aux Limites non Homogenes a Donnees Irregulieres,” *Numerical Analysis of Partial Differential Equations*, pp. 283–292 (2011).
- [157] 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.
- [158] D. N. ARNOLD, “An interior penalty finite element method with discontinuous elements,” *SIAM Journal on Numerical Analysis*, **19**, 4, 742–760 (1982).

- [159] 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).
- [160] J. RUGE and K. STÜBEN, “Algebraic multigrid,” *Multigrid Methods*, **3**, 73–130 (1987).
- [161] W. L. BRIGGS, S. F. MCCORMICK, ET AL., *A multigrid tutorial*, Siam (2000).
- [162] Y. NOTAY, “Users Guide to AGMG,” *Electronic Transactions on Numerical Analysis*, **37**, 123–146 (2010).
- [163] Y. NOTAY, “An aggregation-based algebraic multigrid method,” *Electronic Transactions on Numerical Analysis*, **37**, 6, 123–146 (2010).
- [164] A. NAPOV and Y. NOTAY, “An algebraic multigrid method with guaranteed convergence rate,” *SIAM Journal on Scientific Computing*, **34**, 2, A1079–A1109 (2012).
- [165] Y. NOTAY, “Aggregation-based algebraic multigrid for convection-diffusion equations,” *SIAM Journal on Scientific Computing*, **34**, 4, A2288–A2316 (2012).
- [166] J. A. NELDER and R. MEAD, “A simplex method for function minimization,” *The Computer Journal*, **7**, 4, 308–313 (1965).
- [167] J. CHANG and M. ADAMS, “Analysis of transport synthetic acceleration for highly heterogeneous problems,” in “Proc. of M&C Topical Meeting,” (2003).
- [168] 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).

- [169] H. KHALIL, “A nodal diffusion technique for synthetic acceleration of nodal S_n calculations,” *Nuclear Science and Engineering*, **90**, 3, 263–280 (1985).
- [170] R. D. FALGOUT and U. M. YANG, “hypre: A Library of High Performance Preconditioners,” in P. SLOOT, A. HOEKSTRA, C. TAN, and J. DON-GARRA, editors, “Computational Science ICCS 2002,” Springer Berlin Heidelberg, *Lecture Notes in Computer Science*, vol. 2331, pp. 632–641 (2002).
- [171] V. HENSON and U. YANG, “BoomerAMG: a parallel algebraic multigrid solver and preconditioner,” *Applied Numerical Mathematics*, **41**, 5, 155–177 (2002).
- [172] T. MUKHERJEE and J. P. WEBB, “Hierarchical Bases for Polygonal Finite Elements,” *IEEE Transactions on Magnetics*, **51**, 3, 1–4 (2015).
- [173] T. A. DRISCOLL and L. N. TREFETHEN, *Schwarz-Christoffel Mapping*, vol. 8, Cambridge University Press (2002).
- [174] T. A. DRISCOLL, “Algorithm 843: improvements to the Schwarz-Christoffel toolbox for MATLAB,” *ACM Transactions on Mathematical Software*, **31**, 2, 239–251 (2005).
- [175] S. NATARAJAN, S. BORDAS, and D. ROY MAHAPATRA, “Numerical integration over arbitrary polygonal domains based on Schwarz–Christoffel conformal mapping,” *International Journal for Numerical Methods in Engineering*, **80**, 1, 103–134 (2009).

APPENDIX A

ADDENDUM TO CHAPTER 2

A.1 Detailed Description of the Spherical Harmonics Expansion of the Scattering Kernel

In Section 2.4, we provided details on the angular discretization of the transport equation. Specifically, we mentioned that the scattering term is modified by a series expansion of the scattering cross sections in terms of Legendre polynomials, P , and a series expansion of the angular flux in terms of the spherical harmonics functions, Y . We now go into further detail on the steps to properly perform these expansions for the scattering kernel.

We first define the scattering kernel, and ignore energy and spatial position, as the following:

$$\int_{4\pi} d\Omega' \sigma_s(\vec{\Omega}' \cdot \vec{\Omega}) \Psi(\vec{\Omega}'). \quad (\text{A.1})$$

We can then define the spherical harmonics as

$$Y_{n,k}(\vec{\Omega}) = \sqrt{C_{n,k}} P_n^k(\mu) e^{ik\theta}, \quad (\text{A.2})$$

where

$$C_{n,k} = \frac{(2n+1)(n-k)!}{4\pi(n+k)!}, \quad (\text{A.3})$$

and μ is the cosine of the polar angle, θ is the azimuthal angle, and P_n^k are the associated Legendre polynomials. These associated Legendre polynomials have the

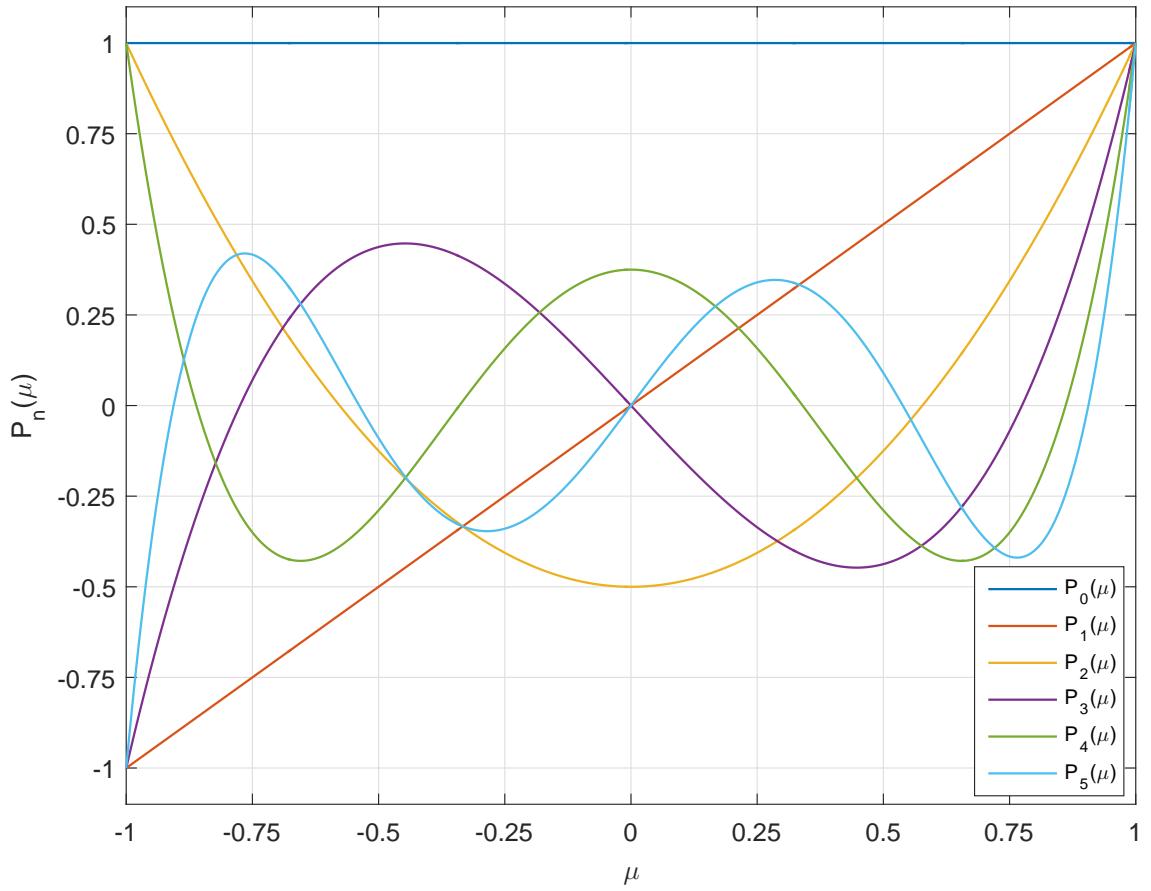


Figure A.1: Legendre polynomials of degrees 0 through 5.

following properties

$$\begin{aligned} P_n^0(\mu) &= P_n(\mu) \\ P_n^{-k}(\mu) &= (-1)^k \frac{(n-k)!}{(n+k)!} P_n^k(\mu) \end{aligned} \tag{A.4}$$

where $P_n(\mu)$ are the Legendre polynomials. The first 5 orders of the Legendre polynomials are given in Figure A.1.

An alternative definition for the spherical harmonics functions can be used that is more amenable to coding since there are no complex numbers involved. We can separate the spherical harmonics functions into their even ($Y_{n,k}^e(\vec{\Omega})$) and odd ($Y_{n,k}^o(\vec{\Omega})$)

components. These definitions now have the form of

$$\begin{aligned} Y_{n,k}^e(\vec{\Omega}) &= \sqrt{C_{n,k}} P_n^k(\mu) \cos(k\theta), & k = 0, \dots, n \\ Y_{n,k}^o(\vec{\Omega}) &= \sqrt{C_{n,k}} P_n^k(\mu) \sin(k\theta), & k = 1, \dots, n \end{aligned} \quad (\text{A.5})$$

where

$$C_{n,k} = \frac{(n-k)!}{(n+k)!} (2 - \delta_{k,0}). \quad (\text{A.6})$$

The orthogonality condition applies to these functions:

$$\begin{aligned} \int_{4\pi} Y_{n,k}^e(\vec{\Omega}) Y_{m,l}^e(\vec{\Omega}) &= \frac{4\pi}{2n+1} \delta_{n,m} \delta_{k,l} \\ \int_{4\pi} Y_{n,k}^o(\vec{\Omega}) Y_{m,l}^o(\vec{\Omega}) &= \frac{4\pi}{2n+1} \delta_{n,m} \delta_{k,l} \\ \int_{4\pi} Y_{n,k}^e(\vec{\Omega}) Y_{m,l}^o(\vec{\Omega}) &= 0 \end{aligned} \quad (\text{A.7})$$

The addition theorem is

$$2\pi P_n(\vec{\Omega}' \cdot \vec{\Omega}) = P_n(\mu_0) = \sum_{k=0}^n Y_{n,k}^e(\vec{\Omega}) Y_{n,k}^e(\vec{\Omega}') + \sum_{k=1}^n Y_{n,k}^o(\vec{\Omega}) Y_{n,k}^o(\vec{\Omega}'), \quad (\text{A.8})$$

where

$$\mu_0 \equiv \vec{\Omega}' \cdot \vec{\Omega}. \quad (\text{A.9})$$

We can then define the following angular flux moments as

$$\begin{aligned}\Phi_{n,k,e} &= \int_{4\pi} \Psi(\vec{\Omega}) Y_{n,k}^e(\vec{\Omega}) \\ \Phi_{n,k,o} &= \int_{4\pi} \Psi(\vec{\Omega}) Y_{n,k}^o(\vec{\Omega})\end{aligned}. \quad (\text{A.10})$$

With these flux moments, the angular flux can then be approximately expanded with the spherical harmonics functions:

$$\Psi(\vec{\Omega}) \approx \sum_{n=0}^{N_f} \frac{2n+1}{4\pi} \left[\sum_{k=0}^n \Phi_{n,k,e} Y_{n,k}^e(\vec{\Omega}) + \sum_{k=1}^n \Phi_{n,k,o} Y_{n,k}^o(\vec{\Omega}) \right]. \quad (\text{A.11})$$

where we have truncated the expansion to N_f .

Now, we define the expansion of the scattering cross section by use of the Legendre polynomials. If we truncate at the N_s term, the scattering cross section can be expanded as the following:

$$\sigma_s(\mu_0) \approx \sum_{n=0}^{N_s} \frac{2n+1}{2} \sigma_{s,n} P_n(\mu_0), \quad (\text{A.12})$$

where

$$\sigma_{s,n} \equiv \int_{-1}^1 d\mu_0 \sigma_s(\mu_0) P_n(\mu_0). \quad (\text{A.13})$$

If we define the term N_p as the minimum integer value between N_s and N_f , then we can then insert Eqs. (A.11) and (A.13) into Eq. (A.1) to yield the following

$$\begin{aligned}
& \int_{4\pi} d\Omega' \sigma_s(\vec{\Omega}' \cdot \vec{\Omega}) \Psi(\vec{\Omega}') \\
&= \int_{4\pi} d\Omega' \left\{ \begin{aligned} & \left[\sum_{n=0}^{N_s} \frac{2n+1}{2} \sigma_{s,n} P_n(\vec{\Omega}' \cdot \vec{\Omega}) \right] \\ & \left[\sum_{m=0}^{N_f} \frac{2m+1}{4\pi} \left(\sum_{l=0}^m \Phi_{m,l,e} Y_{m,l}^e(\vec{\Omega}') + \sum_{l=1}^m \Phi_{m,l,o} Y_{m,l}^o(\vec{\Omega}') \right) \right] \end{aligned} \right\} \\
&= \int_{4\pi} d\Omega' \left\{ \begin{aligned} & \left[\sum_{n=0}^{N_s} \frac{2n+1}{2} \sigma_{s,n} \left[\sum_{k=0}^n Y_{n,k}^e(\vec{\Omega}) Y_{n,k}^e(\vec{\Omega}') + \sum_{k=1}^n Y_{n,k}^o(\vec{\Omega}) Y_{n,k}^o(\vec{\Omega}') \right] \right] \\ & \left[\sum_{m=0}^{N_f} \frac{2m+1}{4\pi} \left(\sum_{l=0}^m \Phi_{m,l,e} Y_{m,l}^e(\vec{\Omega}') + \sum_{l=1}^m \Phi_{m,l,o} Y_{m,l}^o(\vec{\Omega}') \right) \right] \end{aligned} \right\} \\
&= \sum_{n=0}^{N_p} \frac{2n+1}{4\pi} \sigma_{s,n} \left[\sum_{k=0}^n \Phi_{n,k,e} Y_{n,k}^e(\vec{\Omega}) + \sum_{k=1}^n \Phi_{n,k,o} Y_{n,k}^o(\vec{\Omega}) \right]
\end{aligned} \tag{A.14}$$

To ease notation, let us define the following:

$$\begin{aligned}
Y_{n,k}(\vec{\Omega}) &\equiv Y_{n,k}^e(\vec{\Omega}), \quad k = 0, \dots, n \\
Y_{n,-k}(\vec{\Omega}) &\equiv Y_{n,k}^o(\vec{\Omega}), \quad k = 1, \dots, n \\
\Phi_{n,k} &\equiv \Phi_{n,k,e}, \quad k = 0, \dots, n \\
\Phi_{n,-k} &\equiv \Phi_{n,k,o}, \quad k = 1, \dots, n
\end{aligned} \tag{A.15}$$

With this simplification, we can write the final form for the scattering kernel:

$$\int_{4\pi} d\Omega' \sigma_s(\vec{\Omega}' \cdot \vec{\Omega}) \Psi(\vec{\Omega}') = \sum_{n=0}^{N_p} \frac{2n+1}{4\pi} \sigma_{s,n} \sum_{k=-n}^n \Phi_{n,k} Y_{n,k}(\vec{\Omega}) \tag{A.16}$$

APPENDIX B

ADDENDUM TO CHAPTER 3

B.1 Limits of the Linear Polygonal Basis Functions

As it was stated in Chapter 3, the Wachspress, mean value, and maximum entropy coordinates are all undefined on the boundary of the polygonal element. However, these basis functions do have a valid limit on the boundary. This means that, while direct boundary evaluation of the coordinates is impossible (results in divide-by-zero issues), we can demonstrate that the limits of their values are exactly those required of general barycentric coordinates. We use the geometric properties for an arbitrary polygon presented in Figure B.1 to do this.

B.1.1 Limits of the Wachspress Coordinates

Recall from Section 3.1.1 that the Wachspress basis functions on a polygon K with N_K vertices are of the form

$$\lambda_i(\vec{x}) = \frac{w_i(\vec{x})}{\sum_{j=1}^{N_K} w_j(\vec{x})}, \quad (\text{B.1})$$

where the weight function for vertex i , w_i , has the following definition:

$$w_i(\vec{x}) = \frac{A(\vec{x}_{i-1}, \vec{x}_i, \vec{x}_{i+1})}{A(\vec{x}, \vec{x}_{i-1}, \vec{x}_i) A(\vec{x}, \vec{x}_i, \vec{x}_{i+1})}. \quad (\text{B.2})$$

First, we analyze the limiting case as the point of evaluation, \vec{x} , approaches a vertex. The Wachspress coordinates maintain the *Lagrange property*: $\lambda_i(\vec{x}_j) = \delta_{ij}$. We prove this now by first dividing the numerator and denominator of Eq. (B.1)

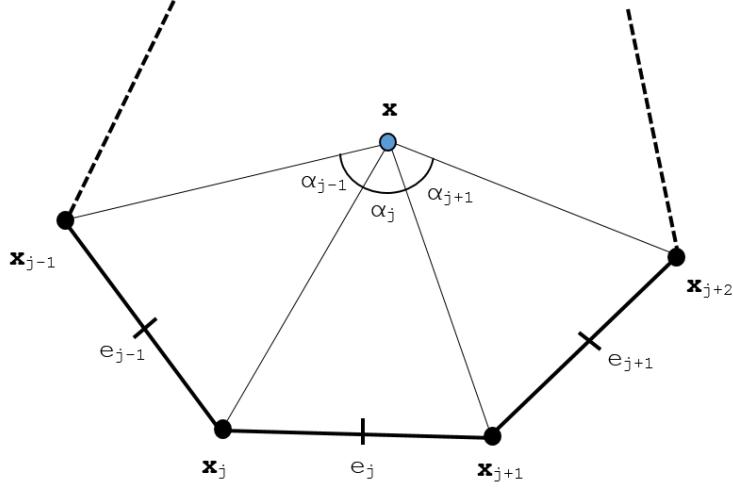


Figure B.1: Arbitrary polygon with geometric properties used for 2D basis function generation.

through by w_j to yield the following:

$$\lambda_i(\vec{x}) = \frac{w_i/w_j}{1 + \sum_{k \neq j} w_k/w_j}, \quad i \neq j \quad \text{and} \quad \lambda_j(\vec{x}) = \frac{1}{1 + \sum_{k \neq j} w_k/w_j}. \quad (\text{B.3})$$

Next, we define the term $R_{i,j}$ to be the following

$$R_{i,j}(\vec{x}) = \frac{w_i(\vec{x})}{w_j(\vec{x})}. \quad (\text{B.4})$$

Using the definition of Eq. (B.4), we can rewrite Eq. (B.3) as the following

$$\lambda_i(\vec{x}) = \frac{R_{i,j}}{1 + \sum_{k \neq j} R_{k,j}}, \quad i \neq j \quad \text{and} \quad \lambda_j(\vec{x}) = \frac{1}{1 + \sum_{k \neq j} R_{k,j}}. \quad (\text{B.5})$$

It is obvious from Eq. (B.5) that if all the $R_{k,j}$ ($k \neq j$) and $R_{i,j}$ ($i \neq j$) terms are zero, then we capture the *Lagrange property* for the Wachspress coordinates. Therefore,

we expand the terms of $R_{i,j}$ to give Eq. (B.6).

$$R_{i,j}(\vec{x}) = \frac{A(\vec{x}_{i-1}, \vec{x}_i, \vec{x}_{i+1})}{A(\vec{x}_{j-1}, \vec{x}_j, \vec{x}_{j+1})} \frac{A(\vec{x}_{j-1}, \vec{x}_j, \vec{x}) A(\vec{x}_j, \vec{x}_{j+1}, \vec{x})}{A(\vec{x}_{i-1}, \vec{x}_i, \vec{x}) A(\vec{x}_i, \vec{x}_{i+1}, \vec{x})} \quad (\text{B.6})$$

From Eq. (B.6), it is clear that the first term is bounded and non-zero and that the following limit is true:

$$\lim_{\vec{x} \rightarrow \vec{x}_j} \left(\frac{A(\vec{x}_{j-1}, \vec{x}_j, \vec{x}) A(\vec{x}_j, \vec{x}_{j+1}, \vec{x})}{A(\vec{x}_{i-1}, \vec{x}_i, \vec{x}) A(\vec{x}_i, \vec{x}_{i+1}, \vec{x})} \right) = 0 \quad (\text{B.7})$$

Therefore, the following is true,

$$\lim_{\vec{x} \rightarrow \vec{x}_j} R_{i,j}(\vec{x}) = 0, \quad (\text{B.8})$$

and the *Lagrange property* holds for the Wachspress coordinates:

$$\lambda_i(\vec{x}) = \delta_{ij}. \quad (\text{B.9})$$

Next, we seek to show that the Wachspress coordinates have piecewise linearity on the boundary of the polygon K . Therefore, we will analyze the limit of the basis functions as the point \vec{x} approaches face e_j . This limit is formally defined as the following

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \lambda_i(\vec{x}). \quad (\text{B.10})$$

From the definitions of the vertex weight functions of Eq. (B.2), we can immediately see that

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} |w_i(\vec{x})| = \infty, \quad i = (j, j+1), \quad (\text{B.11})$$

and

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} |w_i(\vec{x})| < \infty, \quad i \neq (j, j+1). \quad (\text{B.12})$$

Therefore, the following is also true for the basis functions

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \lambda_i(\vec{x}) = 0, \quad i \neq (j, j+1), \quad (\text{B.13})$$

and

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \lambda_j(\vec{x}) = \frac{w_j}{w_j + w_{j+1}}, \quad (\text{B.14})$$

and

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \lambda_{j+1}(\vec{x}) = \frac{w_{j+1}}{w_j + w_{j+1}}. \quad (\text{B.15})$$

We expand the term λ_j as the following,

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \lambda_j(\vec{x}) = \frac{A(\vec{x}_{j-1}, \vec{x}_j, \vec{x}_{j+1}) A(\vec{x}_{j+1}, \vec{x}_{j+2}, \vec{x})}{A(\vec{x}_{j-1}, \vec{x}_j, \vec{x}_{j+1}) A(\vec{x}_{j+1}, \vec{x}_{j+2}, \vec{x}) + A(\vec{x}_j, \vec{x}_{j+1}, \vec{x}_{j+2}) A(\vec{x}_{j-1}, \vec{x}_j, \vec{x})}, \quad (\text{B.16})$$

or

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \lambda_j(\vec{x}) = \frac{1}{1 + \beta}, \quad (\text{B.17})$$

where

$$\beta = \frac{A(\vec{x}_j, \vec{x}_{j+1}, \vec{x}_{j+2}) A(\vec{x}_{j-1}, \vec{x}_j, \vec{x})}{A(\vec{x}_{j-1}, \vec{x}_j, \vec{x}_{j+1}) A(\vec{x}_{j+1}, \vec{x}_{j+2}, \vec{x})}. \quad (\text{B.18})$$

Through extensive algebra along with the use of some trigonometric properties and further limits, the β term can be written as the following

$$\beta = \frac{|\vec{x}_j - \vec{x}|}{|\vec{x}_{j+1} - \vec{x}|}, \quad (\text{B.19})$$

which means that the limit of λ_j can be written as the following as well:

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \lambda_j(\vec{x}) = \frac{|\vec{x}_{j+1} - \vec{x}|}{|\vec{x}_{j+1} - \vec{x}_j|}. \quad (\text{B.20})$$

Through a similar procedure, the limit of λ_{j+1} can be written as the following as well:

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \lambda_{j+1}(\vec{x}) = \frac{|\vec{x}_j - \vec{x}|}{|\vec{x}_{j+1} - \vec{x}_j|}. \quad (\text{B.21})$$

respectively. Therefore, we can write the final result for the boundary analysis of the Wachspress coordinates in Eq. (B.22).

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \lambda_i(\vec{x}) = \begin{cases} \frac{|\vec{x}_{j+1} - \vec{x}|}{|\vec{x}_{j+1} - \vec{x}_j|}, & i = j \\ \frac{|\vec{x}_j - \vec{x}|}{|\vec{x}_{j+1} - \vec{x}_j|}, & i = j + 1 \\ 0, & \text{otherwise} \end{cases} \quad (\text{B.22})$$

B.1.2 Limits of the Mean Value Coordinates

Recall from Section 3.1.3 that the mean value basis functions on a polygon K with N_K vertices are of the form

$$\lambda_i(\vec{x}) = \frac{w_i(\vec{x})}{\sum_{j=1}^{N_K} w_j(\vec{x})} \quad (\text{B.23})$$

where the weight function for vertex i , w_i , has the following definition:

$$w_i(\vec{x}) = \frac{\tan(\alpha_{i-1}/2) + \tan(\alpha_i/2)}{|\vec{x}_i - \vec{x}|} \quad (\text{B.24})$$

First, we analyze the limiting case as the point of evaluation, \vec{x} , approaches the vertex j , \vec{x}_j . The mean value coordinates maintain the *Lagrange property*: $\lambda_i(\vec{x}_j) = \delta_{ij}$. We prove this now by first dividing the numerator and denominator of Eq. (B.23) through by w_j to yield the following:

$$\lambda_i(\vec{x}) = \frac{w_i/w_j}{1 + \sum_{k \neq j} w_k/w_j}, \quad i \neq j \quad \text{and} \quad \lambda_j(\vec{x}) = \frac{1}{1 + \sum_{k \neq j} w_k/w_j}. \quad (\text{B.25})$$

Next, we define the term $R_{i,j}$ to be the following

$$R_{i,j}(\vec{x}) = \frac{w_i(\vec{x})}{w_j(\vec{x})}. \quad (\text{B.26})$$

Using the definition of Eq. (B.26), we can rewrite Eq. (B.25) as the following

$$\lambda_i(\vec{x}) = \frac{R_{i,j}}{1 + \sum_{k \neq j} R_{k,j}}, \quad i \neq j \quad \text{and} \quad \lambda_j(\vec{x}) = \frac{1}{1 + \sum_{k \neq j} R_{k,j}}. \quad (\text{B.27})$$

It is obvious from Eq. (B.27) that if all the $R_{k,j}$ ($k \neq j$) and $R_{i,j}$ ($i \neq j$) terms are zero, then we capture the *Lagrange property* for the mean value coordinates. Therefore, we expand the terms of $R_{i,j}$ to give Eq. (B.28).

$$\begin{aligned}
R_{i,j}(\vec{x}) &= \frac{w_i}{w_j} = \frac{\tan(\alpha_{i-1}/2) + \tan(\alpha_i/2)}{|\vec{x}_i - \vec{x}|} \frac{|\vec{x}_j - \vec{x}|}{\tan(\alpha_{j-1}/2) + \tan(\alpha_j/2)} \\
&= \frac{|\vec{x}_j - \vec{x}|}{|\vec{x}_i - \vec{x}|} \frac{\tan(\alpha_{i-1}/2) + \tan(\alpha_i/2)}{\tan(\alpha_{j-1}/2) + \tan(\alpha_j/2)}
\end{aligned} \tag{B.28}$$

Next, we define the following trigonometric property:

$$\tan(x) + \tan(y) = \frac{\sin(x+y)}{\cos(x)\cos(y)}. \tag{B.29}$$

Inserting Eq. (B.29) into the final line of Eq. (B.28), we obtain:

$$R_{i,j}(\vec{x}) = \frac{|\vec{x}_j - \vec{x}|}{|\vec{x}_i - \vec{x}|} \frac{\sin(\alpha_{i-1}/2 + \alpha_i/2)}{\sin(\alpha_{j-1}/2 + \alpha_j/2)} \frac{\cos(\alpha_{j-1}/2) \cos(\alpha_j/2)}{\cos(\alpha_{i-1}/2) \cos(\alpha_i/2)}. \tag{B.30}$$

In the limit as \vec{x} approaches \vec{x}_j , we see that the term $||\vec{x}_j - \vec{x}||$ approaches zero. However, we need to assess what the other terms in Eq. (B.30) will approach. If we examine the limiting cases where $\vec{x}_i = \vec{x}_{j-1}$ or $\vec{x}_i = \vec{x}_j$, then the limits of the remaining terms of Eq. (B.30) can be written as the following.

$$\begin{aligned}
& \lim_{\vec{x} \rightarrow \vec{x}_j} |\vec{x}_i - \vec{x}| > 0 \\
& \lim_{\vec{x} \rightarrow \vec{x}_j} |\sin(\alpha_{i-1}/2 + \alpha_i/2)| > 0 \\
& \lim_{\vec{x} \rightarrow \vec{x}_j} |\sin(\alpha_{j-1}/2 + \alpha_j/2)| > 0 \\
& \lim_{\vec{x} \rightarrow \vec{x}_j} |\cos(\alpha_{i-1}/2)| > 0 \\
& \lim_{\vec{x} \rightarrow \vec{x}_j} |\cos(\alpha_i/2)| > 0 \\
& \lim_{\vec{x} \rightarrow \vec{x}_j} |\cos(\alpha_{j-1}/2)| > 0 \\
& \lim_{\vec{x} \rightarrow \vec{x}_j} |\cos(\alpha_j/2)| > 0
\end{aligned} \tag{B.31}$$

Therefore, the following is true,

$$\lim_{\vec{x} \rightarrow \vec{x}_j} R_{i,j}(\vec{x}) = 0, \tag{B.32}$$

and the *Lagrange property* holds for the mean value coordinates:

$$\lambda_i(\vec{x}) = \delta_{ij}. \tag{B.33}$$

Next, we seek to show that the mean value coordinates have piecewise linearity on the boundary of the polygon K . Therefore, we will analyze the limit of the basis functions as the point \vec{x} approaches face e_j . This limit is formally defined as the following

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \lambda_i(\vec{x}). \tag{B.34}$$

If we define the signed area function, $A_j(\vec{x})$, for face, e_j ,

$$A_j(\vec{x}) = \frac{1}{2} \begin{vmatrix} 1 & 1 & 1 \\ x & x_j & x_{j+1} \\ y & y_j & y_{j+1} \end{vmatrix} = \frac{|\vec{x}_j - \vec{x}| |\vec{x}_{j+1} - \vec{x}| \sin(\alpha_j)}{2} \quad (\text{B.35})$$

then a modified vertex weight function, \hat{w}_i , can be given as

$$\hat{w}_i(\vec{x}) = w_i(\vec{x}) A_j(\vec{x}). \quad (\text{B.36})$$

This modified weight function can be expanded to

$$\hat{w}_i(\vec{x}) = \frac{\tan(\alpha_{i-1}/2) + \tan(\alpha_i/2)}{|\vec{x}_i - \vec{x}|} \frac{|\vec{x}_j - \vec{x}| |\vec{x}_{j+1} - \vec{x}| \sin(\alpha_j)}{2}, \quad (\text{B.37})$$

and the mean value basis functions can be rewritten in terms of these modified functions:

$$\lambda_i(\vec{x}) = \frac{\hat{w}_i(\vec{x})}{\sum_{j=1}^{N_K} \hat{w}_j(\vec{x})}. \quad (\text{B.38})$$

Analyzing Eq. (B.37), we can immediately discern that $\hat{w}_i(\vec{x}) = 0$ for $(i = j, j + 1)$. This is because as \vec{x} goes to the boundary, α_j goes to π and $\sin(\alpha_j)$ goes to zero. We will show below that the limit of $\hat{w}_i(\vec{x})$ is greater than zero and bounded for $(i = j, j + 1)$. We can then write the mean value basis functions in the modified terms for vertex j and $j + 1$ as

$$\lambda_j(\vec{x}) = \frac{\hat{w}_j}{\hat{w}_j + \hat{w}_{j+1}}, \quad (\text{B.39})$$

and

$$\lambda_{j+1}(\vec{x}) = \frac{\hat{w}_{j+1}}{\hat{w}_j + \hat{w}_{j+1}}, \quad (\text{B.40})$$

respectively. Taking the limits of the modified weight functions for vertex j and $j+1$ gives

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \hat{w}_j(\vec{x}) = |\vec{x}_{j+1} - \vec{x}|, \quad (\text{B.41})$$

and

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \hat{w}_{j+1}(\vec{x}) = |\vec{x}_j - \vec{x}|, \quad (\text{B.42})$$

respectively. Therefore, we can write the final result for the boundary analysis of the mean value coordinates in Eq. (B.56).

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \lambda_i(\vec{x}) = \begin{cases} \frac{|\vec{x}_{j+1} - \vec{x}|}{|\vec{x}_{j+1} - \vec{x}_j|}, & i = j \\ \frac{|\vec{x}_j - \vec{x}|}{|\vec{x}_{j+1} - \vec{x}_j|}, & i = j+1 \\ 0, & \text{otherwise} \end{cases} \quad (\text{B.43})$$

B.1.3 Limits of the Maximum Entropy Coordinates

Recall from Section 3.1.4 that the maximum entropy basis functions on a polygon K with N_K vertices are of the form

$$\lambda_i(\vec{x}) = \frac{w_i(\vec{x})}{\sum_{j=1}^{N_K} w_j(\vec{x})} \quad (\text{B.44})$$

where the weight function for vertex i , w_i , has the following definition

$$w_i(\vec{x}) = m_i(\vec{x}) \exp(-\vec{\kappa} \cdot (\vec{x}_i - \vec{x})). \quad (\text{B.45})$$

In Eq. (B.45), the prior distribution, m_i , has the form

$$m_i(\vec{x}) = \frac{\pi_i(\vec{x})}{\sum_{k=1}^{N_K} \pi_k(\vec{x})}, \quad (\text{B.46})$$

where

$$\pi_i(\vec{x}) = \prod_{k \neq i-1, i}^{N_K} \rho_k(\vec{x}), \quad (\text{B.47})$$

and

$$\rho_j(\vec{x}) = ||\vec{x} - \vec{x}_j|| + ||\vec{x} - \vec{x}_{j+1}|| - ||\vec{x}_{j+1} - \vec{x}_j||, \quad (\text{B.48})$$

and ρ_j is the face weight function corresponding to face e_j . The weight function for face e_j is identically zero along face e_j and strictly positive elsewhere by the Triangle Inequality. This means that the product π_i and the prior distribution m_i for vertex i is zero on all faces that do not touch vertex i . We do not have an explicit proof that $\vec{\kappa}$ has a valid limit on the polygon boundary. However, through numerical testing we confirmed that $\vec{\kappa}$ does have a bounded limit on the polygon boundary. Therefore, the limit of $\exp(-\vec{\kappa} \cdot (\vec{x}_i - \vec{x}))$ in Eq. (B.45) is strictly bounded as well.

For the maximum entropy coordinates, we will only demonstrate the piecewise linearity property since the *Lagrange property* is satisfied by the limit at the face vertices. With the definition of the prior distributions and in the limit as a point, \vec{x} , approaches face e_j , the basis functions for the j and $j+1$ vertices are given by

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \lambda_j(\vec{x}) = \frac{w_j(\vec{x})}{w_j(\vec{x}) + w_{j+1}(\vec{x})}, \quad (\text{B.49})$$

and

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \lambda_{j+1}(\vec{x}) = \frac{w_{j+1}(\vec{x})}{w_j(\vec{x}) + w_{j+1}(\vec{x})}, \quad (\text{B.50})$$

respectively. Dropping the spatial parameter, we write the j basis functions as

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \lambda_j = \frac{1}{1 + \frac{w_{j+1}}{w_j}}, \quad (\text{B.51})$$

where we now simply need to define $\frac{w_{j+1}}{w_j}$. From the linear precision property of the general barycentric coordinates, we can write the following:

$$w_j(\vec{x}) (\vec{x}_j - \vec{x}) + w_{j+1}(\vec{x}) (\vec{x}_{j+1} - \vec{x}) = \vec{0}. \quad (\text{B.52})$$

If we take the norm of Eq. (B.52) and rearrange terms, then we obtain

$$\frac{w_{j+1}}{w_j} = \frac{|\vec{x}_j - \vec{x}|}{|\vec{x}_{j+1} - \vec{x}|}. \quad (\text{B.53})$$

Therefore, the limit of λ_j can be written as

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \lambda_j = \frac{|\vec{x}_{j+1} - \vec{x}|}{|\vec{x}_{j+1} - \vec{x}_j|}. \quad (\text{B.54})$$

In a similar manner, the limit of λ_{j+1} can be written as

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \lambda_{j+1} = \frac{|\vec{x}_j - \vec{x}|}{|\vec{x}_{j+1} - \vec{x}_j|}. \quad (\text{B.55})$$

Therefore, we can write the final results for the boundary analysis of the maximum

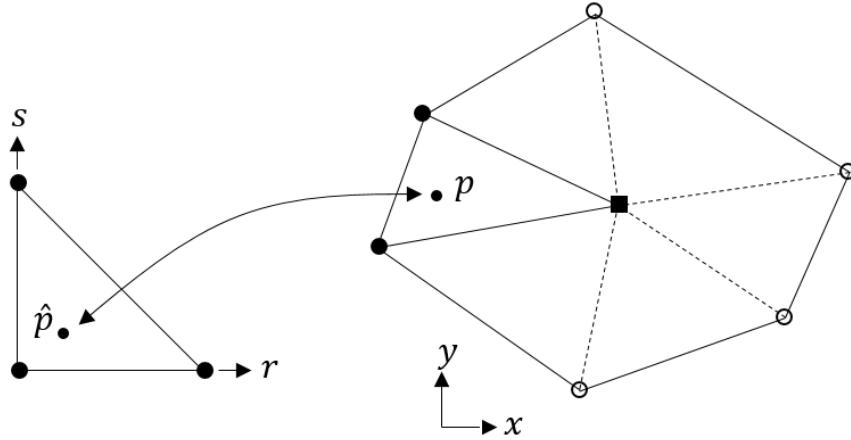


Figure B.2: Mapping a point on the reference triangle onto a sub-triangle of an arbitrary polygon.

entropy coordinates as

$$\lim_{\vec{x} \rightarrow \vec{x}^* \in e_j} \lambda_i(\vec{x}) = \begin{cases} \frac{|\vec{x}_{j+1} - \vec{x}|}{|\vec{x}_{j+1} - \vec{x}_j|}, & i = j \\ \frac{|\vec{x}_j - \vec{x}|}{|\vec{x}_{j+1} - \vec{x}_j|}, & i = j + 1 \\ 0, & \text{otherwise} \end{cases} \quad (\text{B.56})$$

B.2 Jacobian Transformations of the Reference Element

In Section 3.3, we provided the details for numerically integrating the elementary matrices on arbitrary polygons. This was done by triangulation of the polygon K with N_K into N_K distinct sub-triangles. Quadrature nodes and weights were selected on the reference triangle with vertices $[(0, 0), (1, 0), (0, 1)]$. Then, this reference quadrature set was affinely mapped onto the different sub-triangles of K . This mapping of points from the reference space into the global space was performed with a Jacobian transformation. The visual depiction for this mapping is given in Figure B.2. We now provide greater details of this Jacobian transformation.

In general terms for a 2D problem with a reference space defined as (r, s) , the Jacobian matrix can be written as the following

$$J = \begin{bmatrix} \frac{\partial x}{\partial r} & \frac{\partial x}{\partial s} \\ \frac{\partial y}{\partial r} & \frac{\partial y}{\partial s} \end{bmatrix}. \quad (\text{B.57})$$

The 2D reference coordinates can be written in terms of the global position coordinates as

$$\vec{x}(r, s) = \sum_{i=1} \vec{x}_i \hat{b}_i(r, s), \quad (\text{B.58})$$

where \hat{b}_i are the reference triangle basis functions. For this work, we define the reference triangle basis functions and their gradients as

$$\begin{aligned} \hat{b}_1(r, s) &= 1 - r - s \\ \hat{b}_2(r, s) &= r \\ \hat{b}_3(r, s) &= s \end{aligned}, \quad (\text{B.59})$$

and

$$\vec{\nabla} \hat{b}(r, s) = \begin{bmatrix} -1 & -1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad (\text{B.60})$$

respectively. For completeness, we give the 3D reference basis functions and their constant gradients defined on a reference tetrahedron (r, s, t) as

$$\begin{aligned}
\hat{b}_1(r, s, t) &= 1 - r - s - t \\
\hat{b}_2(r, s, t) &= r \\
&\quad , \\
\hat{b}_3(r, s, t) &= s \\
\hat{b}_4(r, s, t) &= t
\end{aligned} \tag{B.61}$$

and

$$\vec{\nabla} \hat{b}(r, s, t) = \begin{bmatrix} -1 & -1 & -1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{B.62}$$

respectively. Therefore, we apply the partial derivate terms of Eq. (B.57) on the coordinates defined in Eq. (B.58). Working through the derivatives yields a Jacobian matrix of the following form

$$J = \begin{bmatrix} (x_2 - x_1) & (x_3 - x_1) \\ (y_2 - y_1) & (y_3 - y_1) \end{bmatrix}. \tag{B.63}$$

Likewise, the corresponding 3D Jacobian using the reference basis functions of Eq. (B.61) is given by

$$J = \begin{bmatrix} (x_2 - x_1) & (x_3 - x_1) & (x_4 - x_1) \\ (y_2 - y_1) & (y_3 - y_1) & (y_4 - y_1) \\ (z_2 - z_1) & (z_3 - z_1) & (z_4 - z_1) \end{bmatrix}. \tag{B.64}$$

The Jacobian can also be used to transform the gradients of the basis functions between the global and reference spaces. The gradient of the reference space can be

computed in terms of the global space by,

$$\nabla_{\hat{x}} = J^T \nabla_x, \quad (\text{B.65})$$

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

$$\nabla_x = J^{-T} \nabla_{\hat{x}}. \quad (\text{B.66})$$

B.3 Analytical Integration of the PWL Basis Functions

In Chapter 3, we provided the functional form for the Piecewise Linear (PWL) coordinates in 2D and 3D. At that time, we simply left the notation for the basis functions and did not give any further information except that a quadrature scheme could be used to integrate the elementary matrices. However, since the PWL coordinates are collections of polynomials over sub-triangles and sub-tetrahedra in 2D and 3D, respectively, direct analytical integration can also be performed. In this section, we detail how the elementary matrices can be integrated with the PWL functions.

B.3.1 2D PWL Basis Functions

In Section 3.1.2, we provided the functional form for the 2D Piecewise Linear (PWL) coordinates. We noted that, of the linearly-complete 2D polygonal coordinates, PWL is the only one that can perform analytical integrations of the elementary matrices. We now describe the procedures to perform these analytical integrations of the elementary matrices. The integral of the $b_i b_j$ term for the mass matrix and streaming matrix are given by

$$\begin{aligned}
\int_K b_i b_j &= \int_K (t_i + \alpha_K t_c) (t_j + \alpha_K t_c) \\
&= \int_K t_i t_j + \alpha_K \int_K t_i t_c + \alpha_K \int_K t_j t_c + \alpha_K^2 \int_K t_c t_c,
\end{aligned} \tag{B.67}$$

and

$$\begin{aligned}
\int_K \vec{\nabla} b_i b_j &= \int_K (\vec{\nabla} t_i + \alpha_K \vec{\nabla} t_c) (t_j + \alpha_K t_c) \\
&= \int_K \vec{\nabla} t_i t_j + \alpha_K \int_K \vec{\nabla} t_i t_c + \alpha_K \int_K t_j \vec{\nabla} t_c + \alpha_K^2 \int_K \vec{\nabla} t_c t_c,
\end{aligned} \tag{B.68}$$

respectively. The t_i function only has local measure on two sub-triangles of the polygon corresponding to $(\vec{x}_{i-1}, \vec{x}_i, \vec{r}_c)$ and $(\vec{x}_i, \vec{x}_{i+1}, \vec{r}_c)$. The t_c function has measure everywhere in the polygon. Therefore, we can loop through the sub-triangles and add contributions of the integral to elementary matrices for the polygon. Specifically, we can make use of the reference basis functions and the Jacobian transformations of Section B.2. If we transform the t_i , t_j , and t_c basis functions into the \hat{b}_1 , \hat{b}_2 , and \hat{b}_3 reference functions on the sub-triangle k , then the mass

$$\int_k b_i b_j = \int_{\hat{K}} \hat{b}_1 \hat{b}_2 |J_k| + \alpha_K \int_{\hat{K}} \hat{b}_1 \hat{b}_3 |J_k| + \alpha_K \int_{\hat{K}} \hat{b}_2 \hat{b}_3 |J_k| + \alpha_K^2 \int_{\hat{K}} \hat{b}_3 \hat{b}_3 |J_k|. \tag{B.69}$$

This integral on the sub-triangle can then be written into a (3×3) matrix

$$\int_k b_i b_j = \begin{bmatrix} \int_{\hat{K}} \hat{b}_1 \hat{b}_1 |J_k| & \int_{\hat{K}} \hat{b}_1 \hat{b}_2 |J_k| & \alpha_K \int_{\hat{K}} \hat{b}_1 \hat{b}_3 |J_k| \\ \int_{\hat{K}} \hat{b}_2 \hat{b}_1 |J_k| & \int_{\hat{K}} \hat{b}_2 \hat{b}_2 |J_k| & \alpha_K \int_{\hat{K}} \hat{b}_2 \hat{b}_3 |J_k| \\ \alpha_K \int_{\hat{K}} \hat{b}_3 \hat{b}_1 |J_k| & \alpha_K \int_{\hat{K}} \hat{b}_3 \hat{b}_2 |J_k| & \alpha_K^2 \int_{\hat{K}} \hat{b}_3 \hat{b}_3 |J_k| \end{bmatrix}. \quad (\text{B.70})$$

This sub-matrix defined on sub-triangle k is then added into the global mass matrix for element K . Likewise, the integral of the streaming term on the sub-triangle k can be written as

$$\int_k \vec{\nabla} b_i b_j = \begin{bmatrix} \int_{\hat{K}} J^{-T} \vec{\nabla} \hat{b}_1 \hat{b}_1 |J_k| & \int_{\hat{K}} J^{-T} \vec{\nabla} \hat{b}_1 \hat{b}_2 |J_k| & \alpha_K \int_{\hat{K}} J^{-T} \vec{\nabla} \hat{b}_1 \hat{b}_3 |J_k| \\ \int_{\hat{K}} J^{-T} \vec{\nabla} \hat{b}_2 \hat{b}_1 |J_k| & \int_{\hat{K}} J^{-T} \vec{\nabla} \hat{b}_2 \hat{b}_2 |J_k| & \alpha_K \int_{\hat{K}} J^{-T} \vec{\nabla} \hat{b}_2 \hat{b}_3 |J_k| \\ \alpha_K \int_{\hat{K}} J^{-T} \vec{\nabla} \hat{b}_3 \hat{b}_1 |J_k| & \alpha_K \int_{\hat{K}} J^{-T} \vec{\nabla} \hat{b}_3 \hat{b}_2 |J_k| & \alpha_K^2 \int_{\hat{K}} J^{-T} \vec{\nabla} \hat{b}_3 \hat{b}_3 |J_k| \end{bmatrix}. \quad (\text{B.71})$$

Again, this sub-matrix can be added into the global streaming matrix for element K .

B.3.2 3D PWL Basis Functions

The integrations of the 3D PWL basis functions on an arbitrary 3D polyhedra can be performed in a similar manner to the integrations of the 2D PWL basis functions. The integral of the $b_i b_j$ term for the mass matrix and streaming matrix are given by

$$\begin{aligned}
\int_K b_i b_j &= \int_K \left(t_i + \sum_{f=1}^{F_i} \beta_f^i t_f + \alpha_K t_c \right) \left(t_j + \sum_{g=1}^{F_j} \beta_g^j t_g + \alpha_K t_c \right) \\
&= \int_K t_i t_j + \sum_{g=1}^{F_j} \beta_g^j \int_K t_i t_g + \alpha_K \int_K t_i t_c \\
&\quad + \sum_{f=1}^{F_i} \beta_f^i \int_K t_f t_j + \sum_{f=1}^{F_i} \sum_{g=1}^{F_j} \beta_f^i \int_K t_f t_g + \sum_{f=1}^{F_i} \beta_f^i \alpha_K \int_K t_f t_c \\
&\quad + \alpha_K \int_K t_c t_j + \sum_{g=1}^{F_j} \beta_g^j \alpha_K \int_K t_c t_g + \alpha_K^2 \int_K t_c t_c
\end{aligned}, \tag{B.72}$$

and

$$\begin{aligned}
\int_K \vec{\nabla} b_i b_j &= \int_K \left(\vec{\nabla} t_i + \sum_{f=1}^{F_i} \beta_f^i \vec{\nabla} t_f + \alpha_K \vec{\nabla} t_c \right) \left(t_j + \sum_{g=1}^{F_j} \beta_g^j t_g + \alpha_K t_c \right) \\
&= \int_K \vec{\nabla} t_i t_j + \sum_{g=1}^{F_j} \beta_g^j \int_K \vec{\nabla} t_i t_g + \alpha_K \int_K \vec{\nabla} t_i t_c \\
&\quad + \sum_{f=1}^{F_i} \beta_f^i \int_K \vec{\nabla} t_f t_j + \sum_{f=1}^{F_i} \sum_{g=1}^{F_j} \beta_f^i \int_K \vec{\nabla} t_f t_g + \sum_{f=1}^{F_i} \beta_f^i \alpha_K \int_K \vec{\nabla} t_f t_c \\
&\quad + \alpha_K \int_K \vec{\nabla} t_c t_j + \sum_{g=1}^{F_j} \beta_g^j \alpha_K \int_K \vec{\nabla} t_c t_g + \alpha_K^2 \int_K \vec{\nabla} t_c t_c
\end{aligned}, \tag{B.73}$$

respectively. In Eqs. (B.72) and (B.73), t_f and t_g are the face tent functions. We then define the sub-tetrahedron k corresponding to two vertices on a face (counter-clockwise orientation), the face center, and the polyhedron center. We transform the t_j , t_j , t_f , and t_c basis functions into the \hat{b}_1 , \hat{b}_2 , \hat{b}_3 , and \hat{b}_4 reference functions on the sub-tetrahedron k . We can write the (4×4) sub-matrix contribution for the sub-tetrahedron k into the global mass matrix of element K as

$$\left[\begin{array}{cccc} \int\limits_{\hat{K}} \hat{b}_1 \hat{b}_1 |J_k| & \int\limits_{\hat{K}} \hat{b}_1 \hat{b}_2 |J_k| & \beta_f \int\limits_{\hat{K}} \hat{b}_1 \hat{b}_3 |J_k| & \alpha_K \int\limits_{\hat{K}} \hat{b}_1 \hat{b}_4 |J_k| \\ \int\limits_{\hat{K}} \hat{b}_2 \hat{b}_1 |J_k| & \int\limits_{\hat{K}} \hat{b}_2 \hat{b}_2 |J_k| & \beta_f \int\limits_{\hat{K}} \hat{b}_2 \hat{b}_3 |J_k| & \alpha_K \int\limits_{\hat{K}} \hat{b}_2 \hat{b}_4 |J_k| \\ \beta_f \int\limits_{\hat{K}} \hat{b}_3 \hat{b}_1 |J_k| & \beta_f \int\limits_{\hat{K}} \hat{b}_3 \hat{b}_2 |J_k| & \beta_f^2 \int\limits_{\hat{K}} \hat{b}_3 \hat{b}_3 |J_k| & \beta_f \alpha_K \int\limits_{\hat{K}} \hat{b}_3 \hat{b}_4 |J_k| \\ \alpha_K \int\limits_{\hat{K}} \hat{b}_4 \hat{b}_1 |J_k| & \alpha_K \int\limits_{\hat{K}} \hat{b}_4 \hat{b}_2 |J_k| & \beta_f \alpha_K \int\limits_{\hat{K}} \hat{b}_4 \hat{b}_3 |J_k| + & \alpha_K^2 \int\limits_{\hat{K}} \hat{b}_4 \hat{b}_4 |J_k| \end{array} \right]. \quad (\text{B.74})$$

The streaming matrix contribution can be computed in an identical manner as Eq. (B.71).

APPENDIX C

ADDENDUM TO CHAPTER 4

In this appendix, we will perform some additional analysis of the MIP diffusion form. First, we provide greater implementation details for Fourier Analysis in Section C.1. Then, we provide some example Fourier Analysis for the 1D MIP DSA scheme in Section C.2. Next, we provide a comparison between the MIP and Modified Four-Step (M4S) DSA schemes in Section C.3. We conclude with a demonstration on the conservation of the SIP diffusion form in Section C.4.

C.1 Extended Fourier Analysis Implementation for MIP in 1D

In this section, we give greater detail on how to implement discretized Fourier Analysis through the use of a 1D example. The 1D, isotropic scattering, continuous transport equation is given by

$$\mu \frac{d}{dx} \psi(x, \mu) + \sigma_t(x) \psi(x, \mu) = \frac{\sigma_s(x)}{2} \phi(x) + \frac{Q(x)}{2}. \quad (\text{C.1})$$

If we define a 1D angular quadrature, $\{\mu_m, w_m\}_{m=1}^M$, suppress the spatial and angular parameters and apply SI, we can rewrite Eq. (C.1) into

$$\mu_m \frac{d}{dx} \psi_m^{(k+1/2)} + \sigma_t \psi_m^{(k+1/2)} = \frac{\sigma_s}{2} \phi^{(k)} + \frac{Q}{2}, \quad (\text{C.2})$$

where

$$\phi^{(k)} = \sum_{m=1}^M w_m \psi_m^{(k)}, \quad (\text{C.3})$$

and we assume constant cross sections and a constant isotropic source. Applying a

spatial discretization, we can express Eq. (C.2) in operator notation as

$$\mathbf{L}\Psi^{(k+1/2)} = \frac{1}{2}\mathbf{S}\Phi^{(k)} + \frac{1}{2}\mathbf{Q}. \quad (\text{C.4})$$

If we perform a transport sweep and integrate over the angular quadrature, then we can write Eq. (C.4) in terms of only the scalar fluxes as

$$\begin{aligned} \Phi^{(k+1/2)} &= \frac{1}{2}\mathbf{D}\mathbf{L}^{-1}\mathbf{S}\Phi^{(k)} + \frac{1}{2}\mathbf{D}\mathbf{L}^{-1}\mathbf{Q}. \\ \Phi^{(k+1/2)} &= \mathbf{T}\Phi^{(k)} + \mathbf{b} \end{aligned} \quad (\text{C.5})$$

In Eq. (C.5), $\mathbf{T} = \frac{1}{2}\mathbf{D}\mathbf{L}^{-1}\mathbf{S}$ and $\mathbf{b} = \frac{1}{2}\mathbf{D}\mathbf{L}^{-1}\mathbf{Q}$.

We now give the low-order diffusion equation for the transport error as

$$-\frac{d}{dx}D\frac{d}{dx}\delta\phi^{(k+1/2)} + \sigma_a\delta\phi^{(k+1/2)} = \sigma_s\left(\phi^{(k+1/2)} - \phi^{(k)}\right), \quad (\text{C.6})$$

where the corresponding discretized operator notation is

$$\mathbf{A}\delta\Phi^{(k+1/2)} = \mathbf{S}\left(\Phi^{(k+1/2)} - \Phi^{(k)}\right). \quad (\text{C.7})$$

Solving for the correction gives

$$\delta\Phi^{(k+1/2)} = \mathbf{A}^{-1}\mathbf{S}\left(\Phi^{(k+1/2)} - \Phi^{(k)}\right). \quad (\text{C.8})$$

We now express the SI+DSA iteration as the following:

$$\begin{aligned}
\Phi^{(k+1)} &= \Phi^{(k+1/2)} + \delta\Phi^{(k+1/2)} \\
&= \mathbf{T}\Phi^{(k)} + \mathbf{b} + \mathbf{A}^{-1}\mathbf{S}\left(\Phi^{(k+1/2)} - \Phi^{(k)}\right) \\
&= \mathbf{T}\Phi^{(k)} + \mathbf{b} + \mathbf{A}^{-1}\mathbf{S}\left(\mathbf{T}\Phi^{(k)} + \mathbf{b} - \Phi^{(k)}\right) \\
&= [\mathbf{T} + \mathbf{A}^{-1}\mathbf{S}(\mathbf{T} - \mathbf{I})]\Phi^{(k)} + (\mathbf{I} + \mathbf{A}^{-1}\mathbf{S})\mathbf{b}
\end{aligned} \tag{C.9}$$

Therefore, our FA iteration matrix to analyze is $[\mathbf{T} + \mathbf{A}^{-1}\mathbf{S}(\mathbf{T} - \mathbf{I})]$. We expand this iteration matrix back out and denote the terms that will require the Fourier transformation with the tilde, $\tilde{\cdot}$, to give

$$\frac{1}{2}\mathbf{D}\tilde{\mathbf{L}}^{-1}\tilde{\mathbf{S}} + \tilde{\mathbf{A}}^{-1}\tilde{\mathbf{S}}\left(\frac{1}{2}\mathbf{D}\tilde{\mathbf{L}}^{-1}\tilde{\mathbf{S}} - \mathbf{I}\right). \tag{C.10}$$

Therefore, all that remains is to define each of these terms in terms of the appropriate elementary matrices and phase transformation matrices.

For this analysis, we will only use a single mesh cell. It will have a cell width of h which gives the physical domain of $[0, h]$. Using the 1D LD basis functions, the cell-wise elementary matrices for the mass, stiffness, and gradient matrices for a cell of width h are

$$\begin{aligned}
\mathbb{M} &= \frac{h}{6} \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}, \\
\mathbb{K} &= \frac{1}{h} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \\
\mathbb{S} &= \frac{1}{2} \begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix},
\end{aligned} \tag{C.11}$$

respectively. The values of the basis functions at the left face and right face are given

by

$$\begin{aligned}\mathbb{E}_L &= \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \\ \mathbb{E}_R &= \begin{bmatrix} 0 \\ 1 \end{bmatrix},\end{aligned}\tag{C.12}$$

respectively. Likewise, the gradients of the basis functions at the left face and the right face are given by

$$\begin{aligned}\mathbb{G}_L &= \frac{1}{h} \begin{bmatrix} -1 \\ 1 \end{bmatrix}, \\ \mathbb{G}_R &= \frac{1}{h} \begin{bmatrix} -1 \\ 1 \end{bmatrix},\end{aligned}\tag{C.13}$$

respectively.

Now, we provide the simple definitions of the Fourier phase transformation matrices. Recall that when the matrix coupling is occurring across a boundary, the periodic boundaries enforce that the degrees of freedom from the other side of the domain be used. However, the Fourier phase that is used is that of the virtual cell just on the other side of the domain (we can view these virtual cells as ghost cells). Therefore, the ghost cell to the left of the domain spans $[-h, 0]$, and the ghost cell to the right of the domain spans $[h, 2h]$. We can write the phase matrices for inside the cell, for the left ghost cell, and for the right ghost cell as

$$\mathbb{P}_{in} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\lambda h} \end{bmatrix},\tag{C.14}$$

$$\mathbb{P}_L = \begin{bmatrix} e^{-i\lambda h} & 0 \\ 0 & 1 \end{bmatrix}, \quad (C.15)$$

and

$$\mathbb{P}_R = \begin{bmatrix} e^{i\lambda h} & 0 \\ 0 & e^{i\lambda 2h} \end{bmatrix}, \quad (C.16)$$

respectively.

We now go term-by-term and give the appropriate forms for each of the terms of Eq. (C.10). Recall that all boundary faces in Fourier Analysis act as interior faces. The scattering term, $\tilde{\mathbf{S}}$, is solely within the domain and is given by

$$\tilde{\mathbf{S}} = \sigma_s \mathbb{M} \mathbb{P}_{in} \quad (C.17)$$

The discretized transport loss operator, $\tilde{\mathbf{L}}$, is a block diagonal matrix where each block corresponds to an angular direction. If we isolate an angular direction m , the m direction loss operator, \tilde{L}_m , is given by

$$\tilde{L}_m = (\sigma_t \mathbb{M} - \mu_m \mathbb{S}) \mathbb{P}_{in} + \mu_m \left(\mathbb{E}_R \mathbb{E}_L^T \mathbb{P}_R - \mathbb{E}_L \mathbb{E}_R^T \mathbb{P}_{in} \right) \quad (C.18)$$

for $\mu_m < 0$ and

$$\tilde{L}_m = (\sigma_t \mathbb{M} - \mu_m \mathbb{S}) \mathbb{P}_{in} + \mu_m \left(\mathbb{E}_R \mathbb{E}_R^T \mathbb{P}_{in} - \mathbb{E}_L \mathbb{E}_R^T \mathbb{P}_L \right) \quad (C.19)$$

for $\mu_m > 0$. Therefore, the $\tilde{\mathbf{T}}$ operator can be given by

$$\tilde{\mathbf{T}} = \frac{1}{2} \sum_{m=1}^M w_m \tilde{L}_m^{-1} \tilde{\mathbf{S}}. \quad (C.20)$$

This just leaves the discretization of $\tilde{\mathbf{A}}$. We choose to assign the face normal for the MIP matrix construction as strictly positive in the x-direction. This leads to $\tilde{\mathbf{A}}$ having the following definition,

$$\begin{aligned}\tilde{\mathbf{A}} = & D\mathbb{K}\mathbb{P}_{in} + \sigma_a \mathbb{M}\mathbb{P}_{in} \\ & + \kappa^{MIP} \left(\mathbb{E}_L \mathbb{E}_L^T \mathbb{P}_{in} + \mathbb{E}_R \mathbb{E}_R^T \mathbb{P}_{in} - \mathbb{E}_L \mathbb{E}_R^T \mathbb{P}_L - \mathbb{E}_R \mathbb{E}_L^T \mathbb{P}_R \right) \\ & + \frac{D}{2} \left(\mathbb{E}_L \mathbb{G}_L^T \mathbb{P}_{in} - \mathbb{E}_R \mathbb{G}_R^T \mathbb{P}_{in} + \mathbb{E}_L \mathbb{G}_R^T \mathbb{P}_L - \mathbb{E}_R \mathbb{G}_L^T \mathbb{P}_R \right) \\ & + \frac{D}{2} \left(\mathbb{G}_L \mathbb{E}_L^T \mathbb{P}_{in} - \mathbb{G}_R \mathbb{E}_R^T \mathbb{P}_{in} + \mathbb{G}_L \mathbb{E}_R^T \mathbb{P}_L - \mathbb{G}_R \mathbb{E}_L^T \mathbb{P}_R \right)\end{aligned}\quad (\text{C.21})$$

where κ^{MIP} is the MIP penalty coefficient. For this FA problem configuration, this penalty coefficient has the following form

$$\kappa^{MIP} = \max\left(\frac{1}{4}, \kappa^{SIP}\right) \quad (\text{C.22})$$

where

$$\kappa^{SIP} = c \frac{D}{h}, \quad (\text{C.23})$$

and c is the penalty constant that is variable.

C.2 1D MIP Fourier Analysis Results

For completeness with the 2D and 3D results presented in Chapter 4, we now provide a set of Fourier Analysis results for the 1D MIP DSA scheme. For all the problems run, we analyze a single mesh cell with $h = 1$ and vary the value of σ_t to change the cell mean free paths. The scattering ratio is set to 0.9999. We also vary the angular quadrature and the constant in the MIP penalty coefficient, c . We present the FA results in Figures C.1, C.2, C.3, C.4, and C.5 for MIP penalty coefficient constant values of 1, 2, 4, 8, and 64, respectively. Like the 3D hexahedral

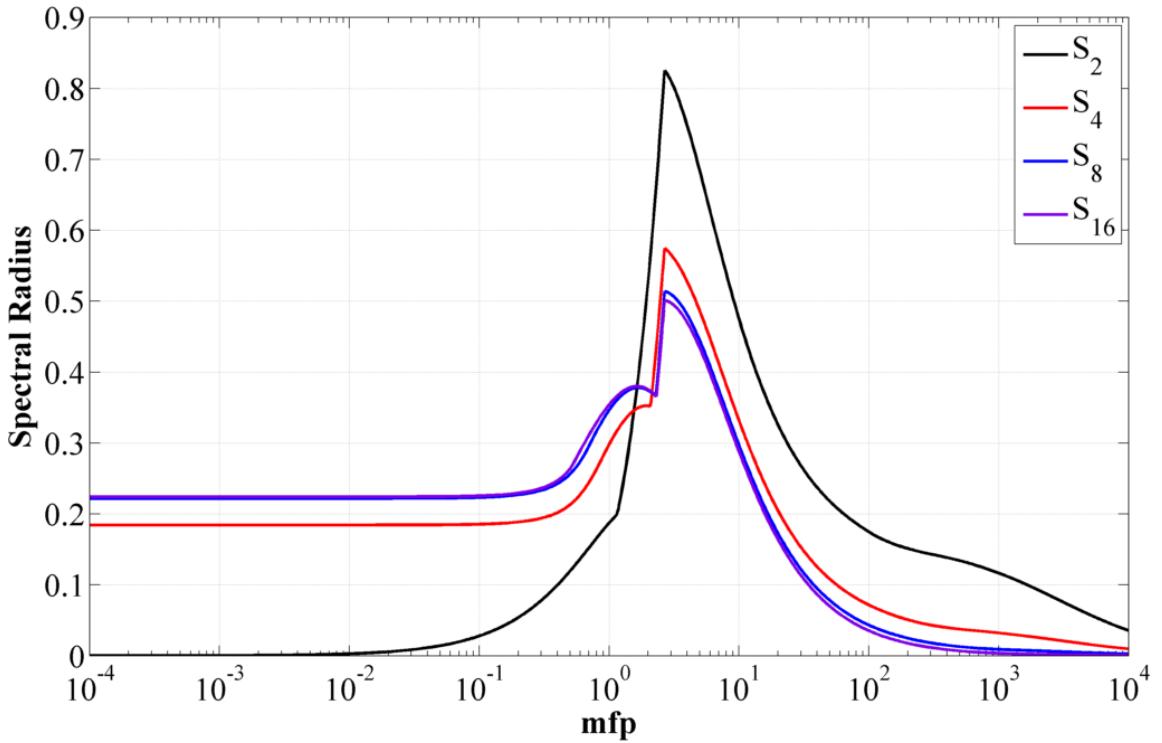


Figure C.1: Spectral radius for the 1D MIP form using $c = 1$.

FA results, a value of $c = 1$ leads to under-penalization and a degradation of the MIP spectral radius. A value of $c = 4$ gives the best spectral radius results just like 3D as well. However, as c becomes large, too much penalization is achieved, and the spectral radius begins to approach unity in the intermediate range of mean free paths. This makes the MIP scheme ineffective in this case.

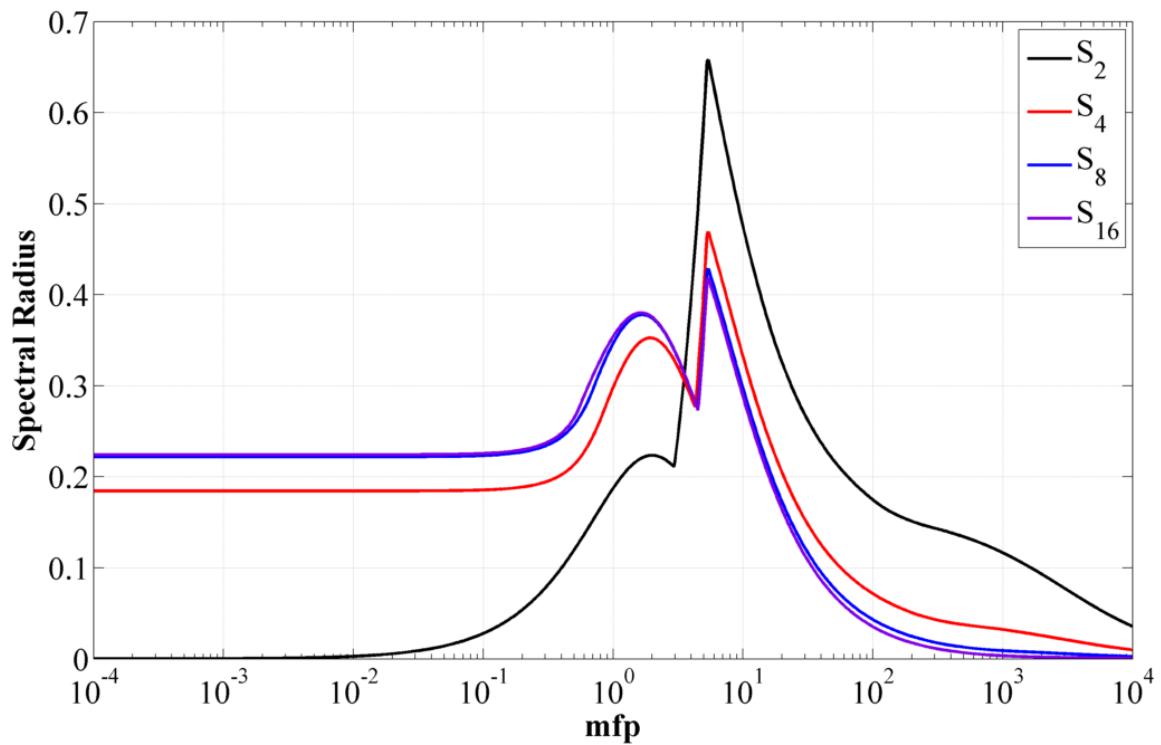


Figure C.2: Spectral radius for the 1D MIP form using $c = 2$.

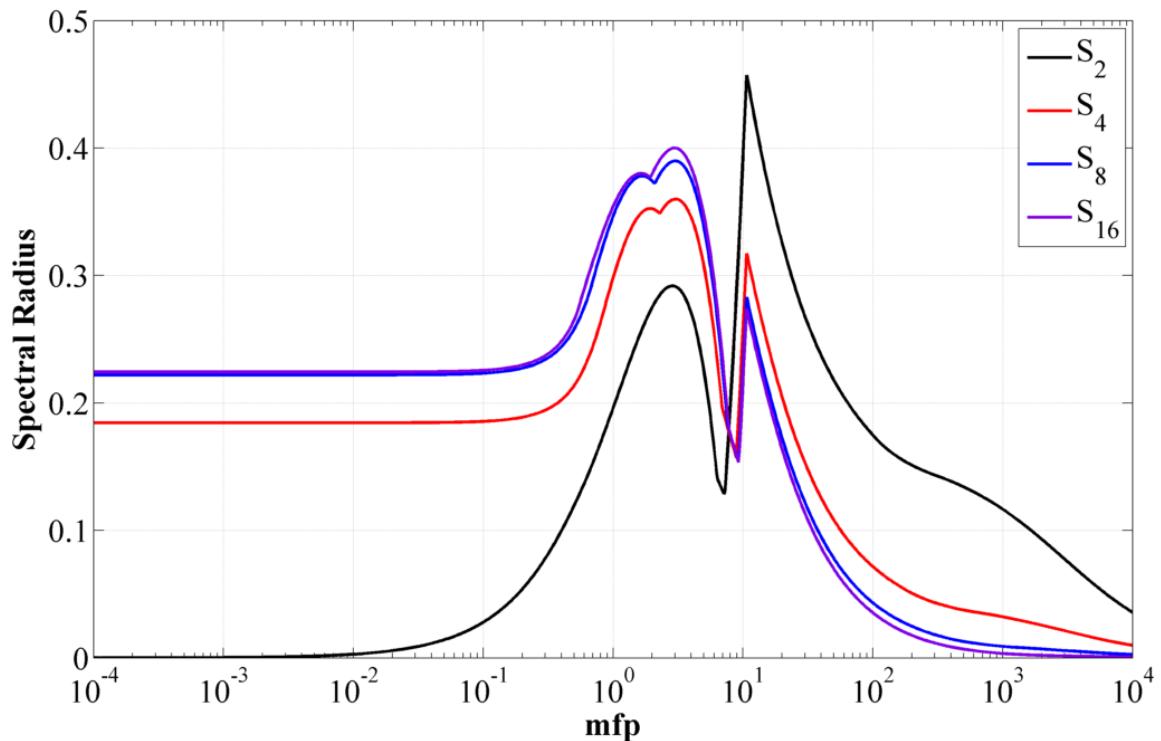


Figure C.3: Spectral radius for the 1D MIP form using $c = 4$.

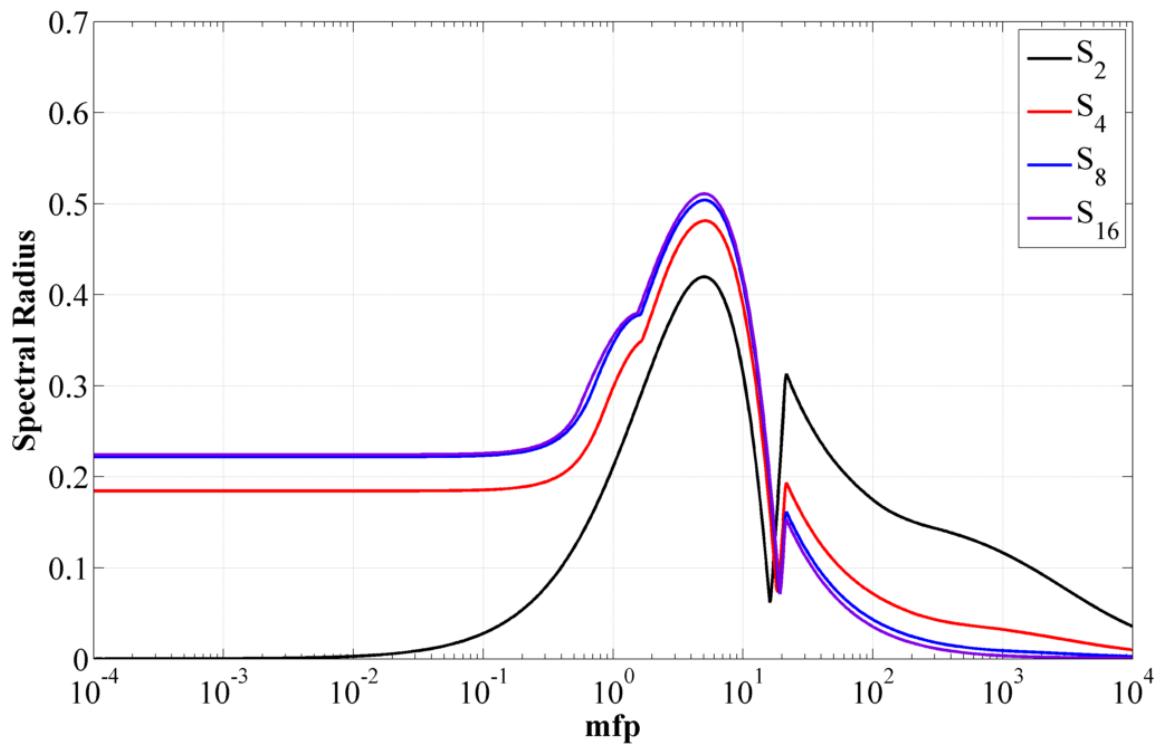


Figure C.4: Spectral radius for the 1D MIP form using $c = 8$.

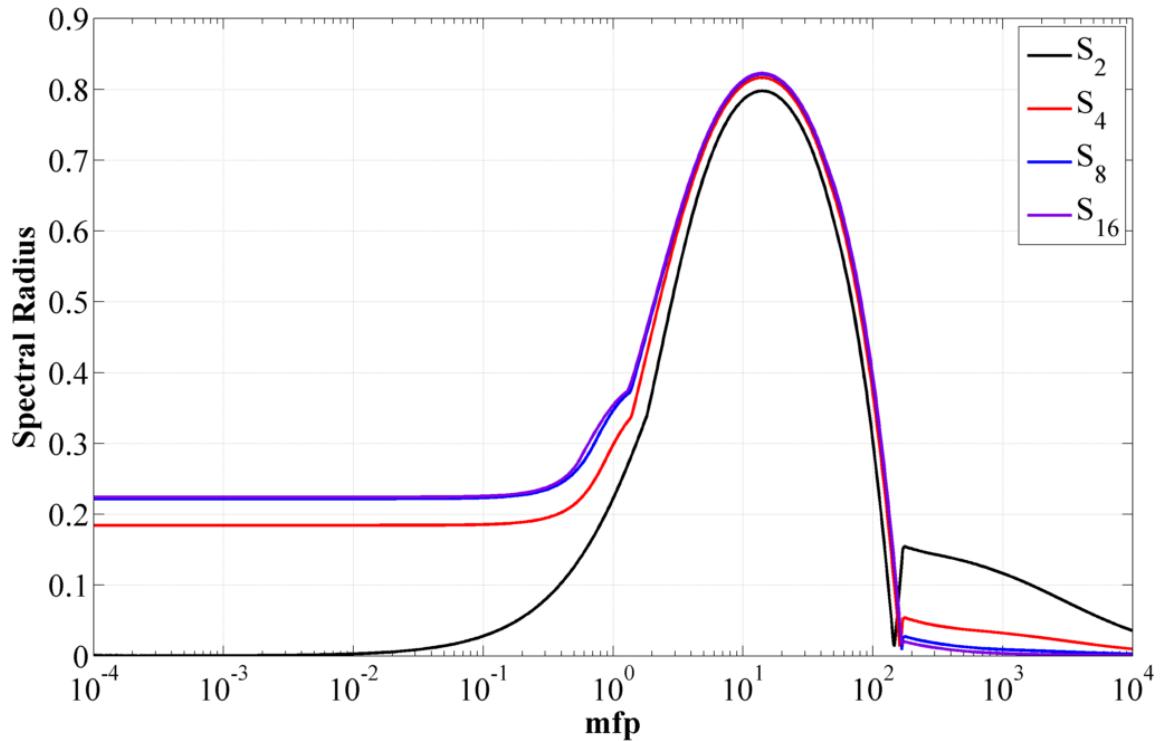


Figure C.5: Spectral radius for the 1D MIP form using $c = 64$.

C.3 Comparison between Modified Interior Penalty and Modified Four-Step DSA Schemes

In this work, our low-order diffusion operator was the discontinuous DFEM MIP form. Another common, fully-discontinuous form of the diffusion equation for use as a DSA scheme is the Modified Four-Step (M4S) method [48]. It is a partially-consistent DSA scheme that can be derived in two separate ways to yield the same functional form. One method is to take the analytic transport equation, expand the angular flux moments, take the moments to form the analytic diffusion equation, and then spatially discretize to form the spatially discretized diffusion equation. Conversely, the derivation can be performed by spatially discretizing the analytic transport equation, expanding the angular flux moments, and taking the moments to form the spatially discretized diffusion equation. Using either of these methods, we can write the equations for a set of N_K basis functions, b_i^K , in element K with N_f^K faces as

$$\sum_f^{N_f^K} \int_f b_i^K \left[\left(\alpha \delta \Phi - \frac{1}{2} D \partial_n \delta \Phi \right)_{r_K^-} - \left(\alpha \delta \Phi + \frac{1}{2} D \partial_n \delta \Phi \right)_{r_K^+} \right] + \int_K D \vec{\nabla} b_i^K \cdot \vec{\nabla} \delta \Phi + \int_K \sigma_a b_i^K \delta \Phi = \int_K b_i^K Q, \quad 1 \leq i \leq N_K \quad (\text{C.24})$$

where

$$\left(\alpha \delta \Phi + \frac{1}{2} D \partial_n \delta \Phi \right)_{r_K^+} \equiv (1 - \beta) \left(\alpha \delta \Phi - \frac{1}{2} D \partial_n \delta \Phi \right)_{r_K^-}, \quad (\text{C.25})$$

and r_K^- and r_K^+ correspond to values taken from within and outside cell K , respectively. In Eq. (C.24), α is approximately 1/4 and Q corresponds to the residual of the DSA scheme. In Eq. (C.25), β has a value of 1 if the face corresponds to an

incoming incident boundary face. Likewise, β has a value of 0 if the face corresponds to an interior face or reflecting boundary face. However, in the case of a reflecting boundary, the r_K^+ terms become the r_K^- terms and no contribution is needed. With the notation that we used for the SIP and MIP forms in Chapter 4, we can write the matrix of the M4S scheme as

$$\begin{aligned} a^{M4S}(b, \delta\Phi) = & \left(D\vec{\nabla}b, \vec{\nabla}\delta\Phi \right)_D + \left(\sigma b, \delta\Phi \right)_D \\ & + \frac{1}{4} \left\langle [\![b]\!], [\![\delta\Phi]\!] \right\rangle_{E_h^i} + \left\langle [\![b]\!], \{ \{ D\partial_n \delta\Phi \} \} \right\rangle_{E_h^i}. \\ & + \frac{1}{4} \left\langle b, \delta\Phi \right\rangle_{\partial\mathcal{D}^d} - \frac{1}{2} \left\langle b, D\partial_n \delta\Phi \right\rangle_{\partial\mathcal{D}^d} \end{aligned} \quad (\text{C.26})$$

We now compare the M4S form with the MIP form. We again write the bilinear matrix of MIP diffusion form as

$$\begin{aligned} a^{MIP}(b, \delta\Phi) = & \left(D\vec{\nabla}b, \vec{\nabla}\delta\Phi \right)_D + \left(\sigma b, \delta\Phi \right)_D \\ & + \left\langle \kappa_f^{MIP} [\![b]\!], [\![\delta\Phi]\!] \right\rangle_{E_h^i} + \left\langle [\![b]\!], \{ \{ D\partial_n \delta\Phi \} \} \right\rangle_{E_h^i} + \left\langle \{ \{ D\partial_n b \} \}, [\![\delta\Phi]\!] \right\rangle_{E_h^i}. \\ & + \left\langle \kappa_f^{MIP} b, \delta\Phi \right\rangle_{\partial\mathcal{D}^d} - \frac{1}{2} \left\langle b, D\partial_n \delta\Phi \right\rangle_{\partial\mathcal{D}^d} - \frac{1}{2} \left\langle D\partial_n b, \delta\Phi \right\rangle_{\partial\mathcal{D}^d} \end{aligned} \quad (\text{C.27})$$

Between Eqs. (C.26) and (C.27), we can immediately see similarities and differences between the M4S and MIP diffusion forms. M4S does not contain the $\left\langle \{ \{ D\partial_n b \} \}, [\![\delta\Phi]\!] \right\rangle_{E_h^i}$ or $\frac{1}{2} \left\langle D\partial_n b, \delta\Phi \right\rangle_{\partial\mathcal{D}^d}$ symmetrization terms. Therefore, the M4S scheme is not symmetric, and PCG cannot be used as the iterative method. Instead, a method like GMRES would be needed to solve the M4S system of equations. We can also see from Eq. (C.26) that the “penalty” term in M4S is a constant value of 1/4.

We now provide a pair of results for the M4S form. First, we give a comparison

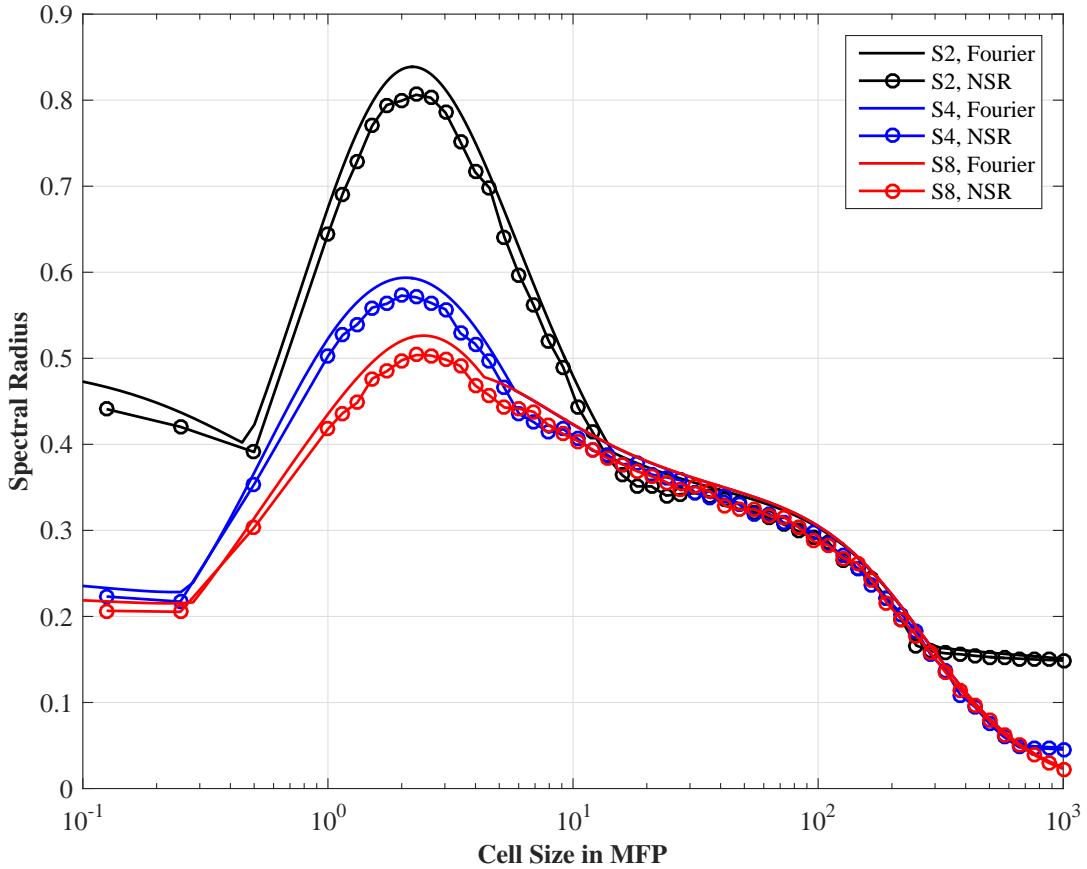


Figure C.6: Comparison between the numerical spectral radii and the theoretical Fourier Analysis spectral radii for the M4S scheme on the unit square.

between the theoretical Fourier Analysis and numerical spectral radii results on unit square mesh cells using the linear PWL basis functions in Figure C.6. Then, we provide the Fourier wave number distributions of the PWL basis functions on the unit square in Figures C.7, C.8, C.9, and C.10 for the S_2 , S_4 , S_8 , and S_{16} level-symmetric quadrature, respectively.

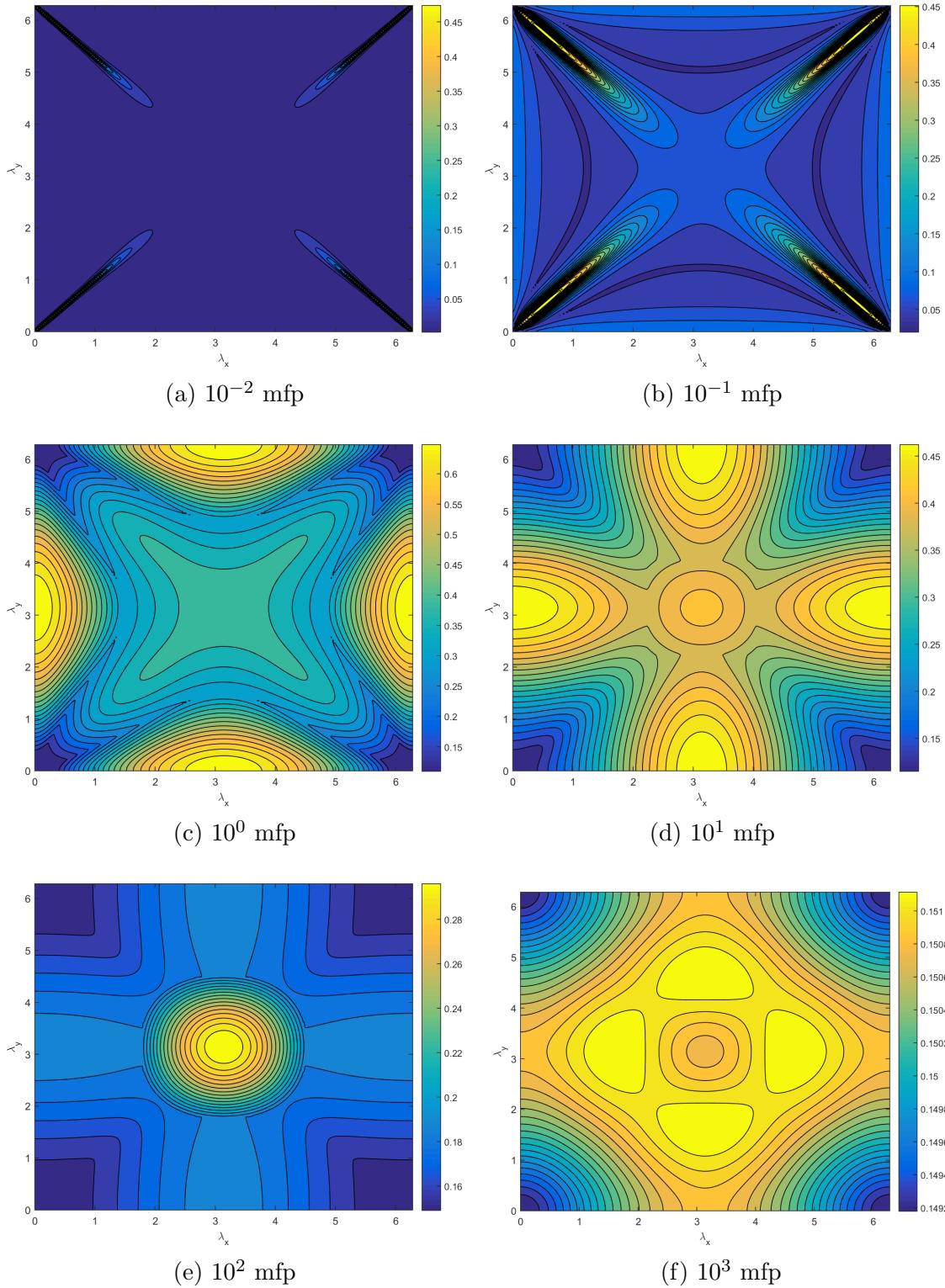


Figure C.7: Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL basis functions and LS_2 quadrature.

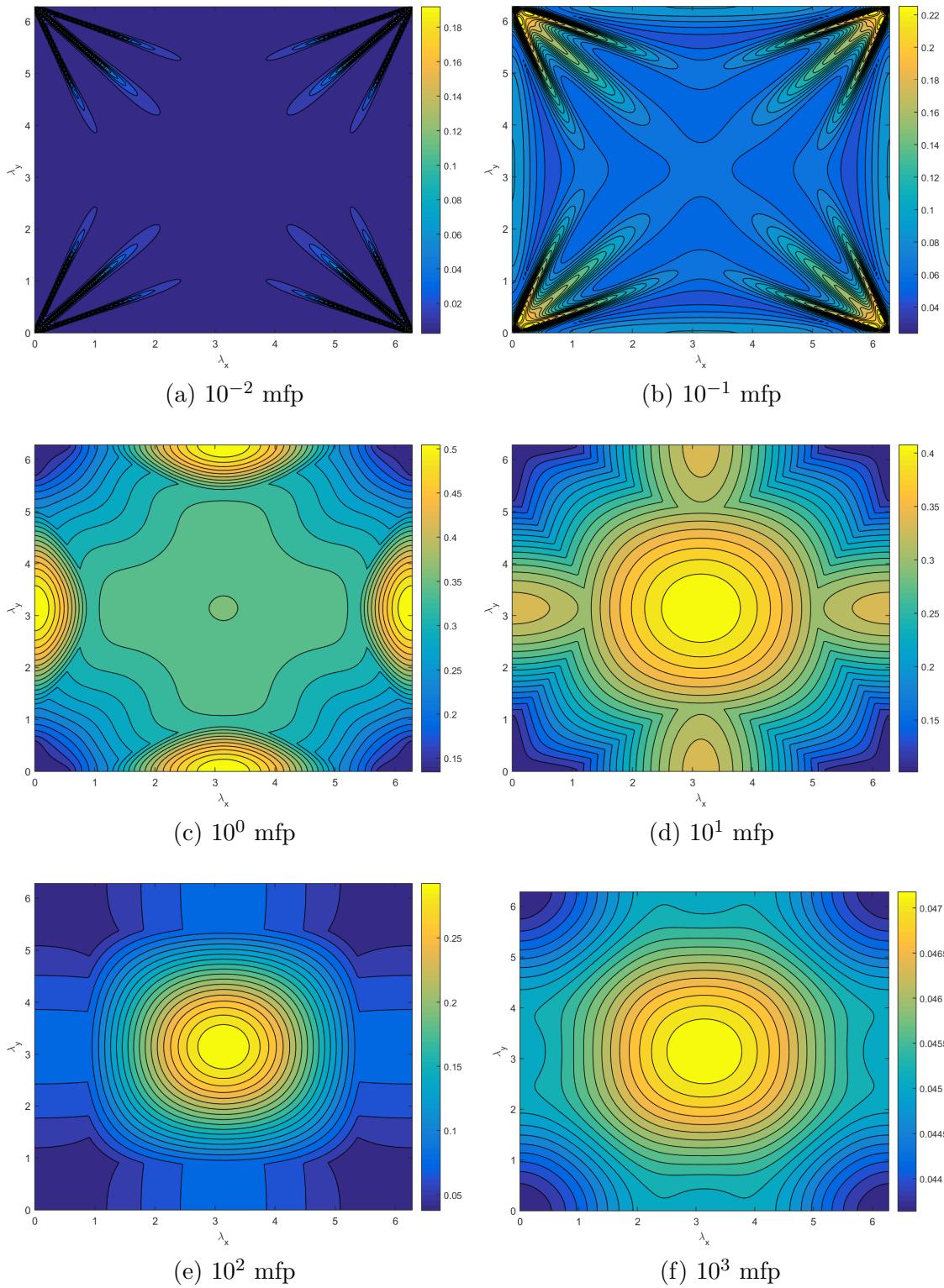


Figure C.8: Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL basis functions and LS_4 quadrature.

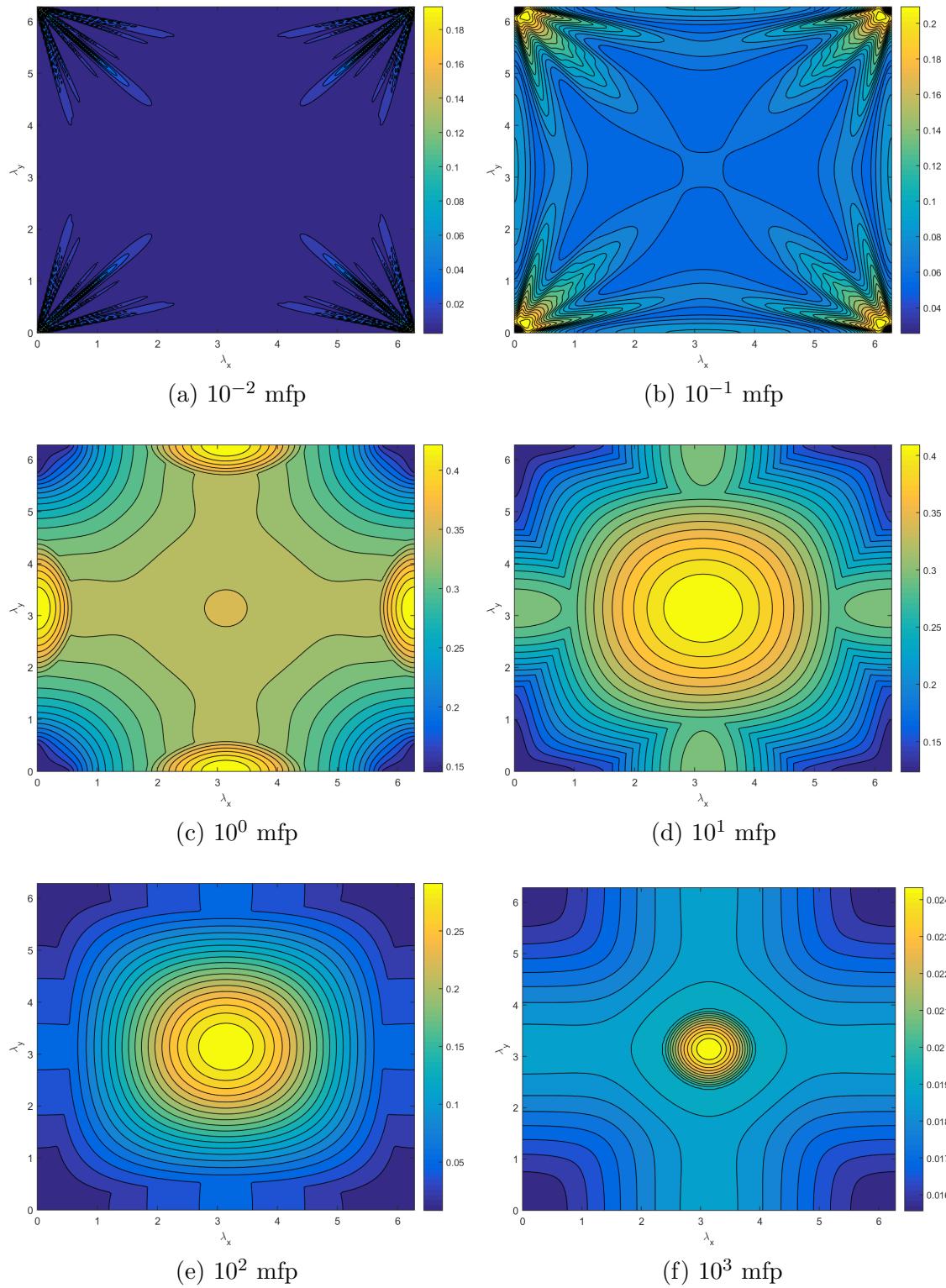


Figure C.9: Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL basis functions and LS_8 quadrature.

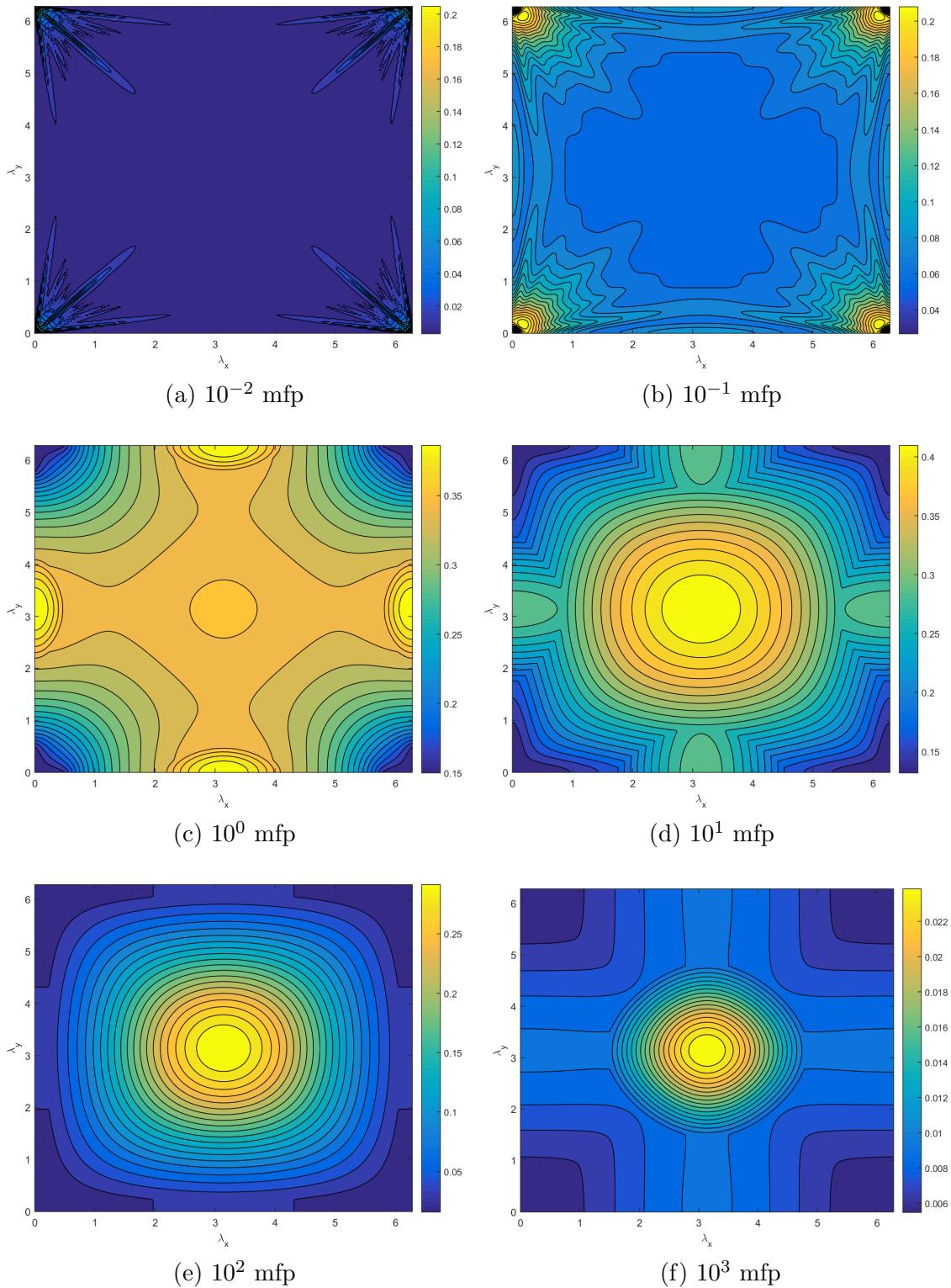


Figure C.10: Fourier wave number distribution for different mesh optical thicknesses of a single 2D square cell with PWL basis functions and LS_{16} quadrature.

C.4 Conservation of the SIP Diffusion Form

We conclude by demonstrating that the SIP form is a conservative discretization of the diffusion equation. This can be done by formulating an appropriate balance equation over any element. We show conservation by performing the following steps:

1. Write the N_K equations for element K in an analogous manner to Eq. (C.24).
2. Perform a summation over the N_K basis functions for element K .
3. Collect like terms and verify that an appropriate balance equation is produced for element K .

We first write the N_K DFEM SIP equations for element K as

$$\int_{\partial K} \kappa^{SIP} b_i^K (\Phi^- - \Phi^+) - \int_{\partial K} b_i^K \left(\frac{1}{2} D^- \partial_n \Phi^- + \frac{1}{2} D^+ \partial_n \Phi^+ \right) - \int_{\partial K} \frac{1}{2} D \partial_n b_i^K (\Phi^- - \Phi^+) \\ + \int_K D \vec{\nabla} b_i^K \cdot \vec{\nabla} \Phi + \int_K \sigma_a b_i^K \Phi = \int_K b_i^K Q, \quad 1 \leq i \leq N_K \quad , \quad (C.28)$$

where the Φ^- and Φ^+ terms correspond to the fluxes within and outside element K and the basis functions, b_i^K , always correspond to element K . Next, we sum over all the basis functions of element K . Performing this summation yields

$$\int_{\partial K} \kappa^{SIP} (\Phi^- - \Phi^+) - \int_{\partial K} \left(\frac{1}{2} D^- \partial_n \Phi^- + \frac{1}{2} D^+ \partial_n \Phi^+ \right) + \int_K \sigma_a \Phi = \int_K Q, \quad (C.29)$$

where we note that the summation of the basis function values and gradients within element K are

$$\sum_{i=1}^{N_K} b_i^K = 1, \quad (\text{C.30})$$

and

$$\sum_{i=1}^{N_K} \vec{\nabla} b_i^K = \vec{0}, \quad (\text{C.31})$$

respectively. Because of Eq. (C.31), the stiffness matrix and the surface matrices containing gradients of the basis functions are eliminated. We collect like terms, and write Eq. (C.29) as

$$\int_{\partial K} \left(\kappa^{SIP} \Phi^- - \frac{1}{2} D^- \partial_n \Phi^- \right) - \int_{\partial K} \left(\kappa^{SIP} \Phi^+ + \frac{1}{2} D^+ \partial_n \Phi^+ \right) + \int_K \sigma_a \Phi = \int_K Q. \quad (\text{C.32})$$

Equation (C.32) constitutes a balance equation for element K where we see that the total source is equal to the absorption plus net leakage. We note that the partial currents (the surface integrals in Eq. (C.32)) are only correct if κ^{SIP} is equal to 1/4. However, if this same procedure is performed to form a balance equation for element K' , then we can write

$$\int_{\partial K'} \left(\kappa^{SIP} \Phi^+ + \frac{1}{2} D^+ \partial_n \Phi^+ \right) - \int_{\partial K'} \left(\kappa^{SIP} \Phi^- - \frac{1}{2} D^- \partial_n \Phi^- \right) + \int_{K'} \sigma_a \Phi = \int_{K'} Q, \quad (\text{C.33})$$

where we used Φ^+ and Φ^- from element K and made use of the opposite face normals of element K' . Summing the net leakage terms between Eqs. (C.32) and (C.33), we see that the net leakage between elements K and K' vanishes. Therefore, every element forms an appropriate balance equation, and SIP is conservative.