ANL/NE-14/5

PROTEUS-SN Methodology Manual

Nuclear Engineering Division

About Argonne National Laboratory
Argonne is a U.S. Department of Energy laboratory managed by UChicago Argonne, LLC under contract DE-AC02-06CH11357. The Laboratory's main facility is outside Chicago, at 9700 South Cass Avenue, Argonne, Illinois 60439. For information about Argonne and its pioneering science and technology programs, see www.anl.gov.

DOCUMENT AVAILABILITY

Online Access: U.S. Department of Energy (DOE) reports produced after 1991 and a growing number of pre-1991 documents are available free via DOE's SciTech Connect (http://www.osti.gov/scitech/)

Reports not in digital format may be purchased by the public from the National Technical Information Service (NTIS):
U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Rd
Alexandra, VA 22312
www.ntis.gov
Phone: (800) 553-NTIS (6847) or (703) 605-6000
Fax: (703) 605-6900
Email: **orders@ntis.gov**

Reports not in digital format are available to DOE and DOE contractors from the Office of Scientific and Technical Information (OSTI):
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831-0062
www.osti.gov
Phone: (865) 576-8401
Fax: (865) 576-5728

Disclaimer
This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor UChicago Argonne, LLC, nor any of their employees or officers, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of document authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof, Argonne National Laboratory, or UChicago Argonne, LLC.

ANL/NE-14/5

PROTEUS-SN Methodology Manual

prepared by
Emily R. Shemon, Micheal A. Smith, and Changho Lee
Nuclear Engineering Division, Argonne National Laboratory

June 30, 2014

ABSTRACT

PROTEUS-SN is a three-dimensional, neutron transport code developed at Argonne National Laboratory. The code is applicable to all spectrum reactor transport calculations, particularly those in which a high degree of geometric detail is needed to resolve solution gradients. PROTEUS-SN solves the second order formulation of the transport equation using the continuous Galerkin finite element method in space, the discrete ordinates approximation in angle, and the multigroup approximation in energy. PROTEUS-SN's parallel methodology permits the efficient decomposition of the problem by both space and angle, permitting large problems to run efficiently (scalable) on hundreds of thousands of cores. PROTEUS-SN can also be used in serial or on smaller compute clusters (10's to 100's of cores) for smaller homogenized problems, although it is generally more computationally expensive than traditional homogenized methodology codes. PROTEUS-SN has been used to model partially homogenized systems, where regions of interest are represented explicitly and other regions are homogenized to reduce the problem size and required computational resources.

PROTEUS-SN is based upon sound numerical methodologies from the 1970s as it was intended to be a demonstration of the available parallelism. It is important to note that these algorithms are not accelerated significantly and thus performance compared with modern algorithms may not be optimal. PROTEUS-SN provides forward and adjoint eigenvalue calculation options using the power method. It treats neutron upscattering (in energy) by using conventional Gauss-Seidel iteration where each within-group system of equations is solved using synthetic diffusion accelerated scattering iterations with a SSOR-preconditioned conjugate gradient method (PETSc library) used to invert the streaming and collision operator.

A kinetics option based upon the adiabatic form of the quasi-static approximation has recently been included for performing simple time-dependent calculations in addition to standard steady state calculations. PROTEUS-SN handles void and reflective boundary conditions. Multigroup cross sections are generated with the MC^2-3 fast reactor multigroup cross section generation code or the cross section API. PROTEUS has been demonstrated to solve 500 billion - 1 trillion degrees of freedom on leadership computing resources.

PROTEUS-SN is written in Fortran 90 and links to the PETSc, METIS, HDF5, and MPICH libraries. It optionally links against the MOAB library and is a part of the SHARP multi-physics suite for coupled multi-physics analysis of nuclear reactors.

This document describes the methodology used in the PROTEUS-SN neutronics code, including theory, derivation, and numerical implementation. A companion user guide describes how to perform analysis with the code.

TABLE OF CONTENTS

Abstract i
Table of Contents iii
List of Figures v
List of Tables v
1 Introduction 7
1.1 Background 7
1.2 Code Summary 7
2 Theory 9
2.1 History 9
2.2 Summary of Approximations in Angle, Space, Energy and Time 9
2.3 Even-Parity Formulation 9
2.4 Derivation of the Second-Order Even-Parity Formulation 10
2.5 Application of Discrete Ordinates Approximation 13
2.6 Weak Form of the Even-Parity Equation 15
2.7 Spatial Approximation 18
2.8 Finite Element Continuity Conditions 21
2.9 Vacuum Boundary Conditions 22
2.10 Reflected Boundary Conditions 23
2.11 Summary 27
3 Numerical Implementation 29
3.1 Overview 29
3.2 Parallelism 29
3.3 Problem Initialization 31
3.3.1 Read Input 31
3.3.2 Mesh Decomposition 31
3.3.3 Computation of the Finite Element Matrices 31
3.4 Multi-Group Solve Procedure 32
3.4.1 Within-Group Parallel Preconditioned Conjugate Gradient Method for each Discrete Ordinate 32
3.4.2 Synthetic Diffusion Acceleration of the Within Group Scattering Source 34
3.4.3 Connection of Multigroup System of Equations 37
3.4.4 Fission Source Iteration 37
3.4.4.1 Power Iteration 37
3.4.4.2 Tchebychev Acceleration of the Fission Source 38
3.5 Methodology Limitations 38
4 Time-Dependent Calculations 39
4.1 Neutron Kinetics Theory 39
4.2 Point Kinetics Approximation 40
4.3 Adiabatic Approximation Implemented in PROTEUS-SN 41
4.4 Implementation 42
5 Cross Sections 45
5.1 Conventional Multigroup Cross Sections 45
5.2 Cross Section API 45

5.2.1 Subgroup Cross Section Method 45
5.2.2 Generalized Cross Section Method 47
6 Using PROTEUS-SN 50
6.1 Input 50
6.2 Output 50
6.2.1 Steady State Calculations 50
6.2.2 Time-Dependent Calculations 50
References 51
Appendix A. Finite Element Spatial Approximation 53
Appendix B. Angular Cubature in PROTEUS-SN 64
Appendix C. Reflected Boundary Condition 92

LIST OF FIGURES

Figure 2.1 Splitting of the Energy Range into Energy Groups 11
Figure 2.2 Example Angular Cubature for the Discrete Ordinates Method 14
Figure 2.3 Imposition of a Finite Element Mesh for a Control Assembly..... 18
Figure 2.4 Adjacent Triangular Finite Elements..... 21
Figure 2.5. Simplified 1-D Diagram of the Reflected Boundary Condition..... 23
Figure 2.6. Example Triangular Meshes of Spherical Cubature..... 24
Figure A.1. Linear (Left) and Quadratic (Right) Master Triangular Elements..... 53
Figure A.2. Linear (Left) and Quadratic (Right) Master Quadrilateral Elements 55
Figure A.3. A Reference Element (Left) and an General Element (Right)..... 56
Figure A.4. A Reference Element (Left) and an General Element (Right)..... 59
Figure B.1. Carlson S_{10}, S_{12}, and S_{16} Even-Moment Level Symmetric (top) 66
Figure B.2. Square (top) and Triangular (bottom) Legendre-Tchebychev 69
Figure B.3. Thurgood Triangulation of the Equilateral Triangle..... 70
Figure B.4. Thurgood (top) and Fitted Thurgood (bottom). 71
Figure B.5. Lebedov-Laikov Cubature 73
Figure B.6. Even Order (top) and Odd Order COBE Cubature..... 74
Figure B.7. Corner Based Equal Weight (top) and Least Squares Fitted (bottom). 75
Figure B.8 Tegmark Hierarchical Resolution of a Triangle on the Icosahedron. 76
Figure B.9. Center Based Equal Weight (top) and Least Squares Fitted (bottom)..... 76
Figure B.10. Number of Abscissas versus Resolution of the Cubature..... 79
Figure B.11. “Order” of the Cubature for a Given Resolution 79
Figure C.1 The Three Possible Intersection Outcomes for a Reflected Direction..... 93

LIST OF TABLES

Table 3.1. Strong Spatial Scaling of PROTEUS-SN on BlueGene/P Architecture..... 30
Table 3.2. Weak Angle Scaling of PROTEUS-SN on BlueGene/P Architecture..... 30
Table B.1. Level-Symmetric Integration Results for Eq. (B.1). 65
Table B.2. Legendre-Tchebychev Integration Error for Equation B.1 69
Table B.3. Thurgood Integration Error for Equation B.1 72
Table B.4. Lebedev-Laikov Integration Error for Equation B.1..... 73
Table B.5. COBE Integration Error for Equation B.1..... 74
Table B.6. Tegmark Integration Error for Equation B.1..... 77
Table B.7. Summary Information about Carlson Cubature in Gauss Library 80
Table B.8. Summary Information about Thurgood Cubature in Gauss Library..... 81
Table B.9. Summary Information about Legendre-Tchebychev Cubature in Gauss Library.. 82
Table B.10. Summary Information about Tegmark Cubature in Gauss Library 83
Table B.11. Summary Information about COBE Cubature in Gauss Library 84
Table B.12. Summary Information about Lebedev-Laikov Cubature in Gauss Library 85
Table B.13. Weight Distributions for Lebedev-Laikov and Carlson Y_l^m Fitted..... 87
Table B.14. Weight Distributions for Thurgood and Tegmark Y_l^m Fitted..... 89

1 Introduction

PROTEUS-SN is a three-dimensional neutron transport code targeted for modeling detailed geometry nuclear reactor applications. This document provides a basic description of the methodology in PROTEUS-SN for steady state and time-dependent calculations.

1.1 Background

PROTEUS-SN is a neutron transport code developed at Argonne National Laboratory that targets problems that require detailed spatial representations in order to resolve difficult spatial gradients. Spatially detailed solutions require large computational resources in order to (a) reduce the memory requirements per node and (b) reduce the wall-clock time to solution. PROTEUS-SN's implementation permits the parallel decomposition of the problem by both space and angle, permitting large problems to run on hundreds of thousands of cores. PROTEUS-SN can also be used in serial or on smaller compute clusters (10's to 100's of cores) for smaller homogenized problems, although it is generally more computationally expensive than traditional homogenized methodology codes for those conventional problems. PROTEUS-SN has been used to model partially heterogeneous systems, where the regions of interest are represented explicitly and other regions are homogenized to reduce the problem size and required computational resources. Providing accurate multigroup cross sections to PROTEUS is key to obtaining accurate results in all cases. For homogenized regions, a considerable amount of work has been done in the past and we rely upon MC^2-3 [1] for fast reactor cross section generation. For heterogeneous cross section, considerable research is still required and is thus a major area of recent work [2].

1.2 Code Summary

PROTEUS-SN is based upon the second order formulation of the transport equation. The second order "even-parity" form of the transport equation is discretized using the continuous Galerkin finite element method in space, the discrete ordinates approximation in angle, and the multigroup approximation in energy. It solves both forward and adjoint eigenvalue problems using conventional power iteration with up-scattering treatments and only has simple boundary condition treatments (vacuum and specular reflection). We have recently added an adiabatic option based on the quasi-static kinetics approximation to allow modeling of simple time-dependent problems.

Multigroup cross sections are read from *.anlxs or *.ISOTXS formatted files. The latter is the native file format for Argonne's MC^2-3 fast reactor multigroup cross section generation code and is also the compatible format for Argonne's well-established neutronics code, DIF3D. PROTEUS does not perform gamma radiation transport. However, PROTEUS accounts for power production by gammas by assuming that the gammas are absorbed at the location where

they are emitted. This is a conventional approximation to account for power production without implementation of full gamma transport.

The overall solution technique in PROTEUS-SN is discretization of the even-parity transport equation into a linear system of equations for the even-parity component of the angular flux. PROTEUS-SN permits parallelization of neutron transport problems in both space and angle which significantly reduces in the per-processor memory load for a given problem. The efficient parallel techniques in PROTEUS-SN permit the solution of very large problems. PROTEUS has been demonstrated to solve 500 billion - 1 trillion degrees of freedom on leadership computing resources.

PROTEUS-SN is written in Fortran 90 source code and also includes C preprocessor definitions. The code links against the PETSc [3], METIS [4], HDF5 [5], and MPICH [6] libraries. It optionally links against the MOAB [7] library and is part of the SHARP [8] multi-physics suite for coupled multi-physics analysis of nuclear reactors.

2 Theory

In this section we describe the history of PROTEUS-SN and provide a detailed description of the methodology used inside the code.

2.1 History

Development of PROTEUS-SN began in 2005 under the DOE Nuclear Energy Advanced Modeling and Simulation (NEAMS) program. PROTEUS-SN is the product of an original research code, termed "UNIC: Ultimate Neutronics Investigation Code", created to study the performance of different neutron transport algorithms for the targeted application of transitioning from homogenized to high-fidelity neutron transport calculations. Several solvers were developed under the UNIC framework [9], including PN2ND (a spherical harmonics/finite element method), SN2ND (a discrete ordinates/finite element method), and MOCFE (a 3D method-of-characteristics on unstructured finite element mesh). The PROTEUS-SN code is equivalent to the SN2ND code, and previous literature references refer to the code as SN2ND. The aforementioned codes were tested on various architectures ranging from small Linux clusters to leadership machines with hundreds of thousands of cores (IBM BG/P and Cray XT5).

Compared to the PN2ND spherical harmonics solver based on the even-parity formulation, PROTEUS-SN takes more iterative time due to the need for scattering iterations. However, PROTEUS-SN has a large reduction in memory usage compared to PN2ND due to the greatly reduced coupling in the angular space of the system of linear equations. Additionally, the PN2ND solver breaks down for transport problems which have discontinuities in the angular flux, while the SN2ND solver does not suffer the same consequence. The memory requirements in MOCFE were considered too large, and an alternative solver PROTEUS-MOC was developed [10] as an alternative method-of-characteristics technique. PROTEUS-MOC is not discussed in this manual.

2.2 Summary of Approximations in Angle, Space, Energy and Time

PROTEUS-SN solves the even-parity formulation of the neutron transport equation using the following approximations for the angular, spatial, and energy variables:

- Discrete ordinates approximation for the angular variable
- Continuous finite element approximation for the spatial variable
- Multigroup approximation for the energy variable

PROTEUS-SN does not include explicit time dependence. However, an adiabatic time-dependent capability has recently been implemented [11].

2.3 Even-Parity Formulation

PROTEUS-SN is based on the second-order even-parity formulation of the transport equation which is less widely used than the familiar first-order form. The second-order form was chosen because it works well with the continuous finite element method for spatial discretization and

there are considerable problems with scaling sweeping methods based on first order treatments on unstructured meshes. When the second order formulation is discretized with the continuous finite element method, the resulting within-group linear system is symmetric positive definite which allows the well-established parallel conjugate gradient algorithm to be applied. The parallel conjugate gradient method has been well-established in other physics fields and is known to have efficient parallel performance.

The second-order formulation has known drawbacks. First, the presence of the inverse total cross section term results in an ill-posed system of equations for void mediums (due to division by zero). This issue can be circumvented by homogenization of void regions with neighboring materials. However, this required homogenization inherently limits the heterogeneous capability of the code. Finally, the condition number of the matrix corresponding to the discretized system of equations increases quadratically as the element size is reduced. Further, the condition number increases considerably as the polynomial order of the element basis functions increase(linear, quadratic, cubic, etc.). Finally, the methodology does not display more than second order convergence with respect to spatial mesh refinement, whereas first order forms can display second or third order accuracy. All of these issues pose serious problems with regard to modeling, but the even-parity method was chosen as it posed the best case scenario for performance and scalability (if it performed worse than first order sweeping codes then there is no point in looking at better continuous finite element treatments). Based on experience, we recommend using a larger number of linear elements rather than using quadratic elements for heterogeneous geometry modeling. We note that there are known continuous finite element schemes which avoid all of the preceding problems, but they have been studied far less than the current research.

2.4 Derivation of the Second-Order Even-Parity Formulation

To derive the even-parity form of the transport equation upon which PROTEUS-SN is based, we begin with the time-independent form of the Boltzmann transport equation taken from Lewis [12].

$$\begin{aligned}\hat{\Omega}\cdot\vec{\nabla}\psi(\vec{r},\hat{\Omega},E)+\Sigma_t(\vec{r},E)\psi(\vec{r},\hat{\Omega},E) &= \int\int\Sigma_s(\vec{r},\hat{\Omega}'\to\hat{\Omega},E'\to E)\psi(\vec{r},\hat{\Omega}',E')d\Omega' dE' \\ &\quad +\chi(E)\int_0^\infty \nu\Sigma_f(\vec{r},E')\phi(\vec{r},E')dE'+q(\vec{r},\hat{\Omega},E)\end{aligned} \tag{2.1}$$

In Eq. (2.1), the quantity $\psi(\vec{r},\hat{\Omega},E)$ represents the neutron angular flux which is a function of three space variables (x, y, and z in $\vec{r}$), two angular variables (θ and ϕ in $\hat{\Omega}$), and one energy variable (E). The quantity $\phi(\vec{r},E)$ is the angular flux integrated over all angles on the unit sphere. The macroscopic total cross section, $\Sigma_t(\vec{r},E)$, represents the sum of all possible neutron reaction probabilities with energy E at the point $\vec{r}$. Similarly, the scattering kernel $\Sigma_s(\vec{r},\hat{\Omega}'\to\hat{\Omega},E'\to E)$ represents the probability that a particle at $\vec{r}$ with energy E' traveling in

the direction $\hat{\Omega}'$ is scattered into energy dE about E with direction $d\Omega$ about $\hat{\Omega}$. The term $\nu\Sigma_f(\vec{r},E')$ represents the total number of fission neutrons being born as a result of a fission occurring at energy E' at the point $\vec{r}$. These secondary fission source neutrons are emitted with energy distribution $\chi(E)$. Finally, $q(\vec{r},\hat{\Omega},E)$ is a generic external fixed neutron source. Note the implementation of PROTEUS-SN does not currently allow fixed external sources, but we include the term for completeness.

The first step in a deterministic formulation is to apply a multigroup approach [12] to the energy dependence of the neutron flux. This approach divides the energy range of interest into G intervals with an upper energy cutoff, E_0, and lower energy cutoff, E_G, as seen in Figure 2.1.

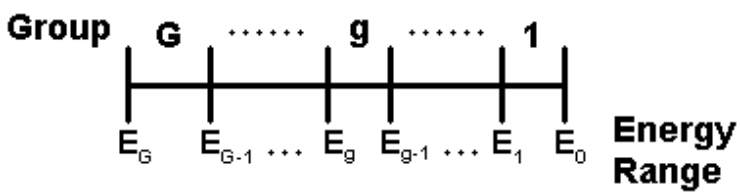


Figure 2.1 Splitting of the Energy Range into Energy Groups

Integrating the transport equation over energy group g results in G equations where the degrees of freedom are the group angular fluxes $\psi_g(\vec{r},\hat{\Omega})$ defined by Eq. (2.2),

$$\begin{aligned}\psi_g(\vec{r},\hat{\Omega}) &= \int_{E_g}^{E_{g-1}} \psi(\vec{r},\hat{\Omega},E)dE \\ &= \int_g \psi(\vec{r},\hat{\Omega},E)dE, \quad g=1,\ldots,G.\end{aligned} \tag{2.2}$$

The multi-group cross sections in the resulting equations are defined in more detail elsewhere [12]. Using these relationships, the multigroup Boltzmann transport equation is derived exactly as

$$\begin{aligned}\hat{\Omega}\cdot\vec{\nabla}\psi_g(\vec{r},\hat{\Omega})+\Sigma_{t,g}(\vec{r})\psi_g(\vec{r},\hat{\Omega}) &= \sum_{g'=1}^{G}\int \Sigma_{s,g'\to g}(\vec{r},\hat{\Omega}\cdot\hat{\Omega}')\psi_{g'}(\vec{r},\hat{\Omega}')d\hat{\Omega}' \\ &\quad +\chi_g\sum_{g'=1}^{G}\nu\Sigma_{f,g'}(\vec{r})\phi_{g'}(\vec{r})+q_g(\vec{r},\hat{\Omega})\end{aligned} \tag{2.3}$$

It is convenient to lump the group-to-group scattering sources (except the within group scattering term) into one term called the group source in order to obtain Eq. (2.4), a familiar form for the within group transport equation:

$$\hat{\Omega}\cdot\vec{\nabla}\psi_g(\vec{r},\hat{\Omega})+\Sigma_{t,g}(\vec{r})\psi_g(\vec{r},\hat{\Omega})=W_g(\vec{r},\hat{\Omega})+S_g(\vec{r},\hat{\Omega}) \tag{2.4}$$

The term $W_g(\vec{r},\hat{\Omega})$ represents the within-group scattering source, and the term $S_g(\vec{r},\hat{\Omega})$ represents the fission source, external source, and in-scattering source to energy group g from other groups. Expanding the differential scattering cross section into Legendre basis functions of Mth order, $\Sigma_{s,g'\to g}(\vec{r},\hat{\Omega}\cdot\hat{\Omega}') = \sum_{m=0}^{M}\Sigma_{s,g\to g,m}(\vec{r})P_m(\hat{\Omega}\cdot\hat{\Omega}')$, we define the within-group and other sources by

$$\begin{aligned} W_g(\vec{r},\hat{\Omega}) &= \int \Sigma_{s,g\to g}(\vec{r},\hat{\Omega}\cdot\hat{\Omega}')\psi_g(\vec{r},\hat{\Omega}')d\hat{\Omega}' \\ &= \sum_{m=0}^{M}\int \Sigma_{s,g\to g,m}(\vec{r})P_m(\hat{\Omega}\cdot\hat{\Omega}')\psi_g(\vec{r},\hat{\Omega}')d\hat{\Omega}' \end{aligned} \tag{2.5}$$

$$\begin{aligned} S_g(\vec{r},\hat{\Omega}) &= \sum_{g'\neq g}^{G}\int \Sigma_{s,g'\to g}(\vec{r},\hat{\Omega}\cdot\hat{\Omega}')\psi_{g'}(\vec{r},\hat{\Omega}')d\hat{\Omega}' + \chi_g\sum_{g'=1}^{G}\nu\Sigma_{f,g'}(\vec{r})\phi_{g'}(\vec{r}) + q_g(\vec{r},\hat{\Omega}) \\ &= \sum_{g'\neq g}^{G}\sum_{m=0}^{M}\int \Sigma_{s,g\to g,m}(\vec{r})P_m(\hat{\Omega}\cdot\hat{\Omega}')\psi_{g'}(\vec{r},\hat{\Omega}')d\hat{\Omega}' + \chi_g\sum_{g'=1}^{G}\nu\Sigma_{f,g'}(\vec{r})\phi_{g'}(\vec{r}) + q_g(\vec{r},\hat{\Omega}) \end{aligned} \tag{2.6}$$

To derive the second order even-parity formulation, we define the even-parity component $\psi_g^+(\vec{r},\hat{\Omega})$ to be the sum of the angular flux evaluated at directions $\hat{\Omega}$ and $-\hat{\Omega}$. Similarly, the odd-parity angular flux $\psi_g^-(\vec{r},\hat{\Omega})$ is defined as the difference of the true angular flux evaluated at directions $\hat{\Omega}$ and $-\hat{\Omega}$.

$$\psi_g^{\pm}(\vec{r},\hat{\Omega}) \equiv \frac{1}{2}\left(\psi_g(\vec{r},\hat{\Omega}) \pm \psi_g(\vec{r},-\hat{\Omega})\right) \tag{2.7}$$

It follows directly that the angular flux can be written as the sum of the even- and odd-parity components:

$$\psi_g(\vec{r},\hat{\Omega}) = \psi_g^+(\vec{r},\hat{\Omega}) + \psi_g^-(\vec{r},\hat{\Omega}) \tag{2.8}$$

The even- and odd-parity components of the flux can be shown to have the following properties, where the function $\phi_g(\vec{r})$ represents the group scalar flux.

$$\psi_g^+(\vec{r},\hat{\Omega}) = \psi_g^+(\vec{r},-\hat{\Omega}) \tag{2.9}$$

$$\int \psi_g^+(\vec{r},\hat{\Omega})d\Omega = \phi_g(\vec{r}) \tag{2.10}$$

$$\psi_g^-(\vec{r},\hat{\Omega}) = -\psi_g^-(\vec{r},-\hat{\Omega}) \tag{2.11}$$

$$\int \psi_g^-(\vec{r},\hat{\Omega})d\Omega = 0 \tag{2.12}$$

Inserting the expression for the angular flux (Eq. (2.8)) into the within-group first order transport equation (Eq. (2.4)) yields the following equation.

$$\hat{\Omega}\cdot\vec{\nabla}\left[\psi_g^+(\vec{r},\hat{\Omega})+\psi_g^-(\vec{r},\hat{\Omega})\right]+\Sigma_{t,g}(\vec{r})\left[\psi_g^+(\vec{r},\hat{\Omega})+\psi_g^-(\vec{r},\hat{\Omega})\right]=W_g(\vec{r},\hat{\Omega})+S_g(\vec{r},\hat{\Omega}) \quad (2.13)$$

Evaluating Eq. (2.13) at $-\hat{\Omega}$ and adding to Eq. (2.13) yields the even-parity form of the transport equation in Eq. (2.14). Evaluating Eq. (2.13) at $-\hat{\Omega}$ and subtracting from Eq. (2.13) yields the odd-parity form of the transport equation given by Eq. (2.15).

$$\hat{\Omega}\cdot\vec{\nabla}\psi_g^-(\vec{r},\hat{\Omega})+\Sigma_{t,g}(\vec{r})\psi_g^+(\vec{r},\hat{\Omega})=W_g^+(\vec{r},\hat{\Omega})+S_g^+(\vec{r},\hat{\Omega}) \quad (2.14)$$

$$\hat{\Omega}\cdot\vec{\nabla}\psi_g^+(\vec{r},\hat{\Omega})+\Sigma_{t,g}(\vec{r})\psi_g^-(\vec{r},\hat{\Omega})=W_g^-(\vec{r},\hat{\Omega})+S_g^-(\vec{r},\hat{\Omega}) \quad (2.15)$$

We have defined the even- and odd-parity components of the within group source by:

$$\begin{aligned} W_g^+(\vec{r},\hat{\Omega}) &= \sum_{m\in even}\int_{4\pi}\Sigma_{s,g\to g,m}(\vec{r})P_m(\hat{\Omega}\cdot\hat{\Omega}')\psi_g^+(\vec{r},\hat{\Omega}')d\hat{\Omega}' \\ W_g^-(\vec{r},\hat{\Omega}) &= \sum_{m\in odd}\int_{4\pi}\Sigma_{s,g\to g,m}(\vec{r})P_m(\hat{\Omega}\cdot\hat{\Omega}')\psi_g^-(\vec{r},\hat{\Omega}')d\hat{\Omega}' \end{aligned} \quad (2.16)$$

These expressions are derived exactly due to the even and odd dependence on angle of the Legendre polynomials and the even- and odd-parity fluxes. Written compactly, this is equivalent to:

$$\begin{aligned} W_g^\pm(\vec{r},\hat{\Omega}) &= \sum_{m^\pm}\int_{4\pi}\Sigma_{s,g\to g,m^\pm}(\vec{r})P_{m^\pm}(\hat{\Omega}\cdot\hat{\Omega}')\psi_g^\pm(\vec{r},\hat{\Omega}')d\hat{\Omega}' \\ &= \int_{4\pi}\Sigma_{s,g\to g}^\pm(\vec{r},\hat{\Omega}\cdot\hat{\Omega}')\psi_g^\pm(\vec{r},\hat{\Omega}')d\hat{\Omega}'. \end{aligned} \quad (2.17)$$

Similarly, the remaining even- and odd-parity components of the source term are defined in Eq. (2.18), where the fission source term drops out of the odd-parity component because it is isotropic.

$$\begin{aligned} S_g^+(\vec{r},\hat{\Omega}) &= \sum_{g'\neq g}^{G}\sum_{m=even}^{M}\int\Sigma_{s,g\to g,m}(\vec{r})P_m(\hat{\Omega}\cdot\hat{\Omega}')\psi_{g'}(\vec{r},\hat{\Omega}')d\hat{\Omega}'+q_q^+(\vec{r}) \\ &\quad +\chi_g\sum_{g'=1}^{G}\nu\Sigma_{f,g'}(\vec{r})\phi_{g'}(\vec{r}) \\ S_g^-(\vec{r},\hat{\Omega}) &= \sum_{g'\neq g}^{G}\sum_{m=odd}^{M}\int\Sigma_{s,g\to g,m}(\vec{r})P_m(\hat{\Omega}\cdot\hat{\Omega}')\psi_{g'}(\vec{r},\hat{\Omega}')d\hat{\Omega}'+q_q^-(\vec{r}) \end{aligned} \quad (2.18)$$

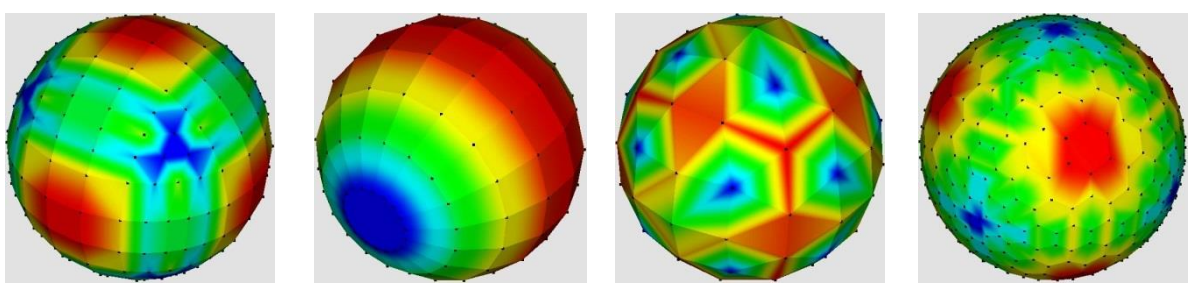
2.5 Application of Discrete Ordinates Approximation

We now make the following two approximations of the angular dependence of the flux, collectively known as the discrete ordinates approximation.

$$\begin{aligned}\psi_g^+(\vec{r},\hat{\Omega}) &\approx \psi_g^+(\vec{r},\hat{\Omega}_n) \\ &= \psi_{g,n}^+(\vec{r})\end{aligned} \tag{2.19}$$

$$\begin{aligned}\varphi_{g,l,m}^+(\vec{r}) &= \int Y_{l,m}(\hat{\Omega})\psi_g^+(\vec{r},\hat{\Omega})d\Omega \\ &\approx \sum_n Y_{l,m}(\hat{\Omega}_n)\psi_g^+(\vec{r},\hat{\Omega}_n)w_n\end{aligned} \tag{2.20}$$

Eq. (2.19) assumes the angular flux is a 0th order finite element approximation in the angular space. With this approximation, the flux is solved for a set of directions $\Omega_n, n = 1 \ldots N$ on the unit sphere, and area weights $w_n, n = 1 \ldots N$ are assigned to each direction. The choice of these directions and weights is arbitrary, but they are typically derived such that Eq. (2.20) is exact. Eq. (2.20) represents the spherical harmonic projection of the discrete ordinate solution which is needed for compatibility with the scattering kernel representation. Figure 2.2 depicts several angular cubatures available in PROTEUS-SN, where the black dots represent the intersections of each direction with the unit sphere and the coloring indicates the relative magnitudes of the weights assigned to each direction.



Carlson Level-Symmetric S_{16} | Square Legendre-Tchebychev S_{16} | Lebedev-Laikov S_{15} | Thurgood S_{15}

Figure 2.2 Example Angular Cubature for the Discrete Ordinates Method

Using the discrete ordinates approximation, we rewrite the even- and odd-parity transport equations using the shorthand notation for each direction in the cubature and obtain the within-group discrete ordinates equations for the even-parity formulation:

$$\hat{\Omega}_n \cdot \vec{\nabla}\psi_{g,n}^-(\vec{r}) + \Sigma_{t,g}(\vec{r})\psi_{g,n}^+(\vec{r}) = W_{g,n}^+(\vec{r}) + S_{g,n}^+(\vec{r}) \tag{2.21}$$

$$\hat{\Omega}_n \cdot \vec{\nabla}\psi_{g,n}^+(\vec{r}) + \Sigma_{t,g}(\vec{r})\psi_{g,n}^-(\vec{r}) = W_{g,n}^-(\vec{r}) + S_{g,n}^-(\vec{r}) \tag{2.22}$$

In the following section, we discuss discretization of the spatial variable.

2.6 Weak Form of the Even-Parity Equation

In order to apply the finite element approximation, we weight the even-parity transport equation, Eq. (2.21), with a set of spatial trial functions, $\lambda(\vec{r})$, and integrate over space to obtain the weak form,

$$\int dV \lambda(\vec{r})\left[\hat{\Omega}_n \cdot \vec{\nabla} \psi_{g,n}^{-}(\vec{r}) + \Sigma_{t,g}(\vec{r})\psi_{g,n}^{+}(\vec{r}) = W_{g,n}^{+}(\vec{r}) + S_{g,n}^{+}(\vec{r})\right]. \tag{2.23}$$

Applying the divergence theorem to the gradient term, we transform the derivative on the odd-parity flux term into a surface integral term and a term where the derivative is applied to the trial function,

$$\begin{aligned}\int\left[-\hat{\Omega}_n \cdot \vec{\nabla}\lambda(\vec{r})\psi_{g,n}^{-}(\vec{r}) + \lambda(\vec{r})\Sigma_{t,g}(\vec{r})\psi_{g,n}^{+}(\vec{r})\right]dV = \int\left[\lambda(\vec{r})W_{g,n}^{+}(\vec{r}) + \lambda(\vec{r})S_{g,n}^{+}(\vec{r})\right]dV \\ - \oint \hat{\Omega}_n \cdot \hat{n}\ \lambda(\vec{r})\psi_{g,n}^{-}(\vec{r})d\Gamma\end{aligned} \tag{2.24}$$

where $\hat{n}$ is the outward normal from the domain surface, Γ. We now seek to replace the term with odd-parity component dependence, $-\oint \hat{\Omega}_n \cdot \hat{n}\ \lambda(\vec{r})\psi_{g,n}^{-}(\vec{r})d\Gamma$, by an equivalent definition using boundary conditions.

It can be shown using angular parity arguments that the following expression holds:

$$\oint d\Gamma\, \hat{\Omega}_n \cdot \hat{n}\ \lambda(\vec{r})\psi_{g,n}^{-}(\vec{r}) = \oint d\Gamma \left|\hat{\Omega}_n \cdot \hat{n}\right| \lambda(\vec{r})\psi_{g,n}^{+}(\vec{r}) + 2\oint d\Gamma\, \hat{\Omega}_n \cdot \hat{n}\lambda(\vec{r})\psi_{\Gamma,g}^{\hat{\Omega}\cdot\hat{n}<0}(\vec{r},\hat{\Omega}) \tag{2.25}$$

Let the boundary condition on the outer domain surface $\vec{r} \in \Gamma$ be given by $\psi_g(\vec{r},\hat{\Omega}_n) = \psi_{\Gamma,g}(\vec{r},\hat{\Omega}_n)$ for incoming directions, $\hat{\Omega}_n \cdot \hat{n} < 0$. We can express the angular flux on the boundary as:

$$\begin{aligned}\hat{\Omega}_n \cdot n_\gamma \psi_{\Gamma,g}(\vec{r},\hat{\Omega}_n) &= \hat{\Omega}_n \cdot n_\gamma \psi_{\Gamma,g}^{\hat{\Omega}_n\cdot\hat{n}<0}(\vec{r},\hat{\Omega}_n) + \hat{\Omega}_n \cdot n_\gamma \psi_{\Gamma,g}^{\hat{\Omega}_n\cdot\hat{n}>0}(\vec{r},\hat{\Omega}_n) \\ &= \hat{\Omega}_n \cdot n_\gamma \psi_{\Gamma,g}^{+}(\vec{r},\hat{\Omega}_n) + \hat{\Omega}_n \cdot n_\gamma \psi_{\Gamma,g}^{-}(\vec{r},\hat{\Omega}_n)\end{aligned}. \tag{2.26}$$

We can expand the incoming and outgoing angular fluxes into even- and odd parity components as

$$\begin{aligned}\hat{\Omega}_n \cdot n_\gamma \psi_{\Gamma,g}^{\hat{\Omega}_n\cdot\hat{n}<0}(\vec{r},\hat{\Omega}_n) &= \hat{\Omega}_n \cdot n_\gamma \psi_{\Gamma,g}^{+,\hat{\Omega}_n\cdot\hat{n}<0}(\vec{r},\hat{\Omega}_n) + \hat{\Omega}_n \cdot n_\gamma \psi_{\Gamma,g}^{-,\hat{\Omega}_n\cdot\hat{n}<0}(\vec{r},\hat{\Omega}_n) \\ \hat{\Omega}_n \cdot n_\gamma \psi_{\Gamma,g}^{\hat{\Omega}_n\cdot\hat{n}>0}(\vec{r},\hat{\Omega}_n) &= \hat{\Omega}_n \cdot n_\gamma \psi_{\Gamma,g}^{+,\hat{\Omega}_n\cdot\hat{n}>0}(\vec{r},\hat{\Omega}_n) + \hat{\Omega}_n \cdot n_\gamma \psi_{\Gamma,g}^{-,\hat{\Omega}_n\cdot\hat{n}>0}(\vec{r},\hat{\Omega}_n)\end{aligned}. \tag{2.27}$$

By definition, we cannot apply angular restraints to the even- and odd-parity fluxes and thus find Eq. (2.27) must be modified into something like

$$\begin{aligned}2\cdot\hat{\Omega}_n \cdot n_\gamma \psi_{\Gamma,g}^{\hat{\Omega}_n\cdot\hat{n}<0}(\vec{r},\hat{\Omega}_n) &= \hat{\Omega}_n \cdot n_\gamma \psi_{\Gamma,g}^{+}(\vec{r},\hat{\Omega}_n) + \hat{\Omega}_n \cdot n_\gamma \psi_{\Gamma,g}^{-}(\vec{r},\hat{\Omega}_n) \\ 2\cdot\hat{\Omega}_n \cdot n_\gamma \psi_{\Gamma,g}^{\hat{\Omega}_n\cdot\hat{n}>0}(\vec{r},\hat{\Omega}_n) &= \hat{\Omega}_n \cdot n_\gamma \psi_{\Gamma,g}^{+}(\vec{r},\hat{\Omega}_n) + \hat{\Omega}_n \cdot n_\gamma \psi_{\Gamma,g}^{-}(\vec{r},\hat{\Omega}_n)\end{aligned}. \tag{2.28}$$

such that Eq. (2.26) can still be satisfied. For vacuum boundary conditions we can write

$$\begin{aligned} 2\cdot\hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{\hat{\Omega}_n\cdot\hat{n}<0}(\vec{r},\hat{\Omega}_n) &= 0 = \hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{+}(\vec{r},\hat{\Omega}_n)+\hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{-}(\vec{r},\hat{\Omega}_n) \\ \hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{+}(\vec{r},\hat{\Omega}_n) &= -\hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{-}(\vec{r},\hat{\Omega}_n) \end{aligned} . \tag{2.29}$$

It is important to note that while this was explicitly defined for incoming directions, we have in fact defined a rule for the parity system that must be obeyed for both incoming and outgoing directions. Thus we can evaluate Eq. (2.29) for the outgoing direction set and find

$$\begin{aligned} -\hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{+}(\vec{r},-\hat{\Omega}_n) &= -1\cdot -\hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{-}(\vec{r},-\hat{\Omega}_n) \\ -\hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{+}(\vec{r},-\hat{\Omega}_n) &= -\hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{-}(\vec{r},\hat{\Omega}_n) \end{aligned} . \tag{2.30}$$

In this case, we see that for the "outgoing" directions, we must impose a different relation between the even- and odd-parity fluxes which cannot be true. Because Eq. (2.29) is trying to equivalence an odd*even function on the left hand side with an odd*odd function on the right hand side, we know that we must physically convert either the left hand side to even*even or right hand side to odd*odd such that we obey the original constraint given by Eq. (2.29). In this case, the sign resulting from $\hat{\Omega}_n\cdot n_\gamma$ for $\hat{\Omega}_n\cdot n_\gamma<0$ cancels out on both sides of the equation and we are free to modify Eq. (2.29) into one of

$$\begin{aligned} -\left|\hat{\Omega}_n\cdot n_\gamma\right|\psi_{\Gamma,g}^{+}(\vec{r},\hat{\Omega}_n) &= -\hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{-}(\vec{r},\hat{\Omega}_n) \\ \hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{+}(\vec{r},\hat{\Omega}_n) &= \left|\hat{\Omega}_n\cdot n_\gamma\right|\psi_{\Gamma,g}^{-}(\vec{r},\hat{\Omega}_n) \end{aligned} . \tag{2.31}$$

It is very important to note that the signs are preserved with respect to Eq. (2.29) in both cases with respect to $\hat{\Omega}_n\cdot n_\gamma<0$. The first of these is used for the even-parity transport methodology while the second is used for the odd-parity one. Looking at Eq. (2.28), we can see that conversion of Eq. (2.27) into Eq. (2.28) was actually invalid and, for the even-parity system, we must modify it into the form

$$\begin{aligned} 2\cdot\hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{\hat{\Omega}_n\cdot\hat{n}<0}(\vec{r},\hat{\Omega}_n) &= -\left|\hat{\Omega}_n\cdot n_\gamma\right|\psi_{\Gamma,g}^{+}(\vec{r},\hat{\Omega}_n)+\hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{-}(\vec{r},\hat{\Omega}_n) \\ 2\cdot\hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{\hat{\Omega}_n\cdot\hat{n}>0}(\vec{r},\hat{\Omega}_n) &= \left|\hat{\Omega}_n\cdot n_\gamma\right|\psi_{\Gamma,g}^{+}(\vec{r},\hat{\Omega}_n)+\hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{-}(\vec{r},\hat{\Omega}_n) \end{aligned} . \tag{2.32}$$

For completeness, it is not hard to show that

$$\left\{-\left|\hat{\Omega}_n\cdot n_\gamma\right|\psi_{\Gamma,g}^{+}(\vec{r},\hat{\Omega}_n)\right\}_{\hat{\Omega}_n\cdot n_\gamma<0}+\left\{\left|\hat{\Omega}_n\cdot n_\gamma\right|\psi_{\Gamma,g}^{+}(\vec{r},\hat{\Omega}_n)\right\}_{\hat{\Omega}_n\cdot n_\gamma>0}=\hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{+}(\vec{r},\hat{\Omega}_n) . \tag{2.33}$$

Given these relations, we can generically write an incoming flux boundary condition as:

$$\hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{-}(\vec{r},\hat{\Omega}_n)=\left|\hat{\Omega}_n\cdot n_\gamma\right|\psi_{\Gamma,g}^{+}(\vec{r},\hat{\Omega}_n)+2\cdot\hat{\Omega}_n\cdot n_\gamma\psi_{\Gamma,g}^{\hat{\Omega}_n\cdot\hat{n}<0}(\vec{r},\hat{\Omega}_n) . \tag{2.34}$$

Or, given a weighting function and integrals on the surface, we write:

$$\oint d\Gamma\, \hat{\Omega}_n \cdot \hat{n}\ \lambda(\vec{r})\psi^-_{g,n}(\vec{r}) = \oint d\Gamma \left|\hat{\Omega}_n \cdot \hat{n}\right| \lambda(\vec{r})\psi^+_{g,n}(\vec{r}) + 2\oint d\Gamma\, \hat{\Omega}_n \cdot \hat{n}\lambda(\vec{r})\psi^{\hat{\Omega}\cdot\hat{n}<0}_{\Gamma,g}(\vec{r},\hat{\Omega}) \quad (2.35)$$

We substitute this expression into Eq. (2.24) and obtain the second-order transport equation with modified natural boundary conditions:

$$\begin{aligned}\int\left[-\hat{\Omega}_n \cdot \vec{\nabla}\lambda(\vec{r})\psi^-_{g,n}(\vec{r}) + \lambda(\vec{r})\Sigma_{t,g}(\vec{r})\psi^+_{g,n}(\vec{r})\right]dV \\ = \int\left[\lambda(\vec{r})W^+_{g,n}(\vec{r}) + \lambda(\vec{r})S^+_{g,n}(\vec{r})\right]dV \\ - \oint d\Gamma\left|\hat{\Omega}_n \cdot \hat{n}\right|\lambda(\vec{r})\psi^+_{g,n}(\vec{r}) - 2\oint d\Gamma\ \hat{\Omega}_n \cdot \hat{n}\ \lambda(\vec{r})\psi^{\hat{\Omega}_n\cdot\hat{n}<0}_{\Gamma,g,n}(\vec{r})\end{aligned} \quad (2.36)$$

Next, we solve the odd-parity transport equation, Eq. (2.22), for the odd-parity flux,

$$\psi^-_{g,n}(\vec{r}) = \frac{1}{\Sigma_{t,g}(\vec{r})}\left(-\hat{\Omega}_n \cdot \vec{\nabla}\psi^+_{g,n}(\vec{r}) + W^-_{g,n}(\vec{r}) + S^-_{g,n}(\vec{r})\right). \quad (2.37)$$

We note that the odd parity within-group and in-scattering sources $W^-_{g,n}(\vec{r})$ and $S^-_{g,n}(\vec{r})$ are still dependent on the odd-parity component of the flux, so the above equation is not an explicit definition for the odd-parity component of the flux. However, we can update these sources iteratively later in the solver rather than evaluating them explicitly with the odd-parity flux.

Substituting Eq. (2.37) into Eq. (2.36) to eliminate dependence on the odd-parity flux, we obtain the weak form of the even-parity transport formulation with modified natural boundary conditions,

$$\begin{aligned}\int\left[\hat{\Omega}_n \cdot \vec{\nabla}\lambda(\vec{r})\frac{1}{\Sigma_{t,g}(\vec{r})}\hat{\Omega}_n \cdot \vec{\nabla}\psi^+_{g,n}(\vec{r}) + \lambda(\vec{r})\Sigma_{t,g}(\vec{r})\psi^+_{g,n}(\vec{r})\right]dV \\ = \int\left[\lambda(\vec{r})W^+_{g,n}(\vec{r}) + \lambda(\vec{r})S^+_{g,n}(\vec{r})\right]dV \\ + \int\left[\hat{\Omega}_n \cdot \vec{\nabla}\lambda(\vec{r})\frac{1}{\Sigma_{t,g}(\vec{r})}W^-_{g,n}(\vec{r}) + \hat{\Omega}_n \cdot \vec{\nabla}\lambda(\vec{r})\frac{1}{\Sigma_{t,g}(\vec{r})}S^-_{g,n}(\vec{r})\right]dV \\ - \oint d\Gamma\left|\hat{\Omega}_n \cdot \hat{n}\right|\lambda(\vec{r})\psi^+_{g,n}(\vec{r}) - 2\oint d\Gamma\ \hat{\Omega}_n \cdot \hat{n}\ \lambda(\vec{r})\psi^{\hat{\Omega}_n\cdot\hat{n}<0}_{\Gamma,g,n}(\vec{r}).\end{aligned} \quad (2.38)$$

We can also put this in the following functional form and define the generic boundary condition contribution term.

$$
\begin{aligned}
\mathrm{T}\left[\psi_g^+(\vec{r}), S_g^{\pm}(\vec{r})\right] = & \int\left[\hat{\Omega}_n \cdot \vec{\nabla}\lambda(\vec{r})\frac{1}{\Sigma_{t,g}(\vec{r})}\hat{\Omega}_n \cdot \vec{\nabla}\psi_{g,n}^+(\vec{r}) + \lambda(\vec{r})\Sigma_{t,g}(\vec{r})\psi_{g,n}^+(\vec{r})\right]dV \\
& - \int\left[\lambda(\vec{r})W_{g,n}^+(\vec{r}) + \lambda(\vec{r})S_{g,n}^+(\vec{r})\right]dV \\
& - \int\left[\hat{\Omega}_n \cdot \vec{\nabla}\lambda(\vec{r})\frac{1}{\Sigma_{t,g}(\vec{r})}W_{g,n}^-(\vec{r}) + \hat{\Omega}_n \cdot \vec{\nabla}\lambda(\vec{r})\frac{1}{\Sigma_{t,g}(\vec{r})}S_{g,n}^-(\vec{r})\right]dV \\
& + BC_{g,n}
\end{aligned}
\tag{2.39}
$$

$$
\begin{aligned}
BC_{g,n} &= \oint \hat{\Omega}_n \cdot \hat{n}\ \lambda(\vec{r})\psi_{g,n}^-(\vec{r})d\Gamma \\
&= \oint d\Gamma \left|\hat{\Omega}_n \cdot \hat{n}\right| \lambda(\vec{r})\psi_{g,n}^+(\vec{r}) + 2\oint d\Gamma\ \hat{\Omega}_n \cdot \hat{n}\ \lambda(\vec{r})\psi_{\Gamma,g,n}^{\hat{\Omega}_n \cdot \hat{n}<0}(\vec{r})
\end{aligned}
\tag{2.40}
$$

2.7 Spatial Approximation

In the finite element method, the problem domain is subdivided using a finite element mesh as seen in Figure 2.3. Each finite element in the mesh is defined through the use of spatial vertex points (typically called nodes in the finite element literature) and the material cross sections are assumed to be spatially flat within each element.

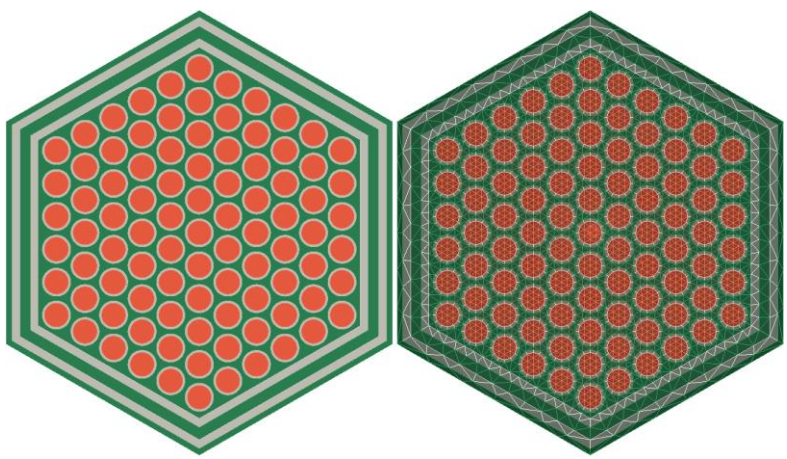


Figure 2.3 Imposition of a Finite Element Mesh for a Control Assembly

We define the total volume of the domain as the sum of all element volumes,

$$
V = \sum_e V_e \,. \tag{2.41}
$$

In two dimensions, the volume can refer to area (for example, the area of a triangular or quadrilateral element). On each finite element, there exists a number of polynomial trial functions equal to the number of nodes on the element (e.g. N=3 nodes for linear triangular elements). The N polynomial trial functions on element e hold the property that $L_e^i(\vec{r}_j) = \delta_{ij}$,

meaning that the ith trial function evaluates to unity at the ith node (and to zero at every other node) on the element.

Within each finite element, we approximate the spatial dependence of the flux, source, and weighting functions as a linear combination of the polynomial trial functions $L_e^T(\vec{r}) = \left[L_e^1(\vec{r}), \ldots L_e^N(\vec{r}) \right]$ that exist on the element. If the same sets of trial functions are used in adjoining elements, the flux approximation will be continuous across the interfaces. The following expressions show this spatial approximation within an element, i.e. for $\vec{r} \in V_e$:

$$\psi_g^{\pm}(\vec{r}, \hat{\Omega}_n) = L_e^T(\vec{r}) \tilde{\psi}_{g,e,n}^{\pm} = \left[L_e^1(\vec{r}), \ldots L_e^N(\vec{r}) \right] \begin{bmatrix} \tilde{\psi}_{g,e,n}^{\pm,1} \\ \vdots \\ \tilde{\psi}_{g,e,n}^{\pm,N} \end{bmatrix} = \sum_{i=1}^{N} L_e^i(\vec{r}) \tilde{\psi}_{g,e,n}^{\pm,i} \tag{2.42}$$

$$\varphi_g^{\pm}(\vec{r}, \hat{\Omega}) = Y_{\pm}^T(\hat{\Omega}) \otimes L_e^T(\vec{r}) \tilde{\varphi}_{g,e}^{\pm} \tag{2.43}$$

$$W_g^{\pm}(\vec{r}, \Omega) = Y_{\pm}^T(\hat{\Omega}) \otimes L_e^T(\vec{r}) \tilde{W}_{g,e}^{\pm} \tag{2.44}$$

$$S_g^{\pm}(\vec{r}, \Omega) = Y_{\pm}^T(\hat{\Omega}) \otimes L_e^T(\vec{r}) \tilde{S}_{g,e}^{\pm} \tag{2.45}$$

$$\lambda(\vec{r}) = L_e(\vec{r}) \tag{2.46}$$

The notation $\otimes$ represents a tensor product of the spatial and angular vectors of trial functions. (The tensor product of two vectors x and y is equal to $x \otimes y = xy^T$, i.e. a rank-1 matrix whose columns are multiples of the first vector.) Eqs. (2.43)-(2.45) involve the evaluation of spherical harmonics functions at a given angle. Eq. (2.46) states that the spatial trial functions in the weak-form of the even-parity transport equation will in fact be identical to the finite element trial functions.

We substitute Eqs. (2.43)-(2.46) into Eq. (2.39) and modify the functional to be defined over a single element volume instead of the entire volume. This substitution, as well as the assumption of constant cross sections within an element, permits all terms except the trial functions and their derivatives to be moved outside the spatial integrals. Only the following types of spatial integrals, called spatial matrices, remain after this substitution:

$$P_{e,K,L} = \int \vec{\nabla}_K L_e(\vec{r}) \vec{\nabla}_L L_e^T(\vec{r}) dV_e \tag{2.47}$$

$$U_{e,K} = \int L_e(\vec{r}) \vec{\nabla}_K L_e^T(\vec{r}) dV_e \tag{2.48}$$

$$F_e = \int L_e(\vec{r}) L_e^T(\vec{r}) dV_e \,. \tag{2.49}$$

These spatial matrices are dependent on known features of the element (node locations, shape, trial function definitions). The integrals can be evaluated using Gauss quadrature. Using the spatial matrix notation, the functional expression over a single element volume can be written:

$$\begin{aligned} T_e\left[\tilde{\psi}^{\pm}_{g,e,n}(\vec{r}), S^{\pm}_{g,e,n}(\vec{r})\right] = & \left[\sum_{K,L} \frac{\hat{\Omega}_{K,n}\hat{\Omega}_{L,n}}{\Sigma_{t,g,e}} P_{e,K,L} + \Sigma_{t,g,e} F_e\right] \tilde{\psi}^{\pm}_{g,e,n} - F_e \tilde{W}^{+}_{g,e,n} + F_e \tilde{S}^{+}_{g,e,n} \\ & - \left[\sum_{K} \frac{\hat{\Omega}_{K,n}}{\Sigma_{t,g,e}} U^{T}_{e,K}\right]\left(\tilde{W}^{-}_{g,e,n} + \tilde{S}^{-}_{g,e,n}\right) + BC_{g,e,n} \end{aligned} \tag{2.50}$$

To evaluate the original functional, T, we sum the element-wise trial functions into a global representation,

$$\mathrm{T}\left[\tilde{\psi}^{+}_{g,n}, \tilde{S}^{\pm}_{g,n}\right] = \sum_{e} \mathrm{T}_e\left[\tilde{\psi}^{+}_{g,e,n}, \tilde{S}^{\pm}_{g,e,n}\right]. \tag{2.51}$$

We can simplify Eq. (2.50) by defining the matrices and vectors

$$A_{g,e,n} = \sum_{K,L} \frac{\hat{\Omega}_{K,n}\hat{\Omega}_{L,n}}{\Sigma_{t,g,e}} P_{e,K,L} + \Sigma_{t,g,e} F_e \tag{2.52}$$

$$\tilde{s}^{+}_{g,e,n} = F_e \tilde{S}^{+}_{g,e,n} \tag{2.53}$$

$$\tilde{w}^{+}_{g,e,n} = F_e \tilde{W}^{+}_{g,e,n} \tag{2.54}$$

$$\tilde{s}^{-}_{g,e,n} = \sum_{K} \frac{\hat{\Omega}_{K,n}}{\Sigma_{t,g}} U^{T}_{e,K} \tilde{S}^{-}_{g,e,n} \tag{2.55}$$

$$\tilde{w}^{-}_{g,e,n} = \sum_{K} \frac{\hat{\Omega}_{K,n}}{\Sigma_{t,g}} U^{T}_{e,K} \tilde{W}^{-}_{g,e,n} \tag{2.56}$$

to obtain

$$\mathrm{T}_e\left[\tilde{\psi}^{+}_{g,e,n}, \tilde{s}^{\pm}_{g,e,n}\right] = A_{g,e,n}\tilde{\psi}^{+}_{g,e,n} - \tilde{s}^{+}_{g,e,n} - \tilde{s}^{-}_{g,e,n} - \tilde{w}^{+}_{g,e,n} - \tilde{w}^{-}_{g,e,n} + BC_{g,e,n} \tag{2.57}$$

The expression in Eq. (2.57) is valid for all discrete ordinates, Ω_n and elements e. This discrete functional can be assembled over the domain of interest and has only one solution for $\tilde{\psi}^{+}_{g,e,n}$ that minimizes the error in the functional. However, before solving the equation, we must define the element boundary conditions $BC_{g,e,n}$. In the following sections, we show that the boundary condition term cancels for interior elements due to the continuity conditions that are applied for the angular flux.

2.8 Finite Element Continuity Conditions

In order to completely characterize the boundary condition term defined in Eq. (2.40), we must explicitly define the continuity conditions on each element. The even-parity angular flux is taken to be continuous along element surfaces. We show that the boundary term in Eq. (2.57) will exist only for those elements that lie along the problem domain boundary.

Enforcement of the continuity condition on the angular flux between elements is guaranteed by Eq. (2.51). To show this we inspect two adjacent elements sharing a surface Γ_e shown in Figure 2.4.

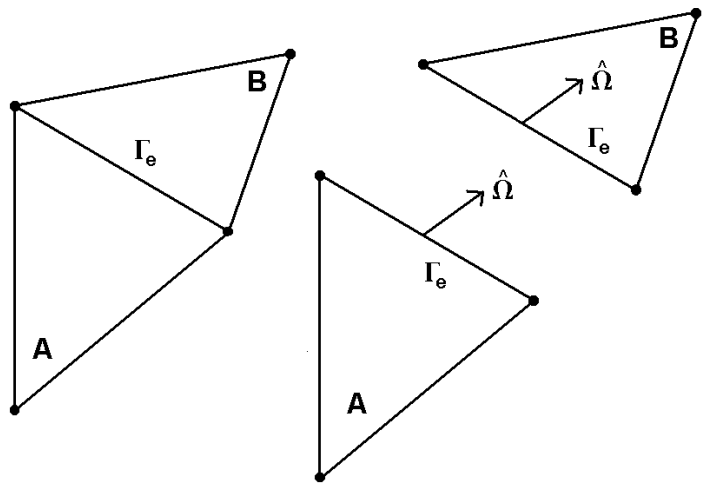


Figure 2.4 Adjacent Triangular Finite Elements

Picking an arbitrary point $\vec{r}$ and angular direction $\hat{\Omega}$ along the boundary Γ_e, as shown on the right in Figure 2.4, the following relations can be written for the even- and odd-parity flux and weight functions for the two elements.

$$\begin{aligned}
\psi_A^+\left(\vec{r}\in\Gamma_e,\hat{\Omega}\right) &= \psi_B^+\left(\vec{r}\in\Gamma_e,\hat{\Omega}\right)\\
\psi_A^-\left(\vec{r}\in\Gamma_e,\hat{\Omega}\right) &= \psi_B^-\left(\vec{r}\in\Gamma_e,\hat{\Omega}\right)\\
\lambda_A^+\left(\vec{r}\in\Gamma_e,\hat{\Omega}\right) &= \lambda_B^+\left(\vec{r}\in\Gamma_e,\hat{\Omega}\right)
\end{aligned} \tag{2.58}$$

We note that the first expression (continuity of the even-parity flux) is enforced in this methodology. The second expression (continuity of the odd-parity flux) is not enforced, but only assumed.

The boundary condition on surface Γ_e can be expressed for element A and element B as

$$\begin{aligned}
BC_{g,n,e}^{A} &= \oint \hat{\Omega}_n \cdot \hat{n}\ \lambda_A(\vec{r}\in\Gamma_e)\psi_{A,g,n}^{-}(\vec{r}\in\Gamma_e)d\Gamma_e,\\
BC_{g,n,e}^{B} &= \oint \hat{\Omega}_n \cdot -\hat{n}\ \lambda_B(\vec{r}\in\Gamma_e)\psi_{B,g,n}^{-}(\vec{r}\in\Gamma_e)d\Gamma_e.
\end{aligned} \tag{2.59}$$

The major difference between these two relations is the change in sign on the outward normal $\hat{n}$ for surface Γ_e because the outward normal for each element is the opposite of the other's. Writing out the sum of the two boundary conditions that appear for adjacent elements in Eq. (2.51), and then imposing the continuity conditions in Eq. (2.58) gives the following identity,

$$\begin{aligned}\oint \hat{\Omega}_n \cdot \hat{n}\, \lambda_A(\vec{r} \in \Gamma_e)\psi^-_{A,g,n}(\vec{r} \in \Gamma_e)d\Gamma_e + \oint \hat{\Omega}_n \cdot -\hat{n}\, \lambda_B(\vec{r} \in \Gamma_e)\psi^-_{B,g,n}(\vec{r} \in \Gamma_e)d\Gamma_e \\ = \oint \hat{\Omega}_n \cdot \hat{n}\left[\lambda_A(\vec{r} \in \Gamma_e)\psi^-_{A,g,n}(\vec{r} \in \Gamma_e) - \lambda_B(\vec{r} \in \Gamma_e)\psi^-_{B,g,n}(\vec{r} \in \Gamma_e)\right]d\Gamma_e \\ = \oint \hat{\Omega}_n \cdot \hat{n}\left[\lambda_A(\vec{r} \in \Gamma_e)\psi^-_{B,g,n}(\vec{r} \in \Gamma_e) - \lambda_B(\vec{r} \in \Gamma_e)\psi^-_{B,g,n}(\vec{r} \in \Gamma_e)\right]d\Gamma_e \\ = 0.\end{aligned} \tag{2.60}$$

As a consequence, the surface term in Eq. (2.57) only exists for those elements that lie along the outer problem boundary. With the boundary condition term removed for all those elements that do not lie upon the problem domain boundary, the boundary conditions that have been implemented on those elements along the outer problem boundary can now be investigated.

2.9 Vacuum Boundary Conditions

The vacuum boundary condition in Eq. (2.61) is a standard boundary condition imposed in neutron transport problems.

$$\psi_g\left(\vec{r},\hat{\Omega}\right) = 0, \quad \vec{r} \in \Gamma, \quad \hat{\Omega}\cdot\hat{n} < 0 \tag{2.61}$$

Since Eq. (2.61) defines the incoming angular flux to be zero, Eq. (2.40) reduces to

$$\begin{aligned}BC_{g,e,n} &= \oint d\Gamma \left|\hat{\Omega}_n \cdot \hat{n}\right| \lambda(\vec{r})\psi^+_{g,e,n}(\vec{r}) + 2\oint d\Gamma\, \hat{\Omega}_n \cdot \hat{n}\, \lambda(\vec{r}) \cdot [0] \\ &= \oint d\Gamma \left|\hat{\Omega}_n \cdot \hat{n}\right| \lambda(\vec{r})\psi^+_{g,e,n}(\vec{r}) \\ &= \left|\hat{\Omega}_n \cdot \hat{n}\right| W_\gamma\, \tilde{\psi}^+_{g,e,n},\end{aligned} \tag{2.62}$$

where W_γ is a new spatial matrix defined as a surface integral of the product of trial functions:

$$W_\gamma = \oint d\Gamma L_e(\vec{r} \in \gamma) L_e^T(\vec{r} \in \gamma)\,. \tag{2.63}$$

This allows the functional form to be rewritten for an element with a vacuum boundary on surface γ as:

$$\begin{aligned}\mathrm{T}_e\left[\psi^+_{g,e,n}, \tilde{s}^\pm_{g,e,n}\right] &= \left(A_{g,e,n} + \sum_{\gamma\in\Gamma_v}\left|\hat{\Omega}_n \cdot \hat{n}_\gamma\right| W_\gamma\right)\tilde{\psi}^+_{g,e,n} - \tilde{s}^+_{g,e,n} - \tilde{s}^-_{g,e,n} - \tilde{w}^+_{g,e,n} - \tilde{w}^-_{g,e,n} \\ &= A^v_{g,e,n}\tilde{\psi}^+_{g,e,n} - \tilde{s}^+_{g,e,n} - \tilde{s}^-_{g,e,n} - \tilde{w}^+_{g,e,n} - \tilde{w}^-_{g,e,n}\end{aligned} \tag{2.64}$$

Note that for curvilinear boundary surfaces which are possible with higher order finite elements (quadratic, cubic, super-parametric, etc.), the outward normal is a complicated function

of space and angle and thus the evaluation in Eq. (2.62) can be quite complicated. To avoid this complication in PROTEUS-SN, the outward normal is approximated by an average outward normal for each element surface on the problem domain boundary. This approximation should be sufficient for reactor geometries.

2.10 Reflected Boundary Conditions

The generally oriented reflected boundary condition is the only other boundary condition currently implemented in PROTEUS-SN. The reflected boundary condition, shown in Eq. (2.65), assigns the incoming angular flux on the surface $\vec{r} \in \Gamma$ to be equal to the outgoing angular flux at the corresponding "reflected" angle.

$$\begin{gathered}\psi_g(\vec{r},\hat{\Omega}') = \psi_g(\vec{r},\mu',\eta',\zeta') = \psi_g(\vec{r},\mu,\eta,\zeta) = \psi_g(\vec{r},\hat{\Omega}) \\ \vec{r} \in \Gamma,\ \hat{\Omega}' = \hat{\Omega}_{out,reflected},\ \hat{\Omega} = \hat{\Omega}_{out}\end{gathered} \tag{2.65}$$

The explicit dependence of the vector $\hat{\Omega}$ on the azimuthal and polar angles is explicitly written in Eq. (2.66) as a reminder.

$$\begin{aligned}\hat{\Omega} &= \mu\,\hat{i} + \eta\,\hat{j} + \zeta\,\hat{k} \\ &= \cos(\theta)\,\hat{i} + \cos(\phi)\sin(\theta)\,\hat{j} + \sin(\phi)\sin(\theta)\,\hat{k}\end{aligned} \tag{2.66}$$

In Eq. (2.65), the incoming angular flux, $\psi_g(\vec{r},\hat{\Omega}')$, is set equal to the outgoing angular flux $\psi_g(\vec{r},\hat{\Omega})$. These two angles are graphically represented in Figure 2.5 for the surface Γ, which in this example lies perpendicular to the x-axis and has an outward normal $\hat{n}$ extending towards the exterior of the domain. In this picture, the reflected boundary condition is given by $\psi_g(\vec{r},-\mu,\eta,\zeta) = \psi_g(\vec{r},\mu,\eta,\zeta)$, where the only difference is the sign of the first vector component μ because the surface lies perpendicular to the x-axis.

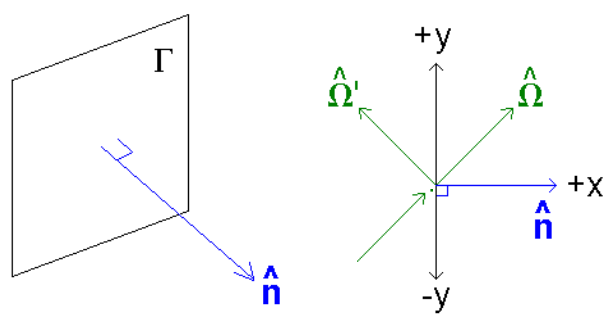
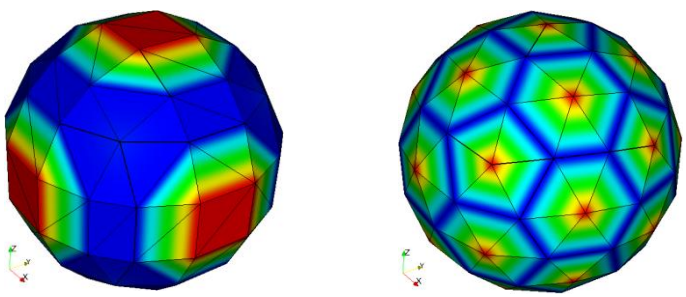


Figure 2.5. Simplified 1-D Diagram of the Reflected Boundary Condition

The conditions required to meet the reflected boundary condition are problematic because of the nature of the discrete ordinates approximation. For every outgoing discrete angle $\hat{\Omega}_n$ in the

cubature set, there may not exist another discrete angle in the cubature set which is the exact reflection of $\hat{\Omega}_n$ off a generally oriented surface. In the literature, a common approach is to utilize a cubature on the sphere which obeys all of the symmetry assumed in the geometry. In such an approach, a "one-to-one" mapping between the ordinates exists for reflected boundary conditions and thus all that remains is to define a set of dependent directions (those which are not solved for) and a set of independent directions (those which are solved for) at a given point on the surface of the domain.

However, to handle a generally orientated boundary condition, as desired in PROTEUS-SN, we must approximate Eq. (2.65) when the cubature fails to satisfy all of the problem symmetry. The first task is to define an interpolation scheme on the surface of the sphere such that the reflection of any angle in the cubature off a generally oriented surface can be defined in terms of the other angles in the cubature. PROTEUS-SN maps the cubature on the sphere with a set of triangular finite elements (spherical triangles) as shown in Figure 2.6.



Carlson Even Moment S_8 Tegmark Y_l^m fitted S_9

Figure 2.6. Example Triangular Meshes of Spherical Cubature

A given outward normal defines a plane that slices the unit sphere into incoming, outgoing, and parallel directions. Using the interpolation scheme outlined in Appendix C, we can produce a set of first-order equations of the form shown in Eq. (2.67), where each incoming direction is dependent upon at most 3 outgoing directions. These three directions are the nodes of the triangular surface which the exactly reflected angle passes through. For cubatures that exactly meet the symmetry conditions of the domain, the outward normal will cut the sphere such that a "one-to-one" mapping between the directions is created for a given reflected boundary condition.

$$\begin{aligned} \psi_{in} &= \sum_{j=1,3} \mathcal{P}_j \, \psi_{out,j} = \mathcal{P} \, \psi_{out} \\ \psi_{in} &\in \left\{ \psi_n, \;\; \hat{\Omega}_n \cdot \hat{n} < 0 \right\} \\ \psi_{out} &\in \left\{ \psi_n, \;\; \hat{\Omega}_n \cdot \hat{n} \geq 0 \right\} \end{aligned} \tag{2.67}$$

Because we are working with continuous finite element trial functions, we must impose these constraints at the vertices of the finite element mesh. Under these circumstances, a vertex could

lie at the apex of more than one reflected boundary surface with different orientations, thus we must combine the sets of first-order equations defined in Eq. (2.67) to properly constrain the system. A standard least squares approach was implemented to resolve the final set of first-order equations. The first step in this procedure is to identify the set of incoming and outgoing directions which are incident with regard to all of the reflected surfaces associated with a given spatial vertex indicated by summation over γ in Eq. (2.68).

$$\breve{\psi}_{in} \in \left\{ \psi_n, \bigcup_{\gamma} \Omega_n \cdot \hat{n}_\gamma < 0 \right\} \qquad \breve{\psi}_{out} \in \left\{ \psi_n, \bigcup_{\gamma} \Omega_n \cdot \hat{n}_\gamma \geq 0 \right\} \tag{2.68}$$

Using this set of directions, we recast the individual first-order systems of equations into the non-square system

$$\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_\gamma \end{bmatrix} \breve{\psi}_{in} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_\gamma \end{bmatrix} \breve{\psi}_{out} \tag{2.69}$$

$$\alpha \breve{\psi}_{in} = \beta \breve{\psi}_{out},$$

where α_γ and β_γ are obtained by rearranging the individual first-order systems of equations, $\mathcal{P}_\gamma$, such that they correspond to Eq. (2.68). This system of equations enforces the reflective boundary condition on all incoming angles at a given vertex. There are often more equations than degrees of freedom, so a least squares method must be used to solve this system. The least squares procedure is then implemented as shown in Eq. (2.70).

$$\begin{aligned} \alpha \breve{\psi}_{in} &= \beta \breve{\psi}_{out} \\ \alpha^T \alpha \breve{\psi}_{in} &= \alpha^T \beta \breve{\psi}_{out} \\ \breve{\psi}_{in} &= \left(\alpha^T \alpha \right)^{-1} \alpha^T \beta \breve{\psi}_{out} \\ &= \breve{\mathcal{P}} \, \breve{\psi}_{out} \end{aligned} \tag{2.70}$$

Note that for a single reflected boundary condition on a vertex, this least squares procedure simply reverts to the original first-order system defined in Eq. (2.67). Additionally, for cases where the cubature meets all of the symmetry of the problem, the resulting system of equations defined by Eq. (2.70) will be a Boolean matrix which properly defines the one-to-one mapping between the incoming and outgoing direction set. Finally, for high order cubatures that do not meet the symmetry of the problem but which have been implemented for a vertex associated with multiple reflected surfaces, the above least squares projection has been observed to produce very small positive and negative numbers. Such dependencies illustrate a very weak coupling between these directions through the reflection operator, and PROTEUS-SN truncates such dependencies and rebalances the equation.

Since we are solving the even-parity form of the transport equation rather than the first-order form, we still have one more transformation to apply. We must recast the system of equations to enforce boundary conditions on the even-parity fluxes (rather than the first order fluxes) as shown in Eq. (2.71), where the even parity fluxes are solved only for angles in a specific part of the sphere dependent on problem dimension. Recall that the even-parity fluxes for angles in the other part of the sphere can be obtained through angular parity relationships.

$$\begin{aligned} &\breve{\psi}_{in} = \breve{\mathscr{P}}\ \breve{\psi}_{out} \Rightarrow B\psi^{+} = C\psi^{+} \\ &\psi^{+} \in \left\{ \psi_n,\ \Omega_n \in \frac{\pi}{2}(1\mathrm{D}),\ \pi(2\mathrm{D})\ \text{or}\ 2\pi(3\mathrm{D}) \right\} \end{aligned} \tag{2.71}$$

Matrix B is a Boolean matrix corresponding to the vector reassignment from the set of incoming directions to the corresponding set of even-parity directions. Matrix C is the reordered $\breve{\mathscr{P}}$ matrix such that it corresponds to the set of even-parity directions. For one-to-one mappings, the B and C matrices are both Boolean matrices, and several of the equations are duplicates. To develop a system of valid constraints, we sweep through the system of equations defined by Eq. (2.71) and remove duplicate ones such that we end up with the maximal set of unique equations which constrain the maximal set of angular directions producing a new equation of the form given as Eq. (2.72). Note that this does not involve the inversion of B, but rather a simple rearrangement of B and C such that the Π matrix is defined and redundant equations are eliminated.

$$\psi^{+} = \Pi\psi^{+} \tag{2.72}$$

This Π matrix can typically be rearranged using a Gauss elimination algorithm with row pivoting.

$$\begin{bmatrix} \psi_D^{+} \\ \psi_I^{+} \end{bmatrix} = \begin{bmatrix} \Pi_D & \Pi_I \\ 0 & I \end{bmatrix} \begin{bmatrix} \psi_D^{+} \\ \psi_I^{+} \end{bmatrix} \tag{2.73}$$

This final system defines a subset of the even-parity directions, ψ_I^{+}, which are independent, and a subset which are dependent, ψ_D^{+}. The submatrix Π_D is strictly upper triangular and the structure of Π_I is irrelevant. This final system of equations states mathematically that for a given set of independent even-parity directions (those that we must solve for), the solution for the dependent set of directions is known from the reflection operator. This operation connects the dependent angular directions to the independent angular directions for all those vertices on the boundary of the domain.

For elements with vacuum boundary conditions, we can assemble the functional in Eq. (2.57) over all directions as shown in Eq. (2.74), where the A matrix for vacuum boundary conditions was defined earlier.

$$\mathrm{T}_e\left[\psi_{g,e}^{+},\tilde{s}_{g,e}^{\pm}\right]=\begin{bmatrix} A_{g,e,1}^{v} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & A_{g,e,N}^{v} \end{bmatrix}\begin{bmatrix} \tilde{\psi}_{g,e,1}^{+} \\ \vdots \\ \tilde{\psi}_{g,e,N}^{+} \end{bmatrix}-\begin{bmatrix} \tilde{s}_{g,e,1}^{+} \\ \vdots \\ \tilde{s}_{g,e,N}^{+} \end{bmatrix}-\begin{bmatrix} \tilde{s}_{g,e,1}^{-} \\ \vdots \\ \tilde{s}_{g,e,N}^{-} \end{bmatrix}-\begin{bmatrix} \tilde{w}_{g,e,1}^{+} \\ \vdots \\ \tilde{w}_{g,e,N}^{+} \end{bmatrix}-\begin{bmatrix} \tilde{w}_{g,e,1}^{-} \\ \vdots \\ \tilde{w}_{g,e,N}^{-} \end{bmatrix} \tag{2.74}$$
$$=A_{g,e}^{v}\psi_{g,e}^{+}-\tilde{s}_{g,e}^{+}-\tilde{s}_{g,e}^{-}-\tilde{w}_{g,e,n}^{+}-\tilde{w}_{g,e,n}^{-}$$

The application of equation Eq. (2.73) for a given vertex on the surface eliminates the block diagonal nature of the system in Eq. (2.74), but only for those vertices on a reflected surface. The application of Eq. (2.73) is straightforward and leads to the new system given by Eq. (2.75).

$$\mathrm{T}_e\left[\psi_{g,e}^{+},\tilde{s}_{g,e}^{\pm}\right]=A_{g,e}^{r}\psi_{g,e}^{+}-\tilde{s}_{g,e}^{+}-\tilde{s}_{g,e}^{-}-\tilde{w}_{g,e}^{+}-\tilde{w}_{g,e}^{-} \tag{2.75}$$

The details of the $A_{g,e}^{r}$ matrix are not shown for brevity (the "r" superscript stands for "reflected"). In forming this matrix, the dependent space-angle degrees of freedom are not removed from Eq. (2.74), but instead the row and column of the dependent space-angle degrees of freedom are zeroed out and the scaling factor of $\frac{1}{\Sigma_t}+\Sigma_t$ is placed on the diagonal to maintain good matrix properties for the preconditioner. This trick allows the vector space to be kept constant for all elements, regardless of the presence of boundary conditions.

Note that the sources for the dependent directions have to be redirected as sources for the independent directions using Eq. (2.73), and, after Eq. (2.75) is solved, the dependent discrete ordinate flux moments must be rebuilt using Eq. (2.73). These operations are not only computationally cheap given the sparse nature of Eq. (2.73), but they are scalable so long as the ratio of work to communication is known for the problem of interest ahead of time. Eq. (2.75) can be assembled over all elements as defined by Eq. (2.51) to give the final within group equation

$$\mathrm{T}\left[\psi_{g}^{+},\tilde{s}_{g}^{\pm}\right]=A_{g}^{r}\psi_{g}^{+}-\tilde{s}_{g}^{+}-\tilde{s}_{g}^{-}-\tilde{w}_{g}^{+}-\tilde{w}_{g}^{-} \tag{2.76}$$

2.11 Summary

In this chapter, we have derived the second-order even-parity transport equation which governs the even-parity flux. We have applied the multigroup approximation to discretize the energy domain into the typical "within-group" equations, which are coupled to each other the scattering and fission source terms. Most energy groups receive contributions from higher energy groups (via down-scattering). Additionally, a few energy groups may also receive contributions from lower energy groups (if up-scattering is present). Finally, the fission source causes neutrons to be born in high-energy groups due to fission reactions occurring in other energy groups.

We have also applied the discrete ordinates approximation to the multigroup equations in order to discretize the angular dependence. Functionals were derived to express the multigroup flux along a particular discrete ordinate. These functionals are connected to each other by the scattering source, which causes neutrons to change direction. It is very important to note that in

the presence of up-scattering, the even-parity operator is iterative as the odd-parity source must be iteratively based upon the even-parity flux. In a sense, the odd-parity flux is not really the odd-parity flux as the even-parity methodology assumes the odd-parity flux is completely eliminated via substitution. This makes the application of GMRES and other solvers on the complete multi-group system somewhat unwise and impractical for the even-parity methodology.

Finally, the continuous Galerkin finite element method is applied to discretize the spatial domain. The degrees of freedom therefore become the values of the even-parity angular flux at each "node" or "vertex", for a given energy group and discrete ordinate. The boundary conditions for each element vary depending on whether the element is interior (no extra boundary condition term), or has a void or reflective surface.

In the following chapter, we discuss the numerical implementation of these equations and the algorithms used to solve them.

3 Numerical Implementation

In this chapter we discuss the flow of a PROTEUS calculation, starting with the problem set-up and continuing with the solution of the within-group equations and global system. PROTEUS-SN can be used in two different modes: steady state and kinetics (adiabatic time dependence). We primarily discuss the steady state mode here and later make mention of the time-dependent mode.

3.1 Overview

In PROTEUS-SN, the multigroup second order even-parity transport equation is discretized using the angular discrete ordinates approximation and the continuous finite element approximation in space. Boundary conditions are not applied to the odd-parity flux in the even-parity method as it assumed to be entirely removed via substitution. Available finite element types include bar, triangle, quadrilateral, tetrahedron, triangular prism, and hexahedron. Lagrangian, serendipity and Gauss-Lobatto polynomial trial functions are available up to ninth, second, and sixth orders, respectively.

As derived in the previous chapter, the discrete ordinate and finite element approximations reduce the second order even-parity transport equation to a set of within-group linear algebra equations for the even-parity multigroup flux,

$$A_g \psi_g^+ = B_g \psi_g^+ + S_g^+ + C_g (W_g^- + S_g^-), \tag{3.1}$$

where B_g represents the within-group scattering matrix. The odd-parity sources are iteratively calculated. The coefficient matrix A_g is positive definite which enables Eq. (3.1) to be solved using the conjugate gradient (CG) method [13] known to be scalable on large parallel systems.

The solution methodology in PROTEUS-SN consists of the following iterations listed in order from outer to inner, where the PETSc package is used to perform the innermost iteration:

- Fission source (inverse power) iteration for the eigenvalue or fixed source
- Gauss-Seidel iteration over energy groups
- Scattering iterations for within-group scattering system
- Conjugate gradient over whole space-angle system
- PETSc preconditioned conjugate gradient for each discrete ordinate

We describe these iterations in further detail in this chapter.

3.2 Parallelism

The computational method used in PROTEUS-SN is highly parallelizable in both space and angle, due to the applicability of the conjugate gradient method and the block diagonal connection in angle of the matrix A_g. This method permits the efficient utilization of hundreds of thousands of processors which in turn allows the efficient solution of high-fidelity problems with billions of degrees of freedom. While PROTEUS-SN was built to enable parallelization in the

energy domain, parallelization in energy is not currently permitted, nor needed, as the current space-angle decomposition scheme already saturates the capabilities of leadership computing machines.

The parallel decomposition of the problem in both space and angle is accomplished by assigning sections of the phase space to different processors which each calculate the solution on their "owned" portion of the domain. Therefore, if a problem runs on 4 processors, each processor might own ¼ of the spatial domain and all of the angles. Alternatively, each processor might own ½ of the spatial domain and ½ of the angles. The number of domains to decompose into is specified at runtime by the user.

PROTEUS-SN uses the Message Passing Interface (MPI) standard to communicate information from one processor to another, since the physical boundaries between locally owned domains are shared. Also, scattering causes a connection in angle (and between different processors). The code runs on machines ranging from desktop to leadership computing machines. It has been previously demonstrated [14] to scale efficiently to 300,000 processors (IBM's BlueGene/P architecture) and 222,000 processors (Cray XT5) which enables the solution of very large problems up to 10^{12} degrees of freedom on leadership computing resources. The scaling results shown in Table 3.1 and Table 3.2 were performed for different space-angle discretizations of the ZPR6/7 fast reactor experiment and indicate the expected scaling efficiencies for typical simulations with PROTEUS-SN.

Table 3.1. Strong Spatial Scaling of PROTEUS-SN on BlueGene/P Architecture

Total Cores	Vertices Per Process	Total Time (s)	Strong Scaling
8,192	7,324	2,402	100%
16,384	3,662	1,312	92%
24,576	2,441	873	92%
32,768	1,831	637	94%

Table 3.2. Weak Angle Scaling of PROTEUS-SN on BlueGene/P Architecture

Total Cores	Angular Directions	Total Time (s)	Weak Scaling
32,768	32	579	100%
73,728	72	572	101%
131,072	128	581	100%
163,840	160	691	84%
294,912	288	763	76%

Experience has shown that the number of vertices per process (i.e. size of a local domain on a processor) should be kept greater than 1,500 in order to maintain a reasonable work to communication ratio. Each processor should also be assigned work for at least two angles in the cubature to balance the PETSc work. A combined >90% strong space-angle scaling efficiency is expected in this regime.

3.3 Problem Initialization

Before the multigroup system of equations are solved, PROTEUS-SN performs some initializations and calculations, including reading the input data, translating material information, decomposing the mesh, and building the finite element matrices.

3.3.1 Read Input

At the beginning of a PROTEUS-SN calculation, the root processor reads in the driver input file and initializes the PETSc library using the requested angular resolution, parallelization in angle, scattering order, and other iterative values. The driver input information is communicated to all other processors. Next, the root processor looks at the multigroup cross section library and broadcasts the control information (number of groups and anisotropic scattering information) to all other processors. Alternatively, PROTEUS-SN can generate multigroup cross section data from a generalized library using the subgroup application programming interface (API). The implementation of the subgroup API is a new feature is being verified. For more information on the subgroup API, contact nera-software@anl.gov.

3.3.2 Mesh Decomposition

In the current implementation, the root processor reads in the user-provided mesh and sends each piece to the rest of the parallel system. A partitioning mesh file can optionally be provided in order to specify the desired spatial decomposition. If the partitioning file is not included, PROTEUS-SN partitions the mesh on-the-fly using the METIS software and the number of requested spatial partitions. At the end of this process, each processor owns a piece of the mesh, called the local mesh. It sees its local mesh and a one-ring ghost set of elements, combined these are called the visible piece of the mesh. A basic bandwidth optimization can be performed on the local piece of the mesh to improve preconditioning if requested in the driver input.

3.3.3 Computation of the Finite Element Matrices

After the control data, cross section and mesh data are read in, the mesh is processed and auxiliary boundary conditions provided in the control input are applied. This is necessary to identify the necessary communication to carry out with respect to reflected boundary conditions. During this process the finite element spatial matrices are generated on each processor for the local visible mesh using Gaussian numerical integration.

After processing the finite element mesh, PROTEUS determines the material compositions on the local mesh and eliminates recursive definitions in the material assignment file in favor of direct definitions using the isotopes included in the cross section file.

The PETSc library set-up is performed by defining the vector scatter/gather operation for a single space-angle segment of the flux vector. The spatial matrices are stenciled and the preconditioner is set. Finally, the multigroup solve routine is called to solve either a forward problem, adjoint problem, or both.

3.4 Multi-Group Solve Procedure

The solution of the global system of equations has several nested loops. The outermost loop is the fission source iteration, in which iterates of the fission source and eigenvalue are calculated based on the current estimate of the global multi-group flux for all groups. Tchebychev acceleration is optionally applied to accelerate fission source convergence.

To obtain the multigroup fluxes needed for the fission source iteration, each within-group equation is solved in order, starting with the highest energy group, in Gauss-Seidel fashion. For each within-group system, the group sources (fission, fixed source, in-scattering and within-group scattering) are calculated based on current estimates of the multigroup flux values, and the within-group solver is then called to solve the matrix equation in Eq. (3.1).

The within-group solver contains a two-step iterative procedure since the left hand side and right hand side both depend on the within-group flux. The first step involves performing a scattering (Richardson) iteration for the discrete ordinates even parity flux. The second step follows with a synthetic diffusion acceleration calculation to correct the P0 moment of angular flux. The scattering source is updated using the diffusion correction, and the procedure repeats. The synthetic diffusion scheme can be considered a multigrid preconditioner in angle.

After the within-group flux is obtained, the fission and scattering group sources are updated and the next energy group is solved. This group-by-group solution proceeds until all of the down-scattering-only groups have been solved. The within-group equations for up-scattering groups are then solved, but they require an extra outer iteration loop called the “up-scattering iteration” since the flux in a group with both down-scattering and up-scattering is dependent on both the groups above and below it.

We now discuss the details of how to solve the within-group equations (innermost iteration).

3.4.1 Within-Group Parallel Preconditioned Conjugate Gradient Method for each Discrete Ordinate

The within-group flux equation for the entire space-angle system has the form

$$\begin{aligned} A_g^r \left(\psi_g^+\right)^{i+1/2} &= \tilde{s}_g^+ + \tilde{s}_g^- + \left(\tilde{w}_g^+\right)^i + \left(\tilde{w}_g^-\right)^i \\ &= \tilde{s}_g^+ + \tilde{s}_g^- + L_g \left(\psi_g^+\right)^i \\ &= b \end{aligned} \tag{3.2}$$

where the A matrix has convenient properties which enable the use of highly parallel iterative schemes. First, the A matrix is real, symmetric ($A^T = A$), and positive definite ($x^T Ax > 0$ for any vector x), and therefore the conjugate gradient method can be applied to solve the system. CG is an iterative method which is excellent for large, sparse systems such as those that result from finite element discretizations. Second, the A matrix is block-diagonal in angle for vacuum boundary conditions and contains only minor coupling between the angles for reflected boundaries. The method is relatively scalable in angle so long as the cost associated with communicating the reflected independent moment information is accounted for in the load balancing. The preconditioner of the CG solve of Eq. (3.2), is forced to be block diagonal by eliminating the connections in angle and thus can be solved independently with respect to angle (discrete ordinate). Each angle sub-system in the preconditioner is symmetric positive definite and thus again we can use CG to solve it. For problems with vacuum boundary conditions only, the upper level CG solve can be omitted.

For the preconditioner, PROTEUS-SN links to and uses the parallel SSOR-preconditioned conjugate gradient solver in the PETSc package [3] to solve the preconditioner system. The PETSc package was chosen due to its development history and optimization on supercomputing platforms. The preconditioned conjugate gradient method can be summarized as follows, where the matrix C is the preconditioner:

Let $x_o \in R^n$ be an initial vector.

Let $r_o = b - Ax_o$ be the initial residual.

Let $d_o = z_o = C^{-1} r_o$ be the initial direction of correction.

do $k = 0,1,\ldots$

$\alpha_k = \dfrac{z_k^T r_k}{d_k^T A d_k}$ (compute magnitude of correction)

$x_{k+1} = x_0 + \alpha_k d_k$ (apply correction)

$r_{k+1} = r_k - \alpha_k A d_k$ (recompute residual)

$z_{k+1} = C^{-1} r_{k+1}$

$\beta_{k+1} = \dfrac{z_{k+1}^T r_{k+1}}{z_k^T r_k}$

$d_{k+1} = z_{k+1} + \beta_{k+1} d_k$ (recompute new direction)

end do

The SSOR (Symmetric Successive Over Relaxation) preconditioner defines the matrix C in terms of the symmetric matrix $A = L + D + L^T$,

$$C = \left(\frac{1}{\omega}D + L\right)\frac{\omega}{2-\omega}D^{-1}\left(\frac{1}{\omega}D + L^T\right),\ \omega \in [0,2] \tag{3.3}$$

One scattering iteration therefore consists of multiple iterations within the conjugate gradient solver. The two largest computational components of the CG method are the application of the coefficient matrix A and the application/inversion of the preconditioner matrix C. In most cases, the computational effort for the preconditioner is more costly than all of the other solver operations. PROTEUS-SN reports the number of iterations required to invert the preconditioner in the output file.

3.4.2 Synthetic Diffusion Acceleration of the Within Group Scattering Source

The within-group scattering source for the within group equation is directly dependent upon the flux solution for that group, and therefore scattering source iterations are required to properly compute the flux. Eq. (3.4) presents the standard scattering source iteration of the within group equation, where *i* gives the index number of the iteration.

$$A_g^r\left(\psi_g^+\right)^{i+1} = \tilde{s}_g^+ + \tilde{s}_g^- + \left(\tilde{w}_g^+\right)^i + \left(\tilde{w}_g^-\right)^i \tag{3.4}$$

We modify this source iteration slightly by expanding the definition of the within-group sources in terms of the spherical harmonic projection of the flux to develop the following relationships.

$$\begin{aligned}
\tilde{w}_{g,e,n}^+ &= Y_+^T(\hat{\Omega}_n)\Sigma_{s,g\to g}^+ Y_+(\hat{\Omega}_n) w_n \otimes F_e \tilde{\psi}_{g,e,n}^+ \\
&= Y_+^T(\hat{\Omega}_n)\Sigma_{s,g\to g}^+ \otimes F_e \tilde{\varphi}_{g,e}^+ \\
&= L_{+,n}\tilde{\varphi}_{g,e}^+ \\
&\Rightarrow \tilde{w}_g^+ = L_+\tilde{\varphi}_g^+
\end{aligned} \tag{3.5}$$

$$\begin{aligned}
\tilde{w}_{g,e,n}^- &= \sum_K Y_-^T(\hat{\Omega}_n)\frac{\hat{\Omega}_{K,n}\Sigma_{s,g\to g}^-}{\Sigma_{t,g}} \otimes U_{e,K}^T\,\tilde{\varphi}_{g,e}^- \\
&= L_{-,n}\tilde{\varphi}_{g,e}^- \\
&\Rightarrow \tilde{w}_g^- = L_-\tilde{\varphi}_g^-
\end{aligned} \tag{3.6}$$

We insert Eqs. (3.5) and (3.6) into Eq. (3.4) and introduce the superscript $i + \frac{1}{2}$ on the flux vector to indicate the intermediate estimate of the flux solution obtained by source iteration.

$$A_g^r\left(\psi_g^+\right)^{i+1/2} = \tilde{s}_g^+ + \tilde{s}_g^- + L_+\left(\tilde{\varphi}_g^+\right)^i + L_-\left(\tilde{\varphi}_g^-\right)^i \tag{3.7}$$

We can rewrite Eq. (3.7) as the more general equation Eq. (3.8), where it is understood that the definition of the L matrix is rather complicated.

$$A_g^r \left(\psi_g^+\right)^{i+1/2} = \tilde{s}_g^+ + \tilde{s}_g^- + L_g \left(\psi_g^+\right)^i \tag{3.8}$$

The exact solution, ψ_g^+, therefore satisfies Eq. (3.9):

$$\begin{aligned} \left(A_g^r - L_g\right)\psi_g^+ &= \tilde{s}_g^+ + \tilde{s}_g^- \\ D_g\ \psi_g^+ &= \tilde{s}_g^+ + \tilde{s}_g^- \end{aligned} \tag{3.9}$$

The source iteration shown in Eq. (3.7) can be slow to converge. Therefore, as is conventional in discrete ordinates methods, we apply a diffusion synthetic acceleration (DSA) scheme in order to accelerate the convergence of the flux vector. The general steps in the DSA acceleration scheme are (1) solve the transport equation to obtain an intermediate estimate of the flux vector, (2) solve a simplified diffusion-like system (the synthetic equation) to estimate the correction needed to the flux vector, and (3) update the intermediate estimate of the flux vector with the correction. The DSA iteration steps are then repeated until convergence. This process reduces the number of high-order transport calculations that are needed.

To define the synthetic equation, we first define the error in the flux solution at $i+\frac{1}{2}$ due to the scattering iteration defined by Eq. (3.7) as

$$\left(\varepsilon\right)^{i+1/2} = \psi_g^+ - \left(\psi_g^+\right)^{i+1/2} \tag{3.10}$$

We then develop a system of equations to express the error at iteration $i+\frac{1}{2}$ as a function of the previous two flux solutions:

$$\begin{aligned} D_g\ \left(\varepsilon\right)^{i+1/2} &\equiv D_g \left(\psi_g^+ - \left(\psi_g^+\right)^{i+1/2}\right) \\ &= \tilde{s}_g^+ + \tilde{s}_g^- - D_g \left(\psi_g^+\right)^{i+1/2} \\ &= \tilde{s}_g^+ + \tilde{s}_g^- - A_g^r \left(\psi_g^+\right)^{i+1/2} + L_g \left(\psi_g^+\right)^{i+1/2} \\ &= \tilde{s}_g^+ + \tilde{s}_g^- - \left(\tilde{s}_g^+ + \tilde{s}_g^- + L_g\ \right) + L_g \left(\psi_g^+\right)^{i+1/2} \\ &= L_g \left(\left(\psi_g^+\right)^{i+1/2} - \left(\psi_g^+\right)^i\right) \end{aligned} \tag{3.11}$$

If the error term $\left(\varepsilon\right)^{i+1/2}$ in Eq. (3.11) could be solved exactly, the discrete ordinates flux could be updated immediately and the exact solution would be obtained by Eq. (3.10). However, solving the discrete ordinates system in Eq. (3.11) for the exact discrete ordinates error $\left(\varepsilon\right)^{i+1/2}$ is expensive. Instead, we project the system of equations to the smaller spherical harmonic P_1 space

in order to evaluate the error in that space. To accomplish this we utilize the relationship between the discrete ordinate solution vector $\psi_{g,n}^{+}$ and the spherical harmonic moment coefficients $\tilde{\varphi}_g^{+}$,

$$\tilde{\varphi}_g^{+} = \sum_n Y_+(\hat{\Omega}_n) w_n \psi_{g,n}^{+}. \tag{3.12}$$

We note that the discrete ordinate angular flux can be recovered by using the formula $\psi_g^{+}(\hat{\Omega}_n) = Y_+^T(\hat{\Omega}_n)\tilde{\varphi}_g^{+}$ where the spherical harmonic functions are simply evaluated the prescribed angle. We therefore obtain the following system of equations for the spherical harmonic projection error $\left(\varepsilon_{+,Y}\right)^{i+1/2}$, which can be applied to the intermediate spherical harmonics moments estimate as a correction term.

$$\begin{gathered}\sum_n Y_+^T(\hat{\Omega}_n) w_n D_{g,n} Y_+(\hat{\Omega}_n)\left(\varepsilon_{+,Y}\right)^{i+1/2} = \sum_n Y_+^T(\hat{\Omega}_n) w_n L_g Y_+(\hat{\Omega}_n)\left(\left(\tilde{\varphi}_g^{+}\right)^{i+1/2} - \left(\tilde{\varphi}_g^{+}\right)^{i}\right) \\ D_{g,Y}\left(\varepsilon_{+,Y}\right)^{i+1/2} = L_{g,Y}\left(\left(\tilde{\varphi}_g^{+}\right)^{i+1/2} - \left(\tilde{\varphi}_g^{+}\right)^{i}\right)\end{gathered} \tag{3.13}$$

PROTEUS-SN accelerates (apply corrections to) the P_0 even-parity moment, which corresponds to writing Eq. (3.13) only for the zeroth moment:

$$D_{g,0}\left(\varepsilon_{+,0}\right)^{i+1/2} = L_{g,0}\left(\left(\tilde{\varphi}_{g,0}^{+}\right)^{i+1/2} - \left(\tilde{\varphi}_{g,0}^{+}\right)^{i}\right) \tag{3.14}$$

Solving Eq. (3.14) for $\left(\varepsilon_{+,0}\right)^{i+1/2}$ therefore yields the correction to the P_0 even-parity moment, i.e. $\left(\tilde{\varphi}_{g,0}^{+}\right)^{i+1} = \left(\tilde{\varphi}_{g,0}^{+}\right)^{i+1/2} + \left(\varepsilon\right)^{i+1/2}$. Once this correction is applied, the next source iteration takes place using Eq. (3.7).

We note that Eq. (3.13) and consequently Eq. (3.14) are exact for elements with reflected boundary conditions provided a one-to-one angular mapping exists between the incident and reflected angles. However, for elements with vacuum boundary conditions, an additional term must be added which we do not describe here.

For each scattering iteration, a transport solve is done for all angles. The diffusion acceleration equation is not partitioned over the entire set of processors (ideal), but only over a subset equivalent to the selected spatial decomposition. The solution of the diffusion equation is also done using the PETSc SSOR preconditioned CG algorithm. After it is solved to a given tolerance, the updated P0 solution is transmitted to all of the angle specific processors for use in completing the acceleration process. In most problems, the code only needs to do a single DSA iteration to successfully accelerate the convergence as the diffusion equation is an exact projection of the even-parity discrete ordinates transport equation.

3.4.3 Connection of Multigroup System of Equations

The within-group systems are connected to each other by the scattering source which can include both down-scattering and up-scattering. The global multigroup system is solved by Gauss-Seidel iteration which converges in a single iteration when no up-scattering is present. The Gauss-Seidel iterative matrix inversion scheme can be summarized as

$$\begin{aligned} (D+L+U)x &= b \\ \left(D+L\right)x &= b-Ux \\ \rightarrow \quad x^{k+1} &= \left(D+L\right)^{-1}\left(b-Ux^{k}\right) \end{aligned} \tag{3.15}$$

In the above scheme, the coefficient matrix A is split into the diagonal (*D*), strictly upper triangular (*U*), and strictly lower (*L*) triangular matrices. For the case of no up-scattering, the matrix is lower triangular.

3.4.4 Fission Source Iteration

The outermost iteration in PROTEUS-SN iterates on the entire multigroup fission source and eigenvalue using the power method, with optional Tchebychev acceleration. While there are better acceleration methods available, the PROTEUS-SN code was only meant to be an initial test to see if the continuous finite element method would perform well on petascale computing machines. Possible improvements may be made to the acceleration scheme in the future.

3.4.4.1 Power Iteration

The initial fission source is calculated based on an initial flux vector guess of unity (flat uniform flux). To briefly summarize the power method, we write the global system of equations as $T\psi^{+} = \frac{1}{k}F\psi^{+}$, where the term $F\psi^{+}$ is the fission source, . Applying the matrix product FT^{-1} to both sides of the equation yields the eigenvalue problem $F\psi^{+} = \frac{1}{k}FT^{-1}F\psi^{+}$, or $k\theta = FT^{-1}\theta$, where $\theta = F\psi^{+}$ is the fission source. The fission source (eigenvector) and k-effective value (eigenvalue) can be solved using the power method:

$$\begin{aligned} \theta^{(i+1)} &= \frac{1}{k^{(i)}}FT^{-1}\theta^{(i)} \\ k^{(i+1)} &= k^{(i)}\frac{\left\langle W,\theta^{(i+1)}\right\rangle}{\left\langle W,\theta^{(i)}\right\rangle} \end{aligned} \tag{3.16}$$

In Eq. (3.16), W is some positive weight function and the brackets indicate integration over the reactor volume. The easy choices are to use W=1 (i.e. L_1 norm) or the current fission source distribution (i.e. L_2 norm). While one can choose any arbitrary function, these two choices ensure that the resulting system converges to the dominant eigenmode(s) of the system.

3.4.4.2 Tchebychev Acceleration of the Fission Source

Various extrapolation methods can be used to improve the flux or fission source iterates. One such extrapolation method is Tchebychev acceleration. In traditional mode (no Tchebychev acceleration applied to the fission source), the fission source error, eigenvalue, and iterative flux error decrease for each iteration according to the spectral radius (or dominance ratio) of the eigenvalue problem. A plot of these errors with respect to fission source iteration will display straight curves on a log scale where the slope is the negative of the spectral radius.

Tchebychev acceleration speeds the convergence of the fission source by altering the spectral radius of the eigenvalue problem and hence decreasing the number of required outer iterations (usually by a factor of 2). Using Tchebychev acceleration, the accelerated fission source iterate, θ_{i+1}, is calculated based on the intermediate (non-accelerated) iterate $\tilde{\theta}_{i+1}$ calculated by $\tilde{\theta}^{(i+1)} = \frac{1}{k^{(i)}} FT^{-1}\theta^{(i)}$, as well as the previous estimates of the fission source, θ_i and θ_{i-1}.

$$\tilde{\theta}_{i+1} = \theta_i + \alpha_{i+1}(\tilde{\theta}_{i+1} - \theta_i) + \beta_i(\theta_i - \theta_{i-1}) \tag{3.17}$$

The values of α and β are iteration-dependent and determined by the order of the Tchebychev polynomial fitting that is being applied.

3.5 Methodology Limitations

While the continuous finite element with the second order even-parity methodology offers benefits in terms of computational scalability and efficient use of computing resources, it also has currently unavoidable caveats which we describe here.

PROTEUS-SN cannot model void or very optically thin regions due to the Σ_t^{-1} coefficient appearing in the second order even-parity equation. The workaround is to homogenize void or optically thin regions with surrounding non-void regions.

Additionally, the condition number of the within-group matrix increases as $O(1/h^2)$ with mesh size h decreases, and $O(p^2)$ with element order p increases. This causes the computational burden in PROTEUS-SN to increase at a fast rate for finely detailed problems compared to other methodologies. For most problems, we advise using a large number of linear elements rather than increasing the element order.

PROTEUS-SN also has convergence issues when neighboring materials in the problem have strongly contrasting properties (i.e. alternating optically thick and optically thin regions). There is no current workaround other than homogenization.

4 Time-Dependent Calculations

In addition to the steady state mode, PROTEUS-SN models time-dependence using the adiabatic approximation. This approximation is more accurate than point kinetics but less accurate and simpler to implement than the quasi-static or full kinetics methods.

4.1 Neutron Kinetics Theory

A fission event in a nuclear reactor produces both fission products and prompt neutrons, which are emitted into the system within 10^{-14} seconds of the fission event. Some fission products are unstable (these are called precursors) and undergo a beta decay process which instigates the emission of delayed neutrons. The production of delayed neutrons plays a crucial part in the control of nuclear reactors, and this process must be accounted for in the time dependent transport equation. There are numerous precursor isotopes and complete physics data is not well-known for each one. Therefore, in computational methods, the precursors are typically categorized into 6 "delay groups" based on their half-lives. Each delay group is assigned a characteristic decay constant, λ_m, and a characteristic energy dependent delay yield, ν_m. For example, a single fissionable isotope (i) may produce different precursors ($C_{m,i}$) belonging to any (or all) of the six delay groups. The characteristic delay yield for group *m* is computed as a sum over all contributing fissionable isotopes to group *m*, $\nu_m = \sum_i \beta_{m,i}\nu_{m,i}$.

Eq. (4.1) is the time dependent neutron transport equation with delayed neutrons, which includes a source term corresponding to the production of delayed neutrons by precursor decay. The change in precursor concentration over time is given by Eq. (4.2), which contains the production of delayed neutron precursors via fission events and the loss of delayed neutron precursors via radioactive decay.

$$\begin{aligned}\left[\frac{1}{v}\frac{\partial}{\partial t}+\hat{\Omega}\cdot\nabla+\Sigma_t(\vec{r},E,t)\right]\psi(\vec{r},E,\hat{\Omega},t) &= \int dE'\int d\Omega'\sigma_s(\vec{r},E'\to E,\hat{\Omega}'\cdot\hat{\Omega},t)\psi(\vec{r},E',\hat{\Omega}',t)\\ &+\sum_i \chi_{p,i}(E)\int dE'\nu_{p,i}\Sigma_{f,i}(\vec{r},E',t)\phi(\vec{r},E',t)\\ &+\sum_i\sum_{m=1}^{6}\lambda_m C_{m,i}(\vec{r},E,t)\end{aligned} \tag{4.1}$$

$$\frac{\partial}{\partial t}C_{m,i}(\vec{r},E,t)=\chi_{m,i}(E)\int dE'\nu_{m,i}\Sigma_{f,i}(\vec{r},E',t)\phi(\vec{r},E',t)-\lambda_m C_{m,i}(\vec{r},E,t) \tag{4.2}$$

Various definitions are stated below.

$$\chi_{p,i}(E) = \text{Prompt fission spectrum due to fission of isotope i} \tag{4.3}$$

$$\chi_{m,i}(E) = \text{Delayed neutron spectrum emitted by precursor m of isotope i} \tag{4.4}$$

$$\nu_{p,i} = (1-\beta_i)\nu_i = \text{Prompt neutron yield due to fission of isotope i} \tag{4.5}$$

$$\nu_{m,i} = \beta_{m,i}\nu_i = \text{Delayed neutron yield emitted by precursor m of isotope i} \tag{4.6}$$

$$C_{m,i}(\vec{r},E,t) = \text{Concentration of precursor group m due to fission of isotope i that always emit a delayed neutron at energy E} \tag{4.7}$$

It is important to note that in this formulation, the change in physical position of the precursors is not considered and thus these equations cannot be used to treat "moving fuel" reactor types.

4.2 Point Kinetics Approximation

The point kinetics approximation assumes that the spatial shape of the flux and precursor concentrations do not change with time and can be described with a solution to the steady state problem, $\psi_0(\vec{r},E,\hat{\Omega})$. This approximation allows the flux and precursor densities to be written as functions separable in space and time:

$$\psi(\vec{r},E,\hat{\Omega},t) = n(t)\psi_0(\vec{r},E,\hat{\Omega}) \tag{4.8}$$

$$C_m(\vec{r},E,t) = \sum_i C_{m,i}(\vec{r},E,t) = c_m(t)\int d\Omega\,\psi_0(\vec{r},E,\hat{\Omega}), \quad m = 1,\ldots 6 \tag{4.9}$$

where $n(t)$ and $c_m(t)$ describe the magnitude of the shape functions throughout time. Substitution of the point kinetics approximation into the kinetics equations along with algebraic manipulations leads to the point kinetics equations, where both spatial and isotopic dependence are eliminated:

$$\begin{aligned} \frac{\partial}{\partial t}n(t) &= \frac{\rho(t)-\beta}{\Lambda}n(t) + \sum_{m=1}^{M}\lambda_m c_m(t) \\ \frac{\partial}{\partial t}c_m(t) &= \frac{\beta_m}{\Lambda}n(t) - \lambda_m c_m(t) \end{aligned} \tag{4.10}$$

The reactivity quantity $\rho(t) = \frac{1}{k_0} - \frac{1}{k(t)}$ is the reactivity change from the base state at time t. The value $\beta = \sum_m \beta_m$ is the total fraction of fission neutrons which are delayed, and Λ is the prompt neutron lifetime, i.e. the average time between fission neutron birth to death (absorption or leakage).

In point kinetics, the time-dependent reactivity change $\rho(t)$ is calculated as a function of pre-calculated reactivity coefficients, meaning that the actual transport equation is not solved for the true reactivity at later time steps. Additionally, the kinetics parameters Λ and β are calculated only based on the base configuration and not updated as a function of time. These assumptions limit the accuracy of the point kinetics method to perturbations in which the reactivity insertions are fairly small (<100 pcm) and do not change the kinetics parameters

significantly. It is very important to understand that point kinetics can still be very accurate for large changes in reactivity so long as β and Λ do not change significantly during the transient.

4.3 Adiabatic Approximation Implemented in PROTEUS-SN

The adiabatic approximation implemented in PROTEUS-SN is more accurate and versatile than the point kinetics equations, but less accurate than the quasi-static approximation on which it is based.

In the quasi-static class of kinetics methods [11,15], the time-dependence of the flux is assumed to be dominated by a function independent of space, but smaller fluctuations in the shape over time are also permitted. Therefore, the flux and precursor densities are expanded in a factorizing approach as

$$\psi(\vec{r},E,\hat{\Omega},t) = n(t)\varphi(\vec{r},E,\hat{\Omega},t)\text{, and} \tag{4.11}$$

$$C_m(\vec{r},E,t) = \sum_i C_{m,i}(\vec{r},E,t) = c_m(t)\int d\Omega\varphi(\vec{r},E,\hat{\Omega},t)\,. \tag{4.12}$$

The time-dependence of the shape function $\varphi(\vec{r},E,\hat{\Omega},t)$ is assumed to be weak and bounded in time by some constant, a condition which typically introduces a weighting function like the adjoint flux. Insertion of these approximations into the full kinetics equations and performing algebraic manipulations results in two sets of coupled equations. The first set of equations looks like the point kinetics equations, where the reactivity and kinetics parameters are defined in terms of the shape function $\varphi(\vec{r},E,\hat{\Omega},t)$ and another weighting function, typically either the initial steady state or time-dependent adjoint flux. The second set of equations are a transport-like system for the shape function including the delayed neutron source and other terms unfamiliar to the steady state transport equation. Solving these equations by replacing the time derivative of the shape function with a backward differencing scheme is known as the "improved quasi-static approximation".

To obtain the adiabatic approximation, a series of approximations are applied to the equation for the shape function.

1) The time-derivative of the shape function is negligible, i.e. $\frac{\partial}{\partial t}\varphi(\vec{r},E,\hat{\Omega},t) \approx 0$ (quasi-static approximation).
2) The shape of the delayed neutron source is not distinguished from the shape of the prompt source, i.e. the time delay in the shape of the precursor distribution is neglected. This results in only one fission source which is identical to the steady state fission source (adiabatic approximation).
3) The time derivative of the magnitude function is neglected, i.e. $\frac{\partial}{\partial t}n(t) \approx 0$ (adiabatic approximation).

These collective approximations reduce the shape equation to the familiar steady state transport equation. Recall that in addition to the steady state transport equation, two other equations are derived which differ from point kinetics because the kinetics parameters, $\Lambda(t)$ and $\beta(t)$ are functions of time:

$$\begin{aligned}\frac{\partial}{\partial t}n(t) &= \frac{\rho(t)-\beta(t)}{\Lambda(t)}n(t)+\sum_{m=1}^{M}\lambda_m c_m(t)\\ \frac{\partial}{\partial t}c_m(t) &= \frac{\beta_m(t)}{\Lambda(t)}n(t)-\lambda_m c_m(t)\end{aligned} \tag{4.13}$$

In the adiabatic approximation, the time-dependent kinetics parameters are defined in terms of inner products (integral over space, angle, energy) on the adjoint and forward fluxes $\varphi^*(\vec{r},E,\hat{\Omega},t)$ and $\varphi(\vec{r},E,\hat{\Omega},t)$, which are solved at each time step using the steady state adjoint and forward transport equations:

$$\begin{aligned}\Lambda(t) &= \frac{\left\langle \varphi(t)^* \tfrac{1}{v}\varphi(t)\right\rangle}{\left\langle \varphi(t)^* \underline{\underline{F}}(t)\varphi(t)\right\rangle} \approx \frac{\left\langle \varphi_0^* \tfrac{1}{v}\varphi(t)\right\rangle}{\left\langle \varphi_0^* \underline{\underline{F}}(t)\varphi(t)\right\rangle} \approx \frac{\left\langle \varphi_0^* \tfrac{1}{v}\varphi_0\right\rangle}{\left\langle \varphi_0^* \underline{\underline{F}}_{0,m,i}\varphi_0\right\rangle}\\ \beta_{m,i}(t) &= \frac{\left\langle \varphi(t)^* \underline{\underline{F}}_{m,i}(t)\varphi(t)\right\rangle}{\left\langle \varphi(t)^* \underline{\underline{F}}(t)\varphi(t)\right\rangle} \approx \frac{\left\langle \varphi_0^* \underline{\underline{F}}_{m,i}(t)\varphi(t)\right\rangle}{\left\langle \varphi_0^* \underline{\underline{F}}(t)\varphi(t)\right\rangle} \approx \frac{\left\langle \varphi_0^* \underline{\underline{F}}_{0,m,i}\varphi_0\right\rangle}{\left\langle \varphi_0^* \underline{\underline{F}}_{0}\varphi_0\right\rangle}\\ \rho(t) &= \frac{1}{k_o}-\frac{1}{k(t)}\end{aligned} \tag{4.14}$$

$$\begin{aligned}\underline{\underline{A}}(t)\varphi(t) &= \frac{1}{k(t)}\underline{\underline{F}}(t)\varphi(t)\\ \underline{\underline{A}}^*(t)\varphi^*(t) &= \frac{1}{k(t)}\underline{\underline{F}}^*(t)\varphi^*(t)\end{aligned} \tag{4.15}$$

The term $\underline{\underline{F}}_{m,i}$ denotes the delayed fission source due to precursors in group m from isotope i, $\underline{\underline{F}}_{m,i}(\vec{r},t)=\int dE\nu_{m,i}\Sigma_{f,i}(\vec{r},E,t)\int d\Omega\varphi(\vec{r},E,\hat{\Omega},t)$. The term $\underline{\underline{F}}(t)$ is the total fission source from both prompt and delayed neutrons. All of the basic approximations typically used to update β and Λ are shown in Eq. (4.14), where the first expression denotes recomputing the adjoint flux and Λ and β at every timestep. We note that computing the adjoint flux at each time step is also optional (middle column). Similarly, the re-computation of β and Λ can be made optional (last column).

4.4 Implementation

At the beginning of the adiabatic kinetics calculation, the initial condition steady state forward and adjoint calculations are performed to obtain the initial condition forward and adjoint solutions, as well as initial power level ($\approx n_o$). These values are used to obtain the initial kinetics

parameters $\beta_{m,0}$ and Λ_o and k_o from Eq. (4.14). The initial precursor concentrations are obtained by

$$c_{m,0} = \frac{\beta_{m,0}}{\Lambda_o \cdot \lambda_m} n_o \tag{4.16}$$

For each subsequent time step, steady state forward and adjoint calculations are performed as in Eq. (4.15). The solution of these steady state equations follows the same implementation as described in previous chapters of this manual. The kinetics parameters $\Lambda(t)$ and $\beta(t)$ as well as the reactivity are updated for the time step according to Eq. (4.14).

Given the updated kinetics parameters for the time step, the point kinetics equations in Eq. (4.13) are solved over the time step using one of two techniques

One option in PROTEUS-SN is to use a fully implicit differencing scheme over the time step t_j to t_{j+1}, where $\Lambda(t)$ and $\beta(t)$ are assumed to be constant over a time steps. This approach is least accurate and efficient but is provided for verification purposes and simplicity. Applying this scheme to Eq. (4.13) yields the discretized equations

$$\frac{n_{j+1} - n_j}{t_{j+1} - t_j} = \frac{\rho_{j+1} - \beta}{\Lambda} n_{j+1} + \sum_{m=1}^{M} \lambda_m c_{m,j+1} \tag{4.17}$$

$$\frac{c_{m,j+1} - c_{m,j}}{t_{j+1} - t_j} = \frac{\beta_m}{\Lambda} n_{j+1} - \lambda_m c_{m,j+1} \tag{4.18}$$

The reactivity ρ_{j+1} is assumed to change linearly over the time step and can be written in terms of the eigenvalue as $\rho_{j+1} = \frac{1}{k_j} - \frac{1}{k_{j+1}}$. Rearranging Eqs. (4.17) and (4.18) to combine unknowns yields the linear system

$$\left\{ \frac{1}{t_{j+1} - t_j} - \frac{\rho_{j+1} - \beta}{\Lambda} \right\} n_{j+1} + \sum_{m=1}^{M} \left\{ -\lambda_m \right\} c_{m,j+1} = \left\{ \frac{1}{t_{j+1} - t_j} \right\} n_j \tag{4.19}$$

$$\left\{ \frac{1}{t_{j+1} - t_j} + \lambda_m \right\} c_{m,j+1} + \left\{ \frac{-\beta_m}{\Lambda} \right\} n_{j+1} = \left\{ \frac{1}{t_{j+1} - t_j} \right\} c_{m,j} \tag{4.20}$$

$$m = 1, \ldots 6$$

These equations are implemented into a linear algebra solver and propagated for each time step in PROTEUS-SN. For more details on the derivation of the adiabatic methodology in PROTEUS-SN, we refer the reader to Reference 11.

The alternative time dependent approach to solving this system is to use the Radau [16,17] solver. The Radau solver is based on implicit Runge-Kutta methods of orders 5, 9 and 13 which

are L-stable. In the approach, the Jacobian of the matrix system is evaluated at different points over the time step which is conveniently provided by the preceding implicit differencing scheme. The Radau solver is time step adaptive and auto corrects to the desired convergence accuracy. While there are other approaches to solving this system which may yield more robust performance, we feel that these two approaches are sufficient for the stated needs.

5 Cross Sections

Multigroup cross sections for PROTEUS-SN can be generated externally using the MC^2-3 fast reactor multigroup cross section generation code (or any codes which can generate multigroup cross sections in the ISOTXS format) or internally using the cross section application programming interface (API) which can handle the subgroup or resonance table libraries. This section briefly describes the generation of multigroup cross sections for use in PROTEUS-SN.

5.1 Conventional Multigroup Cross Sections

PROTEUS-SN requires multigroup cross sections for each isotope of all the compositions in a problem. Those multigroup cross sections can be provided from external codes in the ISOTXS format. The MC^2-3 code has been normally used to generate multigroup cross sections for fast reactor analysis. In MC^2-3, resonance self-shielding of isotopes in homogeneous (0D) or 1D geometries is performed based on the numerical integration of the point-wise cross sections using the narrow resonance approximation. The heterogeneity effect for the 1D geometry is accounted for in the resonance self-shielding by introducing the escape cross sections estimated from the 1D collision probability calculation. The ultrafine group (UFG) transport calculation is conducted and then the resulting flux and moment solutions are utilized to condense UFG cross sections to broad group (BG) cross sections. If necessary, RZ-geometry core calculations may be carried out to obtain more realistic region-wise UFG spectra for group condensation. Typically, 33 to 230 group cross sections with the historically optimized group structures are used for fast reactor analyses.

Assembly-homogenized cross sections are typically used and verified through numerous benchmark problems. The assembly-homogenized cross sections are generated using the 0D or 1D calculations discussed above. However, MC^2-3 is also capable of providing heterogeneous cross sections which are to be used for heterogeneous-geometry core calculations with PROTEUS-SN. The procedure using the heterogeneous cross sections is under verification.

5.2 Cross Section API

The cross section API is an external library developed by Argonne National Laboratory which processes multigroup cross sections using the subgroup method or the generalized cross section (resonance table) method. The cross section API linked to PROTEUS-SN produces multigroup cross sections online for cross section regions in a problem. To do this, PROTEUS-SN should provide a one-group fixed source transport solver to the cross section API. The multigroup cross sections calculated from the cross section API are stored in the memory. This capability is still being under verification.

5.2.1 Subgroup Cross Section Method

The subgroup method [18] determines the effective resonance cross sections without the intermediate calculation of Dancoff factors, and thus it is useful for arbitrary geometry or direct

whole core transport calculations in which spatially dependent self-shielding should be properly considered. The flux solution is represented as below in terms of the background cross section σ_b and the microscopic absorption cross section σ_a^r of resonance *r*.

$$\phi(u) = \frac{\sigma_b}{\sigma_a^r(u) + \sigma_b}, \tag{5.1}$$

where $\sigma_b = \lambda\sigma_p + \sigma_e$, and λ, σ_p and σ_e are the intermediate resonance parameter, the potential cross section, and the escape cross section, respectively.

The resonance integral is approximated by quadratures and thus the effective absorption cross section $\bar{\sigma}_a^r$ of resonance r is determined in terms of the subgroup weight w_n and level σ_n and flux solution ϕ_n for the broad group as

$$\bar{\sigma}_a^r = \frac{\sum_n w_n \phi_n \sigma_n}{\sum_n w_n \phi_n}. \tag{5.2}$$

The subgroup parameters w_n for individual isotopes are prepared as a function of background cross section and temperature. Therefore, the resonance interference effect due to the presence of other resonant isotopes in a mixture is accounted for using the Bondarenko iteration in which other isotope cross sections are treated as a constant over the energy group. In the conventional subgroup method, the parameters are determined using the background cross sections estimated from the solution of a fixed source problem (FSP). From there, the Bondarenko iteration is performed only with the subgroup parameters already determined from the FSP,

$$\Sigma_a^k(T) = \frac{\sum_n w_{an}^k(T)\Sigma_{an}^k \dfrac{\Sigma_{bn}^k}{\Sigma_{an}^k + \hat{\Sigma}_a^k(T) + \Sigma_{bn}^k}}{1 - \sum_n w_{an}^k(T) \dfrac{\Sigma_{an}^k}{\Sigma_{an}^k + \hat{\Sigma}_a^k(T) + \Sigma_{bn}^k}}. \tag{5.3}$$

where $\hat{\Sigma}_a$ = absorption cross section of other resonant isotopes, w_{an}^k, Σ_{an}^k, Σ_{bn}^k = weighting factor, absorption cross section, and background cross section, respectively, of subgroup level n of isotope k.

The subgroup method requires considerable effort to prepare the subgroup parameters using the least square method. It is also known that the subgroup method is appropriate for systems with compositions in which there are only a few dominant resonant isotopes and there is minimal or no resonance overlapping between the dominant resonant isotopes. Accordingly, it is noted that the subgroup method has worked well for thermal systems where U-238 is a dominant isotope, and the accurate treatment of the resonance cross sections of structural material is not that important. The subgroup method may not work well for fast systems where there are multiple dominant resonant isotopes and accurate estimation of resonance cross sections of structural material is important.

5.2.2 Generalized Cross Section Method

The conventional subgroup method has been widely and successfully used for light water reactor analysis. However, it requires a significant effort to generate the subgroup parameters with good performance, and thus those subgroup parameters and cross sections need to be re-evaluated or re-generated for reactor cores with different spectrum conditions. Therefore, the generalized cross section method [2] has been developed for its application to various reactor types including LWR, HTR, and SFR.

Unlike the aforementioned subgroup method, the flux representation includes the contribution of the scattering resonance cross section σ_s^r for resonance r.

$$\phi(u) = \frac{\sigma_b}{\sigma_a^r(u) + \lambda\sigma_s^r(u) + \sigma_b}, \tag{5.4}$$

where the background cross section is defined as

$$\sigma_b^r = \frac{1}{N^r}\left(\sum\nolimits_{k\neq r}[\Sigma_a^k(u) + \lambda_k\Sigma_s^k(u)] + \Sigma_e\right). \tag{5.5}$$

Similarly to the subgroup method, the escape cross sections Σ_e accounting for the heterogeneity effect are calculated by solving the one-group fixed source problem. Because of the consideration of the scattering resonance cross section shown in Eq. (5.4), we use the collision probability formation to calculate the background cross section as

$$\sigma_{0,i}^g = \frac{\sum_j \sum_{k\neq r} P_{ji}^g N_{k,j}\lambda\sigma_{p,k}V_j}{\sum_j P_{ji}^g N_{r,j}V_j}. \tag{5.6}$$

where P_{ji}^g is the collision probably from region j to i. The escape cross section representing the heterogeneity effect is defined as

$$\Sigma_{e,i}^g = N_{r,i}\sigma_{0,i}^g - \sum\nolimits_{k\neq r}\lambda\Sigma_{p,k,i}^g. \tag{5.7}$$

Since calculating the collision probabilities in Eq. (5.6) is very expensive, the following two fixed source problems for the resonant isotope r are solved

$$\Omega\cdot\nabla\psi_1^g(\mathbf{r},\Omega) + [\Sigma_a^g(\mathbf{r}) + \lambda\Sigma_s^g(\mathbf{r})]\psi_1^g(r,\Omega) = \sum\nolimits_{k\neq r}\lambda\Sigma_{p,k}^g(\mathbf{r}), \tag{5.8}$$

$$\Omega\cdot\nabla\psi_2^g(\mathbf{r},\Omega) + [\Sigma_a^g(\mathbf{r}) + \lambda\Sigma_s^g(\mathbf{r})]\psi_2^g(\mathbf{r},\Omega) = N_r(\mathbf{r}),$$

Using the solutions of the two equations above, the background cross section in Eq. (5.6) can be calculated as

$$\sigma_{0,i}^g = \frac{\int_{V_i} dV \int_{4\pi} d\Omega\,\psi_1^g(r,\Omega)}{\int_{V_i} dV \int_{4\pi} d\Omega\,\psi_2^g(r,\Omega)} = \frac{\phi_1^g}{\phi_2^g}. \tag{5.9}$$

Since the total cross section including resonance cross sections should be determined by the background cross section, the escape cross section in Eq. (5.7) can be determined by performing

$2 \cdot N$ whole-core transport calculations, Eq. (5.8), where N is the number of resonant isotopes. One to two more iterations may be normally sufficient to get the total cross sections converged.

For its general application, the base cross section library is constructed with ultrafine groups (UFGs). The base UFG cross section library is generated using MC^2-3 and NJOY. The library includes the resonance cross section tables for absorption, nu-fission, and scattering cross sections as a function of the background cross section and temperature for each isotope. The number of the background cross sections for use in the resonance table should be different depending upon isotopic characteristics, but maximum 20 data points are used in this study. The UFG structure is composed of total 2158 groups from 20 MeV to 10^{-4} eV. Relatively large lethargies in the thermal energy range are assigned because there is no significant cross section variation as in the resonance energy range. We use MC^2-3 to obtain all cross section data above 0.414 eV and NJOY to collect thermal cross section below 0.414 eV and update scattering data below 3 eV. The resonance cross section tables are calculated using the hyperfine group (~600,000) slowing-down calculation option of MC^2-3.

Preliminary verification tests indicate that the base UFG cross section library is able to accurately estimate eigenvalue and cross sections for various compositions with different characteristics. Since, however, an UFG is too many for the practical use, it is necessary to be reduced to a practical number of groups. Thus, we use a group condensation optimization algorithm to condense an UFG library to a broad group (BG) library with minimizing the accuracy loss. Once a reactor of interest is selected, the UFG transport calculation is first performed with a representative homogeneous composition to determine the UFG neutron spectrum. Next, multiple homogeneous and heterogeneous compositions available for the specific core(s) are prepared to determine the BG boundaries which can best approximate the solutions with the UFG cross section library in terms of partial reaction rates (absorption and nu-fission) and eigenvalue.

In the group condensation optimization process [2], the group condensation progresses from the highest group until the differences of the following quantities reach a user criterion: $\varepsilon_{keff} = \Delta k_{eff} / k_{eff}^{ref}$, $\varepsilon_a = \Delta\Sigma_{aG}^i \phi_G / \Sigma_{aG}^i \phi_G^{ref}$, and $\varepsilon_{nf} = \Delta \nu\Sigma_{fG}^i \phi_G / \nu\Sigma_{fG}^i \phi_G^{ref}$, where G is a broad group containing UFGs and i is an isotope. The evaluation of k_{eff} during condensation is important since it allows us to check the contribution of the scattering cross section between UFG and BG structures. Once the difference is greater than a user-specified criterion, the group condensation process moves on to the next ultrafine group and starts evaluating the next broad group. Note that the ultrafine group structure is used for the previous broad groups such that the contribution of group condensation to eigenvalue and reaction rates is limited only to the broad group of evaluation.

The UFG-based intermediate resonance (IR) parameters for an isotope mixed with U-238 and hydrogen are calculated by comparing the accurate resonance cross section derived from the hyperfine-group reference calculation and that estimated from the pre-calculated resonance table

with the background cross section. In fact, the IR parameters should be updated depending on the broad group structure.

The followings summarize the keys in the cross section library generation procedure:

- The master cross section library is based on the 2158 ultrafine group structure, which includes the resonance cross section tables for absorption, nu-fission, and scattering reactions. The resonance cross section tables are produced by MC^2-3 with the hyperfine group slowing-down calculation capability.
- The broad group cross section library for a reactor or reactor type of interest is determined by the group condensation optimization algorithm minimizing the change of eigenvalue and reaction rates from the ultrafine group solutions. In this process, various compositions of a reactor or reactor type of interest are used. With the resulting broad group structure and the reactor-representative neutron spectrum, the base ultrafine group cross section library is condensed to a broad group library.
- For the local heterogeneity effect, the escape cross sections are determined iteratively as described in the previous section - Eqs. (5.7), (5.8), and (5.9).
- This method still relies on the Bondarenko iteration to account for the resonance interference effect. It is noticed that the errors in the detailed broad group cross sections are often reduced with a smaller number of energy groups due to error cancellation.

6 Using PROTEUS-SN

For a detailed description of input, output, and usage, we refer the reader to the companion user guide to PROTEUS-SN [19]. However, for convenience, we provide a brief summary of these items here.

6.1 Input

PROTEUS-SN requires 4 input files: multi-group cross section data file, finite element mesh, material assignment (mapping compositions to mesh) file, and a driver input file. The first two files must appear is specific file formats described in the user manual. The MC^2-3 fast reactor cross section generation code is typically used to produce the cross section file. The finite element mesh can be generated using a number of tools, but must be converted by the user to the PROTEUS-compatible format. The recommended mesh-generator options at this time are MeshKit (.h5m) and Cubit (.e), as these mesh files can easily be read in or converted to PROTEUS-compatible formats using the MOAB library. The material assignment file and driver input file are simple readable text files created by the user.

6.2 Output

6.2.1 Steady State Calculations

PROTEUS-SN produces two main types of output files for forward calculations. A simple text file echoes all the input options, parallel decomposition information, the outer iteration convergence history, final eigenvalue, and simple parallel timing data. Additionally, the full flux solution can be exported. The "UNIC" plugin in VisIt can be used to directly import the full flux solution for visualization and analysis. In addition to containing the full flux solution, the output file also includes mesh, composition, power, and certain reaction rates (namely absorption). The output is described in more detail in the companion manual (user guide). For adjoint calculations, only the simple text file is produced.

6.2.2 Time-Dependent Calculations

PROTEUS-SN produces output files for time-dependent calculations at each time step. For example, the eigenvalue history is output along with updated power level, and updated Λ and β (upon request). The full solution is exported to file upon request at each time step.

REFERENCES

1. C. H. Lee and W. S. Yang, "Development of Multi-group Cross Section Generation Code MC^2-3 for Fast Reactor Analysis," *Proc. of Int. Conf. on Fast Reactors and Related Fuel Cycles (FR09)*, Kyoto, Japan, Dec. 7-11, 2009.
2. C. H. Lee, A. Marin-Lafleche, and M. A. Smith, "Development of Cross Section Library and Application Programming Interface (API)," ANL/NE-13/15, Argonne National Laboratory, 2013.
3. S. Balay, et al, "PETSc Webpage," http://www.mcs.anl.gov/petsc/, 2013.
4. G. Karypis and V. Kumar, "A Fast and Highly Quality Multilevel Scheme for Partitioning Irregular Graphs", SIAM Journal on Scientific Computing, **20**, *1*, 359-392, 1999.
5. The HDF5 Group. Hierarchical Data Format, version 5, 1997-2014. http://www.hdfgroup.org/HDF5/.
6. MPICH Website, http://www.mpich.org, 2014.
7. T. J. Tautges, R. Meyers, K. Merkley, C. Stimpson, and C. Ernst, "MOAB: A Mesh-Oriented Database," SAND2004-1592, Sandia National Laboratories, April 2004.
8. W. S. Yang, M. A. Smith, C. H. Lee, A. Wollaber, D. Kaushik, and A. S. Mohamed, "Neutronics Modeling and Simulation of SHARP for Fast Reactor Analysis," *Nuclear Engineering and Technology*, **42**, *5*, 520-545, 2010.
9. M. A. Smith, D. Kaushik, A. Wollaber, W. S. Yang, B. Smith, C. Rabiti, and G. Palmiotti, "Recent Research Progress on UNIC at Argonne National Laboratory," *Proc. of Int. Conf. on Mathematics, Computational Methods & Reactor Physics*, Saratoga Springs, New York, May 3-7, 2009.
10. A. Marin-Lafleche, M. A. Smith, and C. H. Lee, "PROTEUS-MOC: A 3D Deterministic Solver Incorporating 2D Method of Characteristics," *Proc. Of International Conf. on Mathematics and Computational Methods Applied to Nuclear Science and Engineering (M&C 2013)*, Sun Valley, Idaho, May 5-9, 2013
11. M. A. Smith, V. S. Mahadevan, and E. R. Wolters, "Continued Research for Improvement of PROTEUS-SN", ANL/NE-13/12, Argonne National Laboratory, September 30, 2013.
12. E. E. Lewis and W. F. Miller. Computational Methods of Neutron Transport. Wiley, 1984.
13. Y. SADD, *Iterative Methods for Sparse Linear Systems*, Second Edition, Society of Industrial and Applied Mathematics, 2003.
14. D. Kaushik, M. A. Smith, A. Wollaber, B. Smith, A. Siegel, and W. S. Yang, "Enabling High-Fidelity Neutron Transport Simulations on Petascale Architectures", SC2009 Conference, Portland, Oregon, Nov 14-20, 2009.
15. K. Ott and D. Meneley, "Accuracy of the Quasistatic Treatment of Spatial Reactor Kinetics", Nuclear Science and Engineering, **36**, 402-411, 1969.
16. E. Hairer and G. Wanner. Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems. Springer Series in Mathematics 14, Spring-Verlag 1991, Second Edition, 1996.
17. E. Hairer and G. Wanner, "Stiff Differential Equations Solved by Radau Methods", Journal of Computational and Applied Mathematics, **111**, 93-111, 1999.

18. HELIOS Methods, Studsvik Scandpower, November, 2003.
19. E. R. Shemon, M. A. Smith, and C.H. Lee, "PROTEUS-SN User Manual", ANL/NE-14/6, June 30, 2014.

APPENDIX A. FINITE ELEMENT SPATIAL APPROXIMATION

This appendix is focused on the general procedures used in the finite element method used to evaluate the spatial integrals. It is common in finite element approximations to define a reference element to which all of the finite elements in the mesh are individually transformed. The reference elements used in this work were taken from reference 1, an example of which is shown in Figure A.1.

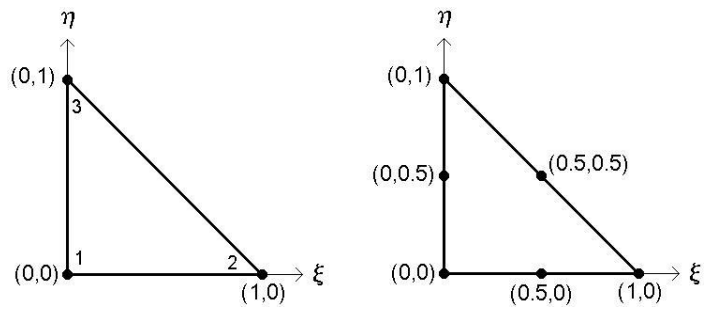


Figure A.1. Linear (Left) and Quadratic (Right) Master Triangular Elements

The class of finite elements chosen for this work is termed iso-parametric. Iso-parametric elements use the same trial functions to approximate the independent variable and to perform the transformation from the master element to any arbitrary element. To reduce the complexity of the spatial integration only the linear triangular finite element scheme will be displayed. Equation A.1 provides the linear triangular finite element trial functions.

$$L(\xi,\eta) = \begin{pmatrix} 1-\xi-\eta \\ \xi \\ \eta \end{pmatrix} \tag{A.1}$$

The ordering of the trial functions is important and it corresponds to the vertex indices indicated in Figure A.1 (the vertex index labels are found on the interior of the master element).

The transformation from an arbitrary finite element to the master element is accomplished using the following relations, where x_k and y_k are the spatial coordinates of the element vertices.

$$x(\xi,\eta) = \sum_k x_k L_k(\xi,\eta) \tag{A.2}$$

$$y(\xi,\eta) = \sum_k y_k L_k(\xi,\eta) \tag{A.3}$$

Using these relationships, the Jacobian transformation matrix and inverse Jacobian matrix can be defined as

$$\mathcal{J}(\xi,\eta)=\begin{bmatrix}\sum_k x_k \frac{d}{d\xi}L_k(\xi,\eta) & \sum_k y_k \frac{d}{d\xi}L_k(\xi,\eta)\\ \sum_k x_k \frac{d}{d\eta}L_k(\xi,\eta) & \sum_k y_k \frac{d}{d\eta}L_k(\xi,\eta)\end{bmatrix} \tag{A.4}$$

$$\mathcal{K}=\mathcal{J}(\xi,\eta)^{-1}. \tag{A.5}$$

The components of the two-dimensional $\vec{\nabla}$ operator can now be expressed as

$$\frac{d}{dx}L_k(\vec{r})=\mathcal{K}_{1,1}\frac{d}{d\xi}L_k(\xi,\eta)+\mathcal{K}_{1,2}\frac{d}{d\eta}L_k(\xi,\eta) \tag{A.6}$$

$$\frac{d}{dy}L_k(\vec{r})=\mathcal{K}_{2,1}\frac{d}{d\xi}L_k(\xi,\eta)+\mathcal{K}_{2,2}\frac{d}{d\eta}L_k(\xi,\eta). \tag{A.7}$$

Using these relationships we can show the calculation of the volumetric based matrices typically carried out in methods that utilize a finite element approximation. In what follows, $|\mathcal{J}(\xi,\eta)|$ represents the determinant of the Jacobian matrix.

$$\begin{aligned}\int \vec{\nabla}_1 L_e(\vec{r})\vec{\nabla}_1 L_e^T(\vec{r})dV_e &= \int\int\left[\mathcal{K}_{1,1}\frac{d}{d\xi}L_i(\xi,\eta)\mathcal{K}_{1,1}\frac{d}{d\xi}L_j(\xi,\eta)|\mathcal{J}(\xi,\eta)|\right]d\xi d\eta\\ &+2\int\int\left[\mathcal{K}_{1,1}\frac{d}{d\xi}L_i(\xi,\eta)\mathcal{K}_{1,2}\frac{d}{d\eta}L_j(\xi,\eta)|\mathcal{J}(\xi,\eta)|\right]d\xi d\eta\\ &+\int\int\left[\mathcal{K}_{1,2}\frac{d}{d\eta}L_i(\xi,\eta)\mathcal{K}_{1,2}\frac{d}{d\eta}L_j(\xi,\eta)|\mathcal{J}(\xi,\eta)|\right]d\xi d\eta\end{aligned} \tag{A.8}$$

$$\begin{aligned}\int \vec{\nabla}_2 L_e(\vec{r})\vec{\nabla}_2 L_e^T(\vec{r})dV_e &= \int\int\left[\mathcal{K}_{2,1}\frac{d}{d\xi}L_i(\xi,\eta)\mathcal{K}_{2,1}\frac{d}{d\xi}L_j(\xi,\eta)|\mathcal{J}(\xi,\eta)|\right]d\xi d\eta\\ &+2\int\int\left[\mathcal{K}_{2,1}\frac{d}{d\xi}L_i(\xi,\eta)\mathcal{K}_{2,2}\frac{d}{d\eta}L_j(\xi,\eta)|\mathcal{J}(\xi,\eta)|\right]d\xi d\eta\\ &+\int\int\left[\mathcal{K}_{2,2}\frac{d}{d\eta}L_i(\xi,\eta)\mathcal{K}_{2,2}\frac{d}{d\eta}L_j(\xi,\eta)|\mathcal{J}(\xi,\eta)|\right]d\xi d\eta\end{aligned} \tag{A.9}$$

$$\begin{aligned}\int \vec{\nabla}_1 L_e(\vec{r})\vec{\nabla}_2 L_e^T(\vec{r})dV_e &= \int\int\left[\mathcal{K}_{1,1}\frac{d}{d\xi}L_i(\xi,\eta)\mathcal{K}_{2,1}\frac{d}{d\xi}L_j(\xi,\eta)|\mathcal{J}(\xi,\eta)|\right]d\xi d\eta\\ &+\int\int\left[\mathcal{K}_{1,1}\frac{d}{d\xi}L_i(\xi,\eta)\mathcal{K}_{2,2}\frac{d}{d\eta}L_j(\xi,\eta)|\mathcal{J}(\xi,\eta)|\right]d\xi d\eta\\ &+\int\int\left[\mathcal{K}_{1,2}\frac{d}{d\eta}L_i(\xi,\eta)\mathcal{K}_{2,1}\frac{d}{d\xi}L_j(\xi,\eta)|\mathcal{J}(\xi,\eta)|\right]d\xi d\eta\\ &+\int\int\left[\mathcal{K}_{1,2}\frac{d}{d\eta}L_i(\xi,\eta)\mathcal{K}_{2,2}\frac{d}{d\eta}L_j(\xi,\eta)|\mathcal{J}(\xi,\eta)|\right]d\xi d\eta\end{aligned} \tag{A.10}$$

$$\int \vec{\nabla}_2 L_e(\vec{r})\vec{\nabla}_1 L_e^T(\vec{r})dV_e=\left[\int \vec{\nabla}_1 L_e(\vec{r})\vec{\nabla}_2 L_e^T(\vec{r})dV_e\right]^T \tag{A.11}$$

$$\int \vec{\nabla}_1 L_e(\vec{r}) L_e^T(\vec{r}) dV_e = \int\int \left[\mathcal{K}_{1,1} \frac{d}{d\xi} L_i(\xi,\eta) L_j(\xi,\eta) \left| \mathcal{J}(\xi,\eta) \right| \right] d\xi d\eta + \int\int \left[\mathcal{K}_{1,2} \frac{d}{d\eta} L_i(\xi,\eta) L_j(\xi,\eta) \left| \mathcal{J}(\xi,\eta) \right| \right] d\xi d\eta \tag{A.12}$$

$$\int \vec{\nabla}_2 L_e(\vec{r}) L_e^T(\vec{r}) dV_e = \int\int \left[\mathcal{K}_{2,1} \frac{d}{d\xi} L_i(\xi,\eta) L_j(\xi,\eta) \left| \mathcal{J}(\xi,\eta) \right| \right] d\xi d\eta + \int\int \left[\mathcal{K}_{2,2} \frac{d}{d\eta} L_i(\xi,\eta) L_j(\xi,\eta) \left| \mathcal{J}(\xi,\eta) \right| \right] d\xi d\eta \tag{A.13}$$

$$\int L_e(\vec{r}) L_e^T(\vec{r}) dV_e = \int\int L_i(\xi,\eta) L_j(\xi,\eta) \left| \mathcal{J}(\xi,\eta) \right| d\xi d\eta \tag{A.14}$$

The surface integrations are more complicated due to the type of integration that must be performed. In short generic expressions similar to those shown for the volumetric integrations can only be obtained for one-dimensional elements, quadrilateral elements, and hexahedral elements. This is due to the ability of directly utilizing the Jacobian for calculating the cross product in the surface area differential component. For those elements with triangular based geometry, a unique linear combination of the components of the Jacobian occurs.

Fortunately, such occurrences can be exclusively linked to the outward normal associated with the targeted element surface in the reference frame. For clarity, Figure A.2 shows the reference elements for 2-D quadrilateral elements.

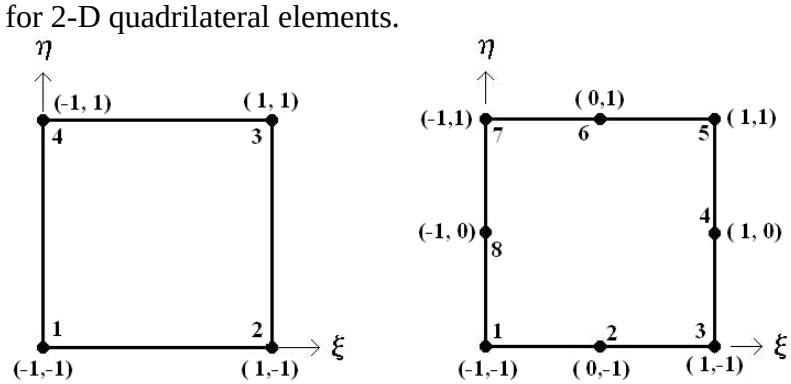


Figure A.2. Linear (Left) and Quadratic (Right) Master Quadrilateral Elements

From Figure A.1, we can see that there are two surfaces for the triangular element which are similar to the quadrilateral and thus will have a similar outward normal. In this way, the surface integration is not element specific, but specific to all of the unique outward normals derived from all of the elements that exist in the element library. This bookkeeping information must obviously be maintained in the code such that the integration type is known. The alternative is to develop a separate set of integration routines for every element type in the library. For completeness, we present the integration information for the triangular elements where that utilized for quadrilaterals and hexahedrons can be found in the literature.

A.1 Surface Integration for Two-dimensional Triangular Elements

Consider a transformation which maps the standard triangular element $P_1P_2P_3$ in onto the reference triangle $Q_1Q_2Q_3$ Figure A.3. Suppose that the line segments P_1P_2, P_2P_3, and P_3P_1 are respectively mapped onto the curves Q_1Q_2, Q_2Q_3, and Q_3Q_1, and that the transformation is represented as:

$$\begin{cases} x = f(\xi,\eta) \\ y = g(\xi,\eta) \\ z = \varsigma \end{cases} \tag{A.15}$$

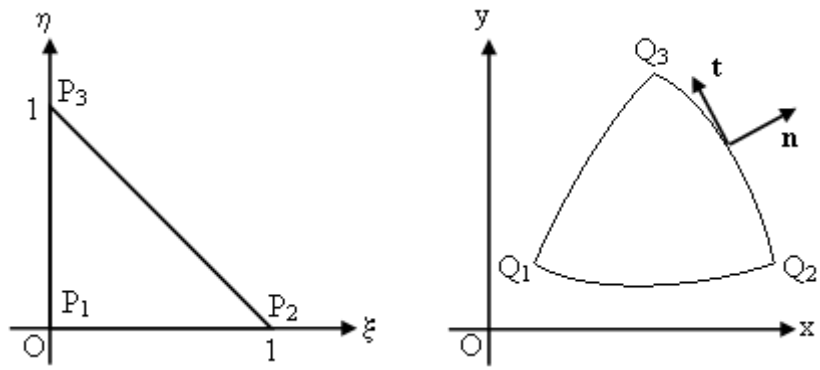


Figure A.3. A Reference Element (Left) and an General Element (Right)

Then, the curves Q_1Q_2, Q_2Q_3, and Q_3Q_1 are represented by functions of a single parameter as

$$\begin{cases} x = f(\xi,0) \\ y = g(\xi,0) \\ z = 0 \end{cases} \tag{A.16}$$

$$\begin{cases} x = f(\xi,1-\xi) = f(1-\eta,\eta) \\ y = g(\xi,1-\xi) = g(1-\eta,\eta) \\ z = 0 \end{cases} \tag{A.17}$$

$$\begin{cases} x = f(0,\eta) \\ y = g(0,\eta) \\ z = 0 \end{cases} \tag{A.18}$$

Denoting the position vector on the curves by $\mathbf{r} = (x, y, z)^t$ and the arc length along the curves by s, the tangential vectors of the curves Q_1Q_2, Q_2Q_3, and Q_3Q_1 can be written as

$$\mathbf{t} = \frac{d\mathbf{r}}{ds} = \begin{pmatrix} \dfrac{\partial f}{\partial \xi} \\ \dfrac{\partial g}{\partial \xi} \\ 0 \end{pmatrix} \frac{d\xi}{ds} \tag{A.19}$$

$$\mathbf{t} = \frac{d\mathbf{r}}{ds} = \begin{pmatrix} \frac{\partial f}{\partial \xi} - \frac{\partial f}{\partial \eta} \\ \frac{\partial g}{\partial \xi} - \frac{\partial g}{\partial \eta} \\ 0 \end{pmatrix} \frac{d\xi}{ds} = -\begin{pmatrix} \frac{\partial f}{\partial \xi} - \frac{\partial f}{\partial \eta} \\ \frac{\partial g}{\partial \xi} - \frac{\partial g}{\partial \eta} \\ 0 \end{pmatrix} \frac{d\eta}{ds} \tag{A.20}$$

$$\mathbf{t} = \frac{d\mathbf{r}}{ds} = \begin{pmatrix} \frac{\partial f}{\partial \eta} \\ \frac{\partial g}{\partial \eta} \\ 0 \end{pmatrix} \frac{d\eta}{ds} \tag{A.21}$$

Therefore, the normal vectors of the vertical surfaces that contain the curves Q_1Q_2, Q_2Q_3, and Q_3Q_1 can be determined by the cross product $\mathbf{t} \times \mathbf{k}$ as

$$\mathbf{n} = \begin{pmatrix} \frac{\partial g}{\partial \xi} \\ -\frac{\partial f}{\partial \xi} \\ 0 \end{pmatrix} \frac{d\xi}{ds} \tag{A.22}$$

$$\mathbf{n} = \begin{pmatrix} \frac{\partial g}{\partial \xi} - \frac{\partial g}{\partial \eta} \\ -\frac{\partial f}{\partial \xi} + \frac{\partial f}{\partial \eta} \\ 0 \end{pmatrix} \frac{d\xi}{ds} = -\begin{pmatrix} \frac{\partial g}{\partial \xi} - \frac{\partial g}{\partial \eta} \\ -\frac{\partial f}{\partial \xi} + \frac{\partial f}{\partial \eta} \\ 0 \end{pmatrix} \frac{d\eta}{ds} \tag{A.23}$$

$$\mathbf{n} = \begin{pmatrix} \frac{\partial g}{\partial \eta} \\ -\frac{\partial f}{\partial \eta} \\ 0 \end{pmatrix} \frac{d\eta}{ds} \tag{A.24}$$

At a point $Q(x, y, z)$ on these vertical surfaces, a differential surface area vector $d\boldsymbol{\sigma}$ is defined by a vector whose length is equal to the element of surface area $d\sigma$ associated with P and whose direction coincides with the normal vector $\mathbf{n}$ at Q. On the aforementioned vertical surfaces, it can be determined as

$$d\boldsymbol{\sigma} = d\mathbf{r} \times (\mathbf{k}dz) = (\mathbf{t}ds) \times (\mathbf{k}d\varsigma) = \mathbf{n}dsd\varsigma \tag{A.25}$$

Thus the differential surface area vectors of the vertical surfaces that contain the curves Q_1Q_2, Q_2Q_3, and Q_3Q_1 can be represented as

$$d\boldsymbol{\sigma} = \begin{pmatrix} \dfrac{\partial g}{\partial \xi} \\ -\dfrac{\partial f}{\partial \xi} \\ 0 \end{pmatrix} d\xi d\varsigma \tag{A.26}$$

$$d\boldsymbol{\sigma} = \begin{pmatrix} \dfrac{\partial g}{\partial \xi} - \dfrac{\partial g}{\partial \eta} \\ -\dfrac{\partial f}{\partial \xi} + \dfrac{\partial f}{\partial \eta} \\ 0 \end{pmatrix} d\xi d\varsigma = -\begin{pmatrix} \dfrac{\partial g}{\partial \xi} - \dfrac{\partial g}{\partial \eta} \\ -\dfrac{\partial f}{\partial \xi} + \dfrac{\partial f}{\partial \eta} \\ 0 \end{pmatrix} d\eta d\varsigma \tag{A.27}$$

$$d\boldsymbol{\sigma} = \begin{pmatrix} \dfrac{\partial g}{\partial \eta} \\ -\dfrac{\partial f}{\partial \eta} \\ 0 \end{pmatrix} d\eta d\varsigma \tag{A.28}$$

The elements of surface area $d\sigma = |\, d\boldsymbol{\sigma} \,|$ are given by

$$d\sigma = \left[\left(\frac{\partial f}{\partial \xi} \right)^2 + \left(\frac{\partial g}{\partial \xi} \right)^2 \right]^{1/2} d\xi d\varsigma \tag{A.29}$$

$$\begin{aligned} d\sigma &= \left[\left(\frac{\partial f}{\partial \xi} - \frac{\partial f}{\partial \eta} \right)^2 + \left(\frac{\partial g}{\partial \xi} - \frac{\partial g}{\partial \eta} \right)^2 \right]^{1/2} d\xi d\varsigma \\ &= \left[\left(\frac{\partial f}{\partial \xi} - \frac{\partial f}{\partial \eta} \right)^2 + \left(\frac{\partial g}{\partial \xi} - \frac{\partial g}{\partial \eta} \right)^2 \right]^{1/2} d\eta d\varsigma \end{aligned} \tag{A.30}$$

$$d\sigma = \left[\left(\frac{\partial f}{\partial \eta} \right)^2 + \left(\frac{\partial g}{\partial \eta} \right)^2 \right]^{1/2} d\eta d\varsigma \tag{A.31}$$

For a linear two-dimensional triangular finite element, we can define the following transformation

$$\begin{aligned} x &= f(\xi,\eta) = a_x \xi + b_x \eta + c_x \\ y &= g(\xi,\eta) = a_y \xi + b_y \eta + c_y \\ z &= \varsigma \end{aligned} \tag{A.33}$$

This transformation maps the points $P_1(0,0)$, $P_2(1,0)$, and $P_3(0,1)$ to $Q_1(c_x, c_y)$, $Q_2(a_x + c_x, a_y + c_y)$, and $Q_3(b_x + c_x, b_y + c_y)$, respectively. Thus, the lengths of line segments Q_1Q_2, Q_2Q_3, and Q_3Q_1 are $\sqrt{a_x^2 + a_y^2}$, $\sqrt{(b_x - a_x)^2 + (b_y - a_y)^2}$, and $\sqrt{b_x^2 + b_y^2}$. The length

elements of the line segments Q_1Q_2, Q_2Q_3, and Q_3Q_1 can be determined from equations A.29 to A.31 as

$$ds = \left[\left(\frac{\partial f}{\partial \xi}\right)^2 + \left(\frac{\partial g}{\partial \xi}\right)^2\right]^{1/2} d\xi = \sqrt{a_x^2 + a_y^2}\, d\xi \tag{A.34}$$

$$ds = \left[\left(\frac{\partial f}{\partial \xi} - \frac{\partial f}{\partial \eta}\right)^2 + \left(\frac{\partial g}{\partial \xi} - \frac{\partial g}{\partial \eta}\right)^2\right]^{1/2} d\xi = \sqrt{(a_x - b_x)^2 + (a_y - b_y)^2}\, d\xi \tag{A.35}$$

$$ds = \left[\left(\frac{\partial f}{\partial \eta}\right)^2 + \left(\frac{\partial g}{\partial \eta}\right)^2\right]^{1/2} d\eta = \sqrt{b_x^2 + b_y^2}\, d\eta \quad \text{(A.36)}$$

Integration of Eqs. A.34 to A.36 yields the exact lengths of line segments Q_1Q_2, Q_2Q_3, and Q_3Q_1.

A.2 Surface Integration for Three-dimensional Tetrahedral Elements

Suppose that a transformation which maps the standard triangular element $P_1P_2P_3$ onto $Q_1Q_2Q_3$ is given by

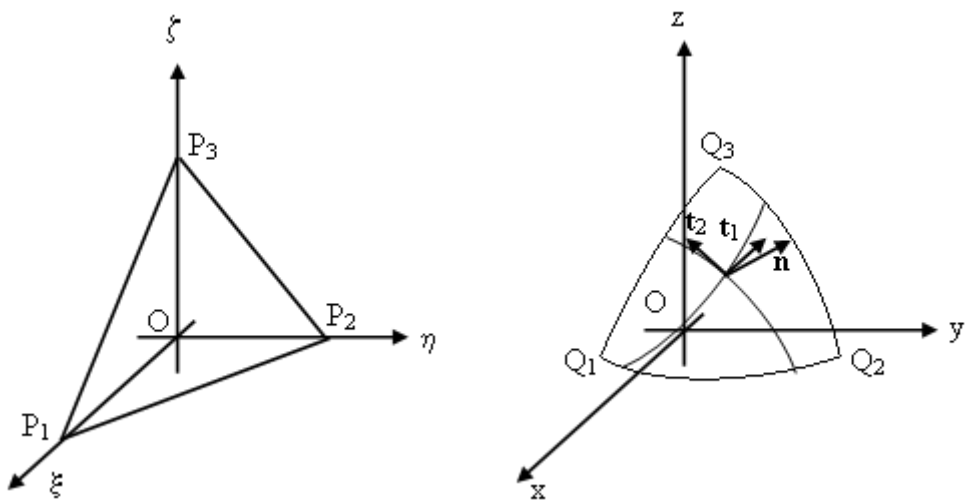


Figure A.4. A Reference Element (Left) and an General Element (Right)

$$\begin{cases} x = f(\xi,\eta,\varsigma) \\ y = g(\xi,\eta,\varsigma) \\ z = h(\xi,\eta,\varsigma) \end{cases} \tag{A.37}$$

and the surface equation for the triangle $P_1P_2P_3$ is given by

$$\alpha(\xi,\eta,\varsigma) = 0. \tag{A.38}$$

The surface $Q_1Q_2Q_3$ is represented by two parameters since the variables ξ, η, and ς are constrained by Eq. A.38. For example, if Eq. A.38 can be solved for ς, it can be represented by two parameters ξ and η. In this case, two tangent vectors $\mathbf{t}_1$ and $\mathbf{t}_2$ can be defined as

$$\mathbf{t}_1 = \left.\frac{d\mathbf{r}}{d\xi}\right|_\eta = \frac{\partial \mathbf{r}}{\partial \xi} + \frac{\partial \mathbf{r}}{\partial \varsigma}\frac{\partial \varsigma}{\partial \xi} = \left(\frac{\partial f}{\partial \xi} + \frac{\partial f}{\partial \varsigma}\frac{\partial \varsigma}{\partial \xi}\right)\mathbf{i} + \left(\frac{\partial g}{\partial \xi} + \frac{\partial g}{\partial \varsigma}\frac{\partial \varsigma}{\partial \xi}\right)\mathbf{j} + \left(\frac{\partial h}{\partial \xi} + \frac{\partial h}{\partial \varsigma}\frac{\partial \varsigma}{\partial \xi}\right)\mathbf{k} \tag{A.39}$$

$$\mathbf{t}_2 = \left.\frac{d\mathbf{r}}{d\eta}\right|_\xi = \frac{\partial \mathbf{r}}{\partial \eta} + \frac{\partial \mathbf{r}}{\partial \varsigma}\frac{\partial \varsigma}{\partial \eta} = \left(\frac{\partial f}{\partial \eta} + \frac{\partial f}{\partial \varsigma}\frac{\partial \varsigma}{\partial \eta}\right)\mathbf{i} + \left(\frac{\partial g}{\partial \eta} + \frac{\partial g}{\partial \varsigma}\frac{\partial \varsigma}{\partial \eta}\right)\mathbf{j} + \left(\frac{\partial h}{\partial \eta} + \frac{\partial h}{\partial \varsigma}\frac{\partial \varsigma}{\partial \eta}\right)\mathbf{k} \tag{A.40}$$

The partial derivatives $\partial\varsigma/\partial\xi$ and $\partial\varsigma/\partial\eta$ can be determined from Eq. A.37 as

$$\frac{\partial \varsigma}{\partial \xi} = -\frac{\partial \alpha/\partial \xi}{\partial \alpha/\partial \varsigma} \tag{A.41}$$

$$\frac{\partial \varsigma}{\partial \eta} = -\frac{\partial \alpha/\partial \eta}{\partial \alpha/\partial \varsigma} \tag{A.42}$$

Substituting Eqs. A.41 and A.42 into Eqs. A.39 and A.40, we obtain

$$\begin{aligned}\mathbf{t}_1 &= \left(\frac{\partial \alpha}{\partial \varsigma}\right)^{-1}\left(\frac{\partial \mathbf{r}}{\partial \xi}\frac{\partial \alpha}{\partial \varsigma} - \frac{\partial \mathbf{r}}{\partial \varsigma}\frac{\partial \alpha}{\partial \xi}\right) \\ &= \left(\frac{\partial \alpha}{\partial \varsigma}\right)^{-1}\left[\left(\frac{\partial f}{\partial \xi}\frac{\partial \alpha}{\partial \varsigma} - \frac{\partial f}{\partial \varsigma}\frac{\partial \alpha}{\partial \xi}\right)\mathbf{i} + \left(\frac{\partial g}{\partial \xi}\frac{\partial \alpha}{\partial \varsigma} - \frac{\partial g}{\partial \varsigma}\frac{\partial \alpha}{\partial \xi}\right)\mathbf{j} + \left(\frac{\partial h}{\partial \xi}\frac{\partial \alpha}{\partial \varsigma} - \frac{\partial h}{\partial \varsigma}\frac{\partial \alpha}{\partial \xi}\right)\mathbf{k}\right]\end{aligned} \tag{A.43}$$

$$\begin{aligned}\mathbf{t}_2 &= \left(\frac{\partial \alpha}{\partial \varsigma}\right)^{-1}\left(\frac{\partial \mathbf{r}}{\partial \eta}\frac{\partial \alpha}{\partial \varsigma} - \frac{\partial \mathbf{r}}{\partial \varsigma}\frac{\partial \alpha}{\partial \eta}\right) \\ &= \left(\frac{\partial \alpha}{\partial \varsigma}\right)^{-1}\left[\left(\frac{\partial f}{\partial \eta}\frac{\partial \alpha}{\partial \varsigma} - \frac{\partial f}{\partial \varsigma}\frac{\partial \alpha}{\partial \eta}\right)\mathbf{i} + \left(\frac{\partial g}{\partial \eta}\frac{\partial \alpha}{\partial \varsigma} - \frac{\partial g}{\partial \varsigma}\frac{\partial \alpha}{\partial \eta}\right)\mathbf{j} + \left(\frac{\partial h}{\partial \eta}\frac{\partial \alpha}{\partial \varsigma} - \frac{\partial h}{\partial \varsigma}\frac{\partial \alpha}{\partial \eta}\right)\mathbf{k}\right]\end{aligned} \tag{A.44}$$

Therefore, the differential surface area vector $d\boldsymbol{\sigma}$ can be determined as

$$d\boldsymbol{\sigma} = \mathbf{n}d\sigma = (\mathbf{t}_1 \times \mathbf{t}_2)d\xi d\eta \qquad \text{(A.45)}$$

where $\mathbf{n}$ is the normal vector at a point $Q(x, y, z)$ of the element $Q_1Q_2Q_3$.

If Eq. A.38 can be solved for ξ, the two tangent vectors $\mathbf{t}_1$ and $\mathbf{t}_2$ are defined as

$$\begin{aligned}\mathbf{t}_1 &= \left.\frac{d\mathbf{r}}{d\eta}\right|_\varsigma \\ &= \left(\frac{\partial \alpha}{\partial \xi}\right)^{-1}\left[\left(\frac{\partial f}{\partial \eta}\frac{\partial \alpha}{\partial \xi} - \frac{\partial f}{\partial \xi}\frac{\partial \alpha}{\partial \eta}\right)\mathbf{i} + \left(\frac{\partial g}{\partial \eta}\frac{\partial \alpha}{\partial \xi} - \frac{\partial g}{\partial \xi}\frac{\partial \alpha}{\partial \eta}\right)\mathbf{j} + \left(\frac{\partial h}{\partial \eta}\frac{\partial \alpha}{\partial \xi} - \frac{\partial h}{\partial \xi}\frac{\partial \alpha}{\partial \eta}\right)\mathbf{k}\right]\end{aligned} \tag{A.46}$$

$$\begin{aligned}\mathbf{t}_2 &= \left.\frac{d\mathbf{r}}{d\varsigma}\right|_\eta \\ &= \left(\frac{\partial \alpha}{\partial \xi}\right)^{-1}\left[\left(\frac{\partial f}{\partial \varsigma}\frac{\partial \alpha}{\partial \xi} - \frac{\partial f}{\partial \xi}\frac{\partial \alpha}{\partial \varsigma}\right)\mathbf{i} + \left(\frac{\partial g}{\partial \varsigma}\frac{\partial \alpha}{\partial \xi} - \frac{\partial g}{\partial \xi}\frac{\partial \alpha}{\partial \varsigma}\right)\mathbf{j} + \left(\frac{\partial h}{\partial \varsigma}\frac{\partial \alpha}{\partial \xi} - \frac{\partial h}{\partial \xi}\frac{\partial \alpha}{\partial \varsigma}\right)\mathbf{k}\right]\end{aligned} \tag{A.47}$$

and the differential surface area vector $d\boldsymbol{\sigma}$ is determined as

$$d\boldsymbol{\sigma} = \mathbf{n}d\sigma = (\mathbf{t}_1 \times \mathbf{t}_2)d\eta d\varsigma \tag{A.48}$$

If Eq. (38) can be solved for η, the two tangent vectors $\mathbf{t}_1$ and $\mathbf{t}_2$ are defined as

$$\mathbf{t}_1 = \left.\frac{d\mathbf{r}}{d\varsigma}\right|_{\xi} = \left(\frac{\partial\alpha}{\partial\eta}\right)^{-1}\left[\left(\frac{\partial f}{\partial\varsigma}\frac{\partial\alpha}{\partial\eta}-\frac{\partial f}{\partial\eta}\frac{\partial\alpha}{\partial\varsigma}\right)\mathbf{i}+\left(\frac{\partial g}{\partial\varsigma}\frac{\partial\alpha}{\partial\eta}-\frac{\partial g}{\partial\varsigma}\frac{\partial\alpha}{\partial\eta}\right)\mathbf{j}+\left(\frac{\partial h}{\partial\varsigma}\frac{\partial\alpha}{\partial\eta}-\frac{\partial h}{\partial\varsigma}\frac{\partial\alpha}{\partial\eta}\right)\mathbf{k}\right] \tag{A.49}$$

$$\mathbf{t}_2 = \left.\frac{d\mathbf{r}}{d\xi}\right|_{\varsigma} = \left(\frac{\partial\alpha}{\partial\eta}\right)^{-1}\left[\left(\frac{\partial f}{\partial\xi}\frac{\partial\alpha}{\partial\eta}-\frac{\partial f}{\partial\eta}\frac{\partial\alpha}{\partial\xi}\right)\mathbf{i}+\left(\frac{\partial g}{\partial\xi}\frac{\partial\alpha}{\partial\eta}-\frac{\partial g}{\partial\eta}\frac{\partial\alpha}{\partial\xi}\right)\mathbf{j}+\left(\frac{\partial h}{\partial\xi}\frac{\partial\alpha}{\partial\eta}-\frac{\partial h}{\partial\eta}\frac{\partial\alpha}{\partial\xi}\right)\mathbf{k}\right] \tag{A.50}$$

and the differential surface area vector $d\boldsymbol{\sigma}$ is determined as

$$d\boldsymbol{\sigma} = \mathbf{n}d\sigma = (\mathbf{t}_1 \times \mathbf{t}_2)d\varsigma d\xi \tag{A.51}$$

Given the proceeding derivations, we can display the implementation results for a four surfaces of a linear tetrahedron. Taking the surface defined as

$$\begin{aligned} &\alpha(\xi,\eta,\varsigma) = \eta = 0 \\ &x = f(\xi,\eta) \\ &y = g(\xi,\eta) \\ &z = \varsigma \end{aligned} \tag{A.52}$$

we obtain the two tangent vectors

$$\mathbf{t}_1 = \frac{\partial f}{\partial\varsigma}\mathbf{i}+\frac{\partial g}{\partial\varsigma}\mathbf{j}+\frac{\partial h}{\partial\varsigma}\mathbf{k} = \mathbf{k} \tag{A.53}$$

$$\mathbf{t}_2 = \frac{\partial f}{\partial\xi}\mathbf{i}+\frac{\partial g}{\partial\xi}\mathbf{j}+\frac{\partial h}{\partial\xi}\mathbf{k} = \frac{\partial f}{\partial\xi}\mathbf{i}+\frac{\partial g}{\partial\xi}\mathbf{j} \tag{A.54}$$

and the differential area

$$d\boldsymbol{\sigma} = \left(\frac{\partial g}{\partial\xi},-\frac{\partial f}{\partial\xi},0\right)^t d\xi d\varsigma \tag{A.55}$$

$$d\sigma = \left[\left(\frac{\partial f}{\partial\xi}\right)^2+\left(\frac{\partial g}{\partial\xi}\right)^2\right]^{1/2} d\xi d\varsigma \tag{A.56}$$

Similarly, we can pick the surface

$$\begin{aligned} &\alpha(\xi,\eta,\varsigma) = \xi = 0 \\ &x = f(\xi,\eta) \\ &y = g(\xi,\eta) \\ &z = \varsigma \end{aligned} \tag{A.57}$$

and obtain the tangent vectors

$$\mathbf{t}_1 = \frac{\partial f}{\partial\eta}\mathbf{i}+\frac{\partial g}{\partial\eta}\mathbf{j}+\frac{\partial h}{\partial\eta}\mathbf{k} = \frac{\partial f}{\partial\eta}\mathbf{i}+\frac{\partial g}{\partial\eta}\mathbf{j} \tag{A.58}$$

$$\mathbf{t}_2 = \frac{\partial f}{\partial\varsigma}\mathbf{i}+\frac{\partial g}{\partial\varsigma}\mathbf{j}+\frac{\partial h}{\partial\varsigma}\mathbf{k} = \mathbf{k} \tag{A.59}$$

and the differential area

$$d\boldsymbol{\sigma} = \left(\frac{\partial g}{\partial \eta}, -\frac{\partial f}{\partial \eta}, 0\right)^t d\eta d\varsigma \tag{A.60}$$

$$d\sigma = \left[\left(\frac{\partial f}{\partial \eta}\right)^2 + \left(\frac{\partial g}{\partial \eta}\right)^2\right]^{1/2} d\eta d\varsigma \tag{A.61}$$

Picking the other surface we have

$$\begin{aligned} &\alpha(\xi,\eta,\varsigma) = \xi + \eta - 1 = 0 \\ &x = f(\xi,\eta) \\ &y = g(\xi,\eta) \\ &z = \varsigma \end{aligned} \tag{A.62}$$

and the tangent vectors

$$\mathbf{t}_1 = \left(\frac{\partial f}{\partial \eta} - \frac{\partial f}{\partial \xi}\right)\mathbf{i} + \left(\frac{\partial g}{\partial \eta} - \frac{\partial g}{\partial \xi}\right)\mathbf{j} + \left(\frac{\partial h}{\partial \eta} - \frac{\partial h}{\partial \xi}\right)\mathbf{k} = \left(\frac{\partial f}{\partial \eta} - \frac{\partial f}{\partial \xi}\right)\mathbf{i} + \left(\frac{\partial g}{\partial \eta} - \frac{\partial g}{\partial \xi}\right)\mathbf{j} \tag{A.63}$$

$$\mathbf{t}_2 = \frac{\partial f}{\partial \varsigma}\mathbf{i} + \frac{\partial g}{\partial \varsigma}\mathbf{j} + \frac{\partial h}{\partial \varsigma}\mathbf{k} = \mathbf{k} \tag{A.64}$$

and the differential area

$$d\boldsymbol{\sigma} = \left(-\frac{\partial f}{\partial \xi} + \frac{\partial f}{\partial \eta}, \frac{\partial g}{\partial \xi} - \frac{\partial g}{\partial \eta}, 0\right)^t d\eta d\varsigma \tag{A.65}$$

$$d\sigma = \left[\left(\frac{\partial f}{\partial \xi} - \frac{\partial f}{\partial \eta}\right)^2 + \left(\frac{\partial g}{\partial \xi} - \frac{\partial g}{\partial \eta}\right)^2\right]^{1/2} d\eta d\varsigma \tag{A.66}$$

Finally, we can pick the remaining surface

$$\begin{aligned} &\alpha(\xi,\eta,\varsigma) = \xi + \eta + \varsigma - 1 = 0 \\ &x = f(\xi,\eta,\varsigma) \\ &y = g(\xi,\eta,\varsigma) \\ &z = h(\xi,\eta,\varsigma) \end{aligned} \tag{A.67}$$

To get the tangent vectors

$$\mathbf{t}_1 = \left(\frac{\partial f}{\partial \xi} - \frac{\partial f}{\partial \varsigma}\right)\mathbf{i} + \left(\frac{\partial g}{\partial \xi} - \frac{\partial g}{\partial \varsigma}\right)\mathbf{j} + \left(\frac{\partial h}{\partial \xi} - \frac{\partial h}{\partial \varsigma}\right)\mathbf{k} \tag{A.68}$$

$$\mathbf{t}_2 = \left(\frac{\partial f}{\partial \eta} - \frac{\partial f}{\partial \varsigma}\right)\mathbf{i} + \left(\frac{\partial g}{\partial \eta} - \frac{\partial g}{\partial \varsigma}\right)\mathbf{j} + \left(\frac{\partial h}{\partial \eta} - \frac{\partial h}{\partial \varsigma}\right)\mathbf{k} \tag{A.69}$$

And the differential areas

$$d\boldsymbol{\sigma} = \begin{bmatrix} \left(\frac{\partial g}{\partial \xi} - \frac{\partial g}{\partial \varsigma}\right)\left(\frac{\partial h}{\partial \eta} - \frac{\partial h}{\partial \varsigma}\right) - \left(\frac{\partial h}{\partial \xi} - \frac{\partial h}{\partial \varsigma}\right)\left(\frac{\partial g}{\partial \eta} - \frac{\partial g}{\partial \varsigma}\right) \\ \left(\frac{\partial h}{\partial \xi} - \frac{\partial h}{\partial \varsigma}\right)\left(\frac{\partial f}{\partial \eta} - \frac{\partial f}{\partial \varsigma}\right) - \left(\frac{\partial f}{\partial \xi} - \frac{\partial f}{\partial \varsigma}\right)\left(\frac{\partial h}{\partial \eta} - \frac{\partial h}{\partial \varsigma}\right) \\ \left(\frac{\partial f}{\partial \xi} - \frac{\partial f}{\partial \varsigma}\right)\left(\frac{\partial g}{\partial \eta} - \frac{\partial g}{\partial \varsigma}\right) - \left(\frac{\partial g}{\partial \xi} - \frac{\partial g}{\partial \varsigma}\right)\left(\frac{\partial f}{\partial \eta} - \frac{\partial f}{\partial \varsigma}\right) \end{bmatrix} d\xi d\eta \tag{A.70}$$

$$d\sigma = \left[\left\{ \left(\frac{\partial g}{\partial \xi} - \frac{\partial g}{\partial \varsigma} \right) \left(\frac{\partial h}{\partial \eta} - \frac{\partial h}{\partial \varsigma} \right) - \left(\frac{\partial h}{\partial \xi} - \frac{\partial h}{\partial \varsigma} \right) \left(\frac{\partial g}{\partial \eta} - \frac{\partial g}{\partial \varsigma} \right) \right\}^2 \right.$$
$$+ \left\{ \left(\frac{\partial h}{\partial \xi} - \frac{\partial h}{\partial \varsigma} \right) \left(\frac{\partial f}{\partial \eta} - \frac{\partial f}{\partial \varsigma} \right) - \left(\frac{\partial f}{\partial \xi} - \frac{\partial f}{\partial \varsigma} \right) \left(\frac{\partial h}{\partial \eta} - \frac{\partial h}{\partial \varsigma} \right) \right\}^2 \quad . \tag{A.71}$$
$$\left. + \left\{ \left(\frac{\partial f}{\partial \xi} - \frac{\partial f}{\partial \varsigma} \right) \left(\frac{\partial g}{\partial \eta} - \frac{\partial g}{\partial \varsigma} \right) - \left(\frac{\partial g}{\partial \xi} - \frac{\partial g}{\partial \varsigma} \right) \left(\frac{\partial f}{\partial \eta} - \frac{\partial f}{\partial \varsigma} \right) \right\}^2 \right]^{1/2} d\xi d\eta$$

Clearly these expressions are complicated, but all of the components are fundamentally in the Jacobian defined in Eq. A.4. In this manner, we can use the differential area and the shape functions to construct derivatives as done in Eqs. A.5 and A.6 as necessary to build matrices similar to Eqs. A.8 through A.14.

References

1. Dhatt, G., Touzot, G., The Finite Element Method Displayed. Chichester, John Wiley & Sons, 1984.

APPENDIX B. ANGULAR CUBATURE IN PROTEUS-SN

Given the long history of discrete ordinates approximations of the neutron transport equation, angular cubature have undergone quite a substantial amount of development. All of the angular cubature that exists cannot obviously be discussed here and thus we only focus on those that are relevant for two- and three-dimensional Cartesian implementations in the PROTEUS code.

B.1 Carlson's Level Symmetric Cubature

A substantial number of angular cubature that are currently used in the field of neutronics can trace their origins to the work of Bengt G. Carlson [1,2]. His focus was to develop sets of angular cubature which could be used in discrete ordinates, finite difference codes for structure Cartesian, hexagonal, and R-Z geometries. We can separate this work into two pieces: level symmetric and the Legendre-Tchebychev product cubature.

The level symmetric cubature is based upon the double tetragon which has a total of eight external triangular surfaces. This surface geometry is very similar to the two-dimensional surface of a sphere when it is split along the coordinate axes. Three separate level-symmetric cubatures were proposed by Carlson. The first Carlson method [1] used even-moment conditions to develop a set of equations which gave the necessary set of weights. The second Carlson method [1] used level-moment conditions to obtain the weights. The third Carlson method [2] imposed that all abscissas weights were equal. To produce a one-to-one angular mapping for reflected boundary conditions (to avoid interpolation), Carlson focused on placing points in a triangular pattern (Pascal's) inside of the triangle segments on the surface of the sphere. In this manner, the points were laid out to be rotationally symmetric about the coordinate axes such that the number of degrees of freedom were reduced, however, additional symmetry conditions were typically required to develop a closed set of equations. These cubature follow the well known even order series 2,4,6,8, etc., which has become synonymous with discrete ordinates methods.

Both the first and second level symmetric cubatures had the drawback of generating negative weights after 20th order. The equal weight cubature defines $R/2$ equations, which when combined, formed an Rth order equation in terms of the largest azimuthal angle (θ). After obtaining the smallest root from this equation, it can be back substituted into the other equations to define the entire set of abscissas. This approach is quite effective and appears to define a gauss rule, but it is very difficult to code up in a general fashion. As stated by Carlson, the set of $R/2$ equations were arrived at via a "trial and error" procedure thus adding more uncertainty to the possibility of producing a viable program to obtain the weights and abscissas at very high orders. We note that there are other level symmetric cubatures proposed by Carlson, but these generally had negative weights or were of insufficient accuracy in the literature.

Given the presence of the spherical harmonics code, some early work indicated problems with the first two level symmetric cubatures based upon even moment conditions (LSQ) and level moment conditions (LSL). In short, those cubature higher than S_{12} (S_{14}, S_{16},…,S_N) were stated to be of higher order, yet they only possessed accuracy of order S_{11}. One can check this by performing the integration of Eq. B.1 on the unit sphere for the set of spherical harmonics that are of order $L=N$. In Table B.1, we have shown the results where $L=N-1$ for comparison purposes to our new work.

$$\int d\Omega\, Y_l^m(\Omega) = \delta_l^m \tag{B.1}$$

Table B.1. Level-Symmetric Integration Results for Eq. (B.1).

N	LSQ[a]		LSL[b] Y_l^m	
	Points	Max Error	Points	Max Error
2	8	0.E+0	8	0.E+0
4	24	2.E-16	24	2E-16
6	48	8.E-16	48	6E-16
8	80	2.E-15	80	1E-16
10	120	3.E-14	120	1E-15
12	168	1.E-14	168	2E-15
14	224	1.E-2	224	2E-15
16	288	2.E-2	288	8E-16

[a]Level symmetric even-moment conditions (Carlson).
[b]Level symmetric level-moment abscissas with Y_l^m rules (follow an odd order series).

It is unclear whether Eq. B.1 was solved by Carlson or others since using his cubatures, but it is clear that errors are immediately present for order S_{12} or higher. It is important to note that this was found only to exist with the even-moment and level-moment cubature and not with the equal weight cubature. After some study, we determined that the additional symmetry conditions implemented by Carlson to obtain the point weights failed to obey the complete rotational symmetry of the polynomials on the unit sphere and thus the set of weights fail to produce the correct integration.

As an alternative approach to Carlson's, we chose to investigate the possibility of using a least squares procedure on Carlson's set of abscissas. As one would expect, the additional symmetry conditions imposed by Carlson (rotational groups on the first octant) reduces the viable set of spherical harmonics that can be used as constraints. First, the upper and lower hemisphere symmetry eliminates all the odd parity spherical harmonics. Second, the octant to octant symmetry eliminates the sine series of the even-parity set of spherical harmonics. Finally, the rotational symmetry within the octant eliminates the odd *m* terms of the cosine series of the even-parity spherical harmonics. The remaining set of spherical harmonics is defined in Eq. B.2 followed with the accompanying equation defining the number of moments in the series of Lth order.

$$\sum_{i=1}^{N} w_i Y_{l,m}^{\cos}\left(\vec{\Omega}_i\right) = \int_{4\pi} Y_{l,m}^{\cos}\left(\vec{\Omega}\right) d\vec{\Omega} = \delta_{l0,m0} \qquad \begin{matrix} l = 0,2,\ldots L \\ m = 0,2\ldots l \end{matrix} \tag{B.2}$$

$$M = \frac{(L+1)(L+3)}{8} \qquad L = 1,3,5,\ldots \tag{B.3}$$

As it turns out, the number of abscissas on the first octant for the Carlson cubature is given by Eq. B.4 and M=N when L=R+1.

$$N = \frac{R(R+2)}{8} \qquad R = 2, 4, 6, \ldots \qquad \text{(B.4)}$$

This means we can exactly define an odd order cubature rule without requiring the use of more spherical harmonic moments than abscissas. It is important to note that this set of spherical harmonics closely resembles the set of monomials chosen by Carlson in the even-moment method. Figure B.1 shows three selected cubature from Carlson along with the fitted results for the equivalent set of abscissas (the level-moment abscissas and those derived from even-moment conditions are effectively the same). As one would expect the set of weights between the two approaches are significantly different. However, the same rotational symmetry properties of the weights are observable in both cubatures. Unfortunately, this approach produces negative weights at order S_{17} and above order S_{21}. Although the resulting set of weights exactly meets the integration requirements defined by Eq. (2), at high orders (S_{45}+), the magnitudes of the individual weights are greater than 1.0 which indicates some degree of non-physical oscillations in the distribution of weights. Consequently, one must question the impact that this will have upon solution of the discrete ordinates or characteristics equations.

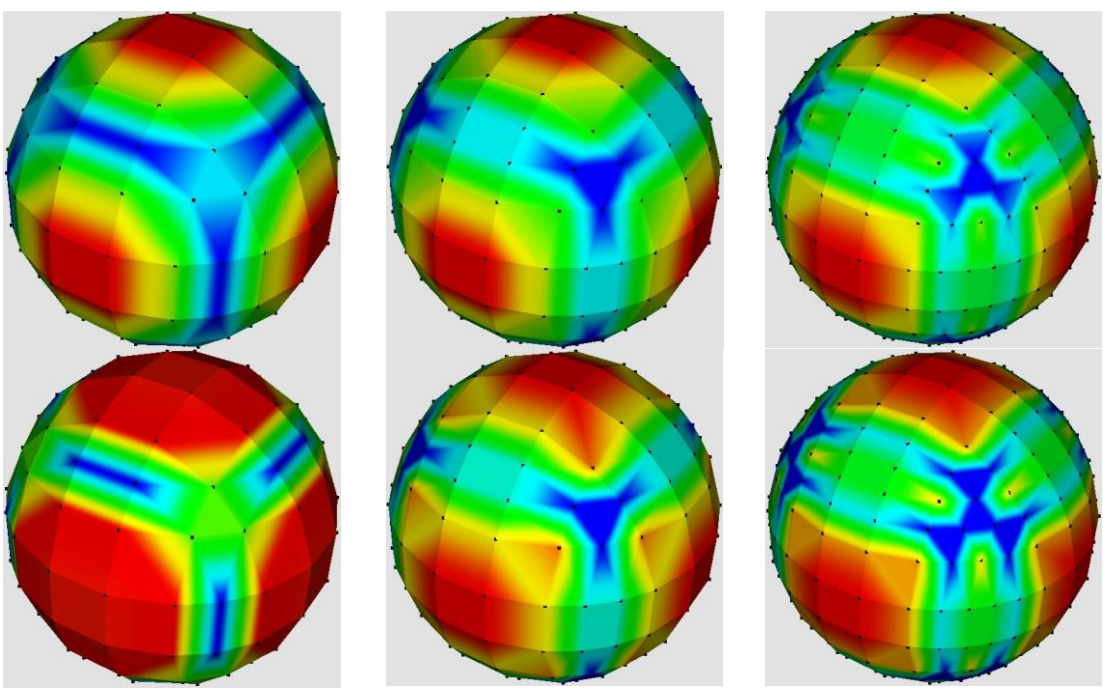


Figure B.1. Carlson S_{10}, S_{12}, and S_{16} Even-Moment Level Symmetric (top)
Least Squares Fitted S_9, S_{11}, and S_{15} Level Symmetric (bottom)

In summary, both our approach and Carlson's, negative weights are generated after order N=20, however, our approach yields cubature whose order of integration is consistent with the specified order N of the cubature. The primary problem with all of Carlson's work is that it is very difficult to construct a general computer algorithm to generate the cubature for any order R where negative weights are avoided. A greater problem is that the first and second methods proposed by Carlson are not consistent with the stated order.

B.2 Legendre-Tchebychev Product Cubature

The origin of the Legendre-Tchebychev product cubature is not clear from the literature, but the earliest record we have found is Carlson [1]. The idea behind the Legendre-Tchebychev cubature is to combine the one-dimensional cubatures to form a two-dimensional cubature. We start with the general targeted integral of Eq. B.5.

$$\int_{4\pi} d\Omega\, f\left(\hat{\Omega}\right) = \int_0^{\pi} d\theta \int_0^{2\pi} d\phi\, f\left(\theta,\phi\right)\sin\left(\theta\right) \tag{B.5}$$

For the θ variable, we utilize the transformation given by Eq. B.6 such that we can implement the well known Legendre cubature where we assume the transformation from $f\left(\theta,\phi\right)$ to $f\left(\mu,\phi\right)$ is smooth.

$$\begin{aligned} \mu &= \cos\left(\theta\right) \\ d\mu &= \sin\left(\theta\right) d\theta \end{aligned} \tag{B.6}$$

Implementing Eq. B.6 into Eq. B.5 reduces the integral to

$$\int_{4\pi} d\Omega\, f\left(\hat{\Omega}\right) = \int_{-1}^{1} d\mu \int_0^{2\pi} d\phi\, f\left(\mu,\phi\right) \tag{B.7}$$

To transform the ϕ variable, we first split the range of integration such that we can write

$$\int_{4\pi} d\Omega\, f\left(\hat{\Omega}\right) = \int_{-1}^{1} d\mu \int_0^{\pi} d\phi\, f\left(\mu,\phi\right) + \int_{-1}^{1} d\mu \int_{\pi}^{2\pi} d\phi\, f\left(\mu,\phi\right) \tag{B.8}$$

To put Eq. B.8 into a form acceptable for Tchebychev integration, we make the following variable substitution

$$\begin{aligned} h &= \cos\left(\phi\right) \\ dh &= \sin\left(\phi\right) d\phi \end{aligned} \tag{B.9}$$

We note the rule defined by Eq. B.10

$$\begin{aligned} \mu^2 + \eta^2 + \zeta^2 &= 1 \\ \mu &= \cos\left(\theta\right) \\ \eta &= \sin\left(\theta\right)\cos\left(\phi\right) \\ \zeta &= \sin\left(\theta\right)\sin\left(\phi\right) \end{aligned} \tag{B.10}$$

This allows us to replace $\sin(\phi)$ in Eq. B.9 such that we can write

$$h = \cos(\phi)$$
$$\frac{dh}{\sqrt{1-h^2}} = d\phi \tag{B.11}$$

and

$$\begin{aligned}\int_{4\pi} d\Omega\, f\left(\hat{\Omega}\right) &= \int_{-1}^{1} d\mu \int_{0}^{\pi} d\phi\, f(\mu,\phi) + \int_{-1}^{1} d\mu \int_{\pi}^{2\pi} d\phi\, f(\mu,\phi) \\ &= \int_{-1}^{1} d\mu \int_{-1}^{1} \frac{dh}{\sqrt{1-h^2}} f(\mu,h) + \int_{-1}^{1} d\mu \int_{-1}^{1} \frac{dh'}{\sqrt{1-h'^2}} f(\mu,h')\end{aligned} \tag{B.12}$$

where it is understood that the transformation h and h' are different due to the range of the ϕ variable in the substitution. To carry out the integration, we can implement the Legendre polynomial based cubature in the μ direction and the Tchebychev cubature in the h integration. The presence of the split integration in Eq. B.12 can be eliminated by proper selection of points in the space of Eq. B.10.

The primary advantage of this product cubature is its ability to exactly integrate a set of spherical harmonics and an easy to write construction algorithm. The most common path used for implementing the preceding cubature is termed Square Legendre-Tchebychev such that the order of the Tchebychev and Legendre series are the same. Another commonly used approach is termed Triangular Legendre-Tchebychev which forms a triangular pattern on the surface of the sphere similarly to the preceding level-symmetric cubature. Other notable uses are the so called double Legendre-Tchebychev where the Legendre series is defined over 0 to π. Of course, the real potential of a product cubature lies in its ability to use different orders for the one-dimensional spaces and thus tailor the number of points on the sphere for the problem of interest. Figure B.2 shows three examples of the square Legendre-Tchebychev along with three examples of the Triangular Legendre-Tchebychev which are of comparable order to those defined earlier for the level-symmetric cubatures. Table B.2 shows the integration results using the Legendre-Tchebychev cubature for Eq. B.1. Note that the poor performance of the Triangular Legendre-Tchebychev cubature is due to the inherent limitation of the construction algorithm.

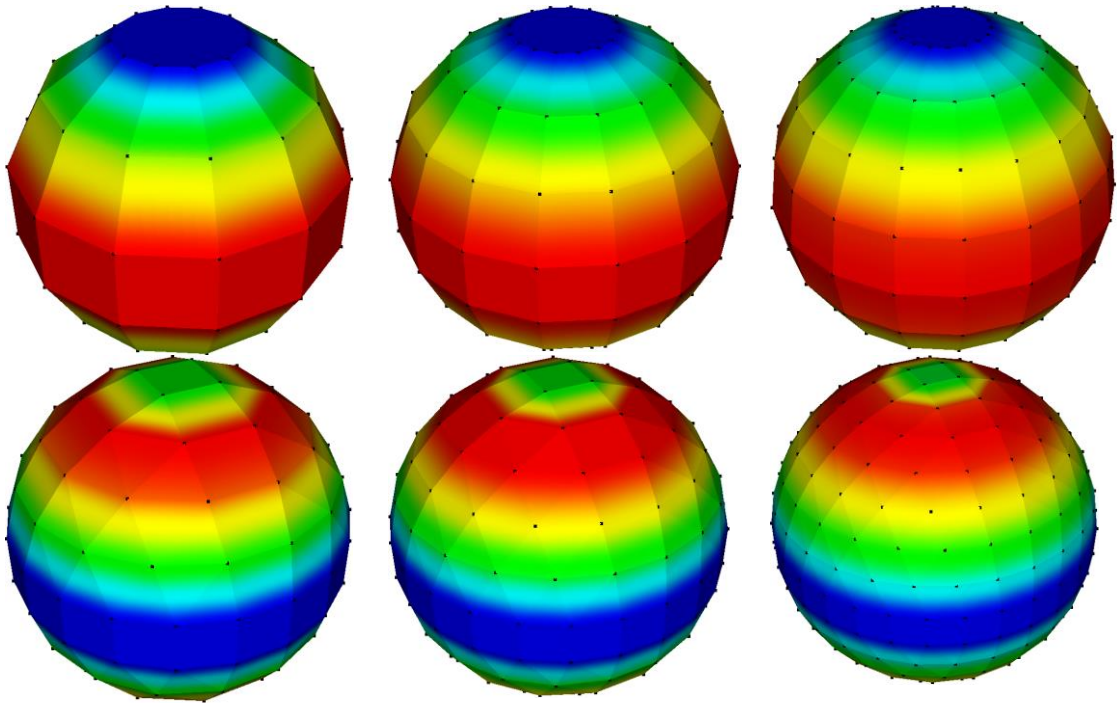


Figure B.2. Square (top) and Triangular (bottom) Legendre-Tchebychev

Table B.2. Legendre-Tchebychev Integration Error for Equation B.1

P_N Order	Square Legendre-Tchebychev	Triangular Legendre-Tchebychev	P_N Order	Square Legendre-Tchebychev	Triangular Legendre-Tchebychev
1	0.0E+00	0.0E+00	27	4.9E-14	5.8E-01
3	0.0E+00	0.0E+00	29	5.8E-14	4.4E-01
5	2.8E-15	1.2E+00	31	6.9E-14	5.3E-01
7	9.2E-13	3.0E-01	33	7.6E-14	4.4E-01
9	7.7E-15	1.1E+00	35	8.1E-14	4.9E-01
11	9.2E-15	5.2E-01	37	7.8E-14	4.3E-01
13	5.7E-14	8.9E-01	39	8.9E-14	4.4E-01
15	7.0E-15	6.2E-01	41	8.5E-14	4.1E-01
17	7.4E-15	7.1E-01	43	8.7E-14	4.0E-01
19	6.7E-15	6.4E-01	45	7.4E-14	4.0E-01
21	1.6E-14	5.7E-01	47	8.6E-14	3.6E-01
23	2.6E-14	6.2E-01	49	8.9E-14	3.8E-01
25	4.2E-14	4.7E-01	51	8.3E-14	3.3E-01

B.3 Thurgood Cubature

The Thurgood cubature [3] was posed as an alternative to Carlson's in the field of mechanical engineering for the solution of the gamma ray transport equation. The goal of course was to avoid the problems associated with generating negative weights past order 20 with the overriding goal of defining more points on the unit sphere. In this regard, they paid no attention to the ability of the cubature to integrate the spherical harmonics past 1st order. Similar to Carlson, they assume rotational symmetry of the abscissa distribution on the first octant. They take a reference equilateral triangle domain as shown in Figure B.3 and split the domain into a series of equal area triangles.

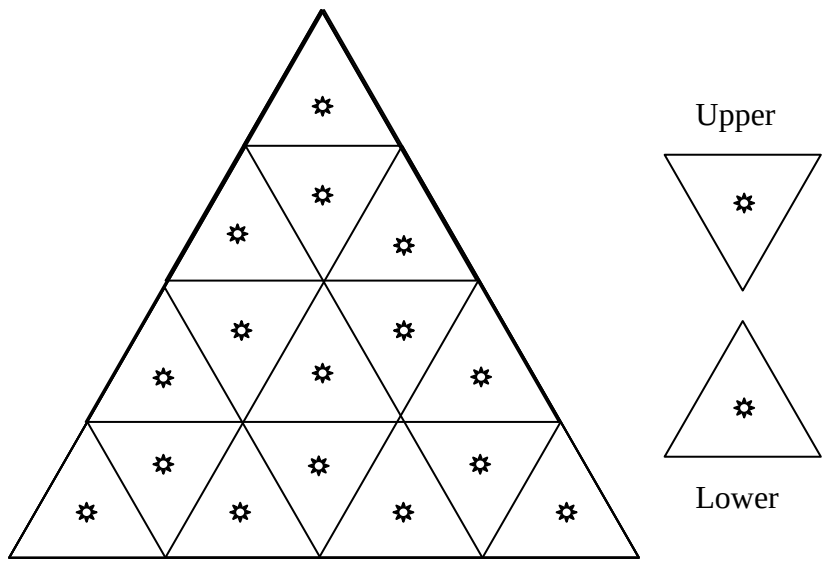


Figure B.3. Thurgood Triangulation of the Equilateral Triangle.

The abscissa locations are defined as the centroids of the equivalent spherical triangles. The mapping between the abscissas in this system to the 1st octant is straightforward and the set of abscissas that are equally spaced on the equilateral triangle turn out to be biased towards the corners of the spherical triangle that defines the 1st octant. This is the opposite of the Carlson approach where the points are biased towards the centroid of the spherical triangle. As it turns out, the Thurgood method imposes an abscissa layout which is basically equivalent to two merged Carleson level symmetric cubatures although the selection of the abscissas is quite different. As an example, the upper triangle abscissa layout is equivalent to the S_6 Carlson layout while the lower triangle abscissas layout is equivalent to the S_8 Carlson layout. As a consequence we can implement the same fitting rules used for Carlson abscissas with respect to the upper and lower set of triangles. As one would expect, this approach also leads to negative weights past order 21. Curiously, the S_{17} cubature does not yield negative weights and the relative size of the negative weights appears to be better distributed in the Thurgood cubature than the Carlson up to order S_{41}. After this order, the weights start to increase dramatically as seen with the Carlson approach. Figure B.4 shows three resolution orders of the Thurgood cubature equivalent to those of the level-symmetric cubatures. Although the cubature may appear to violate the fundamental

symmetry properties, this is only an artifact of the two merged level symmetric cubatures and the resulting meshing of the set of points. Table B.3 shows the integration results for Eq. B.1 for both the fitted and unfitted Thurgood cubature.

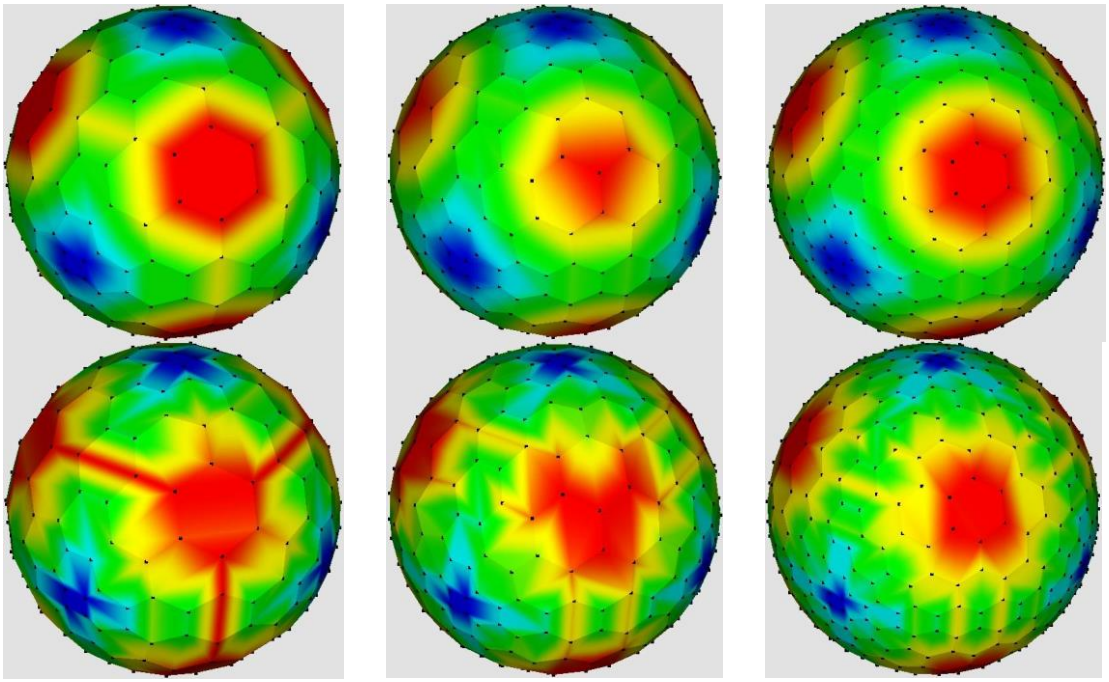


Figure B.4. Thurgood (top) and Fitted Thurgood (bottom).

Table B.3. Thurgood Integration Error for Equation B.1

P_N Order	Thurgood	Thurgood Y_l^m Fitted	P_N Order	Thurgood	Thurgood Y_l^m Fitted
1	0.0E+00	4.4E-16	27	2.0E-03	1.6E-15
3	2.2E-16	4.4E-16	29	1.9E-03	2.7E-15
5	8.5E-02	6.7E-16	31	1.5E-03	3.4E-15
7	2.2E-02	1.8E-15	33	1.5E-03	2.4E-15
9	1.6E-02	6.7E-16	35	1.3E-03	0.0E+00
11	1.1E-02	1.1E-15	37	1.2E-03	6.7E-16
13	8.1E-03	1.8E-15	39	1.0E-03	2.4E-15
15	6.2E-03	4.4E-16	41	1.0E-03	1.6E-15
17	5.7E-03	1.6E-15	43	8.9E-04	5.6E-15
19	3.9E-03	4.4E-16	45	8.6E-04	6.7E-16
21	3.3E-03	6.7E-16	47	7.6E-04	4.9E-15
23	2.7E-03	1.1E-15	49	7.4E-04	0.0E+00
25	2.3E-03	1.1E-16	51	6.6E-04	6.0E-15

B.4 Lebedev-Laikov Cubature

The Lebedev-Laikov cubature [4] is rather interesting because it follows an odd order series and is relatively unused in the neutronics field. The goal of developers of this cubature was to minimize the number of points used to integrate a set of spherical harmonics. As will be shown later, it is vastly superior in this capability when compared to any of the other cubatures. Its primary drawback is the presence of directions along the coordinate axes and planes which complicates the treatment of reflected boundary conditions. Also some negative weights exist in some of the resolutions of the cubature. While a general construction algorithm cannot be built for this cubature, sufficient data exists such that it is not a significant problem. Figure B.5 shows plots of the weight distributions for cubature orders equivalent to the preceding Carlson and Thurgood. As can be seen, the Lebedev-Laikov cubature sets are very coarse compared to the Thurgood or Carlson cubature. Table B.4 shows the excellent integration abilities of the cubature when used on Eq. B.1.

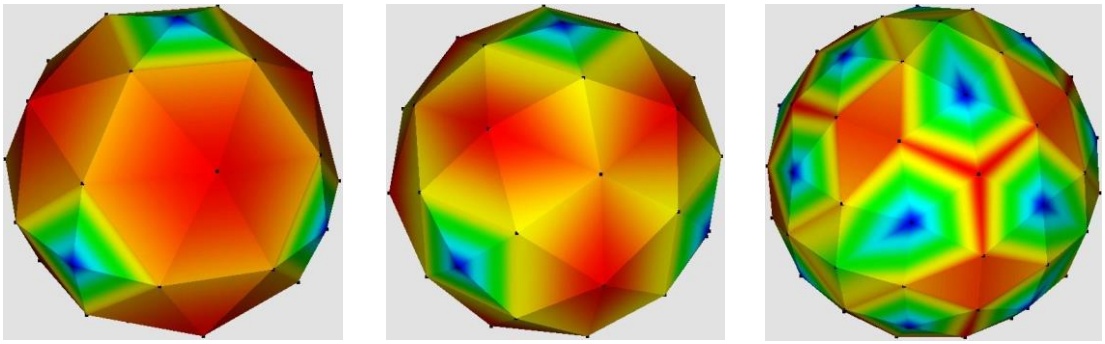


Figure B.5. Lebedov-Laikov Cubature

Table B.4. Lebedev-Laikov Integration Error for Equation B.1.

P_N Order	Lebedev-Laikov	P_N Order	Lebedev-Laikov	P_N Order	Lebedev-Laikov
1	2.2E-16	19	2.0E-15	37	1.0E-15
3	2.2E-16	21	2.2E-15	39	1.0E-15
5	2.2E-16	23	4.4E-16	41	1.0E-15
7	1.1E-16	25	1.9E-15	43	3.6E-15
9	0.0E+00	27	2.7E-15	45	3.6E-15
11	4.4E-16	29	2.1E-15	47	3.6E-15
13	3.3E-16	31	2.0E-15	49	2.4E-15
15	8.9E-16	33	1.1E-15	51	2.4E-15
17	4.4E-16	35	1.1E-15		

B.5 COBE Sky Cube Cubature

The COBE sky cube based cubature [5] is based upon the projection of the cube to the surface of the sphere rather than the double pyramid. The developers created a fitting function which would distribute points from the cube onto the sphere in nearly equidistant manner. The weights for this approach were set to be equal and were quite poor when used to integrate spherical harmonics. While the originating purpose was to create a hierarchical interpolation basis, we modified the code provided by the authors such that we filled in the missing resolutions.

In an attempt to correct the integration problems, we again utilized the least squares procedure. Due to the method employed in projection points from the cube to the sphere, we found that the fitting procedure led to two separate fitting cases which we term even and odd COBE. The results of this fitting are shown in Figure B.6 where the spherical harmonic fitting procedure was used.

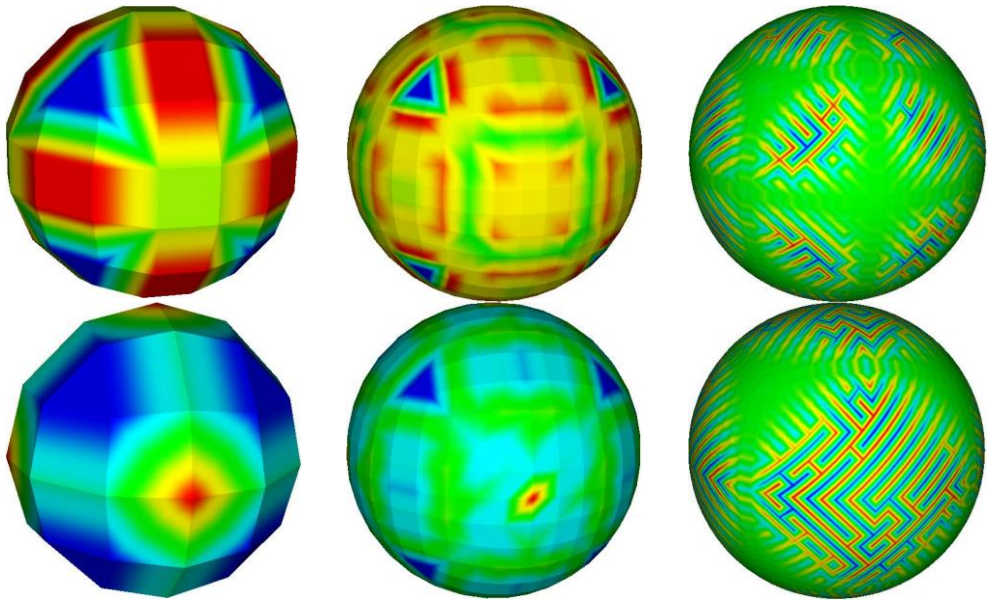


Figure B.6. Even Order (top) and Odd Order COBE Cubature.

The results of the integration for Eq. B.1 are shown in Table B.5 (COBE odd is no more effective than COBE even). Obviously more work is necessary to improve the integration properties.

Table B.5. COBE Integration Error for Equation B.1.

P_N Order	COBE Equal Weight	COBE Even Y_l^m Fitted	P_N Order	COBE Equal Weight	COBE Even Y_l^m Fitted
1	1.1E-16	0.0E+00	27	6.6E-03	5.2E-03
3	1.1E-16	0.0E+00	29	4.0E-03	2.4E-03
5	8.9E-03	8.9E-02	31	4.0E-03	2.4E-03
7	5.7E-02	8.9E-02	33	3.4E-03	1.1E-03
9	1.6E-02	3.3E-02	35	3.4E-03	1.1E-03
11	1.8E-02	3.3E-02	37	5.3E-03	1.1E-03
13	1.3E-02	4.2E-02	39	3.0E-03	3.2E-03
15	1.3E-02	4.2E-02	41	3.0E-03	3.2E-03
17	2.5E-02	4.2E-02	43	2.8E-03	1.5E-03
19	8.0E-03	1.5E-02	45	2.8E-03	1.5E-03
21	1.2E-02	1.5E-02	47	2.8E-03	1.5E-03
23	5.0E-03	5.2E-03	49	2.7E-03	1.2E-03
25	5.0E-03	5.2E-03	51	2.7E-03	1.5E-03

B.6 Tegmark Icosahedron Based Cubature

The Tegmark cubature is based upon an icosahedron projective base suggested by M. Tegmark to improve the COBE sky cube method [6]. The icosahedron is the Platonic solid closest in shape to the sphere, having twenty equilateral triangular faces and twelve vertices. Tegmark showed that by pixelizing the triangular faces into equilateral triangles, shifting the resulting vertices slightly to account for "stretching" effects, and then projecting the vertices onto the sphere, a very uniform distribution of points can be obtained as seen in Figure B.7. The resolution, *R*, of this icosahedron-based cubature leads to a set of abscissas on the unit sphere defined by the equation

$$N = 40R(R-1)+12\,. \tag{B.13}$$

The first three sets consist of *N*=12, 92, and 252 absissas (92, 252, and 1126 are shown in Figure B.7). A construction routine was provided by M. Tegmark and was modified to fit into the framework of the UNÌC Gauss library. For most neutron transport problems, only the first five Tegmark resolutions are conceivably useful with the remainder only appropriate for high order applications (a Tegmark resolution of 10 is comparable to a 60th order level symmetric). Given that Tegmark's cubature is based upon the icosahedron projection to the surface of the sphere, each triangle is broken up into a systematic pattern as shown in Figure B.8 which we term corner based Tegmark.

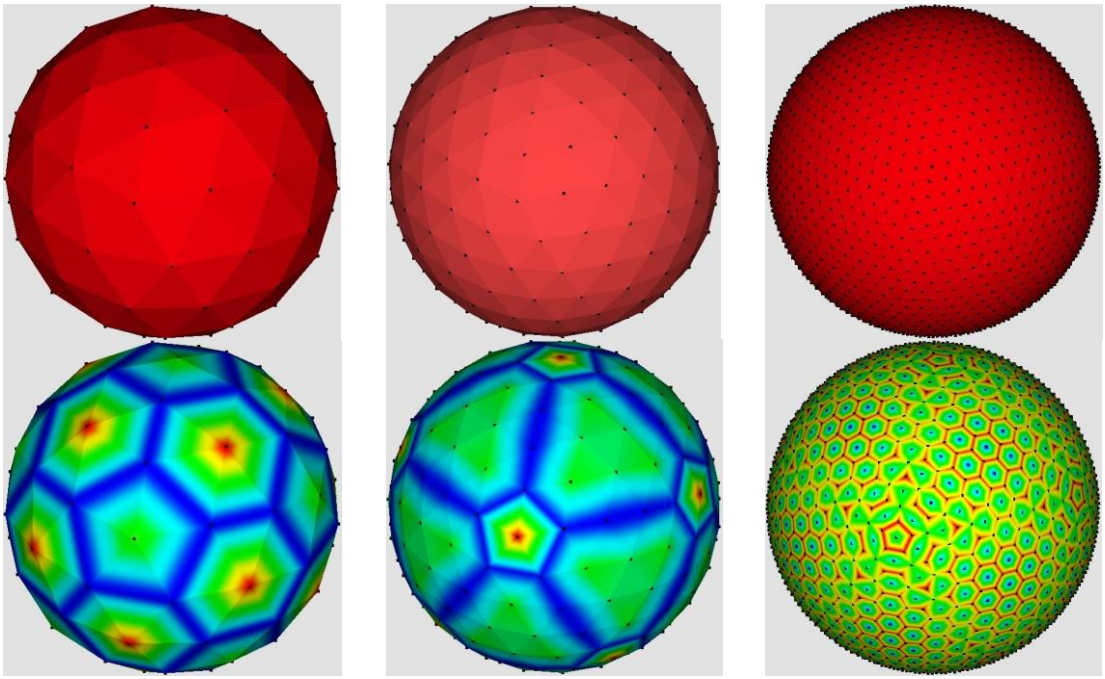


Figure B.7. Corner Based Equal Weight (top) and Least Squares Fitted (bottom).

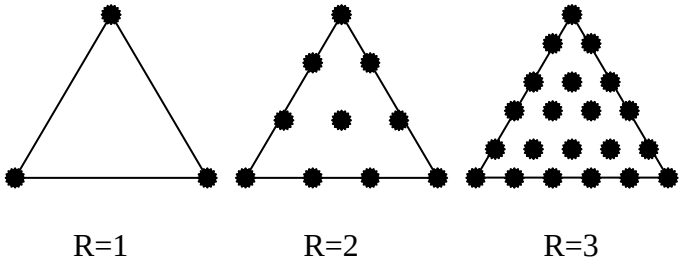


Figure B.8 Tegmark Hierarchical Resolution of a Triangle on the Icosahedron.

Similar to the COBE sky cube cubature, the intent of Tegmark was to define a hierarchical set of abscissas for sky map analysis. The first change we made was to introduce the "missing" resolutions as was done for the COBE cubature such that the hierarchical approach is negated. This alleviated the problem with a rapidly increasing number of points. The second change was to introduce the triangular setup pattern as done by Carlson such that the points lie entirely within the triangle which we term as center based Tegmark.

Overall, the primary drawback of this cubature is that it is only accurate up to 5th order spherical harmonics even though the construction routine can go much higher. As a consequence we again utilize a least squares fitting procedure to try and improve the integration properties. Given the symmetry condition on the hemisphere, we can remove the odd parity spherical harmonic moments from the least squares procedure leaving just the set of even-parity spherical harmonics. From Figure B.7, we can observe remarkably good symmetry properties in the weights for the corner based method which translates well to the integration results for Eq. B.1 as shown in Table B.6. This symmetry reflects the underlying 120 degree symmetry in the icosahedron. The center based scheme displayed in Figure B.9, does not perform as well and more work is required to improve its integration capabilities.

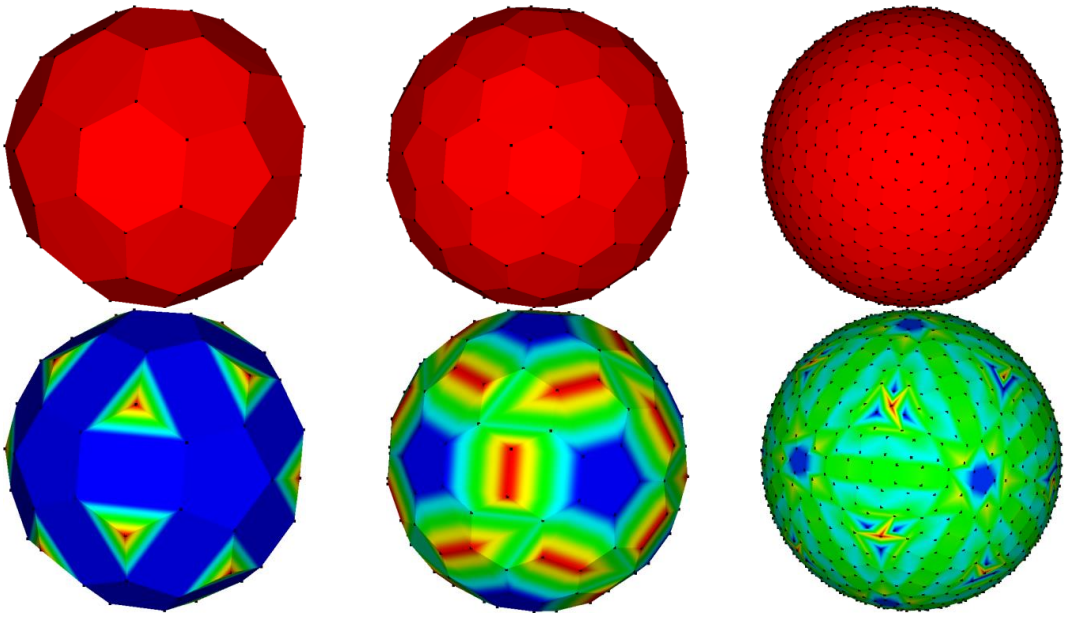


Figure B.9. Center Based Equal Weight (top) and Least Squares Fitted (bottom).

Table B.6. Tegmark Integration Error for Equation B.1.

Corner Based Tegmark			Center Based Tegmark		
P_N Order	Equal Weight	Y_l^m Fitted	P_N Order	Equal Weight	Y_l^m Fitted
1	0.0E+00	2.2E-16	1	2.2E-16	0.0E+00
3	1.2E-08	2.2E-16	3	2.2E-16	0.0E+00
5	2.8E-09	2.4E-09	5	3.1E-15	1.1E-16
7	4.7E-03	3.8E-07	7	2.4E-02	1.1E-16
9	4.7E-03	3.8E-07	9	2.2E-02	3.0E-03
11	1.5E-02	1.9E-07	11	3.2E-02	2.3E-02
13	1.5E-02	3.4E-07	13	2.0E-02	9.8E-03
15	1.5E-02	1.4E-06	15	2.0E-02	9.8E-03
17	1.1E-02	2.8E-07	17	2.5E-02	1.1E-02
19	1.1E-02	3.0E-07	19	2.0E-02	5.8E-03
21	1.1E-02	1.5E-06	21	2.0E-02	5.8E-03
23	8.8E-03	1.6E-03	23	1.9E-02	3.2E-03
25	8.8E-03	2.2E-03	25	1.9E-02	3.2E-03
27	8.8E-03	2.2E-03	27	1.7E-02	2.7E-03
29	4.7E-03	9.9E-04	29	1.7E-02	2.7E-03
31	4.7E-03	1.2E-03	31	1.3E-02	2.6E-03
33	4.7E-03	1.2E-03	33	1.3E-02	2.6E-03
35	4.5E-03	7.5E-04	35	1.3E-02	2.6E-03
37	4.5E-03	7.5E-04	37	1.1E-02	2.8E-03
39	4.7E-03	7.5E-04	39	1.2E-02	2.8E-03
41	3.1E-03	7.3E-07	41	1.0E-02	2.6E-04
43	3.1E-03	1.0E-06	43	1.0E-02	2.6E-04
45	3.8E-03	1.0E-06	45	8.5E-03	7.0E-04
47	3.0E-03	4.7E-04	47	1.1E-02	7.0E-04
49	3.0E-03	4.7E-04	49	8.3E-03	7.9E-04
51	3.1E-03	4.7E-04	51	8.3E-03	7.9E-04

B.7 Summary on Cubature

In conclusion, all of the above mentioned cubatures are currently implemented in the Gauss module of the PROTEUS-SN code. Although cubatures are generated which provide acceptable integration results, the presence of negative weights is generally detrimental to the likely success of individual cubatures. Additional work is necessary to see if this can be fixed. With regard to the problems at 20th order with the Carlson and Thurgood cubatures we believe this is a result of the fact that the abscissa layout doesn't obey the symmetry properties of the spherical harmonics beyond this order. This would seem to indicate that a more preferable layout of points is possible although this may lead to points lying along coordinate axes as is the case in the Lebedev-Laikov and Tegmark cubatures. In such a case the cubature cannot be used in two-dimensional discrete ordinates or characteristics.

Figure B.10 shows the number of points on the sphere for a given resolution of the cubature for the preceding in the current coding (those cubatures that cannot properly integrate the set of spherical harmonics are eliminated). Figure B.11 shows the corresponding "order" of the cubature as a function of the resolution number. These numbers are reproduced in Tables B.7 through B.12 for all existing cubatures on the sphere that are maintained in the Gauss Library. As can be seen, the Lebedev-Laikov cubature produces the least number of points followed directly by the Legendre-Tchebychev product cubature. The Thurgood, Tegmark, and Carlson level symmetric cubatures are next with the Tegmark cubature posing the most rapidly increasing one.

Table B.13 lists the weight distribution properties of the Lebedev-Laikov and fitted Carlson cubature while Table B.14 lists the weight distribution properties for the Thurgood and Tegmark fitted cubatures. The remaining cubatures do not have negative weight properties and thus are eliminated from this discussion. As can be seen all of the cubatures in Tables B.13 and B.14 display different behaviors of the negative weights with increasing order. The Lebedev-Laikov cubature only has negative weights at a few select orders. The fitted Carlson cubature displays the worst distribution with weights rapidly increasing after order S_{25}. The fitted Thurgood cubature has a trend similar to that of the Carlson approach, except the weights start increasing at order S_{29} with a much slower pace than that observed in Carlson. Finally, the Tegmark cubature displays a rather good, nearly equal weight distribution as the order is increased.

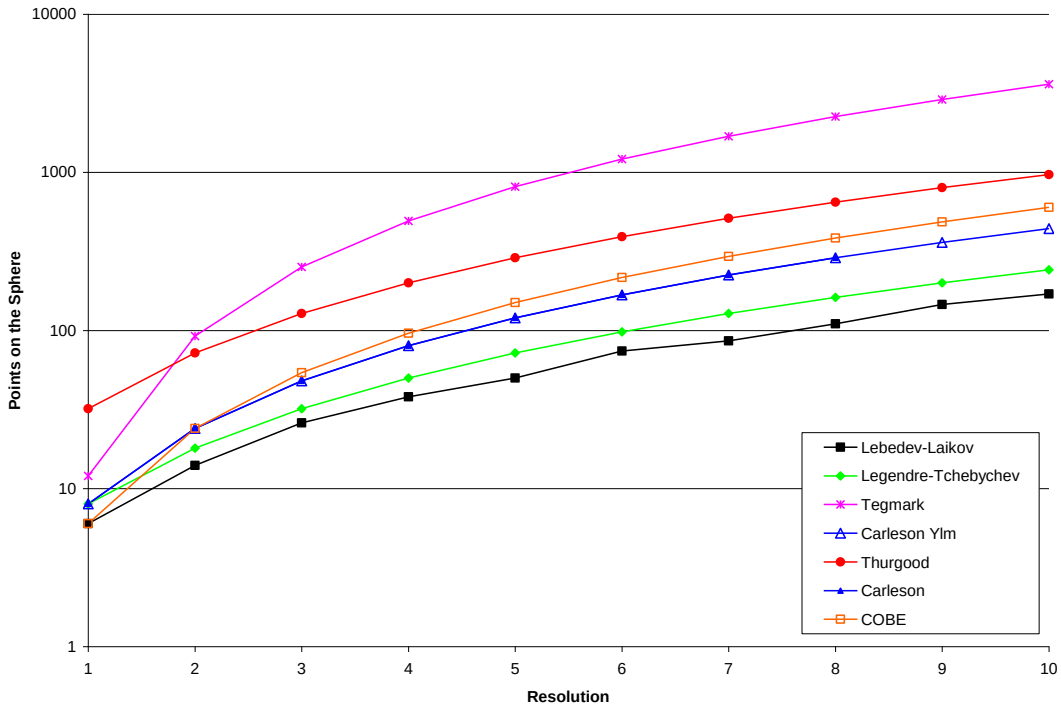


Figure B.10. Number of Abscissas versus Resolution of the Cubature

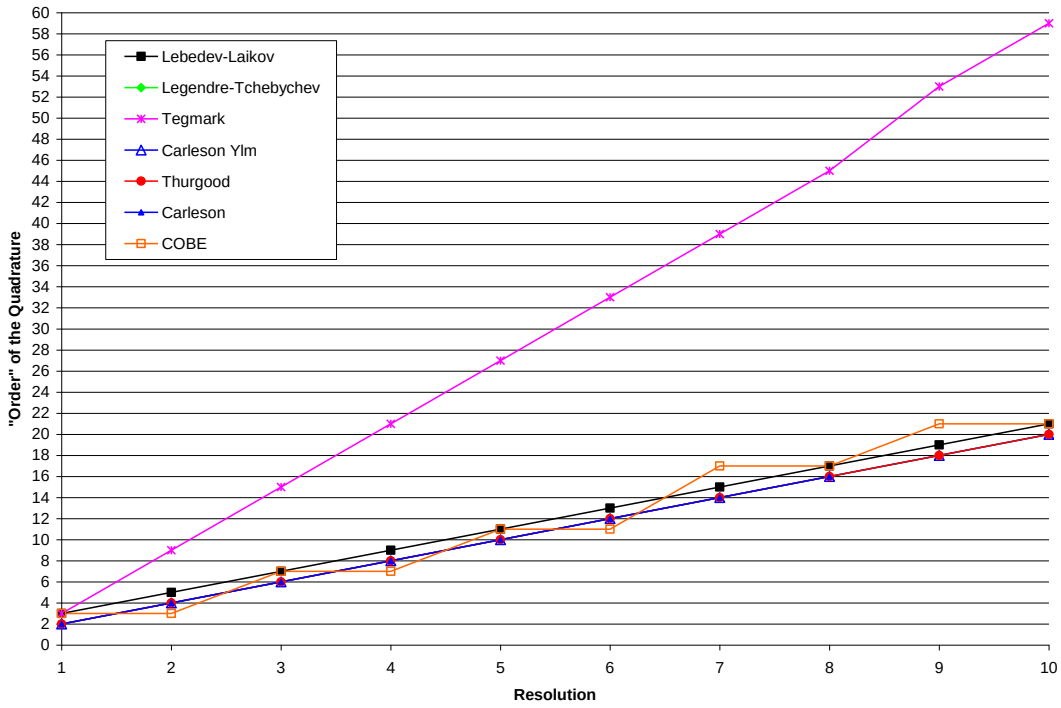


Figure B.11. "Order" of the Cubature for a Given Resolution

Table B.7. Summary Information about Carlson Cubature in Gauss Library

	CARLSON_EM			CARLSON_LM			CARLSON_EQUALW			PYRAMID_CARLSON		
Resolution	Order	2π	4π	Order	2π	4π	Order	2π	4π	Order	2π	4π
0	2	4	8	2	4	8	2	4	8	1	4	8
1	4	12	24	4	12	24	4	12	24	3	12	24
2	6	24	48	6	24	48	6	24	48	5	24	48
3	8	40	80	8	40	80	8	40	80	7	40	80
4	10	60	120				10	60	120	9	60	120
5	12	84	168	12	84	168	12	84	168	11	84	168
6	14	112	224				14	112	224	13	112	224
7	16	144	288	16	144	288	16	144	288	15	144	288

Table B.8. Summary Information about Thurgood Cubature in Gauss Library

	THURGOOD_TN			THURGOOD_LEASTSQ		
Resolution	Order	2π	4π	Order	2π	4π
0	2	16	32	2	16	32
1	4	36	72	4	36	72
2	6	64	128	6	64	128
3	8	100	200	8	100	200
4	10	144	288	10	144	288
5	12	196	392	12	196	392
6	14	256	512	14	256	512
7	16	324	648	16	324	648
8	18	400	800	18	400	800
9	20	484	968	20	484	968

Table B.9. Summary Information about Legendre-Tchebychev Cubature in Gauss Library

	LEG-TCHEBY			X_DIR_LEG-TCHEBY			DOUB_LEG-TCHEBY			TRI_LEG-TCHEBY		
Resolution	Order	2π	4π	Order	2π	4π	Order	2π	4π	Order	2π	4π
0	3	4	8	3	4	8	3	8	16	2	4	8
1	5	9	18	5	9	18	5	18	36	4	12	24
2	7	16	32	7	16	32	7	32	64	6	24	48
3	9	25	50	9	25	50	9	50	100	8	40	80
4	11	36	72	11	36	72	11	72	144	10	60	120
5	13	49	98	13	49	98	13	98	196	12	84	168
6	15	64	128	15	64	128	15	128	256	14	112	224
7	17	81	162	17	81	162	17	162	324	16	144	288
8	19	100	200	19	100	200	19	200	400	18	180	360
9	21	121	242	21	121	242	21	242	484	20	220	440
10	23	144	288	23	144	288	23	288	576	22	264	528
11	25	169	338	25	169	338	25	338	676	24	312	624
12	27	196	392	27	196	392	27	392	784	26	364	728
13	29	225	450	29	225	450	29	450	900	28	420	840
14	31	256	512	31	256	512	31	512	1024	30	480	960
15	33	289	578	33	289	578	33	578	1156	32	544	1088
16	35	324	648	35	324	648	35	648	1296	34	612	1224
17	37	361	722	37	361	722	37	722	1444	36	684	1368
18	39	400	800	39	400	800	39	800	1600	38	760	1520
19	41	441	882	41	441	882	41	882	1764	40	840	1680

20	43	484	968	43	484	968	43	968	1936	42	924	1848
21	45	529	1058	45	529	1058	45	1058	2116	44	1012	2024
22	47	576	1152	47	576	1152	47	1152	2304	46	1104	2208
23	49	625	1250	49	625	1250	49	1250	2500	48	1200	2400
24	51	676	1352	51	676	1352	51	1352	2704	50	1300	2600
25	53	729	1458	53	729	1458	53	1458	2916	52	1404	2808
26	55	784	1568	55	784	1568	55	1568	3136	54	1512	3024
27	57	841	1682	57	841	1682	57	1682	3364	56	1624	3248
28	59	900	1800	59	900	1800	59	1800	3600	58	1740	3480
29	61	961	1922	61	961	1922	61	1922	3844	60	1860	3720

Table B.10. Summary Information about Tegmark Cubature in Gauss Library

	TEG_CORNER_EQWT			TEG_CORNER_LSQ			TEG_CENTROID_EQW			TEG_CENTROID_LSQ		
Resolution	Order	2π	4π	Order	2π	4π	Order	2π	4π	Order	2π	4π
0	3	6	12	3	6	12	3	10	20	3	10	20
1	5	21	42	5	21	42	7	40	80	7	40	80
2	9	46	92	9	46	92	11	90	180	11	90	180
3	11	81	162	11	81	162	17	160	320	17	160	320
4	15	126	252	15	126	252	21	250	500	21	250	500
5	17	181	362	17	181	362	25	360	720	25	360	720
6	21	246	492	21	246	492	29	490	980	29	490	980
7	23	321	642	23	321	642	35	640	1280	35	640	1280
8	27	406	812	27	406	812	39	810	1620	39	810	1620
9	31	501	1002	31	501	1002	43	1000	2000	43	1000	2000

10	33	606	1212	33	606	1212	47	1210	2420	47	1210	2420
11	37	721	1442	37	721	1442	53	1440	2880	53	1440	2880
12	39	846	1692	39	846	1692	57	1690	3380	57	1690	3380
13	43	981	1962	43	981	1962	61	1960	3920	61	1960	3920
14	45	1126	2252	45	1126	2252	65	2250	4500	65	2250	4500

Table B.11. Summary Information about COBE Cubature in Gauss Library

	EVEN_COBE_EQ_WT			**EVEN_COBE_LSQ**			**ODD_COBE_EQ_WT**			**ODD_COBE_LSQ**		
Resolution	Order	2π	4π	Order	2π	4π	Order	2π	4π	Order	2π	4π
0	3	12	24	3	12	24	3	3	6	3	3	6
1	7	48	96	7	48	96	7	27	54	7	27	54
2	11	108	216	11	108	216	11	75	150	11	75	150
3	17	192	384	17	192	384	17	147	294	17	147	294
4	21	300	600	21	300	600	21	243	486	21	243	486
5	27	432	864	27	432	864	27	363	726	27	363	726
6	31	588	1176	31	588	1176	31	507	1014	31	507	1014
7	37	768	1536	37	768	1536	37	675	1350	37	675	1350
8	41	972	1944	41	972	1944	41	867	1734	41	867	1734
9	47	1200	2400	47	1200	2400	47	1083	2166			

Table B.12. Summary Information about Lebedev-Laikov Cubature in Gauss Library

	LEBEDEV-LAIKOV		
Resolution	Order	2π	4π
0	3	3	6
1	5	7	14
2	7	13	26
3	9	19	38
4	11	25	50
5	13	37	74
6	15	43	86
7	17	55	110
8	19	73	146
9	21	85	170
10	23	97	194
11	25	115	230
12	27	133	266
13	29	151	302
14	31	175	350
15	35	217	434
16	41	295	590
17	47	385	770
18	53	487	974
19	59	601	1202

20	65	727	1454
21	71	865	1730
22	77	1015	2030
23	83	1177	2354
24	89	1351	2702
25	95	1537	3074
26	101	1735	3470
27	107	1945	3890

Table B.13. Weight Distributions for Lebedev-Laikov and Carlson Y_l^m Fitted.

	Lebedev-Laikov				Carlson Y_l^m Fitted			
	Positive		Negative		Positive		Negative	
P_N	Number	Average	Number	Average	Number	Average	Number	Average
1	6	1.7E-01	0	0.0E+00	8	1.3E-01	0	0.0E+00
3	6	1.7E-01	0	0.0E+00	24	4.2E-02	0	0.0E+00
5	14	7.1E-02	0	0.0E+00	48	2.1E-02	0	0.0E+00
7	26	3.8E-02	0	0.0E+00	80	1.3E-02	0	0.0E+00
9	38	2.6E-02	0	0.0E+00	120	8.3E-03	0	0.0E+00
11	50	2.0E-02	0	0.0E+00	168	6.0E-03	0	0.0E+00
13	66	1.9E-02	8	-3.0E-02	224	4.5E-03	0	0.0E+00
15	86	1.2E-02	0	0.0E+00	288	3.5E-03	0	0.0E+00
17	110	9.1E-03	0	0.0E+00	336	3.0E-03	24	-4.0E-04
19	146	6.8E-03	0	0.0E+00	440	2.3E-03	0	0.0E+00
21	170	5.9E-03	0	0.0E+00	504	2.0E-03	24	-3.3E-04
23	194	5.2E-03	0	0.0E+00	600	1.7E-03	24	-5.1E-05
25	224	5.9E-03	6	-5.5E-02	560	2.2E-03	168	-1.5E-03
27	248	4.2E-03	18	-2.1E-03	624	2.7E-03	216	-3.1E-03
29	302	3.3E-03	0	0.0E+00	528	8.2E-03	432	-7.7E-03
31	350	2.9E-03	0	0.0E+00	744	8.1E-03	344	-1.5E-02
33	434	2.3E-03	0	0.0E+00	648	2.8E-02	576	-3.0E-02
35	434	2.3E-03	0	0.0E+00	816	3.2E-02	552	-4.5E-02
37	590	1.7E-03	0	0.0E+00	800	9.6E-02	720	-1.0E-01

39	590	1.7E-03	0	0.0E+00	912	1.2E-01	768	-1.4E-01
41	590	1.7E-03	0	0.0E+00	984	3.1E-01	864	-3.6E-01
43	770	1.3E-03	0	0.0E+00	1104	4.1E-01	920	-4.9E-01
45	770	1.3E-03	0	0.0E+00	1128	1.1E+00	1080	-1.1E+00
47	770	1.3E-03	0	0.0E+00	1224	1.5E+00	1176	-1.5E+00
49	974	1.0E-03	0	0.0E+00	1352	3.5E+00	1248	-3.8E+00
51	974	1.0E-03	0	0.0E+00	1488	4.8E+00	1320	-5.5E+00

Table B.14. Weight Distributions for Thurgood and Tegmark Y_l^m Fitted.

	Thurgood Y_l^m Fitted				Tegmark Y_l^m Fitted			
	Positive		Negative		Positive		Negative	
P_N	Number	Average	Number	Average	Number	Average	Number	Average
1	32	3.1E-02	0	0.0E+00	12	8.3E-02	0	0.0E+00
3	72	1.4E-02	0	0.0E+00	12	8.3E-02	0	0.0E+00
5	128	7.8E-03	0	0.0E+00	92	1.1E-02	0	0.0E+00
7	200	5.0E-03	0	0.0E+00	92	1.1E-02	0	0.0E+00
9	288	3.5E-03	0	0.0E+00	92	1.1E-02	0	0.0E+00
11	392	2.6E-03	0	0.0E+00	252	4.0E-03	0	0.0E+00
13	512	2.0E-03	0	0.0E+00	252	4.0E-03	0	0.0E+00
15	648	1.5E-03	0	0.0E+00	252	4.0E-03	0	0.0E+00
17	800	1.3E-03	0	0.0E+00	492	2.0E-03	0	0.0E+00
19	968	1.0E-03	0	0.0E+00	492	2.0E-03	0	0.0E+00
21	1152	8.7E-04	0	0.0E+00	492	2.0E-03	0	0.0E+00
23	1328	7.5E-04	24	-7.3E-05	812	1.2E-03	0	0.0E+00
25	1472	6.9E-04	96	-1.5E-04	812	1.2E-03	0	0.0E+00
27	1656	6.3E-04	144	-3.3E-04	812	1.2E-03	0	0.0E+00
29	1856	6.0E-04	192	-6.1E-04	1212	8.3E-04	0	0.0E+00
31	1976	6.1E-04	336	-6.3E-04	1212	8.3E-04	0	0.0E+00
33	2256	6.2E-04	336	-1.2E-03	1212	8.3E-04	0	0.0E+00
35	2552	6.7E-04	336	-2.1E-03	1692	5.9E-04	0	0.0E+00
37	2720	8.2E-04	480	-2.6E-03	1692	5.9E-04	0	0.0E+00

39	2952	1.0E-03	576	-3.6E-03	1692	5.9E-04	0	0.0E+00
41	3248	1.4E-03	624	-5.8E-03	2252	4.4E-04	0	0.0E+00
43	3488	2.0E-03	744	-8.2E-03	2252	4.4E-04	0	0.0E+00
45	3840	3.0E-03	768	-1.4E-02	2252	4.4E-04	0	0.0E+00
47	4088	4.7E-03	912	-2.0E-02	2892	3.5E-04	0	0.0E+00
49	4304	7.5E-03	1104	-2.8E-02	2892	3.5E-04	0	0.0E+00
51	4752	1.2E-02	1080	-5.1E-02	2892	3.5E-04	0	0.0E+00

We note that the preceding cubatures are primarily focused on Cartesian geometries. In the majority of the work done in this report, the focus has been on hexagonal geometry with 30 degree or 60 degree symmetry conditions. Under these circumstances, the use of a cubature which doesn't satisfy the symmetry requires more directions than one which does satisfy the symmetry conditions. While certain resolutions of the square Legendre-Tchebychev can meet these symmetry conditions, in general there is a need to add additional cubatures which satisfy these symmetry conditions.

We also note that the preceding set of cubatures is not all inclusive and that there are many other cubatures which can be utilized. To note, we can cite the work of Abu-Shumays [7], Heo and Xu [8], Hardin, et. al. [9], and the HEALpix software [10]. The decision to include any such other cubatures will be based upon the need and the potential benefit that can be gained over the existing data in the library.

References

1. Bengt G. Carlson, "Transport Theory: Discrete Ordinates Quadrature Over the Unit Sphere," LA-4554, Los Alamos Scientific Laboratory (April, 1971).
2. Bengt G. Carlson, "Tables of Equal Weight Quadrature EQn Over the Unit Sphere," LA-4734, Los Alamos Scientific Laboratory (April, 1971).
3. C.P. Thurgood, A. Pollard, and H. A. Becker, "The T_N Quadrature Set for the Discrete Ordinates Method," Transactions of the ASME, Vol. 117, November 1995, page 1068.
4. V. I. Lebedev, and D.N. Laikov, "A Quadrature Formula for the Sphere of the 131st Algebraic Order of Accuracy," Doklady Mathematics, Vol. 59, No. 3, 1999, pp. 477-481.
5. F.K. Chan and E.M. O'Neill, "Feasibility Study of a Quadrilateralized Spherical Cube Earth Data Base," Computer Sciences Corp. EPRF Technical Report.
6. Max Tegmark, "An Icosahedron-Based Method for Pixelizing the Celestial Sphere," ApJ Letters, 470, L81-L84 (1996).
7. I. K. Abu-Shumays, "Angular Quadratures for Improved Transport Computations," B-T-3276. Bettis Atomic Power Laboratory, West Mifflin, Pennsylvania 15122 (1999).
8. Sangwoo Heo and Yuan Xu, "Constructing fully symmetric cubature formulae for the sphere," Math. Comp. **70**, 233, p269-279, 2001.
9. R. H. Hardin, N. J. A. Sloane and W. D. Smith, "Minimal Energy Arrangements of Points on a Sphere," **Spherical Codes,** http://www.research.att.com/~njas.
10. Krzysztof M. Górski, Benjamin D. Wandelt, Eric Hivon, Frode K. Hansen, and Anthony J. Banday, "HEALPix, a Hierarchical, Equal Area, and Iso-Latitude Pixelisation of the Sphere," Jet Propulsion Laboratory, California Institute of Technology, http://healpix.jpl.nasa.gov/.

APPENDIX C. REFLECTED BOUNDARY CONDITION

Reflected boundary conditions for the discrete ordinates methods generally produces a mapping between the incoming set of angular directions and the outgoing set of angular directions. For cubature which meet the symmetry conditions that are being imposed, this mapping becomes a one-to-one mapping and is quite easy to setup. When the cubature does not meet the symmetry conditions, more effort is required such that the boundary conditions can be met. While it is entirely likely that existing reactors, which have 30, 45, 90, or 120 degree symmetry, can be properly accounted for using existing cubature, the user must be very well versed in the cubature that they are using. Moreover, if a new symmetry condition is implemented other than those already mentioned, then cubature must be defined which can be used in those applications. The more problematic issue is that when such symmetry conditions are utilized and a one-to-one mapping routine is built, the geometry must be constrained such that it agrees with the scheme employed.

To avoid these issues and make certain the code can be used in a general fashion; we decided to implement a triangular interpolation routine on the surface of the sphere. In this manner, we allow users to utilize the level-symmetric cubature of Carlson, which most users are familiar with, for any problem they want to solve. To accomplish this we first had to build an algorithm which would generate a triangular mesh on the surface of the sphere. We considered using existing Delaney triangular routines, but we found that these routines were more focused on producing well posed meshes and thus arbitrarily added directions. This obviously causes more problems than it is worth and we constructed our own routine which was inevitably less efficient. However, we have found that the meshing procedure we created is computationally irrelevant (less than 1 second) on modern computational platforms for cubatures with less than 2000 directions on the sphere which is more than acceptable for neutronics problems.

A reflected boundary condition is typically defined on a flat surface and the cubature on the sphere can be split into three sets: incoming, parallel, outgoing. The implementation approach for how to deal with these three sets differs between the methods that utilize the cubature, thus we focused on working on the assumption that the outgoing directed directions can only be redirected to the incoming set of directions such that we can define the relationship

$$\begin{aligned}\psi_{in} &= \sum_j \mathcal{P}_j \, \psi_{out,j} \\ &= \mathcal{P} \, \psi_{out}\end{aligned} \qquad \psi_{in} \in \begin{cases} \psi_n \\ \Omega_n \cdot \hat{n} < 0 \end{cases} \qquad \psi_{out} \in \begin{cases} \psi_n \\ \Omega_n \cdot \hat{n} \geq 0 \end{cases}. \tag{C.1}$$

Given this, we can postulate that at most three conditions will occur when an outgoing direction is reflected as shown in Figure C.1. In all three situations we can compute the spherical distance between the reflected outgoing direction (red dot) and the incoming directions which make up the corners of the triangle. In the general case, we obviously want to specify weights which puts the largest fraction to the closest point to the reflected direction which is guaranteed by using Eq. C.2.

$$w_1 = \frac{(L_2 + L_3)}{2(L_1 + L_2 + L_3)} \quad w_2 = \frac{(L_1 + L_3)}{2(L_1 + L_2 + L_3)} \quad w_3 = \frac{(L_1 + L_2)}{2(L_1 + L_2 + L_3)} \tag{C.2}$$

For the case when a line is intersected, we use Eq. C.3 and for a point intersection, we obviously put all of the weight to the point that is intersected.

$$w_A = \frac{L_B}{(L_A + L_B)} \quad w_B = \frac{L_A}{(L_A + L_B)} \tag{C.3}$$

When Eq. C.2 and C.3 produce dependencies which are either self dependent or dependent upon parallel directions, the remaining weights are renormalized such that these dependencies are eliminated. We can obviously define a better procedure which is based upon an intersection area rather than spherical distance, but the preceding one is computationally efficient and appears to be sufficient for producing accurate results.

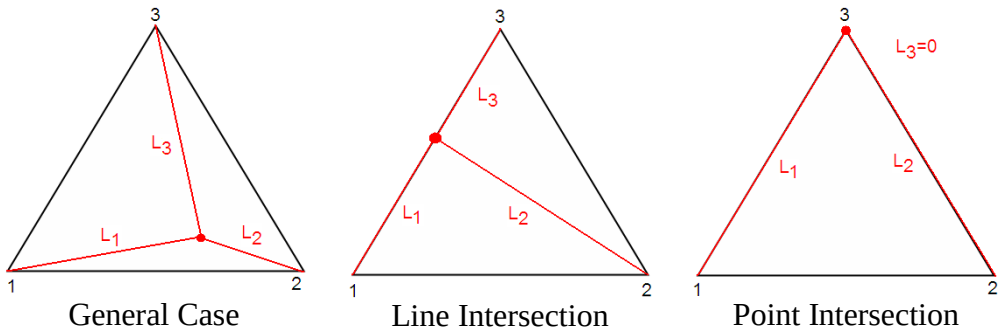


Figure C.1 The Three Possible Intersection Outcomes for a Reflected Direction.

Nuclear Engineering Division
Argonne National Laboratory
9700 South Cass Avenue, Bldg. 208
Argonne, IL 60439-4842

www.anl.gov

Argonne National Laboratory is a U.S. Department of Energy
laboratory managed by UChicago Argonne, LLC