Development of the Adaptive Collision Source Method for Discrete Ordinates Radiation Transport

William J. Walters

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Nuclear Engineering

Alireza Haghighat, Chair
Mark Pierson
Christopher Roy
Glenn Sjoden
Danesh Tafti

March 25, 2015
Blacksburg, Virginia

Keywords: neutron transport theory, discrete ordinates, angular quadrature

Development of the Adaptive Collision Source Method for Discrete Ordinates Radiation Transport

William J. Walters

ABSTRACT

A novel collision source method has been developed to solve the Linear Boltzmann Equation (LBE) more efficiently by adaptation of the angular quadrature order. The angular adaptation method is unique in that the flux from each scattering source iteration is obtained, with potentially a different quadrature order used for each. Traditionally, the flux from every iteration is combined, with the same quadrature applied to the combined flux. Since the scattering process tends to distribute the radiation more evenly over angles (i.e., make it more isotropic), the quadrature requirements generally decrease with each iteration. This method allows for an optimal use of processing power, by using a high order quadrature for the first few iterations that need it, before shifting to lower order quadratures for the remaining iterations. This is essentially an extension of the first collision source method, and is referred to as the adaptive collision source (ACS) method. The ACS methodology has been implemented in the 3-D, parallel, multigroup discrete ordinates code TITAN. This code was tested on a variety of test problems including fixed-source and eigenvalue problems. The ACS implementation in TITAN has shown a reduction in computation time by a factor of 1.5-4 on the fixed-source test problems, for the same desired level of accuracy, as compared to the standard TITAN code.

Acknowledgments

I would like to thank my advisor Dr. Alireza Haghighat for his support and guidance throughout my years in graduate school. His knowledge of transport theory was essential in shaping this work. I am also grateful my other committee members, Dr. Mark Pierson, Dr. Christopher Roy, Dr. Danesh Tafti, and Dr. Glenn Sjoden for their time and inights. In particular, Dr. Sjoden has been a constant source of wisdom since I started graduate school.

This work was helped greatly by Dr. Ce Yi, who not only allowed me to modify his TITAN code but also was essential on understanding its inner workings.

Of course, life would have much more difficult without the friendship and assistance of my fellow graduate students. In particular, I want to thank Mike Wenner, for mentoring me when I first started out; Kevin Manalo, for our many discussions on computational methods; Nathan Roskoff, for comedic relief; and most of all Katherine Royston, for helping me remember all my deadlines.

Finally, I would also like to thank my family and friends for their continued encouragement and support. I credit much of my success to my mother and father; their love and support throughout my educational endeavors has been truly motivating.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The linear Boltzmann Equation (LBE) describes the balance of neutral particle transport (i.e., neutrons and photons) in phase space (position, energy, direction and time)[1]. The solution of this equation is essential for the design and optimization of any physical system involving radiation, such as nuclear reactors, radiation protection and shielding, medical imaging devices, and radiation detectors. An analytical solution to the LBE is usually impossible, except for very simplified cases. Generally, the LBE is solved using computer simulations by discretizing the different variables and employing different numerical techniques to obtain an approximate solution.

In this dissertation, a new method is presented for adapting the angular discretization in order to solve the LBE more efficiently.

1.1 Linear Boltzmann Equation

The Linear Boltzman Equation is the foundation of radiation transport theory. The steady-state (i.e., time independent) LBE describes the angular radiation flux ψ, which is the

variable of interest, in Eq. 1.1. Note that the LBE is presented here for demonstration purposes and will be explained in depth in Chapter 2.

$$\hat{\Omega} \cdot \nabla \Psi(\vec{r}, E, \Omega) + \sigma(\vec{r}, E)\Psi(\vec{r}, E, \Omega) = Q_0(\vec{r}, E, \Omega) + \int_0^\infty dE' \int_{4\pi} d\Omega' \sigma_s(\vec{r}, E' \to E, \hat{\Omega}' \cdot \hat{\Omega})\Psi(\vec{r}, E, \Omega) + \frac{\chi(E)}{4\pi} \int_0^\infty dE' \int_{4\pi} d\Omega' \nu(\vec{r}, E)\sigma_f(\vec{r}, E)\Psi(\vec{r}, E, \Omega) \tag{1.1}$$

In solving the radiation transport problem (essentially, solving the LBE), there are two fundamental approaches - deterministic methods and stochastic methods.

1.1.1 Stochastic Methods

Stochastic methods (also known as Monte Carlo methods) are a way to solve the radiation transport problem in a statistical fashion[2]. Instead of trying to solve the LBE directly, the stochastic processes of many individual particles are simulated in an attempt to determine the average or expected particle behavior. The Monte Carlo approach is generally seen as the most accurate method of computational radiation transport, since it is generally free of any discretization errors, as space, energy and angle can all be treated as continuous functions. One major downside is the computation time; this can be attributed to the need for the simulation of many particles (up to billions or more) in order to get answers with a reasonable statistical uncertainty. Another downside is the amount of information. For deterministic methods, information about the entire phase space is obtained, whereas only certain tallies of integrated quantities get recorded in Monte Carlo. If one tries to get more detailed information from Monte Carlo, then inherently, fewer events will be tallied and the statistical uncertainties will rise resulting in even higher computation times.

1.1.2 Deterministic Methods

Deterministic methods involve trying to solve the LBE numerically by discretizing the problem domain. This method has the advantage of huge amounts of information - the entire phase space is mapped. Generally it is also much faster for complete convergence of the LBE than Monte Carlo. The main downside of deterministic methods is the discretization of the phase space - energy, angle, space and sometimes time. The problem domain must be split up into many small pieces in order to be solved; this discretization process can lead to errors and also consume significant amounts of computer memory, which can be a limiting factor. Another issue is that, for problems with high scattering ratio (i.e., a diffusive medium with little absorption), numerical convergence can be slow for certain types of iteration/convergence methods.

The discrete ordinates (S_N) method is one of the standard methods to discretize the angular variable, and is used in many production radiation transport codes[3, 4, 5, 6]. In the discrete ordinates method, the LBE is solved over a set of angular directions $\hat{\Omega}_i$ (also called ordinates), with corresponding weights w_i[7, 8]. This combination of $(\hat{\Omega}_i, w_i)$ is called an angular quadrature set. A quadrature set allows the angular integrals to be converted into sums over the angles of the set. Selection of an appropriate angular quadrature set is one of the difficulties with the discrete ordinates method[9, 10]. Depending on the problem, the flux may vary greatly in direction (that is, very anisotropic). An anisotropic function would require many quadrature points in order to properly resolve the integrals. Using a quadrature order that is too low will result in large errors from so-called "ray effects", while using an order too high greatly increases computation time[11].

1.2 Literature Review on Angular Quadrature Optimization

There have been several methods developed to optimize the efficiency of the angular quadrature in discrete ordinates calculations. The first is simply in the selection of better general quadrature sets that can more accurately integrate the angular flux for the same total number of directions. Another method is local angular refinement, which involves adding quadrature points in angular directions of high anisotropy, while leaving a coarse distribution of points in the more isotropic directions. There is also the so-called first-collision source, which is more closely related to the subject of this dissertation. This involves calculating the un-collided flux using a high-order transport method and then using this to generate a first-collision source, which is used to start a low order transport calculation to solve for the collided flux and thus save on computation time.

1.2.1 Advanced Quadrature Sets

The first way to achieve a more efficient angular discretization is in the general selection of the quadrature angles and associated weights. The most widely used quadrature sets are the so called level-symmetric quadrature, in which a rotationally invariant (around each axis) set of points are picked at intersecting levels based on the zeros of Legendre polynomials[10]. This can only be used up to order 20, since above this, negative weights appear which can result in un-physical solutions. Above order 20, typically Legendre-Chebyschev quadrature sets are used, which use the Legendre roots for the polar ordinates, and roots of Chebyschev polynomials for the azimuthal ordinates. Recently, new quadrature sets have been developed that can yield higher accuracy for a given number of points in the quadrature set, such as the icosahedral quadrature sets[12, 13] or sets based on linear discontinuous finite elements[14, 15]. While these advances can provide more efficient angular discretizations, they do not

directly impact this work since the method presented here can be used with any quadrature set, including these newly developed ones.

1.2.2 Angular Quadrature Refinement

The Regional Angular Refinement (RAR) technique was first developed by Longoni and Haghighat[16, 17, 18, 19] and incorporated into the PENTRAN transport code[20], where it is referred to as ordinate splitting. In the RAR method, one starts with a base quadrature set, from which one or more quadrature points are split into several additional points. This allows for the quadrature set to be very fine in some directions and very coarse in others. On problems with directional sources, such as x-ray beams, this can be very effective. This method has the drawback that it is manual, i.e., it requires the selection of quadrature and refinement to be performed *a priori* and by hand. A similar technique was also developed by Brown et al., called Locally Refined Quadrature[21].

More recent advances in local angular refinement have been the adaptive quadrature schemes developed by Stone[22, 23] in 2-D and extended to 3-D by Jarrell[14, 15]. In these schemes, the problem is first solved on a base quadrature set. Next, trial quadrature points are added, and the transport equation is solved along these ordinates using the source calculated from the base quadrature. This angular flux is compared to a flux that is interpolated between points on the base mesh. If the difference between these fluxes exceeds a tolerance, then this new quadrature point is added to the quadrature set. This method has the advantage of being completely automatic unlike the first methods, and can lead to a very efficient quadrature set. The disadvantage is that additional transport sweeps must be performed for every trial quadrature point. Further, the computational performance of these methods has not been established, as compared to standard methods.

1.2.3 Spatially Dependent Quadrature

Another method to optimize the solution is to allow for different angular quadrature sets to be used in different spatial regions of the problem. The angular fluxes on the boundary between regions is projected from one quadrature set to the other. This method allows higher order quadrature sets to only be used in areas where they are required, such as areas dominated by streaming, and low quadrature with low angular anisotropy, such as in diffusive media (e.g., carbon or water). However, the user is required to do this manually and *a priori*, which limits the usefulness on complicated problems. TORT[5] and TITAN[4] are two codes that allow for this.

1.2.4 First Collision Source

For fixed-source problems, the uncollided flux (i.e., the flux of particles that comes directly from the source and have yet to undergo a collision) is usually much more anisotropic than collided flux (i.e., the flux that has undergone at least one scattering event). This is because the uncollided flux comes from a source that is usually highly localized in space and/or angle. The collided flux, on the other hand, comes from the scattering of the uncollided, and is necessarily more spatially and angularly dispersed. Logically, one could use a high-order method to compute the very anisotropic uncollided flux, create the first-collision source, and use this to compute the collided flux using a lower-order method[24, 25]. The total flux is then the sum of the collided and uncollided fluxes. This can lead to a much more efficient calculation.

Several production discrete ordinates codes can use a semi-analytic ray-tracing method[9, 26, 27] as the high-order method to compute the uncollided flux, and couple this to the discrete ordinates method. The possibility of using a higher-order S_N calculation has been discussed[24], but has not been explored greatly. The first-collision source method can be

ineffective if the first-collision source is itself localized, as the collided flux will still have high quadrature requirements.

The new adaptive collision source (ACS) method developed in this dissertation builds off the first-collision source method by separating not just the uncollided flux, but also the once-collided flux, twice-collided flux, etc. At each transport iteration, only the i'th collided flux is solved for, and the possibility of using different angular quadrature orders for each is allowed. An intelligent scheme is also developed with which to choose, on-the-fly, an appropriate angular quadrature for each iteration. This can achieve the good speedups of the first-collision source method, but is more robust to a wide variety of problems for which the first collision source is not as effective.

1.3 Overview

The remainder of the document is organized as follows. Section 2 gives some background theory on the Linear Boltzmann Equation and the general steps taken to discretize and solve the deterministic problem in the discrete ordinates method. Section 3 discusses the new ACS methodology and its implementation into a full 3-D transport code. Section 4 gives and discusses the results that have been obtained using ACS and compared to a standard calculation for a variety of test problems. Section 5 gives a summary of the work completed and potential future work.

Chapter 2

Theory

2.1 Linear Boltzmann Equation

The steady-state LBE[1] is repeated again here, and will be discussed in more detail.

$$\begin{aligned}\hat{\Omega}\cdot\nabla\Psi(\vec{r},E,\Omega)+\sigma(\vec{r},E)\Psi(\vec{r},E,\Omega) =& Q_0(\vec{r},E,\Omega)+\int_0^\infty dE'\int_{4\pi} d\Omega'\sigma_s(\vec{r},E'\to E,\hat{\Omega}'\cdot\hat{\Omega})\Psi(\vec{r},E,\Omega)\\ &+\frac{\chi(E)}{4\pi}\int_0^\infty dE'\int_{4\pi} d\Omega'\nu(\vec{r},E)\sigma_f(\vec{r},E)\Psi(\vec{r},E,\Omega)\end{aligned} \tag{2.1}$$

Where,

- $\Psi(\vec{r},E,\Omega)$ is the angular flux. The quantity $\Psi(\vec{r},E,\Omega)\,d^3\vec{r}\,dE\,d\Omega$ represents the track length per second of particles with energy E within dE, traveling in direction Ω within $d\Omega$, and at location $\vec{r}$ within volume $d^3\vec{r}$.
- $\vec{r}$ is the spatial position vector (x,y,z)

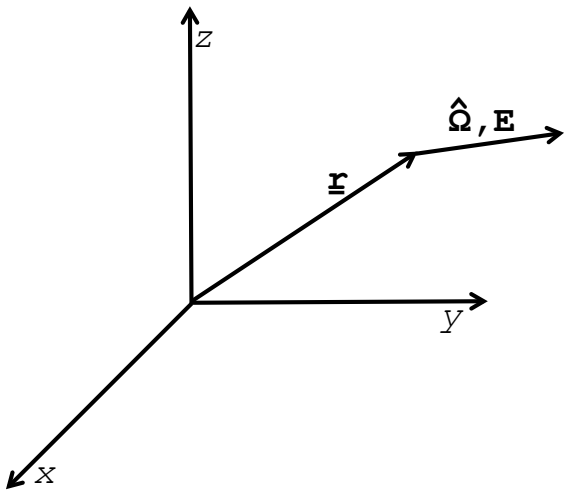


Figure 2.1 Diagram for the location ($\vec{r}$), direction ($\hat{\Omega}$) and energy (E) of the angular flux

- E is the particle energy
- $\hat{\Omega}$ is the unit vector in the direction of travel and is defined as shown by Eq. 2.2. Here, θ and ϕ are the standard polar and azimuthal angles, respectively.

$$\hat{\Omega} = (\mu, \eta, \xi) = (\cos\theta, \sin\theta\cos\phi, \sin\theta\sin\phi) \tag{2.2}$$

This phase space can be visualized in Figure 2.1. Here, the particle is at location $\vec{r}\,(x, y, z)$, traveling in direction $\hat{\Omega}$, with energy E.

- $\sigma(\vec{r}, E)$ is the material total cross-section, representing the probability per unit path length that a particle will have a collision.
- $\sigma_s(\vec{r}, E' \rightarrow E, \hat{\Omega}' \cdot \hat{\Omega})$ is the differential scattering cross section, representing the probability per unit path length that a particle will have a collision during which its angle and energy changes from (E', Ω') to (E, Ω) within the differential $dE\, d\Omega$.
- $Q_0(\vec{r}, E, \Omega)$ is the independent source of radiation.
- $\chi(E)$ is the fission neutron energy spectrum

- $\nu(\vec{r}, E)$ is the mean number of neutrons released per fission
- $\sigma_f(\vec{r}, E)$ is the fission cross-section

The LBE represents a balance equation in phase space ($d^3\vec{r}\,dE\,d\Omega$), with loss on the left side, and production on the right side.

Loss terms:

- $\hat{\Omega} \cdot \nabla\Psi(\vec{r}, E, \Omega)d^3\vec{r}\,dE\,d\Omega$ represents particles streaming out of (or into, if negative) the phase space.
- $\sigma(\vec{r}, E)\Psi(\vec{r}, E, \Omega)d^3\vec{r}\,dE\,d\Omega$ represents collisions that remove the particle from the phase space (either complete absorption or scattering to a different energy and/or angle).

Production terms:

- $\int_0^\infty dE' \int_{4\pi} d\Omega' \sigma_s(\vec{r}, E' \rightarrow E, \hat{\Omega}' \cdot \hat{\Omega})\Psi(\vec{r}, E, \Omega)d^3\vec{r}\,dE\,d\Omega$ represents particles from every other energy E' and angle Ω' within $dE'd\hat{\Omega}'$ scattering into the current phase space within $dEd\hat{\Omega}$ of $(E, \hat{\Omega})$ at location $\vec{r}$.
- $Q_0(\vec{r}, E, \Omega)\vec{r}$, is the independent source of radiation (e.g., radioactive decay) that produces produces particles directly into the phase space.
- $\frac{\chi(E)}{4\pi} \int_0^\infty dE' \int_{4\pi} d\Omega' \nu(\vec{r}, E)\sigma_f(\vec{r}, E)\Psi(\vec{r}, E, \Omega)dE$ is the number of fission neutrons with energy E produced from reactions integrated over all other energies.

An important quantity derived from the angular flux is the scalar flux ϕ, defined in Eq. 2.3.

$$\phi(\vec{r}, E) = \int_{4\pi} d\Omega \Psi(\vec{r}, E, \Omega) \tag{2.3}$$

An important extension of the scalar flux are the flux moments ϕ_l^k, defined in Equation 2.4, which are required for the standard method of describing the anisotropic scattering process.

$$\phi_l^k = \int_{4\pi} d\Omega Y_l^{*k}(\Omega)\Psi(\Omega) \tag{2.4}$$

Where, Y_l^{*k} is the complex conjugate of the standard spherical harmonic function[28].

In order to solve the LBE, we also need to know the boundary condition of Ψ. Generally the boundary condition only applies to the flux entering the volume, where the direction is denoted by $\Omega \cdot \hat{n} < 0$, where $\hat{n}$ is the normal vector to the boundary. The most common boundary condition is the non-reentrant or vacuum condition, on the boundary Γ. This means that there is zero flux entering the system, as in Eq. 2.5.

$$\Psi(\vec{r}, E, \Omega) = 0, \quad \Omega \cdot \hat{n} < 0, \vec{r} \in \Gamma \tag{2.5}$$

The other major boundary condition is albedo or reflective. This is defined as in Eq. 2.6.

$$\Psi(\vec{r}, E, \Omega) = \alpha(E)\Psi(\vec{r}, E, \Omega'), \quad \Omega \cdot \hat{n} < 0, \vec{r} \in \Gamma \tag{2.6}$$

Where, $\alpha(E)$ is called the albedo. The most common use is $\alpha = 1$ which indicates perfect reflection (i.e., symmetry). The angle Ω' is the reflection of Ω across the boundary, i.e., $\Omega \cdot \hat{n} = -\Omega' \cdot \hat{n}$.

2.2 Energy Discretization

To discretize the energy variable, the LBE is integrated over G energy groups with energy boundaries from low to high defined as $E_G, E_{g-1}, ..., E_1, E_0$. The g'th energy group is defined

as $\int_{E_g}^{E_{g-1}}$. This means that the highest energy is group 1, while the lowest energy is group G. This is done because in general, a particle undergoing a scattering interaction will lose energy, and thus move from low to high in energy group. This allows a single calculation loop to be done from $g = 1$ to G. Performing this group-wise integration on the LBE results in the multi-group form of the LBE (considering no fissionable material) in Eq. 2.7.

$$\hat{\Omega} \cdot \nabla \Psi_g(\vec{r}, \Omega) + \sigma_g \Psi_g(\vec{r}, \Omega) = Q_g(\vec{r}, E, \Omega) + \sum_{g'=1}^{G} \int_{4\pi} d\Omega' \sigma_{s,g' \to g}(\vec{r}, \hat{\Omega}' \cdot \hat{\Omega}) \Psi_g(\vec{r}, \Omega) \tag{2.7}$$

The group flux Ψ_g is defined in Eq. 2.8.

$$\Psi_g(\vec{r}, \Omega) = \int_{E_g}^{E_{g-1}} dE \Psi(\vec{r}, E, \Omega) \tag{2.8}$$

Similarly, the group source is defined in Eq. 2.9.

$$Q_g(\vec{r}, \Omega) = \int_{E_g}^{E_{g-1}} dE Q(\vec{r}, E, \Omega) \tag{2.9}$$

The group cross section σ_g is defined in Eq. 2.10.

$$\sigma_g(\vec{r}) = \frac{\int_{E_g}^{E_{g-1}} dE\, \sigma(\vec{r}, E) \phi(\vec{r}, E)}{\int_{E_g}^{E_{g-1}} dE\, \phi(\vec{r}, E)} \tag{2.10}$$

The group-to-group scattering cross section is defined in Eq. 2.11.

$$\sigma_{s,g' \to g}(\vec{r}, \hat{\Omega}' \cdot \hat{\Omega}) = \frac{\int_{E_{g'}}^{E_{g'-1}} dE' \int_{E_g}^{E_{g-1}} dE \sigma_{s,g' \to g}(\vec{r}, E' \to E, \hat{\Omega}' \cdot \hat{\Omega}) \Psi(\vec{r}, E', \hat{\Omega})}{\int_{E_{g'}}^{E_{g'-1}} dE\, \Psi(\vec{r}, E', \hat{\Omega})} \tag{2.11}$$

Generation of multi-group cross sections is problematic because it requires the flux spectrum $\phi(\vec{r}, E)$ (or a reasonable approximation thereof) to be known beforehand. The standard method for obtaining the spectrum is to perform a 1-D transport calculation using continuous-energy cross sections (or very fine multigroup), then use the resulting spectrum to create a multi-group library. This operation can be performed again on 2-D geometry in order to obtain cross sections that are accurate with only a few groups. There are several codes and methodologies available to automate these operations, such as TSUNAMI in the SCALE code package[29] and CPXSD[30, 31].

2.3 Angular Discretization

For the discrete ordinates method, consider that the LBE holds of a set of discrete angles (or ordinates). This allows an integral over angle to be replaced by a weighted sum over the discrete ordinates:

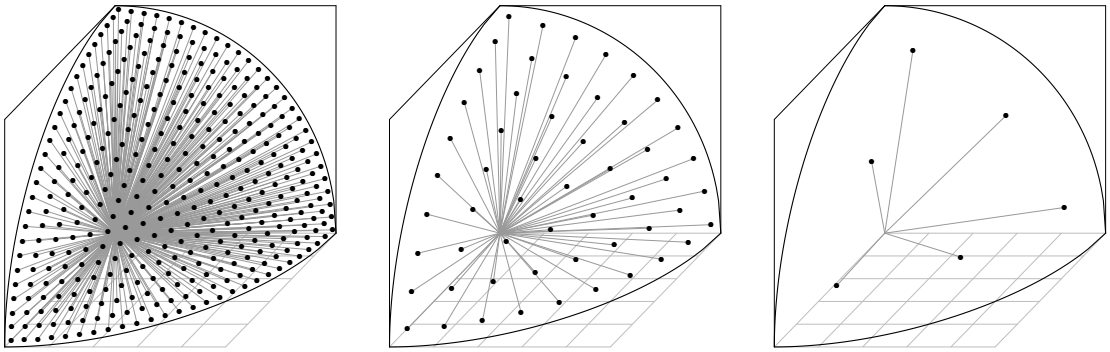
$$\int_{4\pi} d\Omega f(\Omega) = \sum_{m}^{M} w_m f(\Omega_m) \tag{2.12}$$

Where, m is the ordinate index, M is the total number of ordinates, Ω_m is the angle of the m'th ordinate and w_m is the weight of the m'th ordinate. The set of weights and angles (called a quadrature set) must be chosen carefully to ensure conservation of various integral properties such as current and scalar flux moments. Details on the construction and properties of quadrature sets may be found in other references[8, 10]. In this work, Legendre-Chebyschev quadrature sets are used[19, 18], denoted by S_N where N is called the quadrature order. The total number of ordinates M in the set is given by Eq. 2.13.

$$M = N(N+2) \tag{2.13}$$

Some sample Legendre-Chebyschev quadrature set ordinates are shown in Figure 2.2 (for a single octant of the unit sphere only). For example, the S_{20} quadrature set has $M =$

$20(20+2)/8 = 55$ ordinates in the octant shown, while S_6 has only 6. This can mean a huge difference in computation time between these sets.



(a) S_{54} **(b)** S_{20} **(c)** S_6

Figure 2.2 Ordinate directions in an octant for some Legendre-Chebyschev quadrature sets

The more anisotropic the angular flux becomes, the higher the order and thus more ordinates are required to be able to accurately calculate the integral quantities.

The scattering cross section is generally defined as an expansion in Legendre polynomials, truncated at order L. This is called the P_L expansion. The expansion coefficients $\sigma_{s,g\to g',l}$ are called scattering moments. An important thing to note about this is that the scattering probability only depends on the relative direction change $(\hat{\Omega}' \cdot \hat{\Omega})$ and not explicitly on the directions $\hat{\Omega}'$ and $\hat{\Omega}$.

$$\sigma_{s,g\to g'}(\vec{r}, \hat{\Omega}' \cdot \hat{\Omega}) = \sum_{l=0}^{L} \frac{(2l+1)}{4\pi} \sigma_{s,g\to g',l}(\vec{r}) P_l(\hat{\Omega}' \cdot \hat{\Omega}) \tag{2.14}$$

Other methods exist for the treatment of the angular variable, which involve an expansion of the angular dependence using a series of basis functions. If these basis functions are the spherical harmonics, this is called the P_N method[32, 33]. The other alternative is to use angular finite elements[34]. The discrete ordinates method has seen the most wide-spread adoption due to the ease of implementation, robustness and speed. Regardless of the method

used, all angular discretizations suffer from the fact that a finite number of points or functions will always have difficulty representing a highly anisotropic function like the angular flux.

2.4 Spatial Discretization

The multi-group, discrete-ordinates LBE in Cartesian geometry can be written in abbreviated form as in Eq.2.15

$$\left(\mu_m \frac{\partial}{\partial x} + \eta_m \frac{\partial}{\partial y} + \xi_m \frac{\partial}{\partial z} + \sigma_g(\vec{r})\right) \Psi_{m,g}(\vec{r}) = Q_{m,g}(\vec{r}) \tag{2.15}$$

Where, m is the angular index, g is the energy group index, and $(\mu_m, \eta_m, \xi_m) = \hat{\Omega}_m$ is the angular direction for index m. $Q_{m,g}$ represents the sum of the scattering, fission and independent sources.

For the space variable, one commonly used approach is to integrate the LBE over mesh cells (i.e., the finite volume method). This eliminates the derivative term ∇ from the LBE, which reduces to Eq. 2.16 when divided by the cell volume $\Delta x_i \Delta y_j \Delta z_k$.

$$\begin{aligned} &\frac{\mu_m}{\Delta x_i}\left(\Psi_{i+1/2,j,k} - \Psi_{i-1/2,j,k}\right) + \frac{\eta_m}{\Delta y_j}\left(\Psi_{i,j+1/2,k} - \Psi_{i,j-1/2,k}\right) + \\ &\frac{\xi_m}{\Delta z_k}\left(\Psi_{i,j,k+1/2} - \Psi_{i,j,k-1/2}\right) + \sigma_{i,j,k}\Psi_{i,j,k,m,g} = Q_{i,j,k,m,g} \end{aligned} \tag{2.16}$$

In this equation, (i, j, k) represents the (x, y, z) indices of a Cartesian mesh cell. The interfaces are denoted as $i + 1/2$, $i - 1/2$, etc. The cell average flux variable is $\Psi_{i,j,k,m,g}$, while new variables are introduced for the mesh interface incoming/outgoing fluxes (e.g., $\Psi_{i+1/2,j,k}$, $\Psi_{i-1/2,j,k}$. The cell-average flux variable is defined in Eq. 2.17.

$$\Psi_{i,j,k,m,g} = \frac{1}{\Delta x_i \Delta y_j \Delta z_k} \int_{z_{k-1/2}}^{z_{k+1/2}} dz \int_{y_{j-1/2}}^{y_{j+1/2}} dy \int_{x_{i-1/2}}^{x_{i+1/2}} dx \Psi_{m,g}(\vec{r}) \tag{2.17}$$

From this point on, the indices for angle and energy are omitted for brevity. This integration process adds the 3 outgoing fluxes as unknowns. The mesh incoming flux should be known from the boundary of the problem or from the outgoing flux of the adjacent cell. In order to close the set of equations, auxiliary equations are introduced that relate the interface fluxes to the cell average flux. A set of these auxiliary equations is known as a spatial differencing method. A standard class of differencing schemes are the weighted schemes, where the cell average flux is a weighted average of the incoming and outgoing fluxes, as in Eq. 2.18.

$$\begin{aligned} \Psi_{i,j,k} &= a_{i,j,k}\Psi_{i+1/2,j,k} + (1 - a_{i,j,k})\Psi_{i-1/2,j,k} \\ &= b_{i,j,k}\Psi_{i,j+1/2,k} + (1 - b_{i,j,k})\Psi_{i,j-1/2,k} \\ &= c_{i,j,k}\Psi_{i,j,k+1/2} + (1 - c_{i,j,k})\Psi_{i,j,k-1/2} \end{aligned} \tag{2.18}$$

Under such a scheme, the outgoing fluxes can be eliminated from Eq. 2.16 to yield the cell average flux as a function of the incoming fluxes as in Eq. 2.19.

$$\Psi_{i,j,k} = \frac{\frac{\mu_m}{a\Delta x}\Psi_{x,in} + \frac{\eta_m}{b\Delta y}\Psi_{y,in} + \frac{\xi_m}{c\Delta x}\Psi_{z,in} + Q_{i,j,k}}{\frac{\mu_m}{a\Delta x} + \frac{\eta_m}{b\Delta y} + \frac{\xi_m}{c\Delta x} + \sigma_{i,j,k}} \tag{2.19}$$

Where, $\Psi_{(x,y,z),in}$, represents the incoming flux from whichever (x, y, z) face is appropriate. This could be $i/j/k + 1/2$ or $i/j/k - 1/2$ depending on the direction of particle travel m. At this point, the cell average is known so the outgoing fluxes can be calculated from Eq. 2.18.

There are several options for the (a, b, c) coefficients. The simplest and most widely used is the Diamond Difference (DD) method[10] where $a = b = c = 1/2$. It is easy to implement, second-order accurate, and is also linear. It has the downside of occasionally producing negative outgoing fluxes and oscillations[35], both of which are un-physical. Typically, negative fluxes are avoided by setting the outgoing flux to zero and recalculating. This eliminates the negative fluxes, but lowers the accuracy, increases computation time, and makes the method non-linear.

Another weighted method is the step method, in which $a = b = c = 0$. This is always positive and linear, but with only first-order accuracy, and is rarely used. It is examined in this dissertation for some comparisons between methods.

A more sophisticated weighted method is the Directional Theta-Weighted (DTW) method[35, 36]. This is a non-linear method where the weighting coefficients a, b, c are functions of the angles (μ, η, ξ) and the fluxes themselves. This yields an always-positive scheme free from oscillations, but with higher truncation error than DD. The DTW method is implemented in TITAN, and will be examined in this dissertation.

Other differencing schemes include the step-characteristic[37], linear-characteristic[38], linear-discontinuous[39], exponential directional-weighted [40] and exponential directional iterative [41]. All of these offer their own sets of strengths and weaknesses, and the optimal method can vary problem to problem. It is possible to automatically use different methods in different regions of the problem, such as the adaptive differencing scheme in PENTRAN, where an appropriate method is chosen based on the local flux gradient. There are always trade-offs for higher accuracy for a given mesh size, be it increased computational effort or additional memory requirements.

2.5 Source Iteration

The LBE can be written in operator form as:

$$H\Psi = S\Psi + Q_0 \tag{2.20}$$

Where, the streaming-collision operator H is defined as in Eq. 2.21.

$$H = \hat{\Omega} \cdot \nabla + \sigma(\vec{r}, E) \tag{2.21}$$

The scattering operator S is defined in Eq. 2.22.

$$S = \int_0^\infty dE' \int_{4\pi} d\Omega \sigma_s(\vec{r}, E' \rightarrow E, \Omega' \cdot \Omega) \tag{2.22}$$

Both sides of Eq. 2.20 depend on Ψ and can't be solved directly, so in the standard source iteration method, an initial flux $\Psi^{(0)}$ (usually 0) is assumed, then the flux is calculated assuming a constant scattering source $S\Psi$, then the source is updated and the flux recalculated.

$$H\Psi^{(i)} = S\Psi^{(i-1)} + Q_0 \tag{2.23}$$

Where, $\Psi^{(i)}$ is the flux after iteration i. This operation, the solving of Eq 2.23, is commonly referred to as a transport sweep. After enough iterations, both sides of the equation will converge to within some given tolerance.

Chapter 3

Adaptive Collision Source Methodology

This section is divided as follows: first, the first-collision source method is discussed, followed by its extension to the new adaptive collision source (ACS) method[42, 43, 44]. For the ACS method, the general theory will be discussed followed by the iteration procedure, error estimation and the quadrature adaptation procedure. Finally, the implementation of the ACS algorithm into the TITAN transport code is outlined.

3.1 First Collision Source Method

To describe the first-collision source (FCS) method, start with Eq. 3.1, the Linear Boltzmann Equation (LBE).

$$\hat{\Omega} \cdot \nabla \Psi(\vec{r}, E, \Omega) + \sigma \Psi(\vec{r}, E, \Omega) = Q_0(\vec{r}, E, \Omega) + \int_0^\infty dE' \int_{4\pi} d\Omega' \sigma_s(\vec{r}, E' \rightarrow E, \hat{\Omega}' \cdot \hat{\Omega}) \Psi(\vec{r}, E, \Omega) \tag{3.1}$$

Now, the flux is split into the uncollided (Ψ_u) and collided (Ψ_c) fluxes, as in Eq. 3.2.

$$\Psi = \Psi_c + \Psi_u \tag{3.2}$$

If we insert this into Eq. 3.1, we can arrive at a set of coupled transport equations in Eq. 3.3 and Eq. 3.4

$$\hat{\Omega} \cdot \nabla \Psi_u(\vec{r}, E, \Omega) + \sigma \Psi_u(\vec{r}, E, \Omega) = Q_0(\vec{r}, E, \Omega) \tag{3.3}$$

$$\hat{\Omega} \cdot \nabla \Psi_c(\vec{r}, E, \Omega) + \sigma \Psi_c(\vec{r}, E, \Omega) = Q_c(\vec{r}, E, \Omega) + \int_0^\infty dE' \int_{4\pi} d\Omega' \sigma_s(\vec{r}, E' \rightarrow E, \hat{\Omega}' \cdot \hat{\Omega}) \Psi_c(\vec{r}, E, \Omega) \tag{3.4}$$

Where, the first-collision source Q_c is defined in Eq. 3.5

$$Q_c(\vec{r}, E, \Omega) = \int_0^\infty dE' \int_{4\pi} d\Omega' \sigma_s(\vec{r}, E' \rightarrow E, \hat{\Omega}' \cdot \hat{\Omega}) \Psi_u(\vec{r}, E, \Omega) \tag{3.5}$$

As you can see, Eq. 3.3 requires no iterations on the scattering source, and could be solved relatively quickly using a high-order method (e.g., high order S_N or ray-tracing). Eq. 3.4 looks exactly like the standard LBE, except the independent source Q_0 is replaced with the first-collision source from the un-collided flux. This collided flux can then be solved using a lower quadrature order S_N in a standard source iteration method relatively quickly. Again, the motivation behind this is that the un-collided flux has a much higher angular variation than the collided flux, so generally requires a more robust treatment of the angular variable than does the collided flux.

3.2 Adaptive Collision Source Method

The new ACS methodology builds on the first-collision source method to a type of n'th-collision source method using discrete ordinates. Instead of splitting the flux up into un-collided flux and collided flux, we expand the total flux (denoted as $\Psi_{t,n}$) into the fluxes of

different collision source order (i.e., 0 to n).

$$\Psi_{t,n} = \Psi_0 + \Psi_1 + \Psi_2 + ...\Psi_n \tag{3.6}$$

We truncate the series at the n'th collision. The uncollided flux and collided flux can be defined in terms of the i'th collided fluxes as follows:

$$\Psi_u = \Psi_0 \tag{3.7}$$

$$\Psi_c = \Psi_1 + \Psi_2 + ...\Psi_n \tag{3.8}$$

Now we arrive at a similar formulation as the first-collision source method:

$$\hat{\Omega} \cdot \nabla \Psi_i(\vec{r}, E, \Omega) + \sigma \Psi_i(\vec{r}, E, \Omega) = Q_i(\vec{r}, E, \Omega), \quad i = 1, 2, ..., n \tag{3.9}$$

For $i = 0$, Q_0 is the independent source for the problem. For $i > 0$, the i'th-collision source Q_i is defined as:

$$Q_i(\vec{r}, E, \Omega) = \int_0^\infty dE' \int_{4\pi} d\Omega' \sigma_s(\vec{r}, E' \rightarrow E, \hat{\Omega}' \cdot \hat{\Omega}) \Psi_{i-1}(\vec{r}, E, \Omega) \tag{3.10}$$

The idea for the ACS method is to use the discrete ordinates method to solve for each Ψ_i separately, then sum them to get the total flux $\Psi_{t,n}$ in Eq. 3.6. For the calculation of each Ψ_i, i.e., transport sweep, a different angular quadrature set can be used. Since we have separated each i'th collision flux from each other, Ψ_i depends only on the scattering source Q_i, which is dependent only on the flux moments $\phi^k_{l,i-1}$ (as defined in Eq. 2.4), and not explicitly on the angular flux Ψ_{i-1}. This means that the transport sweep for Ψ_i can be performed using a different quadrature set, since Q_i as computed is a continuous function in angle (since it is based on spherical harmonics, unlike Ψ_{i-1}) and can easily be evaluated at the new quadrature angles. The flux moments $\phi^k_{l,i-1}$ are all that need to be stored between iterations (at least for vacuum boundary conditions). Care must be taken to ensure that the

lowest quadrature order can still adequately integrate the highest order of spherical harmonic expansion.

It is important to note that Eq. 3.9 is similar to Eq. 2.1 except that the independent source is only included when $i = 0$. As such, this method can be fairly simply implemented into an existing transport code by including the independent source in only the first iteration and keeping track of the total flux sum.

3.2.1 Iteration Procedure

Recall the formulation for the general source iteration scheme:

$$H\Psi^{(i)} = S\Psi^{(i-1)} + Q_0 \quad \text{for } i = 0 \text{ to } n \tag{2.23}$$

The equations solved at each iteration for the Source Iteration (SI) procedure are shown in Table 3.1. At each iteration (or equivalently, number of scatters), the streaming-collision operator H is acting on the entire flux, and always with the same quadrature level.

Table 3.1 Iteration Progression for the Standard Source Iteration

# Scatters	Equation Solved	Quadrature Level
0	$H\Psi_0 = Q_0$	1
1	$H(\Psi_0 + \Psi_1) = Q_0 + S\Psi_0$	1
2	$H(\Psi_0 + \Psi_1 + \Psi_2) = Q_0 + S(\Psi_0 + \Psi_1)$	1
...	...	...
n	$H(\Psi_0 + \Psi_1 + ...\Psi_n) = Q_0 + S(\Psi_0 + ...\Psi_{n-1})$	1

The equations solved for the First Collision Source (FCS) method are shown in Table 3.2. Here, the uncollided flux (i.e., Ψ_0) is solved for using one quadrature level, while the remaining collided fluxes (Ψ_i) are combined and solved using a second quadrature level (of lower

order). Since Ψ_0 has both the highest magnitude and the most anisotropy, the high level quadrature can be used for it, while saving time using a lower quadrature for the rest.

Table 3.2 Iteration Progression for the First Collision Source Method

# Scatters	Equation Solved	Quadrature Level
0	$H\Psi_0 = Q_0$	1
1	$H\Psi_1 = S\Psi_0$	2
2	$H(\Psi_1 + \Psi_2) = S(\Psi_0 + \Psi_1)$	2
...	...	...
n	$H(\Psi_1 + ...\Psi_n) = S(\Psi_0 + ...\Psi_{n-1})$	2

Finally, the new ACS method is shown in Table 3.3. Every scattering order is solved for separately, potentially each with a different quadrature order (although for this example it is limited to three sets). In many problems, the first-collision source ($S\Psi_0$) can itself be localized and/or anisotropic, and thus still have high quadrature requirements. The ACS method gets around this problem by separating all collided source terms so that the lowest required quadrature can be used at every step. If this would introduce a high amount of error, then ACS will still operate at a high quadrature for the entire calculation, though this is unlikely.

Table 3.3 Iteration Progression for the Adaptive Collision Source Method

# Scatters	Equation Solved	Quadrature Level
0	$H\Psi_0 = Q_0$	1
1	$H\Psi_1 = S\Psi_0$	1 or 2
2	$H\Psi_2 = S\Psi_1$	1 or 2 or 3
...	...	...
n	$H\Psi_n = S\Psi_{n-1}$	1 or 2 or 3

3.2.2 Angular Quadrature Order Adaptation Criteria

In order to change quadrature for each scattering iteration, we need some criteria to decide when and how to do so. If we want to estimate the error produced by changing the quadrature from S_N to $S_{N'}$ at a certain iteration, there are two sources of error that we must consider. The first is the error in the flux of each iteration due to the quadrature change itself, which we will denote as quadrature error or ϵ_{quad}. The second is the iterative error, denoted as ϵ_{iter}, which represents the cumulative amount of flux that has yet to be calculated through the scattering iterations.

We can define the relative angular quadrature error from changing quadrature from order N to N' in Eq 3.11.

$$\epsilon_{quad} = \frac{\phi_i^{N'} - \phi_i^N}{\phi_i^N} \tag{3.11}$$

We define the relative iterative error at iteration i (as compared to an infinite number of iterations) in Eq. 3.12.

$$\epsilon_{iter} = \frac{\phi_{t,i} - \phi_{t,\infty}}{\phi_{t,\infty}} \tag{3.12}$$

The total error is defined in Eq. 3.13.

$$\epsilon_{total} = \frac{(\phi_0^N + ...\phi_i^N + \phi_{i+1}^{N'} + ...\phi_\infty^{N'}) - (\phi_0^N + ...\phi_\infty^N)}{\phi_0^N + ...\phi_\infty^N} \tag{3.13}$$

This can also be written as in Eq. 3.14.

$$\epsilon_{total} = \frac{(\phi_{t,i}^N + \phi_{t,\infty}^{N'} - \phi_{t,i}^{N'}) - \phi_{t,\infty}^N}{\phi_{t,\infty}^N} \tag{3.14}$$

Since the source is spreading out and becoming more isotropic with each iteration, we can assume that the relative quadrature error only goes down with the number of iterations.

$$\epsilon_{quad,i+1} \leq \epsilon_{quad,i} \tag{3.15}$$

Using this bound, we can place an estimate on the upper bound of the total error from changing the quadrature, equal to the quadrature error per iteration (ϵ_{quad}) times the amount of flux yet to be calculated at the new quadrature (ϵ_{iter}), as in Eq. 3.16.

$$\epsilon_{total} \leq \epsilon_{quad} \cdot \epsilon_{iter} \tag{3.16}$$

Once we have this error estimate, then we have the criteria for changing quadrature (i.e., lower the quadrature in iteration $(i+1)$ if ($\epsilon_{total} < \epsilon$, for some user-set tolerance ϵ). Now we will discuss how we calculate the error estimates ϵ_{quad} and ϵ_{iter}.

3.2.2.1 Angular Quadrature Error

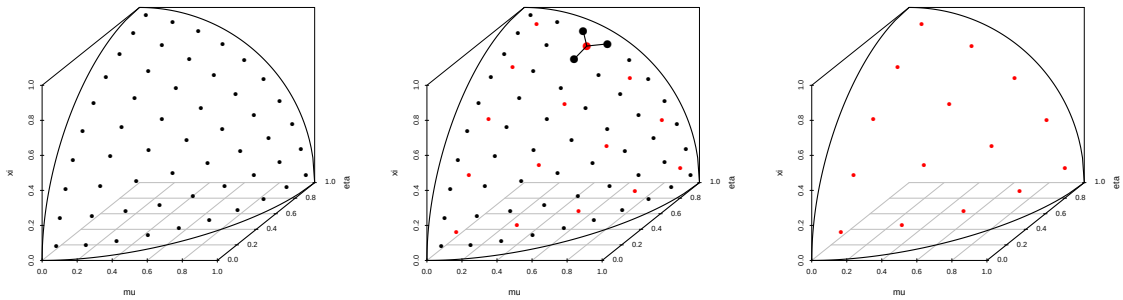
Since radiation from a source necessarily spreads out in space during free-flight, and then becomes spread out in angle during collisions, we can assume that the angular anisotropy (and hence angular quadrature requirement) is a strictly decreasing function of iterations. Using this principle, we can develop a simple technique to decide on what quadrature should be used. If we define the scalar flux (i.e., 0'th order flux moment) for Ψ_i using quadrature S_N:

$$\phi_i^N = \sum_{m=1}^{M} w_m \Psi_{i,m}^N \tag{3.17}$$

If we use an angular projection technique (such as that used in the TITAN code[45, 46, 4], we can project the flux Ψ_i^N onto a new (lower order) quadrature set $S_{N'}$: $\Psi_i^{N'}$. In TITAN, the projection is performed by making the flux in each ordinate of the new quadrature equal a weighted sum of the three nearest neighbours of the previous quadrature, as in Eq. 3.18.

$$\Psi_{i,m}^{N'} = \frac{1}{\frac{1}{\theta_1^2} + \frac{1}{\theta_2^2} + \frac{1}{\theta_3^2}} \left(\frac{\Psi_{i,n_1}^N}{\theta_1^2} + \frac{\Psi_{i,n_2}^N}{\theta_2^2} + \frac{\Psi_{i,n_3}^N}{\theta_3^2} \right) \tag{3.18}$$

Here, n_1, n_2, and n_3 are the nearest three ordinates to m, and θ_1, θ_2 and θ_3 are the respective angles between these ordinates and m. To illustrate this neighbor projection, consider the projection of S_{20} (Figure 3.1a) onto S_{10} (Figure 3.1c). Figure 3.1c shows the nearest neighbors in S_{20} of an ordinate in S_{10} for the projection process. This would be repeated for all ordinates in S_{10}.



(a) S_{20} **(b)** Nearest neighbors **(c)** S_{10}

Figure 3.1 Projection between angular quadrature sets in TITAN

We can calculate the scalar flux using this new angular flux:

$$\phi_i^{N'} = \sum_{m=1}^{M} w_m \Psi_{i,m}^{N'} \tag{3.19}$$

Now we have $\phi_i^{N'}$ and ϕ_i^{N} to use in Eq. 3.11. Note that during the TITAN projection process, the new angular flux would normally be scaled so that scalar flux is conserved (which would make $\epsilon_{quad} = 0$). This scaling is disabled for the error calculation (in fact, the conservation factor is essentially the same as the error estimate).

3.2.2.2 Iterative Error

A source iteration of the LBE should exhibit linear monotone convergence, since each iteration simply adds the next-collided flux, which is always less than the last (unless the system is super-critical). After a sufficient number of iterations, the flux in each successive iteration

should be decaying by a constant factor Λ (i.e., the spectral radius)[47] as in Eq. 3.20.

$$\phi_{i+1} = \Lambda\phi_i \tag{3.20}$$

That Λ is the spectral radius can be more easily seen if we rewrite Eq. 3.20 in terms of the cumulative flux $\phi_{t,i}$.

$$\Lambda = \frac{|\phi_{t,i+1} - \phi_{t,i}|}{|\phi_{t,i} - \phi_{t,i-1}|} \tag{3.21}$$

If we sum up these flux values to infinity, we get an estimate of the remaining flux to be calculated (i.e., the iterative error).

$$\phi_{t,\infty} - \phi_{t,i} = \sum_{j=1}^{\infty} \Lambda^j \phi_i \tag{3.22}$$

This sum can be expressed as follows.

$$\phi_{t,\infty} - \phi_{t,i} = \frac{\Lambda_i}{1 - \Lambda_i}\phi_i \tag{3.23}$$

Where, Λ_i, the estimate of the spectral radius at iteration i, is defined as in Eq. 3.24.

$$\Lambda_i = \frac{\phi_i}{\phi_{i-1}} \tag{3.24}$$

So we can now calculate the iterative error ϵ_{iter} using Eq. 3.25.

$$\epsilon_{iter} = \frac{1}{1 - \phi_i/\phi_{i-1}} \tag{3.25}$$

Of course, these equations are only valid if $\phi_i/\phi_{i-1} < 1$. If this condition is not met, then the solution has not converged sufficiently, and the error is set to 100%.

3.2.3 Angular Quadrature Order Adaptation

To perform the angular quadrature adaptation, we describe a list of allowable quadrature orders for the problem (e.g., [S_{20}, S_{14}, S_{10}, S_6]). This set is arbitrary, user-defined, and the only restriction is that the order be decreasing. Ψ_0 (i.e., the un-collided flux) would be calculated using the first quadrature set (S_{20} in this case). The error estimate described above would then be used to estimate the effect of lowering the quadrature to the next level (S_{14}). If this is below the user-prescribed tolerance ϵ, then Ψ_1 would be calculated using this new lower quadrature set. This angular quadrature adaptation would continue until the lowest quadrature order is reached (S_6 in this example), which would then be used for the remainder of the calculation.

It makes intuitive sense to set the adaptive tolerance to be somewhere on the order of the iterative tolerance for a given problem. We will examine the effect of changing this ϵ parameter later in the paper.

3.3 Algorithm Implementation

3.3.1 TITAN

In order to implement this algorithm, the TITAN 3-D parallel hybrid particle transport code[46, 4, 45] was adapted. TITAN numerically solves the time-independent LBE using the discrete ordinates method, or the ray-tracing method, or a combination thereof. Only the discrete ordinates portion of the code is used here. TITAN has the built-in ability to use different quadrature orders in different regions and perform quadrature projection, so was an ideal choice for this work.

It is important to understand the basic flow of the TITAN code (which is very similar to most discrete ordinates transport codes). A flowchart outlining the standard source iteration

in TITAN for a mono-energetic fixed-source problem is shown in Figure 3.2. The section with the dotted red outline will be called the "inner iterations" in subsequent charts, for brevity. The inner iterations are the portion where the within-group scatter is iterated upon (i.e., $\phi_g \sigma_{s,g\to g}$). The group subscript g is omitted in this flowchart as it is only for a single energy group. The flux tolerance or "inner tolerance" is denoted as tol_ϕ.

3.3.2 Implementation of ACS Modifications

To understand the modifications required to implement the ACS algorithm, a flowchart for the single group case is shown in Figure 3.3. In this chart, q refers to the current quadrature level, and N_q refers to the maximum number of quadrature sets. For example, if there are two quadrature sets used in the problem, S_{10} and S_4, then $N_q = 2$, and $q = 1$ refers to S_{10} while $q = 2$ refers to S_4. As with the SI flowchart in Figure 3.2, the dotted lines identify the "inner iterations".

There are three primary differences from the SI flowchart, which are indicated by dashed red outlines. The first difference is the addition of the cumulative flux $\phi_t^{(i)}$ which is updated every iteration. The second is the block where the quadrature error is estimated and the quadrature order changed if necessary (the adaptive tolerance is denoted as ϵ. Finally, when the source is updated, it is set to be only the scattering source (as opposed to the standard source iteration, where the fixed source Q_0 is included as well). This is the change that separates the collision orders between each iteration. Other small changes include the output (the cumulative flux ϕ_t is the equivalent to ϕ from the source iteration), and the flux change calculation ($\Delta\phi^{(i)}$).

3.3.3 Multi-group Calculations

The previous flowcharts were for only a single energy group. If there are G energy groups with no up-scatter, then the source iteration in TITAN is shown in Figure 3.4. The inner

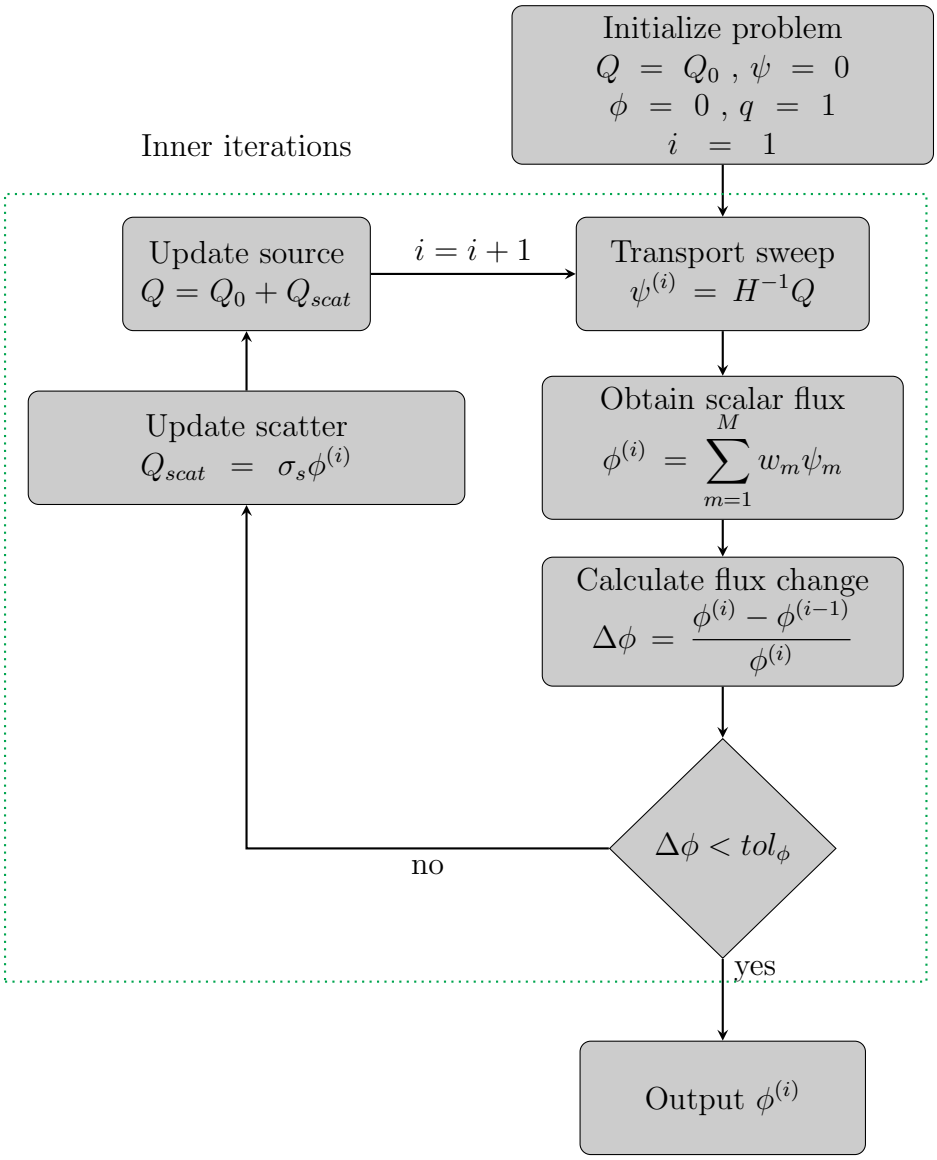


Figure 3.2 Source Iteration in the Standard TITAN Code

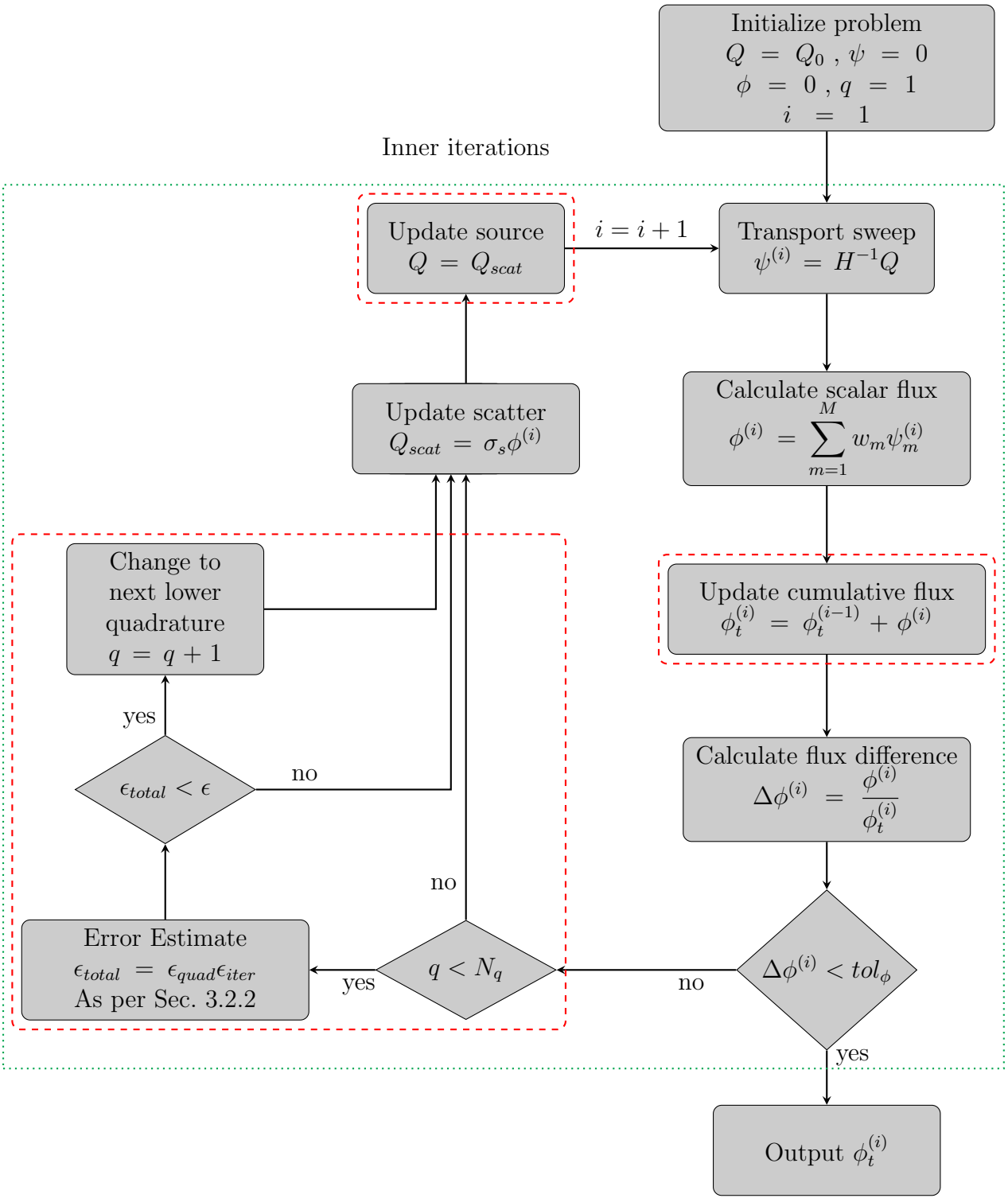


Figure 3.3 Adaptive Collision Source modification to the TITAN code

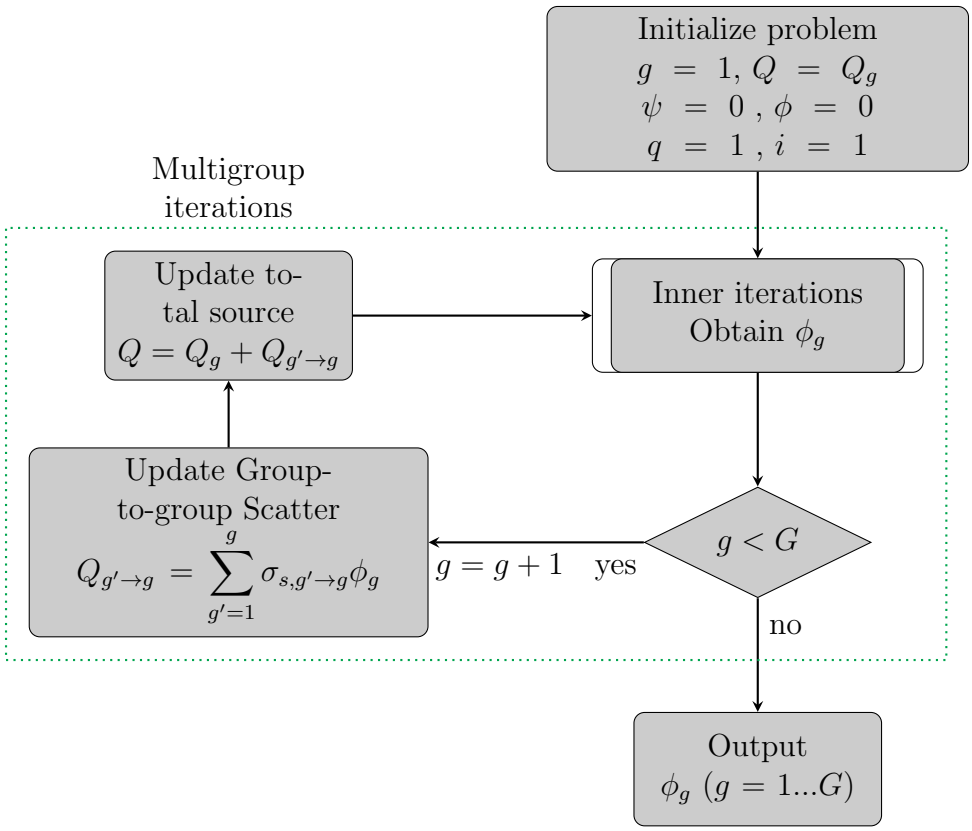


Figure 3.4 Source Iteration for G energy groups in TITAN

iterations as discussed from Figure 3.2 are shown as a single box. If up-scatter is present, then there would be an additional loop, but this is left out for brevity.

For multi-group calculations with ACS, there are not many modifications required. A flowchart is given in Figure 3.5. The first change is that the group-to-group scattering source $Q_{g' \to g}$ is calculated using the cumulative flux $\phi_{t,g}$. The other change is that the quadrature level is reset to the highest level before beginning the inner iterations on the next energy group. It is worth noting that with this implementation, the first iteration on energy groups lower than the first are not, strictly speaking, the uncollided flux. It is the uncollided flux from both the independent source Q_g and the group-to-group scatter $Q_{g' \to g}$. However, this is really just a semantic difference, and does not affect the performance of the method.

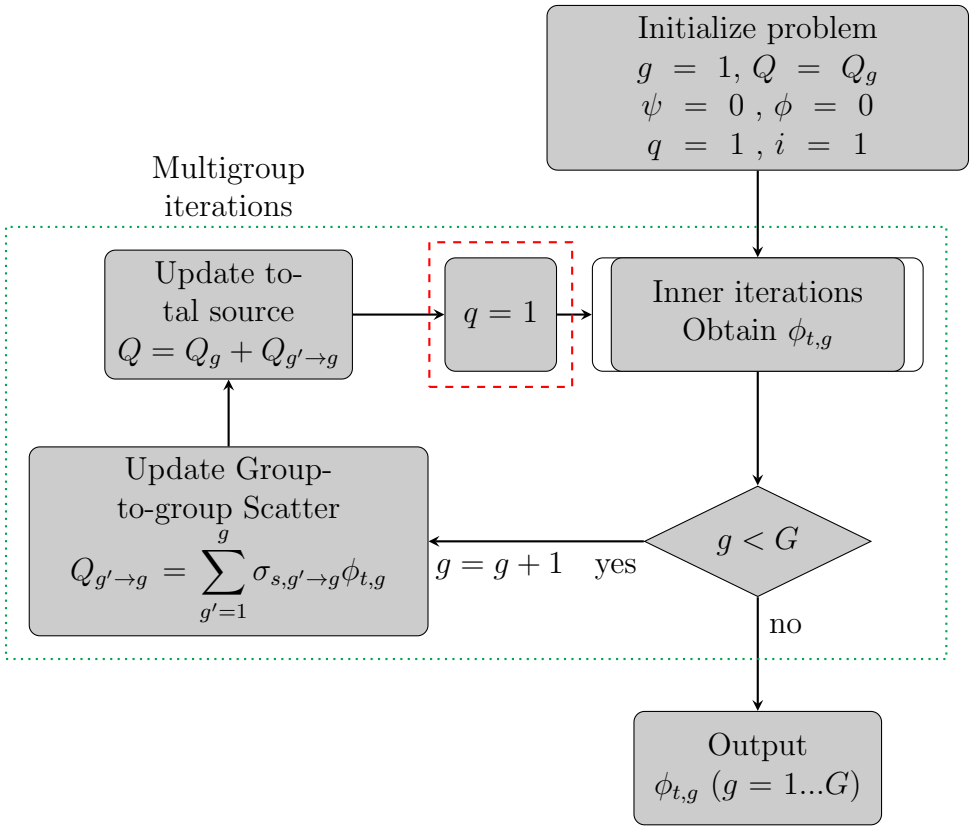


Figure 3.5 Adaptive Collision Source for G energy groups in TITAN

3.3.4 Eigenvalue Calculations

Eigenvalue calculations are another important type of radiation transport problem, in which the source is not independent, but depends on the flux, as in Eq. 3.26. Here, $Q_{f,g}$ is the fission source in group g, k is the eigenvalue, ν is the average number of neutrons per fission, $\sigma_{f,g'}$ is the fission cross section, and χ_g is the fission neutron energy spectrum.

$$Q_{f,g} = \frac{1}{k} \sum_{g'=1}^{G} \frac{1}{4\pi} \nu \sigma_{f,g'} \chi_g \phi_g \tag{3.26}$$

The general procedure for an eigenvalue calculation in standard TITAN is given in Figure 3.6. This eigenvalue procedure is the standard power iteration in linear algebra[48]. This power iteration process involves another loop, outside of the multi-group iterations. These outer loops are generally referred to as "outer iterations". The multi-group iterations from Figure 3.4 are shown as a single box.

The ACS eigenvalue calculation flowchart is shown in Figure 3.7. The procedure for the eigenvalue calculation has one small, yet major difference. When the total source is updated after an outer iteration, it does not include the scattering sources. If it included those, the collision source method would no longer be valid. This presents a practical problem, which is that for most eigenvalue problems, there is slow convergence of the scattering source for the thermal flux, so many iterations are needed. In the standard method, the scattering source is saved between outer iterations, whereas for the ACS method, they must be discarded and re-calculated. This will be a large factor in determining the efficiency of the method.

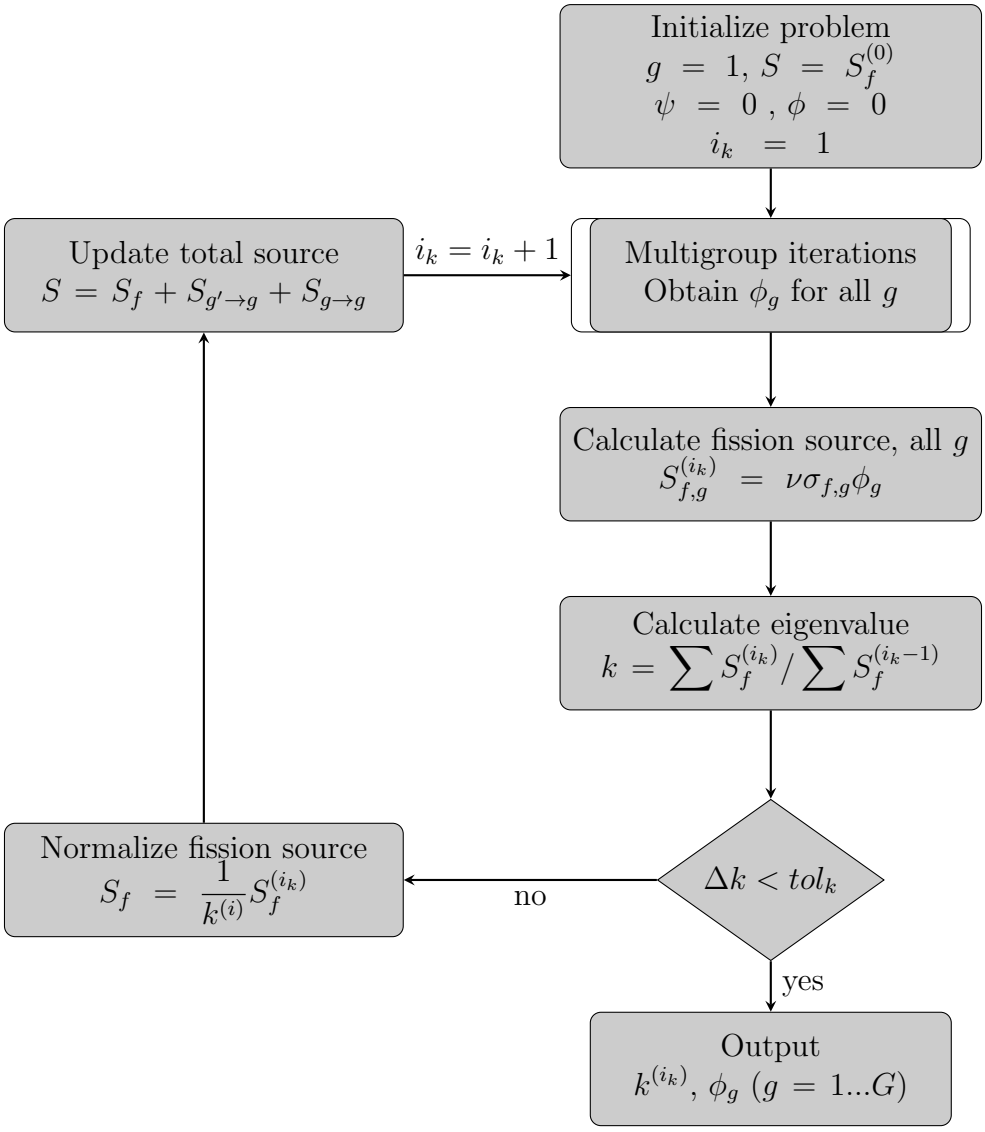


Figure 3.6 Source Iteration for eigenvalue problem in TITAN

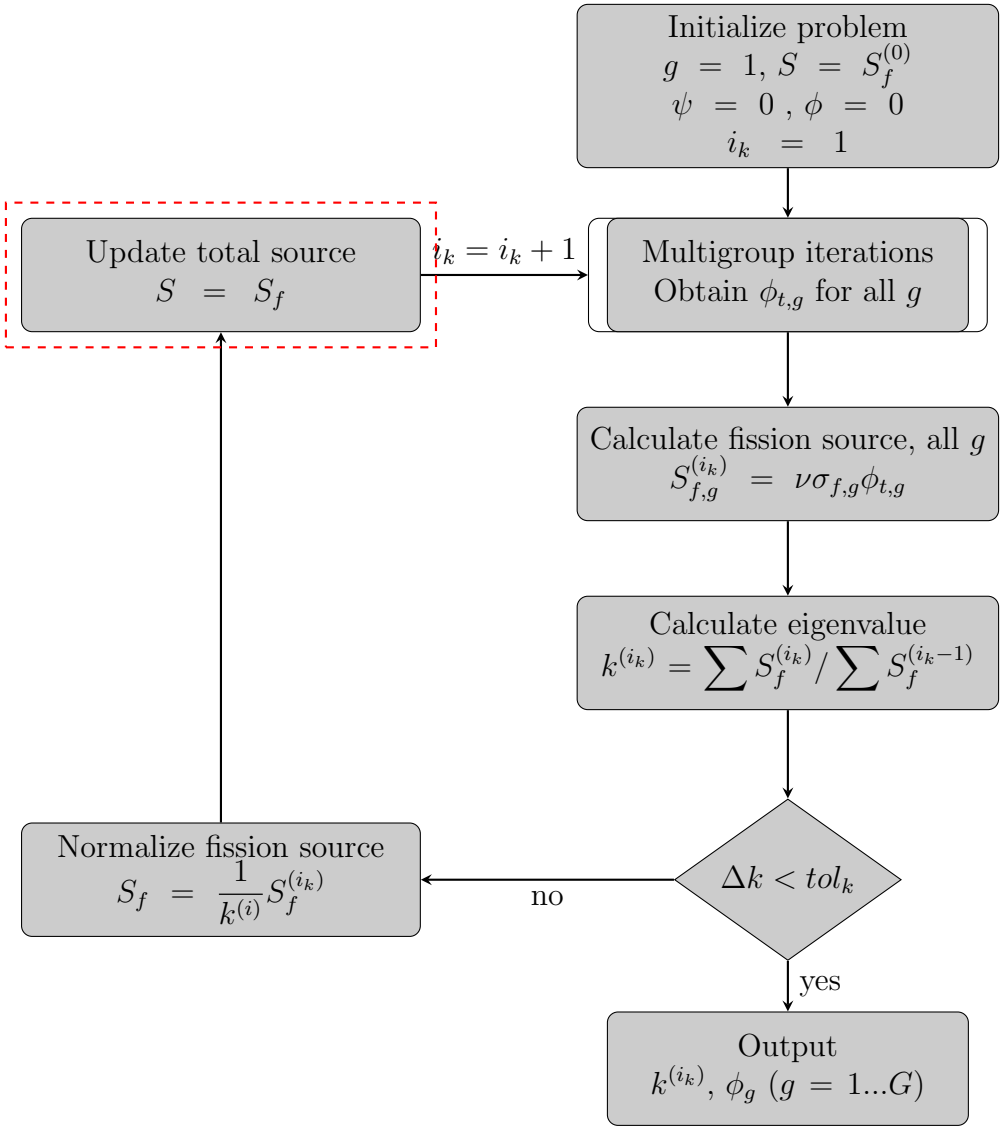


Figure 3.7 Adaptive Collision Source modifications flowchart for eigenvalue problem in TITAN

3.3.5 Parallelization

On large, realistic problems with the computational resources available today, it is generally a requirement of a modern algorithm to have good parallel performance. Good parallel performance will allow larger problems to be completed in a fraction of the time that would be possible on a single processor. Further, it allows for larger problems to be solved, due to the availability of more computer memory. The parallelization of the standard TITAN code is discussed, followed by the adaptations required for the ACS implementation in parallel.

3.3.5.1 Parallelization in TITAN

Parallelization in standard TITAN is done by angular decomposition of individual ordinates. This means that each processor is assigned a set of ordinates. With each iteration, transport sweeps are performed for each ordinate subset and the flux moments are calculated for the subset. At the end of each iteration, the flux moments from all subsets are combined and communicated between all processors using MPI[49] to get the total flux moments. In the TITAN decomposition, if reflective boundary conditions are used, then the number of processors is limited to the number of ordinates per octant. This allows boundary conditions to be applied without any parallel communication by requiring that all possible reflections of an ordinate to every other octant are stored on a single processor.

3.3.5.2 Parallelization in TITAN-ACS

With vacuum boundaries, the angular decomposition is done the same as TITAN, requiring communication of the flux moments (ϕ_l^k). When the quadrature order is changed, the new set of ordinates is re-allocated among the processors. This requires little extra computation and no parallel communication. However, with every iteration, the scalar flux error estimate (ϵ_{quad}) must also be communicated among processors. This increases communication time by a relatively small amount. With reflective boundaries, there is an added complication.

The quadrature of the boundary angular flux (Ψ_{bnd}) that is being reflected may change between iterations. This means that the boundary fluxes need to be combined in order to properly project the flux onto the next iteration. In the current implementation, the full boundary fluxes are communicated at the end of every iteration. This has the downside of always requiring communication, but the upside is that the previous TITAN restriction on the number of processors to be not more than the number of ordinates in an octant is removed.

The amount of parallel communication required in each iteration for the different methods is given in Table 3.4.

Table 3.4 Parallel Communication Required for TITAN and TITAN-ACS

Variable	Method	# Values Communicated
ϕ_l^k	TITAN and ACS	Meshes * $(L+1)^2$
ϵ_{quad}	ACS only	Meshes
Ψ_{bnd}	ACS only (Refl Bnd)	Bnd. Meshes * Ordinates / 2

Here, meshes means the total number of spatial meshes, boundary meshes is the number of spatial meshes located on the boundary, L is the scattering order (making $(L+1)^2$ the total number of flux moments), and ordinates is the number of ordinates in the quadrature set. The ACS method does add more parallel communication requirements, but this should be counteracted by improved efficiency of the method as a whole.

The parallel speedup SP on NP processors is defined as in Eq 3.27, where T_{serial} is the computation time on a single processor, while T_{NP} is the time on NP processors.

$$SP = \frac{T_{serial}}{T_{NP}} \tag{3.27}$$

The parallel efficiency is defined as in Eq. 3.28

$$PE = \frac{SP}{NP} \tag{3.28}$$

Using Amdahl's law[50], the maximum speedup on NP processors can be expressed as Eq 3.29, where f is the parallel fraction of the program (and thus $1 - f$ is the proportion of the program that is not parallelizible). Note that this equation assumes that the parallel communication time is very small compared to the computation time. This may not be a perfect assumption in all cases, but will be used for this work as it still provides a valuable metric.

$$SP = \frac{NP}{1 + f(NP - 1)} \tag{3.29}$$

The parallel fraction can be estimated from the observed speedup SP with Eq 3.30.

$$f = \frac{1/SP - 1}{1/NP - 1} \tag{3.30}$$

These can provide useful metrics with which to examine the parallel performance of a program, as will be discussed in Sec. 4.1.6.

Chapter 4

Results

Here, the performance of the ACS algorithm will be evaluated as compared to the standard uniform-quadrature source iteration in TITAN. In total, four problems will be examined, representing a variety of radiation transport problems. The first is a simple box-in-a-box problem, which due to its simplicity can be examined for a large variety of parameters. It will be examined for spatial and angular mesh convergence, parallel performance and evaluation of adaptive parameters. The next problem is a more realistic shielding problem - reactor dosimetry for the VENUS-2 reactor. The third and fourth problems are to test the criticality capabilities of the ACS method - the Takeda criticality benchmark and a Westinghouse pressurized water reactor (PWR) pin-cell calculation.

4.1 Box-in-a-Box

In order to perform initial tests on the algorithm, a very simple model is used, consisting of a cube of scattering material, with a small cube of source material in the middle. This simple model will allow the testing of various parameters without added difficulties of complex

geometry. A 3-D depiction of the problem is shown in Figure 4.1. The large box (blue) is 11cm· 11cm · 11cm, while the source box (red) is 1cm· 1cm · 1cm.

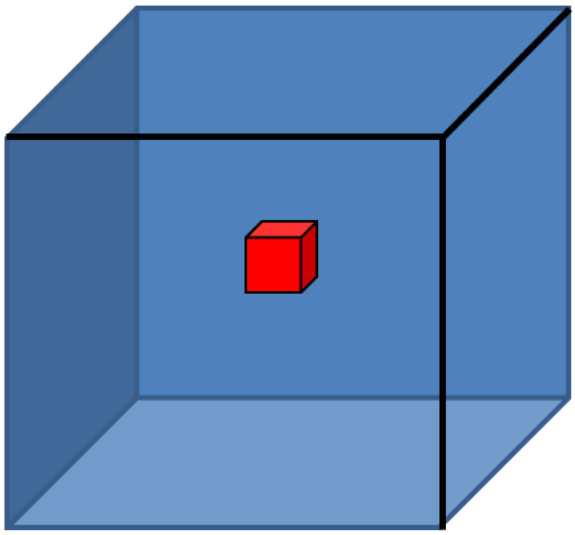


Figure 4.1 Box-in-a-box problem geometry

For simplicity, only one energy group was used, with total cross sections always equal to 1 cm^{-1}. Three materials were examined, with scattering ratios ($c = \sigma_s/\sigma_t$) of 0.1, 0.5 and 0.9. Full cross sections for these materials are given in Table 4.1.

Table 4.1 Cross Sections for Box Problem

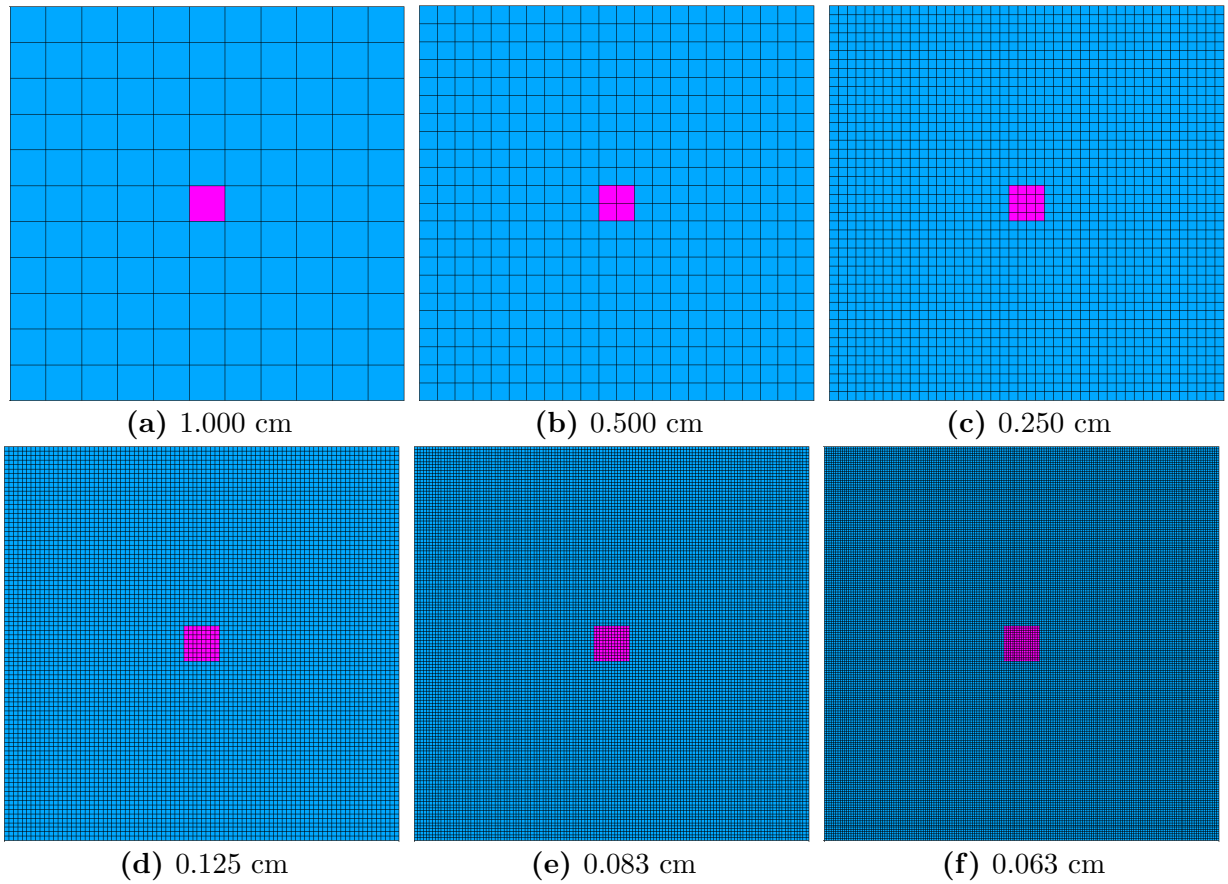
Material ID	Total σ_t (cm^{-1})	Absorption σ_a (cm^{-1})	Scatter σ_{s0} (cm^{-1})	P1 Scattering Moment σ_{s1} (cm^{-1})
1	1.0	0.9	0.1	0.1
2	1.0	0.5	0.5	0.5
3	1.0	0.1	0.9	0.9

In order to test the ACS method, there are many possible parameters to examine. These include quadrature order (including the first and second quadrature orders for ACS), spatial mesh size and adaptive tolerance (ϵ). These parameters and their possible values are shown in Table 4.2. The quadrature orders were selected so as to approximately double the number

of ordinates in each step. The mesh distributions are shown visually in Figure 4.2. The ordinates of the quadrature sets used here are shown in Figure 4.3.

Table 4.2 List of Parameters for Box Problem

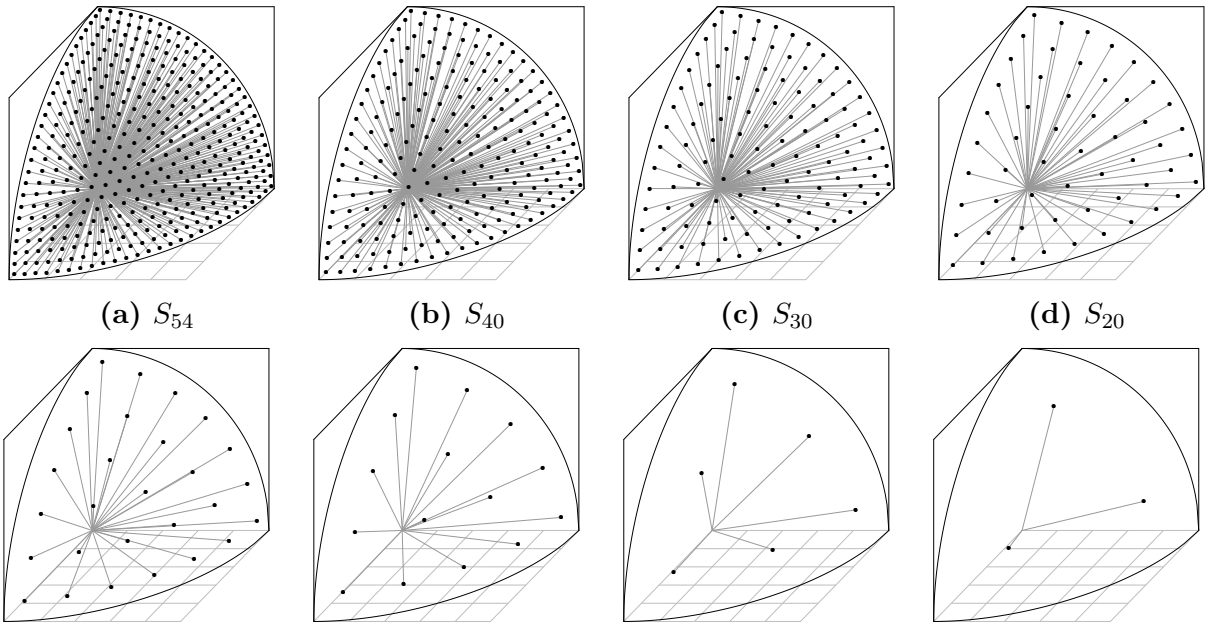
Quadrature Order (# of Ordinates)	Mesh Size (cm)	Adaptive Tolerance
4 (24)	1.000	10^{0}
6 (48)	0.500	10^{-1}
10 (120)	0.250	10^{-2}
14 (224)	0.125	10^{-3}
20 (440)	0.083	10^{-4}
30 (960)	0.063	
40 (1680)		
54 (3024)		



(a) 1.000 cm **(b)** 0.500 cm **(c)** 0.250 cm

(d) 0.125 cm **(e)** 0.083 cm **(f)** 0.063 cm

Figure 4.2 Mesh distributions for box problems at all mesh sizes



(a) S_{54} **(b)** S_{40} **(c)** S_{30} **(d)** S_{20}

(e) S_{14} **(f)** S_{10} **(g)** S_6 **(h)** S_4

Figure 4.3 Ordinate directions in an octant for all quadrature orders used in the box problem

4.1.1 Reference TITAN Calculations

TITAN calculations were performed for all mesh sizes and quadrature orders. Plots of the scalar flux for the finest mesh over all quadratures are shown in Figures 4.4-4.6 for materials 1-3, with scattering ratios of 0.1, 0.5 and 0.9, respectively. The angular discretization errors, seen as "ray effects" are easily visible at the lower quadrature levels. This is more pronounced in streaming-dominated material (i.e., material 1, $c = 0.1$), and much less pronounced in highly scattering material (material 3, $c = 0.9$).

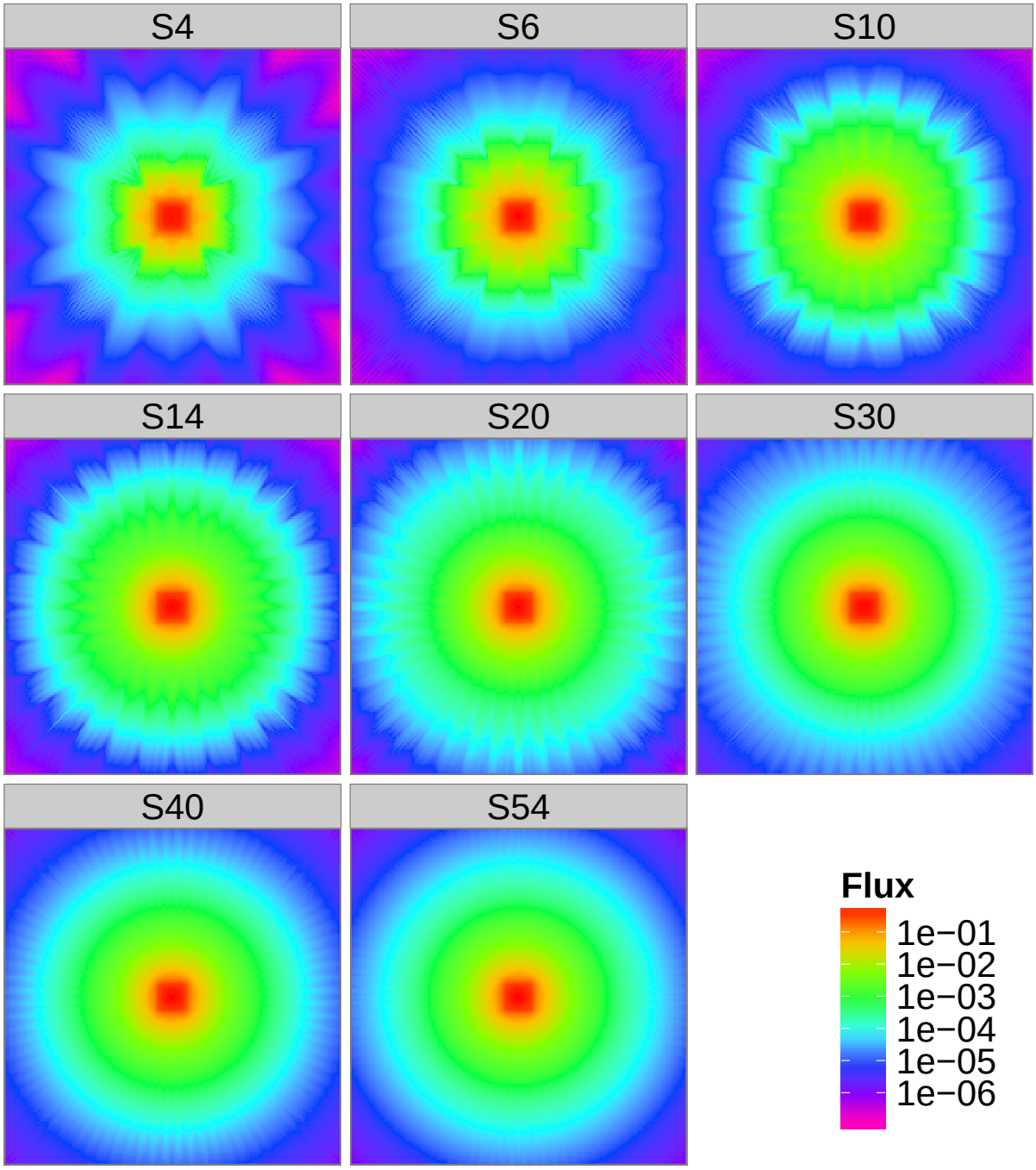


Figure 4.4 TITAN scalar flux plots for the box problem at different quadrature levels, material 1 ($c = 0.1$)

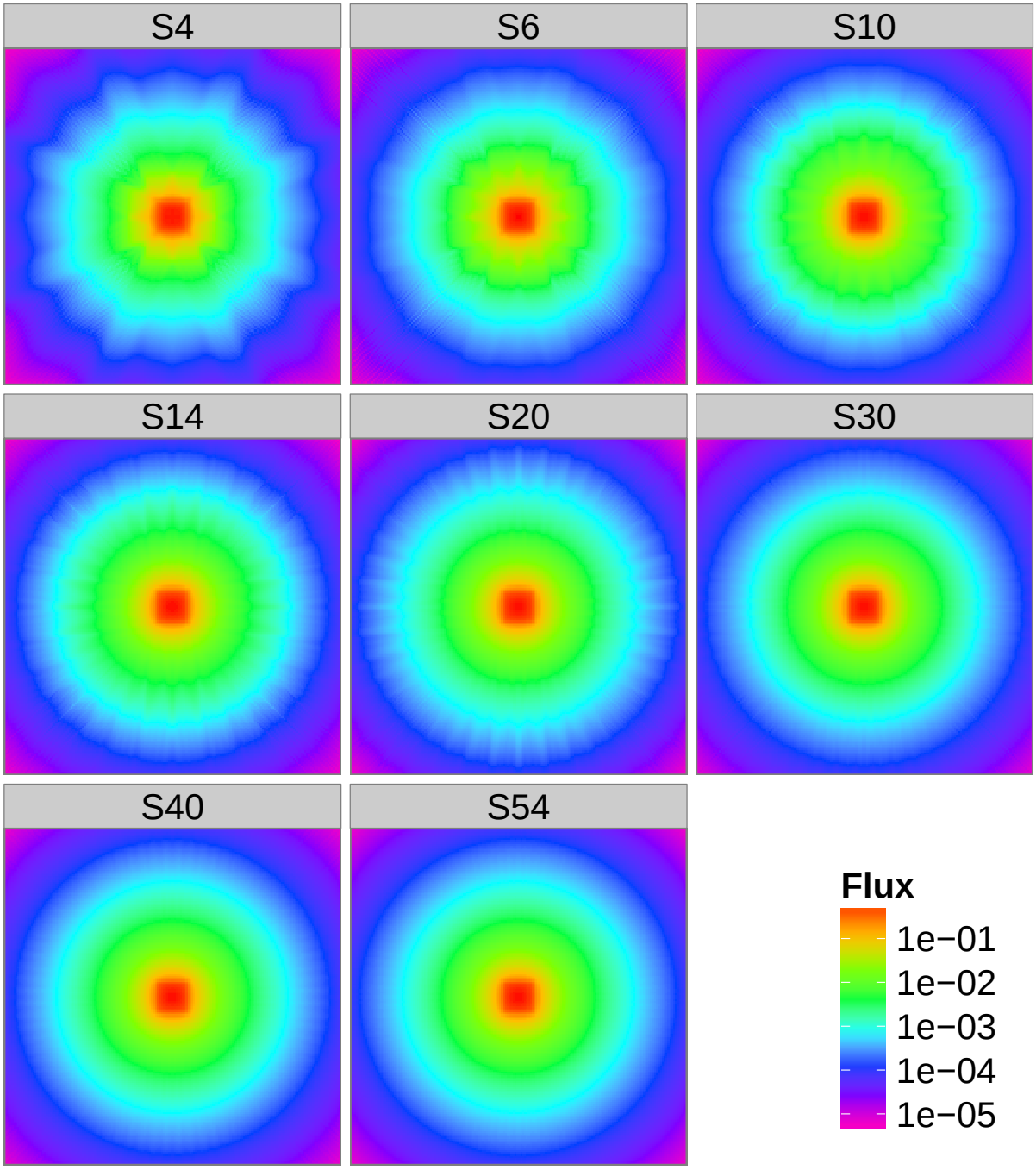


Figure 4.5 TITAN scalar flux plots for the box problem at different quadrature levels, material 2 ($c = 0.5$)

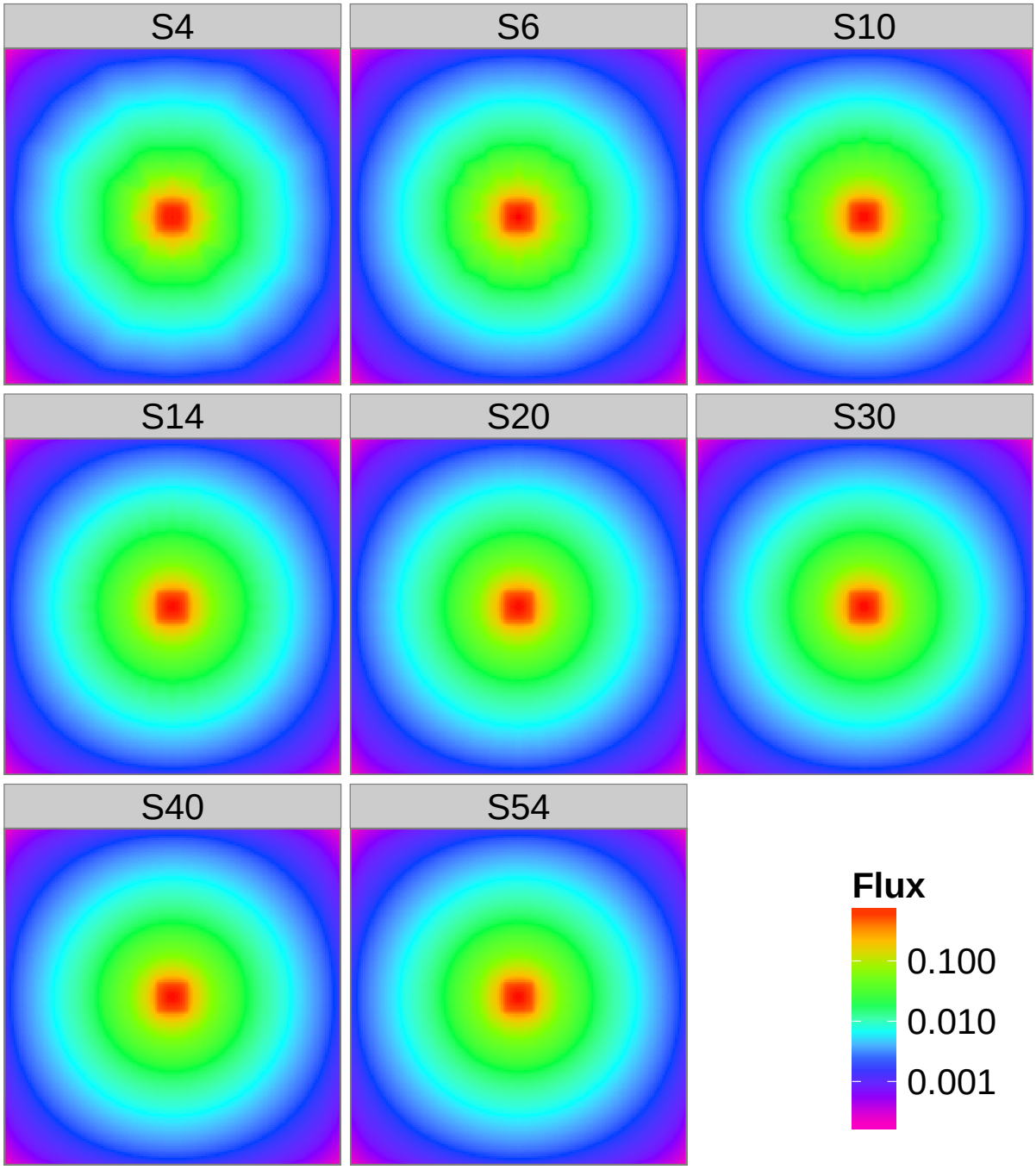


Figure 4.6 TITAN scalar flux plots for the box problem at different quadrature levels, material 3 ($c = 0.9$)

4.1.2 Spatial Discretization Convergence

There are many possible combinations of parameters that can be tested. In order to examine the differences between spatial discretizations, the scalar flux from the finer mesh distributions (i.e., < 1.0 cm) are averaged onto the 1.0 cm mesh. Similarly, to compare between angular discretizations, the scalar flux is used as the metric. As a metric of error, the root mean square relative error (RMSE) of the scalar flux is calculated, as compared to the reference case (standard source iteration, 0.063cm mesh size, S_{54} quadrature). The RMSE is defined in Eq. 4.1.

$$RMSE = \sqrt{\frac{1}{J}\sum_{j=1}^{J}\left(\frac{\phi_j - \phi_{j,ref}}{\phi_{j,ref}}\right)^2} \tag{4.1}$$

Where, j is the spatial index, J is the total number of spatial elements, ϕ_j is the scalar flux of the case in question, and $\phi_{j,ref}$ is the scalar flux of the reference case.

The first test that is performed is to test convergence of the flux under consistent mesh refinement (in both space and angle). This will prove that the spatial and angular discretization schemes are working properly and that ACS is giving the correct result.

To test spatial convergence, the quadrature is fixed to the highest (S_{54}) while the mesh size (h) is changed from highest to lowest. For ACS, S_{54}/S_{30} quadratures with $\epsilon = 10^{-4}$ is used. Diamond Difference (DD), Step and Directional Theta Weighted (DTW) schemes are used for spatial differencing. The flux from these calculations is then compared to the reference case.

Since the angular mesh is so refined, it is assumed that the error in the result is only from the spatial mesh. In the limit of small mesh size, then further refinement should result in the asymptotic behavior given in Eq. 4.2 [47].

$$Err = Ch^p \tag{4.2}$$

Where, C is a constant and p defines the order of the discretization method used, which is typically 1 or 2 for many common methods. If the error is plotted as a function of the mesh size, then a linear result would be expected on a log-log plot, where the slope is equal to the quadrature order. Log-log plots for the mesh refinement for the 3 materials are shown in Figures 4.7-4.9.

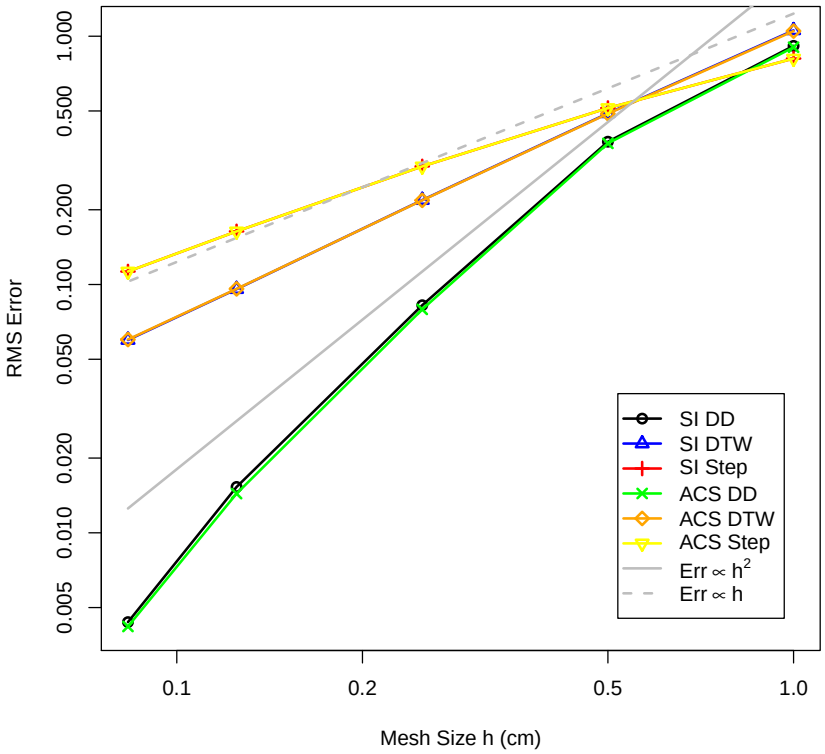


Figure 4.7 Spatial mesh convergence for cox problem material 1 (c=0.1)

It is apparent from these figures that both SI and ACS show similar convergence properties. The step and DTW methods show first order convergence (i.e., $p = 1$). The DD method shows second order convergence ($p = 2$). The error does not exactly fit the h^2 line because the reference case is calculated with a finite h, and has some appreciable discretization error itself. This means that the calculated error has some error itself, especially at small mesh

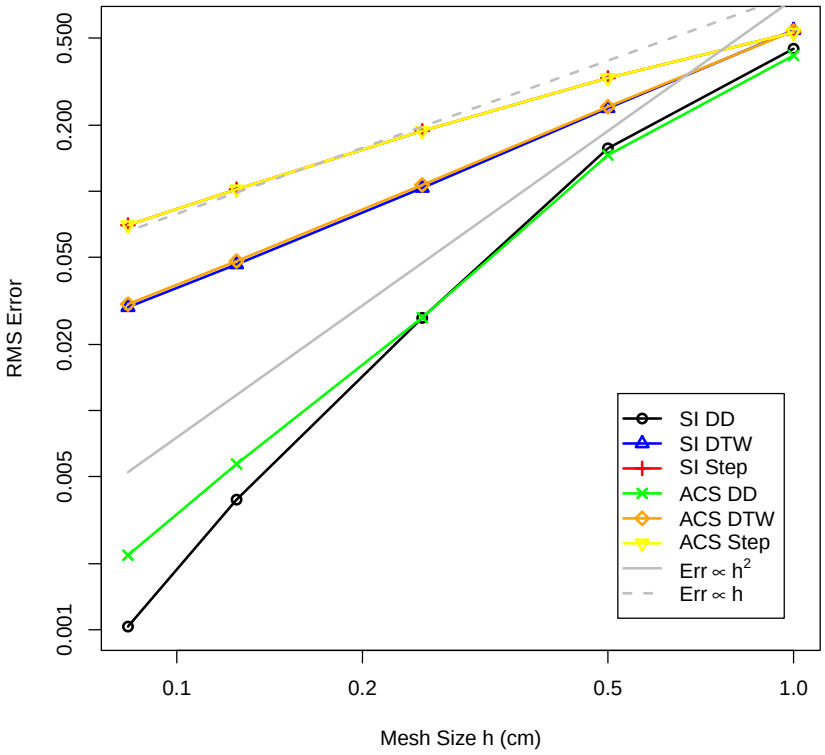


Figure 4.8 Spatial mesh convergence for cox problem material 2 (c=0.5)

sizes. Regardless, in comparison to SI, ACS converges similarly, despite the non-linearity of the spatial discretization operators.

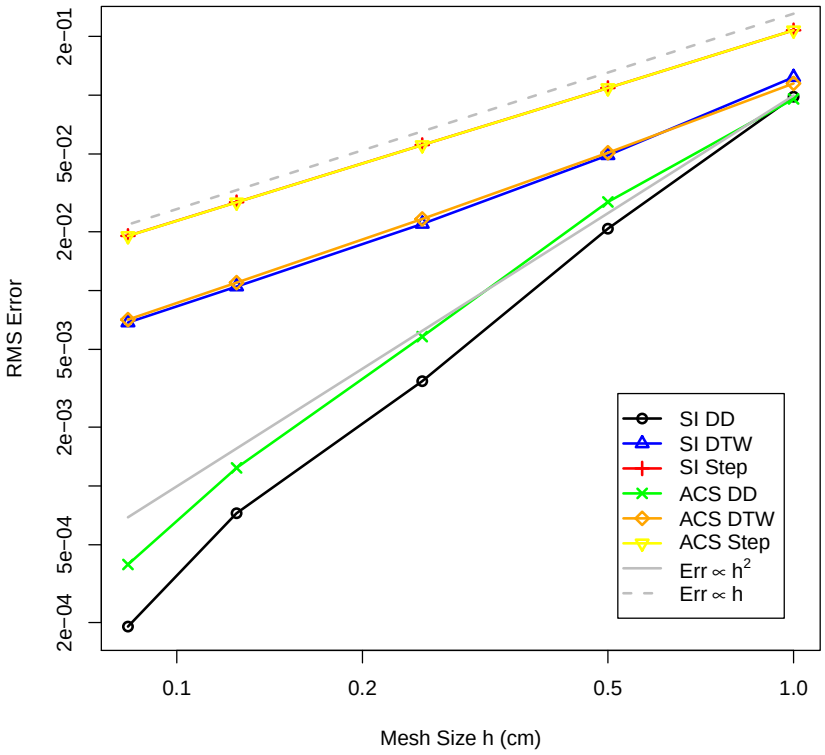


Figure 4.9 Spatialmesh convergence for cox problem material 3 (c=0.9)

4.1.3 Angular Discretization Convergence

The next step is to examine the convergence of the angular discretization. In this case, the spatial mesh size is fixed (at 0.063 cm), and the angular quadrature is varied between S_{30} and S_4. For the ACS cases, the secondary quadrature is chosen to be approximately half the order of the primary quadrature, as shown in Table 4.3.

The ACS method calculations were repeated for two cases, each with a different adaptive quadrature level, $\epsilon = 10^{-4}$ and $\epsilon = 1$. The $\epsilon = 1$ is essentially a first-collision source method, since the quadrature will always shift down after only the first iteration. For the "mesh size" of these calculations, we define this simply as $1/M$ where M is the number of ordinates in the quadrature set. Similar results to the spatial convergence results are expected again here. Convergence results for the three materials are shown in Figures 4.10-4.12.

Table 4.3 Primary and Secondary Quadrature for Angular Convergence of the ACS Method in the Box Problem

Quad. Order (# of ordinates)	
Primary	Secondary
30 (960)	14 (224)
20 (440)	10 (120)
14 (224)	6 (48)
10 (120)	4 (24)
8 (80)	4 (24)
6 (48)	4 (24)

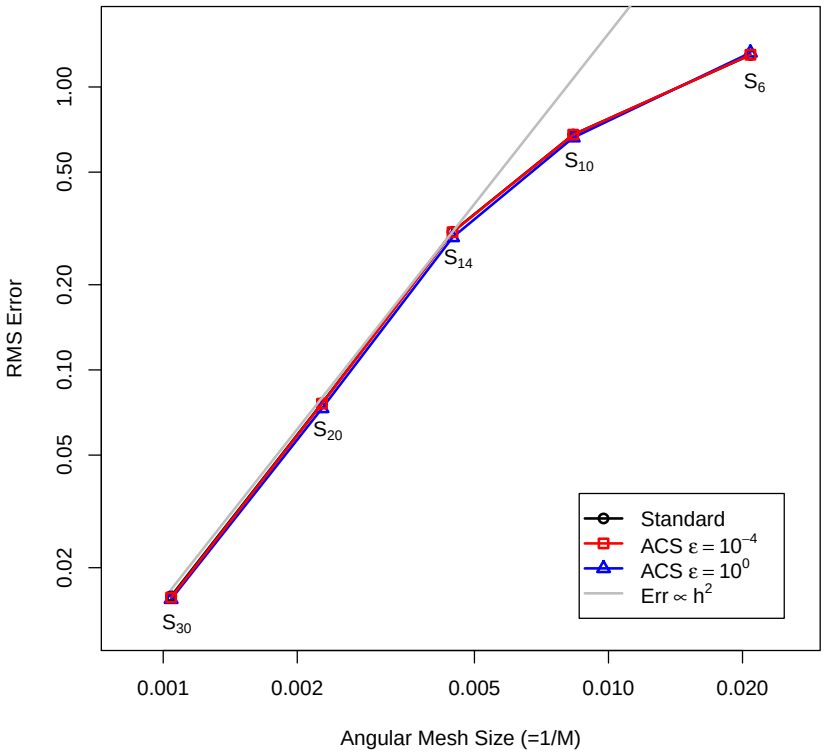


Figure 4.10 Angular mesh convergence for cox problem material 1 (c=0.1)

As seen by the figures, ACS again shows very similar convergence to the standard SI, but with a good computational speedup. ACS with $\epsilon = 1$, which is essentially a first-collision source method, works very well for the low-scattering problem, but quickly grows in error

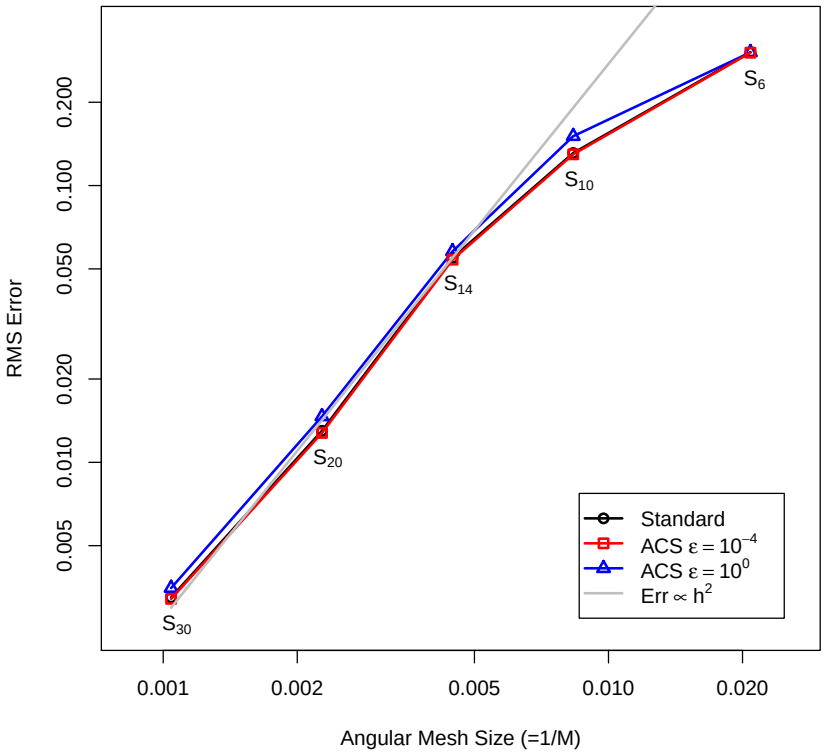


Figure 4.11 Angular mesh convergence for cox problem material 2 (c=0.5)

when more scattering is introduced. It is also apparent that a second-order convergence is achieved using the $P_N - T_N$ quadrature.

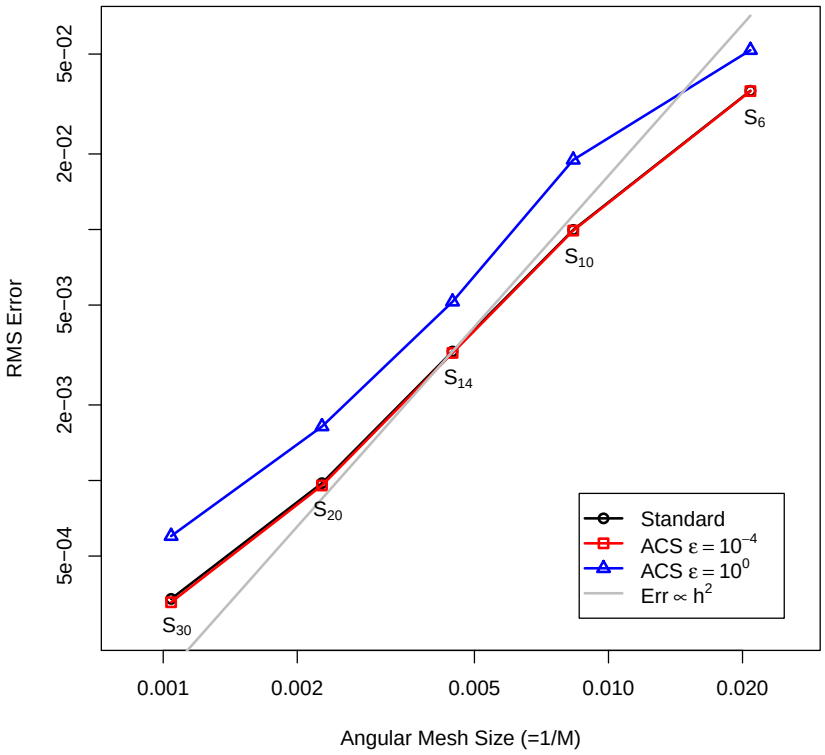


Figure 4.12 Angular mesh convergence for cox problem material 3 (c=0.9)

4.1.4 Effect of parameters on ACS efficiency

For any problem, there are trade-offs between accuracy and speed. Discretization error will always be present to some degree. The angular and spatial discretizations will affect the error in the result and the computation time. A finer discretization yields less error but slower computation time. When comparing methods, it will be important to compare computation time for a given level of error (or vice versa). The best method will be the one that minimizes computation time for a given level of error (or vice versa). This section will examine the computational efficiency of SI and ACS for a variety of parameters, including the adaptive tolerance, material scattering ratio, and quadrature orders. In order to compare the methods, the computation time will be plotted against the RMSE of that calculation for all parameters. For a result to be "good", the ACS results should appear to be below the standard SI results (that is, have a lower computation time for a given error).

In the first test, the initial ACS quadrature is set at S_{30}, and the second quadrature is varied from S_{20} to S_4, for various adaptive tolerances. This is compared to SI with quadrature fixed at between S_{30} and S_4. The results comparing computation time and MRE are shown in Figures 4.13-4.15. These results show that the ACS method for almost all parameters lies below the standard SI. That is, less computation time for a given error. The best results appear to be when the second quadrature is not too far from the first (at or above S_{10} for an S_{30} initial quadrature). If a high second quadrature is chosen, then there is relatively little gain in computation time. If a low second quadrature is chosen, then too much error is introduced. Choosing a "good" set of parameters is more of a factor when there is more scatter, since there is much more variation in the points in Figure 4.15 than in Figure 4.13. However, if "reasonable" parameters are chosen, a good speedup is usually obtained.

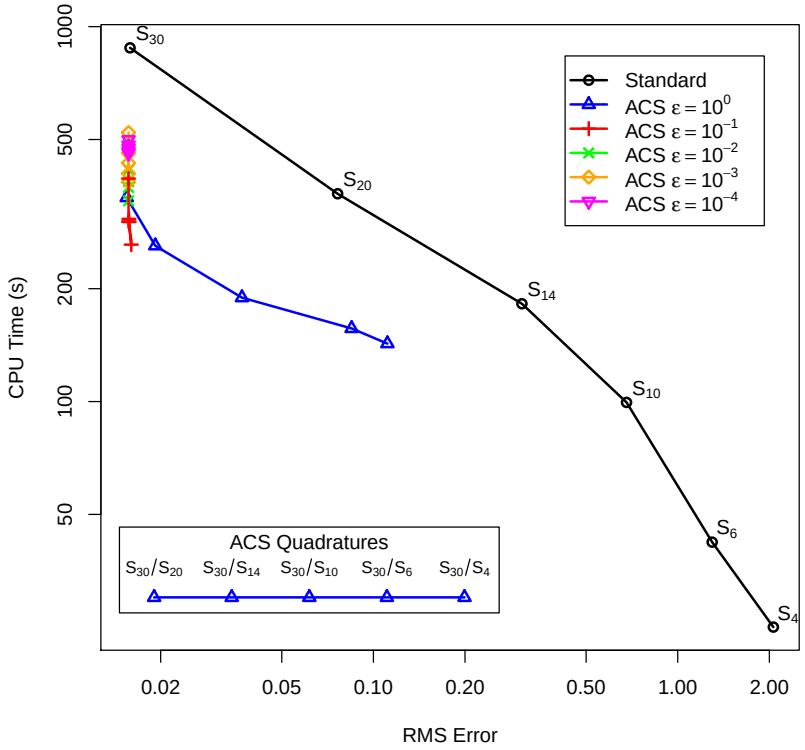


Figure 4.13 Effect of second quadrature on ACS efficiency for box material 1 (c=0.1)

For the next test, the first/second quadratures are set to fixed, reasonable levels (where the second quadrature order is approximately half of the first, as given in Table 4.3). The

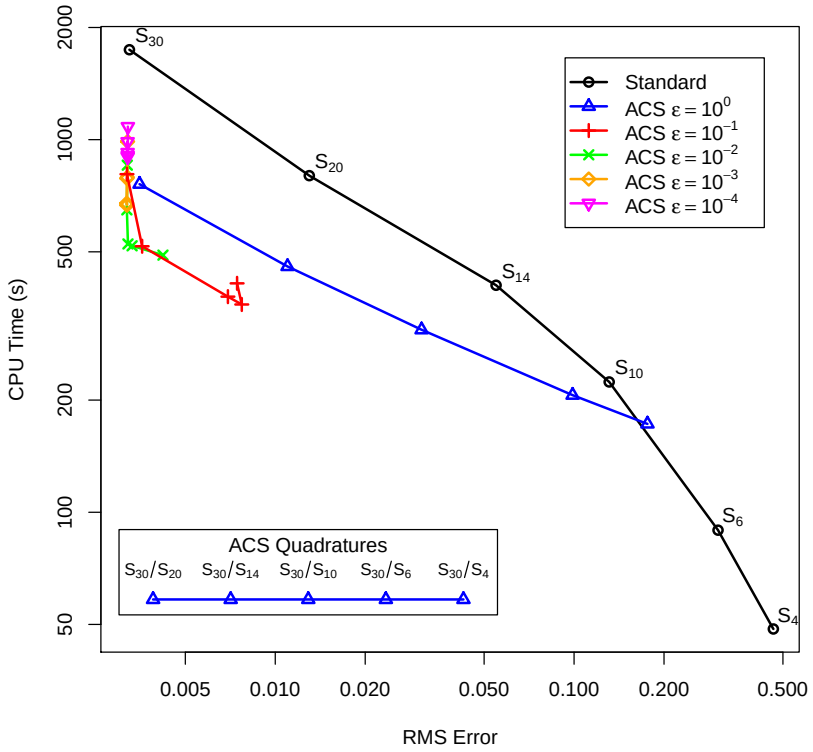


Figure 4.14 Effect of second quadrature on ACS efficiency for box material 2 (c=0.5)

adaptive tolerance ϵ is then varied for each of these combinations. These results are shown in Figures 4.16-4.18. These diagrams show that if a reasonable quadrature combination is chosen, then ACS always shows a good speedup.

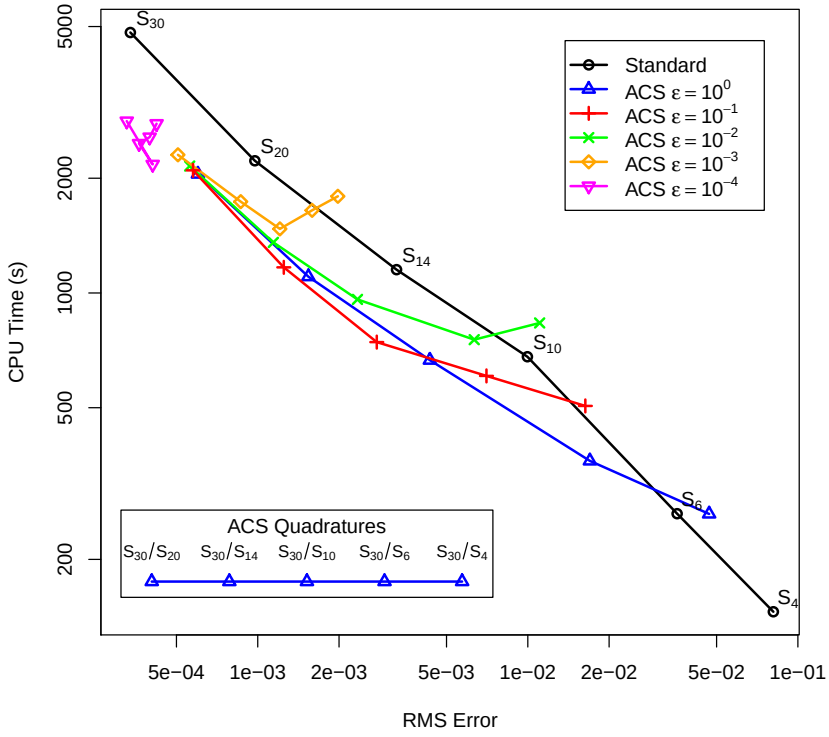


Figure 4.15 Effect of second quadrature on ACS efficiency for box material 3 (c=0.9)

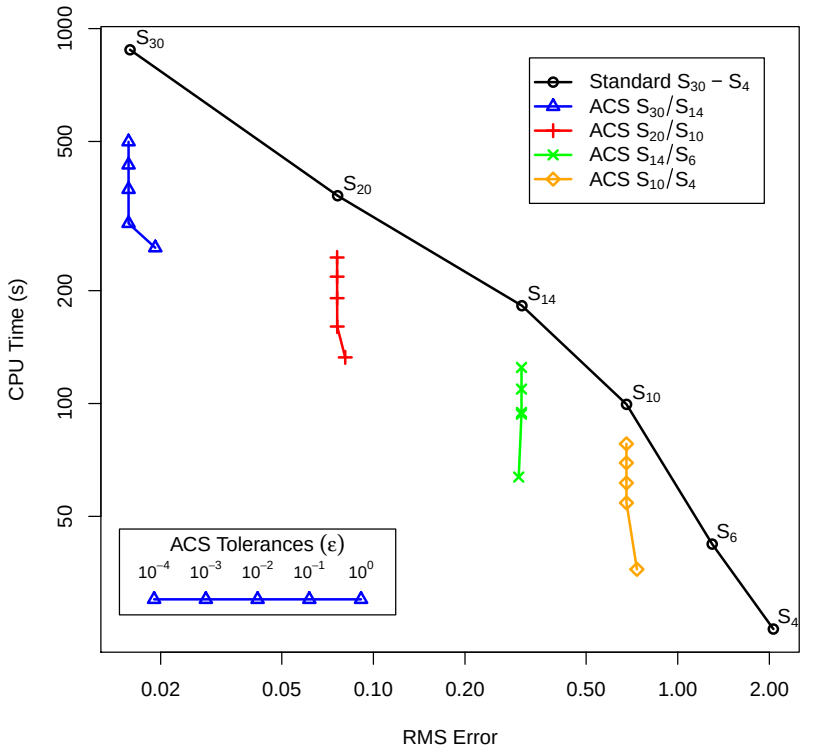


Figure 4.16 Effect of adaptive tolerance on ACS efficiency for box material 1 (c=0.1)

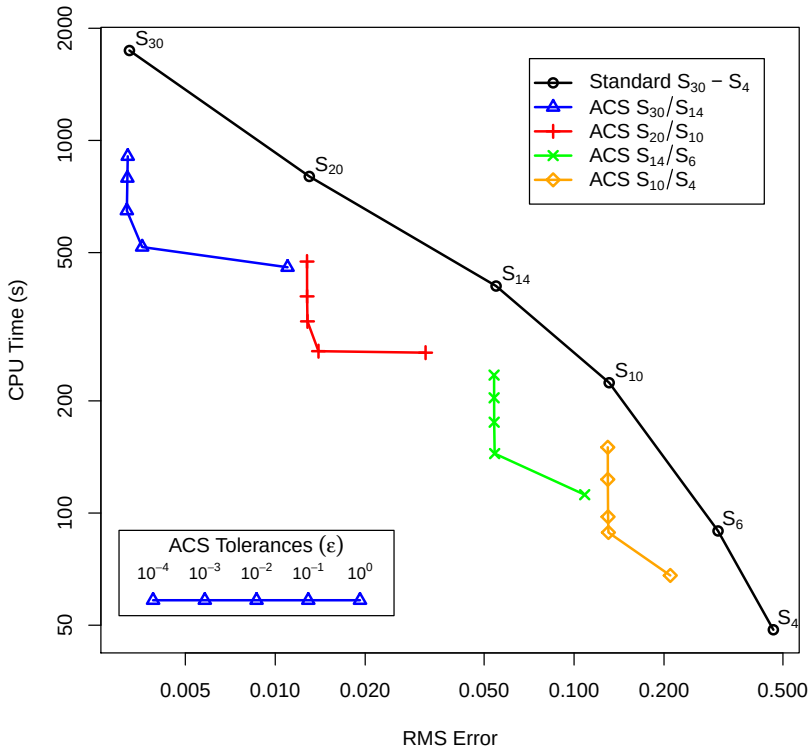


Figure 4.17 Effect of adaptive tolerance on ACS efficiency for box material 2 (c=0.5)

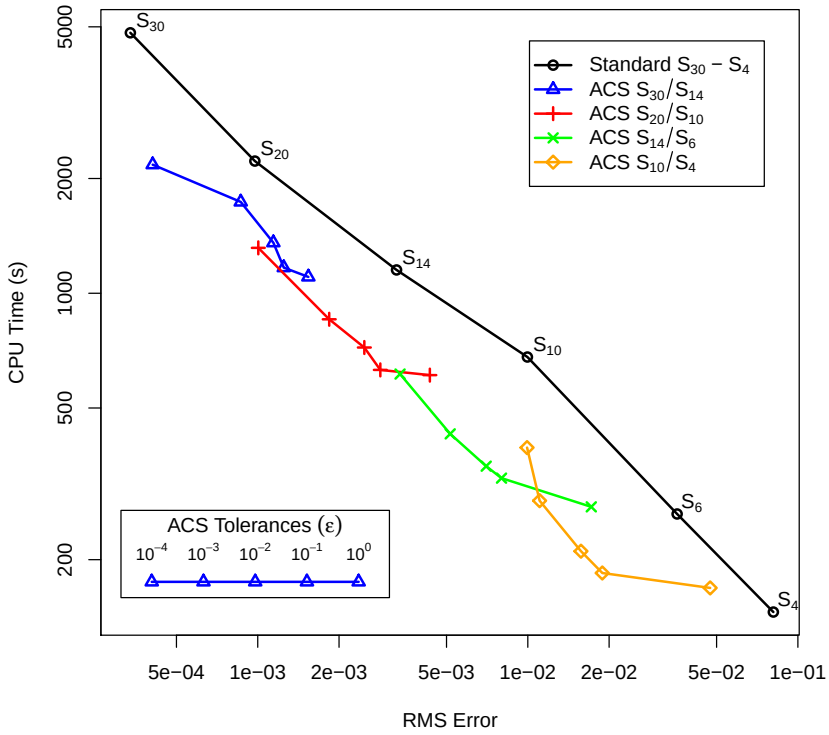


Figure 4.18 Effect of adaptive tolerance on ACS efficiency for box material 3 (c=0.9)

From these results, it's easy to tell that ACS is more efficient, but difficult how to tell by how much, since the calculations have different errors as well as computation times. In order to do a fair comparison, a new metric is introduced, which is termed the equal-error speedup (EES), and is defined for some computation a in Eq. 4.3.

$$EES_a = \frac{T_{SI}(RMSE_a)}{T_a} \tag{4.3}$$

In this equation, T_a and $RMSE_a$ are the computation time and RMSE for the calculation a in question. $T_{SI}(x)$ is the computation time for an SI calculation interpolated to the error value of x. It is equivalent to the black lines connecting the SI points in the Figures 4.13-4.18. By definition, all SI calculations will have $EES = 1$, and ACS calculations will have $EES > 1$ if they are more efficient, and $EES < 1$ if they are less efficient. The higher the EES of the method, the better. For example, an EES of 2 would indicate that for the same error, ACS performed 2 times faster than the standard method.

The calculated EES values for the fixed-quadrature-ratio cases (i.e., all the cases from Figures 4.16-4.18) are shown in Figures 4.19-4.21 for materials 1-3, respectively. For these figures, the shape of the point represents adaptive tolerance value, while the color indicates the primary quadrature order. From these plots, it is fairly clear that almost all the ACS calculations show an equal-error speedup of between 1.5 and 4. The optimal tolerance varies from case to case, but should certainly be less than 1. If high accuracy is required in highly scattering problems, then a low tolerance should be used. Another thing to note is that for the S_6 cases, there is not much speedup possible since the only option below S_6 is S_4, which is not very much faster. Regardless of parameters chosen, ACS can get an answer with the same level of error in a much shorter amount of time.

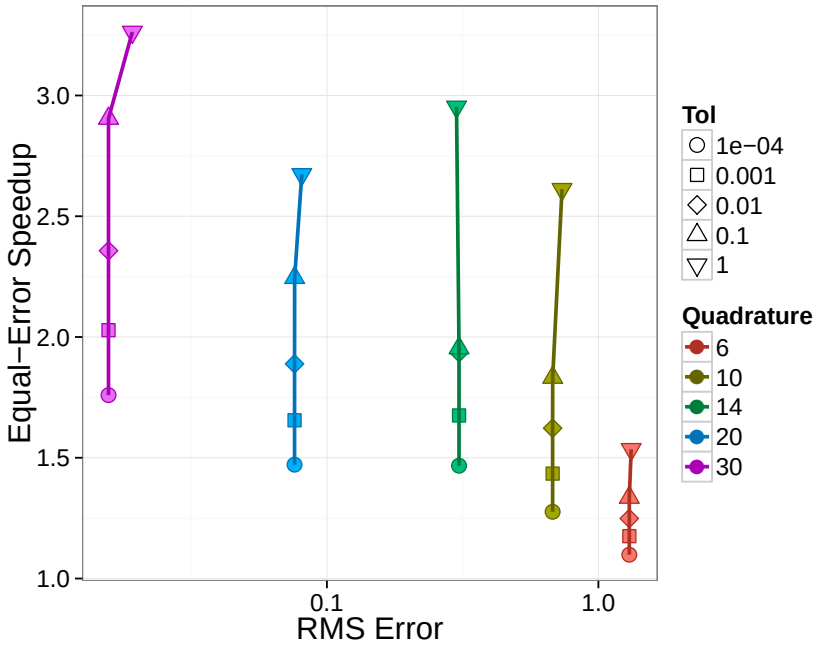


Figure 4.19 Equal-Error Speedup for all ACS parameters on box material 1 (c=0.1)

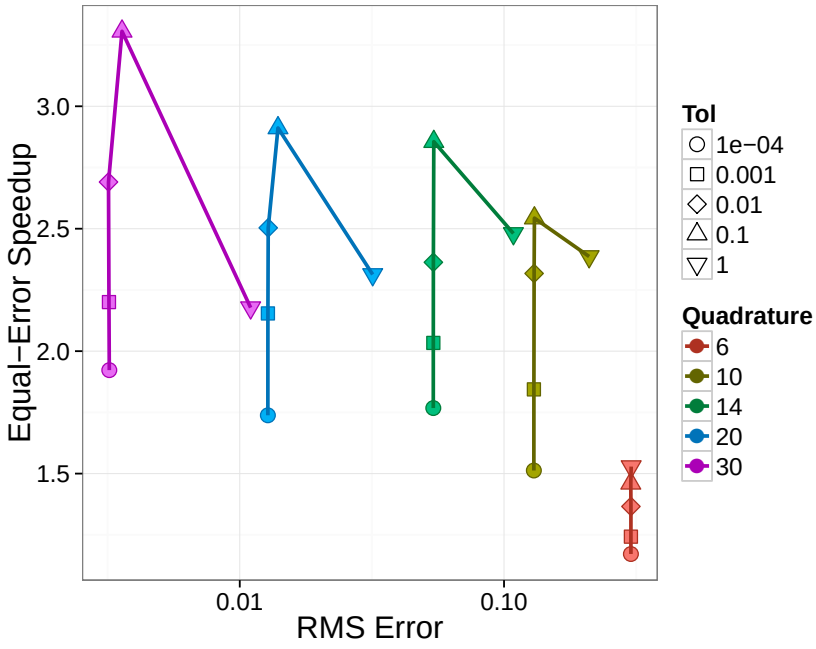


Figure 4.20 Equal-Error Speedup for all ACS parameters on box material 2 (c=0.5)

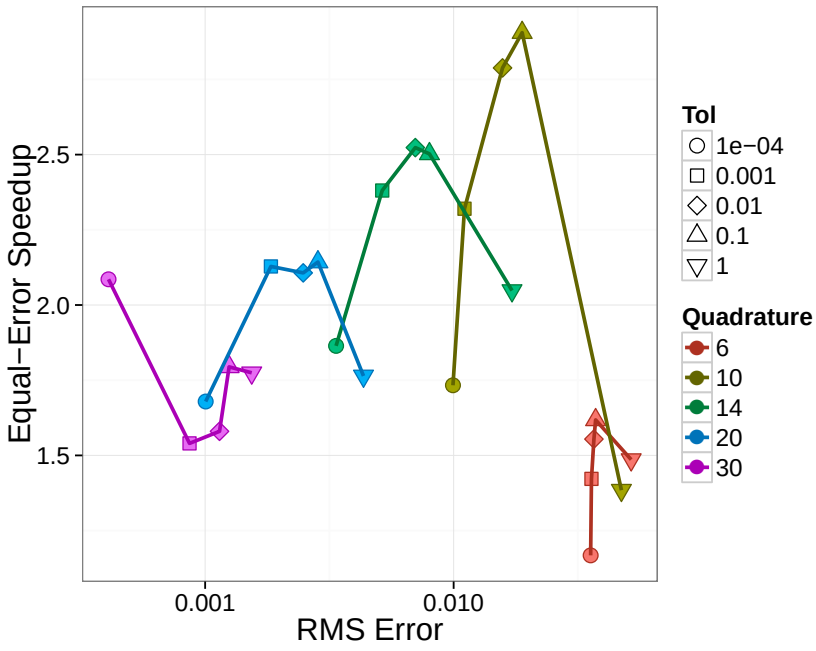


Figure 4.21 Equal-Error Speedup for all ACS parameters on box material 3 (c=0.9)

4.1.5 Performance of error estimates

So far, only final results have been examined. A fundamental part of the method, the error estimation, seemed to be working. When the tolerance is increased, more iterations are performed at the lower quadrature, and thus a lower computation time was observed. Here, a more detailed look will be taken at the quadrature and iterative error estimates.

To examine the quadrature error (ϵ_{quad}, Eq. 3.11), the box problem is examined using S_{30} as the base quadrature and then looking at changing the secondary quadrature. The adaptive tolerance was set at 10^{-9} so that no adaptation occurs. The estimated quadrature error is calculated as a function of the iteration number for each material, as shown in Figures 4.22-4.24.

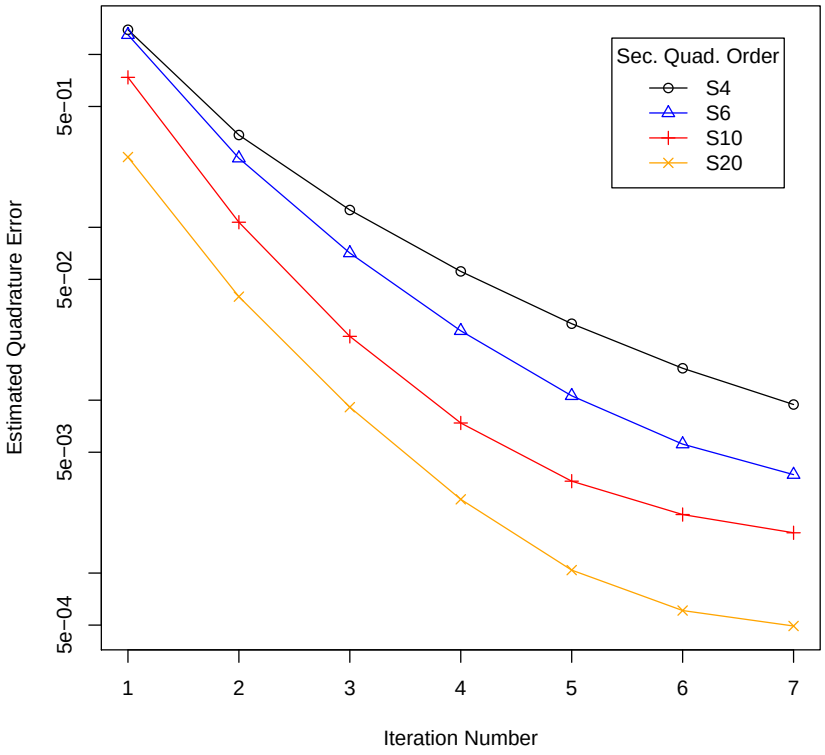


Figure 4.22 Box problem quadrature error estimation for Material 1 (c=0.1)

The differing number of iterations between materials can be attributed to the cases with lower c exhibiting much faster convergence. The first thing to notice is that the estimated quadra-

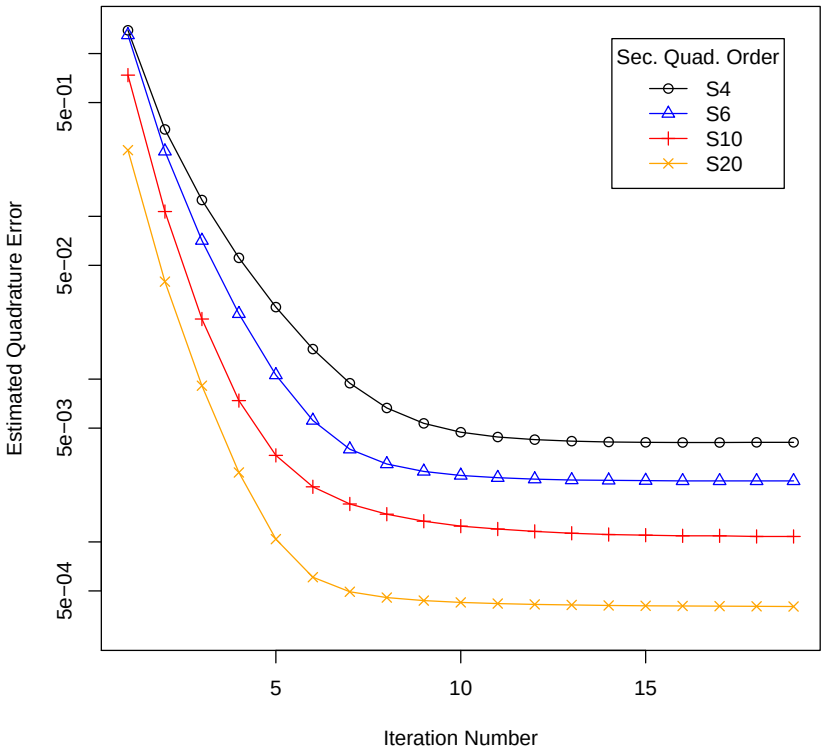


Figure 4.23 Box problem quadrature error estimation for Material 2 (c=0.5)

ture error decreases with iterations. This confirms the prime assumption of the method, i.e., the quadrature requirements (or angular dependencies) decrease with iteration number (or collision number). One interesting observation is that the error decreases over the first several iterations before reaching a constant value. This shows why ACS is more effective than first-collision source, since there is still significant anisotropy after the first collision. Another thing to note is that the error estimate decreases with a higher secondary quadrature order for all cases. This makes sense as there should be less error to lower only slightly the number of quadrature points. Both of these factors support the performance of the error estimation method.

Next, the iterative error (ϵ_{iter}, Eq. 3.25) is examined. A similar analysis to the quadrature error is performed, as shown in Figures 4.25-4.27. It is quickly apparent that the iterative error estimate is independent of the quadrature order, as expected. After a few iterations

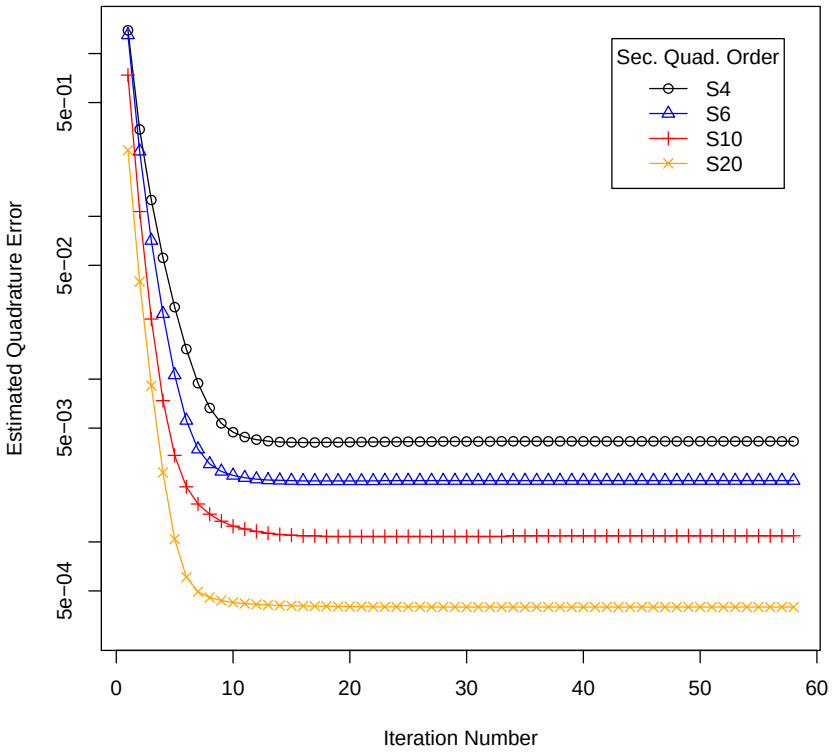


Figure 4.24 Box problem quadrature error estimation for Material 3 (c=0.9)

of settling down, the iterative error appears to reduce exponentially. This supports the assumptions used in Eq. 3.20 to calculate the error.

The total error estimate is the combination of the iterative error and quadrature error estimates. These errors are shown for the S_{30}/S_{14} case in Figures 4.28-4.30. The quadrature error reaches a stable level, while the iterative error drops continuously. This means that overall, the combined error is always decreasing and ensures that the adaptive tolerance will eventually be met for most problems.

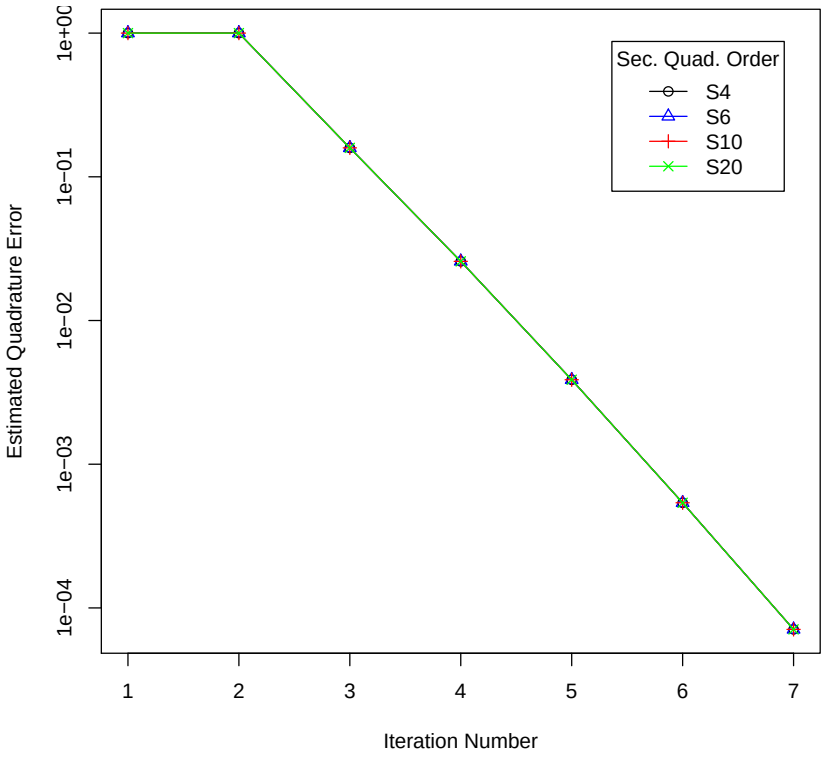


Figure 4.25 Box problem iterative error estimation for Material 1 (c=0.1)

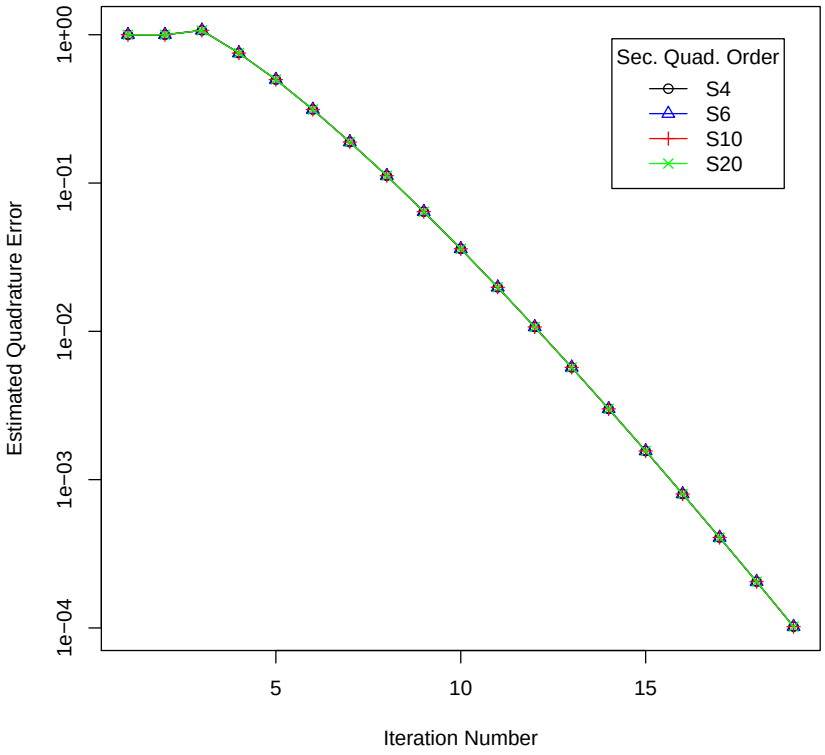


Figure 4.26 Box problem iterative error estimation for Material 2 (c=0.5)

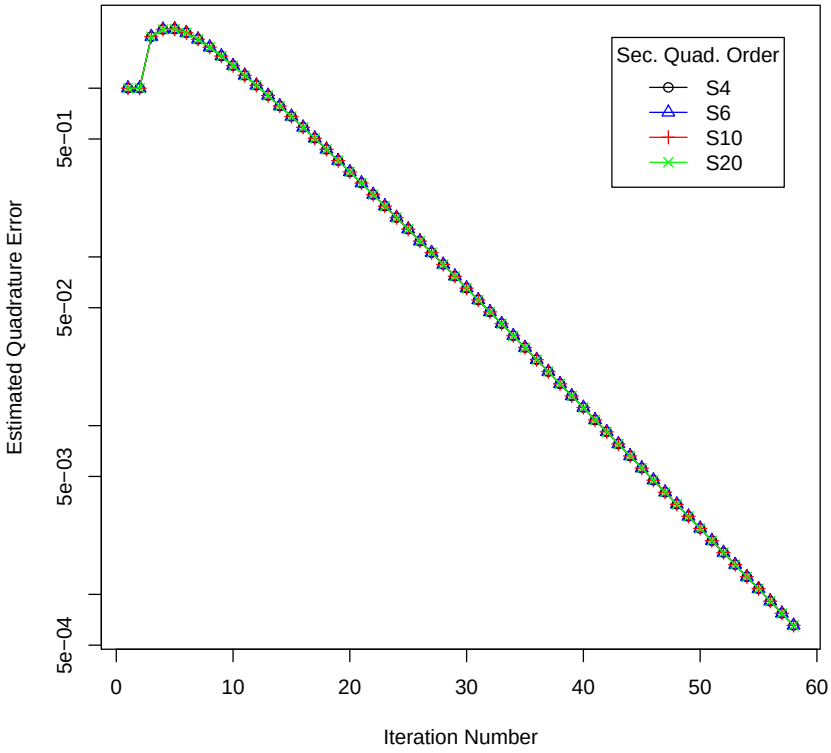


Figure 4.27 Box problem iterative error estimation for Material 3 (c=0.9)

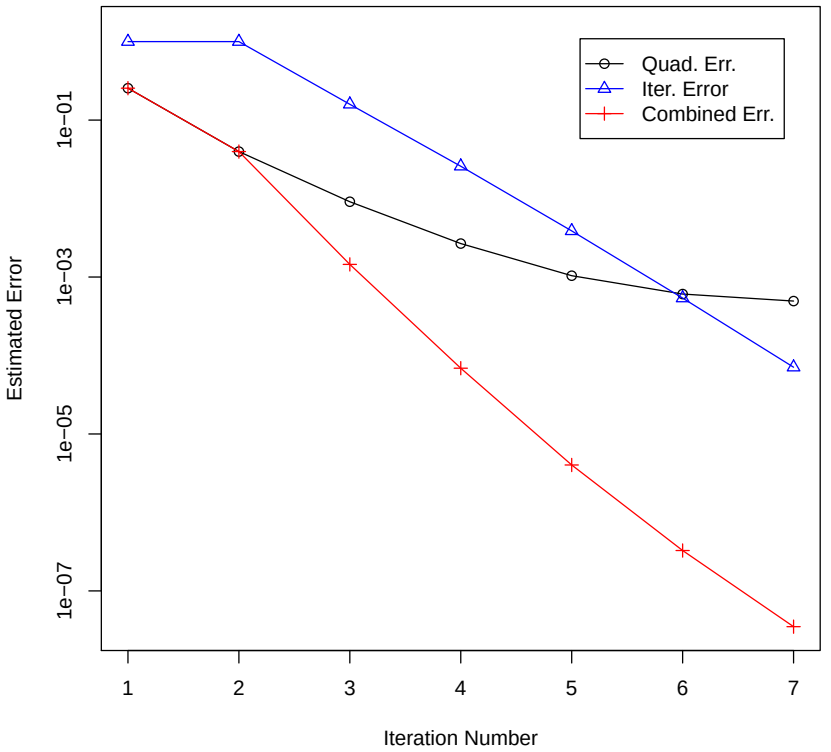


Figure 4.28 Box problem total error estimation for Material 1 (c=0.1)

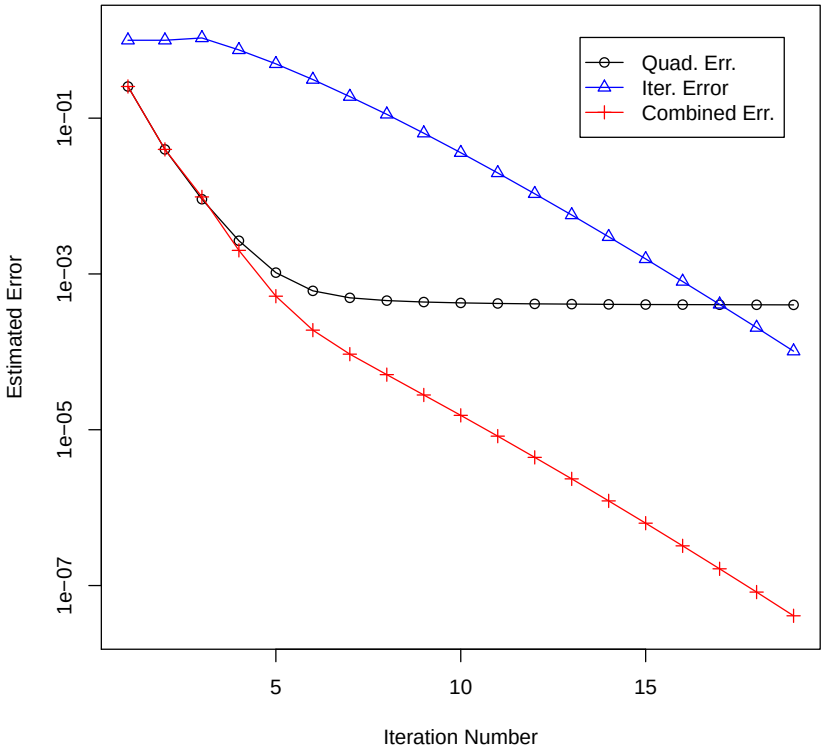


Figure 4.29 Box problem total error estimation for Material 2 (c=0.5)

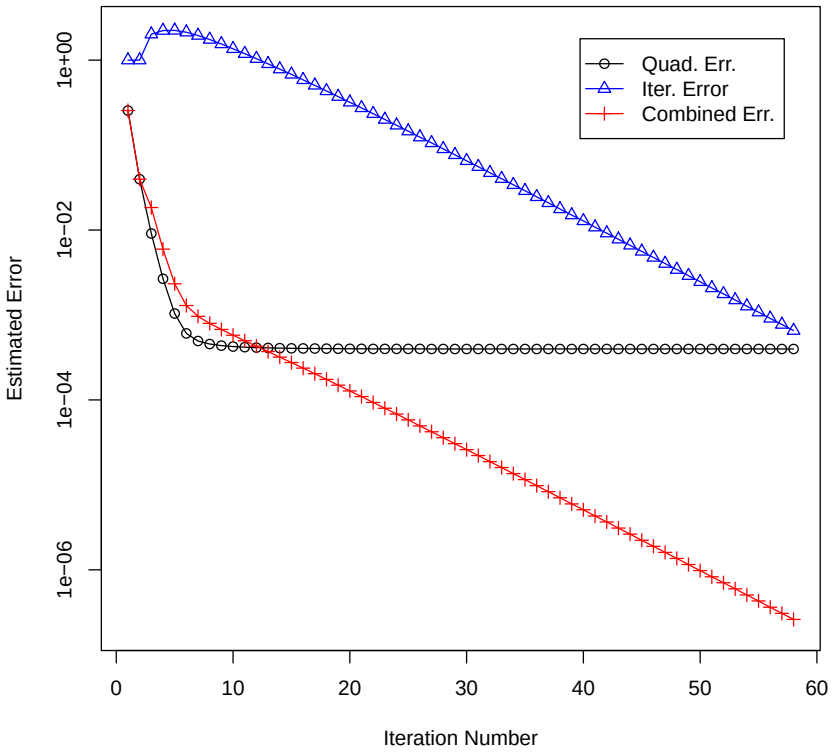


Figure 4.30 Box problem total error estimation for Material 3 (c=0.9)

4.1.6 Parallel Performance

With the ubiquity and potential capabilities of parallel computing, it is essentially a requirement that any new techniques be scalable to many processors. The ACS method has proven to be very effective on a single processor, but the additional requirements for parallel communication should mean a lower parallel efficiency. However, if the general benefits outweigh the (hopefully small) additional parallel costs, it can still be an effective method.

In order to test the parallel performance of the ACS method, the box problem was repeated, with now using three energy groups in order to test the multi-group ACS capabilities as well. These three-group cross sections were taken from a PWR cross section library for water (with a high thermal scattering ratio $c \approx 0.99$). To begin, a reference calculation is performed using the standard source iteration in TITAN at each of the given quadrature levels. As a metric of accuracy, the RMSE of the flux is computed for each case as compared to the base reference case flux of S_{16}. For each quadrature level, the RMSE and computation times are given in Table 4.4 for the vacuum boundary case and Table 4.5 for the reflective case. As expected, the error increases with lower quadrature levels, while the computation time decreases proportional to the number of ordinates.

Table 4.4 TITAN Reference Calculations at Different Quadratures (Vacuum Boundaries)

Quadrature	# of Directions	Time (s)	RMSE
S_{16}	288	1980.3	0
S_{10}	120	836.3	3.74E-02
S_6	48	341.9	1.16E-01

Table 4.5 TITAN Reference Calculations at Different Quadratures (Reflective Boundary)

Quadrature	# of Directions	Time (s)	RMSE
S_{16}	288	2366.0	0
S_{10}	120	983.6	3.35E-02
S_6	48	392.4	9.86E-02

Now, we examine the results for the ACS method using S_{16} as the base quadrature set. The RMSE and computation times are shown in Tables 4.6 and 4.7 for varying values of the adaptive tolerance ϵ.

Table 4.6 ACS Calculations at Different Adaptive Tolerances ϵ (Vacuum Boundaries)

ϵ	# of Calc. at:			RMSE	Comp.
	S_{16}	S_{10}	S_6		Time(s)
10^{-9}	58	0	0	2.30E-07	1872.1
10^{-4}	36	7	21	1.49E-04	1317.7
10^{-3}	15	8	41	1.82E-03	758.9

Table 4.7 ACS Calculations at Different Adaptive Tolerances ϵ (Reflective Boundary)

ϵ	# of Calc. at:			RMSE	Comp.
	S_{16}	S_{10}	S_6		Time(s)
10^{-9}	76	0	0	6.14E-06	2334.1
10^{-4}	48	8	26	1.74E-04	1133.4
10^{-3}	19	9	54	1.81E-03	702.5

There is a small overhead in the ACS method (e.g., computing the error estimates), so one would expect a slight increase in computation time when there is no adaptation (i.e., when the tolerance is set at 10^{-9}). However, since the ACS method only uses the independent source in the first transport sweep, there appears to be an overall time savings even with no adaptation by removing this overhead (normally the source is summed at every iteration). Also given in the table is the number of transport sweeps performed at each quadrature level. As expected, as ϵ increases, more of the computation will be performed at lower quadratures so error and speedup will both increase. It is worth noting that the RMSE for the ACS cases is very close to the adaptive tolerance level, which means that the error estimates are working very well.

The results from the ACS calculations show excellent agreement with the standard source iteration in TITAN. If we compare the intelligent quadrature adaptation in ACS to uniformly lowering the quadrature, ACS performs much better. For an ACS tolerance of $\epsilon = 10^{-3}$, the

computation time is less than the SI calculation at S_{10}, while the RMSE is also over 5 times lower.

To test the parallel scalability, calculations were performed on a small cluster consisting of 2 compute nodes with 8 processor cores each (Intel Xeon 2.4 GHz) for a total of 16 processors. The calculations above were repeated on 2, 4, 8, and 16 processors. TITAN and TITAN-ACS use decomposition of the angular ordinates, with a certain number of ordinates assigned to each processor. For TITAN with reflective boundaries, the amount of decomposition is limited to the number of ordinates within an octant to avoid parallel communication. In TITAN with vacuum boundaries, and TITAN-ACS with any boundary type, the decomposition is only limited to the total number of ordinates. This allows for better load-balance if the number of ordinates is relatively low compared to the total number of processors. The number of ordinates per processor is shown for each quadrature level in Table 4.8, while Table 4.9 shows the ordinate decomposition for the TITAN reflective case which is limited to the number of ordinates per octant.

Table 4.8 Angular Decomposition For TITAN (Vacuum Boundaries) and ACS (All Boundary Types)

Number of Processors	Ordinates Per Proc. S_{16}	S_{10}	S_6
1	288	120	48
2	144	60	20
4	72	30	12
8	36	15	6
16	18	7.5	4

Note the non-integer values, which lead to load imbalances among processors as they cannot process fractions of an ordinate. For example, the S_{16} case on 16 processors from Table 4.8 has 2.25 ordinates per processor. This would lead to 12 processors with 2 ordinates per octant and 4 processors with 3 ordinates. The 12 processors will have to wait for the other 4 to finish the final ordinate before being able to move on to the next iteration. This is

Table 4.9 Angular Decomposition per Octant For TITAN (Reflective Boundaries)

Number of Processors	Ordinates Per Proc. (Per Octant)		
	S_{16}	S_{10}	S_6
1	36	15	6
2	18	7.5	3
4	9	3.75	1.5
8	4.5	1.875	0.75
16	2.25	0.9375	0.375

especially true for the TITAN reflective boundary cases, so poor efficiency is expected for those.

Parallel performance for the TITAN S_{16} calculations are shown in Figures 4.31 (Vacuum) and 4.32 (Reflective). Shown in these plots are TITAN using S_{16}, along with the ACS calculations with different ϵ with S_{16} as the starting quadrature. For an ideal parallel process, the parallel speedup is equal to the number of processors (i.e., the black line). For the vacuum boundary case, both methods have the same angular decomposition, so as expected, the standard TITAN has a higher parallel efficiency due to less communication requirements. However, with reflective boundaries, standard TITAN is limited to the number of ordinates per octant to avoid extra communication. Although it avoids the communication costs, the resulting load imbalance has its own cost. For the ACS calculations, the efficiency goes down with higher tolerance, as more calculations are done at lower quadratures, and thus fewer calculations are performed before each communication step.

It is informative to look at the total speedup as well - that is, time taken on a single processor at S_{16} compared to the time for the given case in parallel. These values for the 16 processors cases are given in Table 4.10 (Vacuum) and Table 4.11 (Reflective). These tables show the large slowdowns due to load imbalance at S_6, especially in the reflective case. In comparing SI to ACS, a good comparison is the S_{10} case against the $\epsilon = 10^{-3}$ ACS case. For the reflective and vacuum cases, ACS has a higher total speedup on 16 processors as well as a MRE that is over 10 times lower.

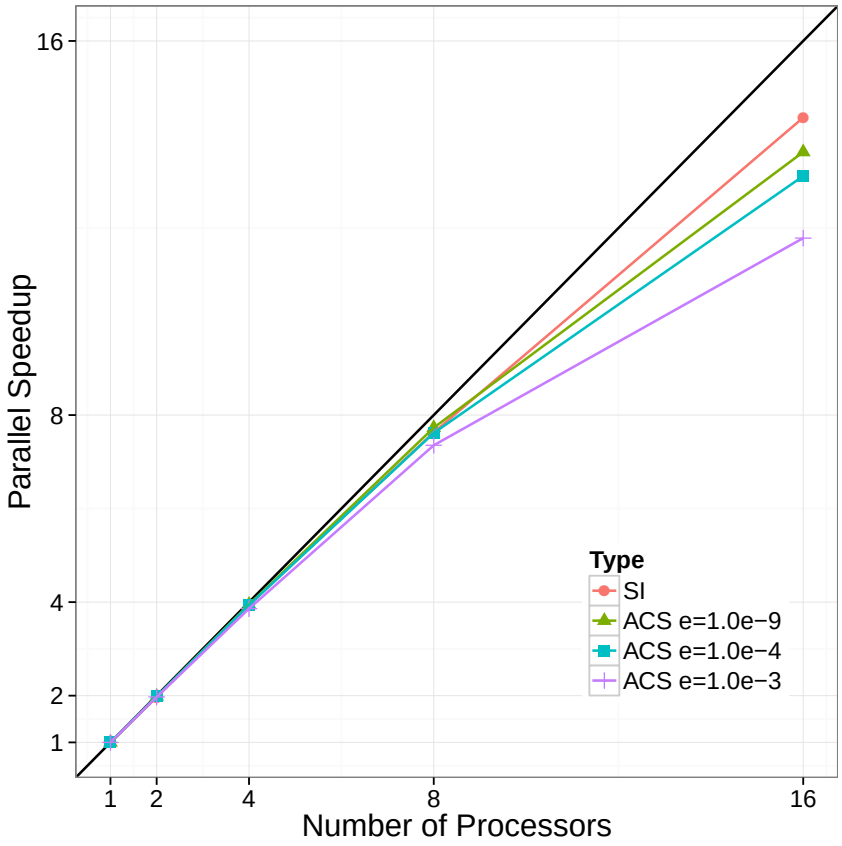


Figure 4.31 Parallel speedup for TITAN and ACS (vacuum boundaries)

Table 4.10 Comparison of parallel performance on 16 processors (Vacuum Boundaries)

Quadrature	Calc. Type	Adap. Tol.	Time (s)	Parallel Speedup	Total Speedup	RMSE	Parallel Efficiency
S_{16}	SI	-	137.9	14.36	14.36	2.05E-07	0.898
S_{10}	SI	-	68.8	12.16	28.78	3.74E-02	0.760
S_6	SI	-	34.5	9.91	57.4	1.16E-01	0.619
$S_{16}/S_{10}/S_6$	ACS	1.00E-09	137.4	13.63	14.41	2.16E-07	0.856
$S_{16}/S_{10}/S_6$	ACS	1.00E-04	100.5	13.11	19.70	1.49E-04	0.819
$S_{16}/S_{10}/S_6$	ACS	1.00E-03	64.4	11.79	30.75	1.82E-03	0.737

More detail for the ACS cases with $\epsilon = 10^{-3}$ are given in Tables 4.12 and 4.13.

These results show excellent parallel scalability for both TITAN and ACS, especially up to 8 processors. At 8 processors, all the speedup numbers equated to a parallel fraction of

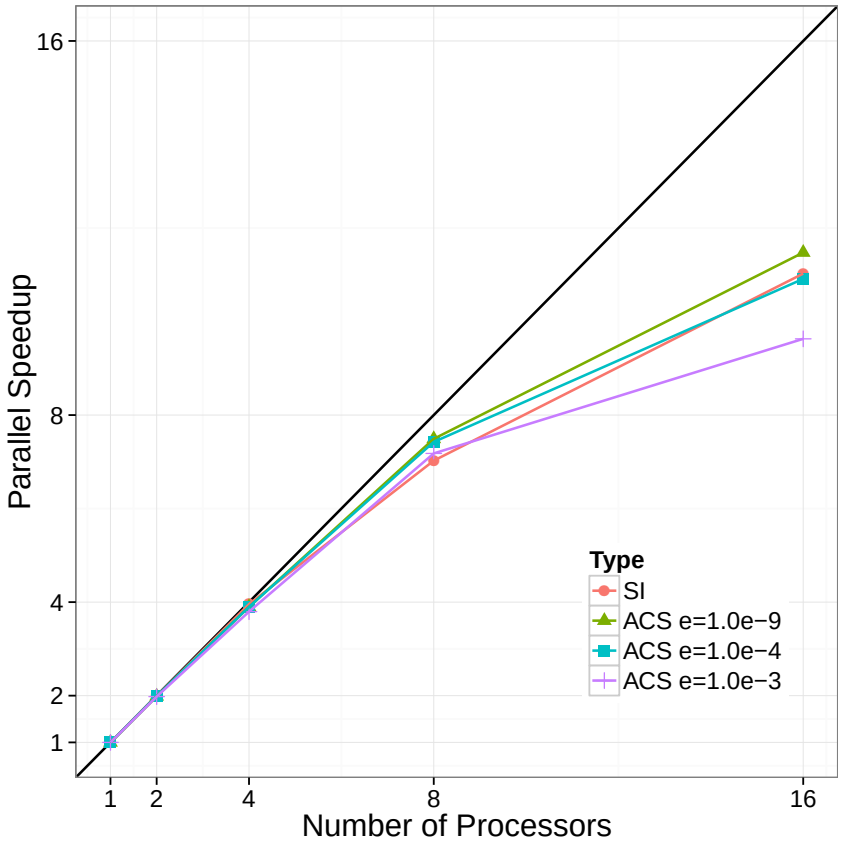


Figure 4.32 Parallel Speedup for TITAN and ACS (reflective boundary)

Table 4.11 Comparison of parallel performance on 16 processors (Reflective Boundaries)

Quadrature	Calc. Type	Adap. Tol.	Time (s)	Parallel Speedup	Total Speedup	RMSE	Parallel Efficiency
S_{16}	SI	-	214.6	11.03	11.03	2.63E-06	0.689
S_{10}	SI	-	148.0	11.97	15.99	3.35E-02	0.748
S_6	SI	-	142.6	4.96	16.59	9.86E-02	0.310
$S_{16}/S_{10}/S_6$	ACS	1.00E-09	213.6	11.49	11.08	6.14E-06	0.717
$S_{16}/S_{10}/S_6$	ACS	1.00E-04	159.9	10.91	14.80	2.22E-04	0.682
$S_{16}/S_{10}/S_6$	ACS	1.00E-03	100.8	9.63	23.47	1.82E-03	0.602

$> 98\%$ (as calculated by Eq. 3.30). For the 16 processor cases, the calculations must be distributed over 2 separate nodes, and is an inherent increased communication time. Here, the parallel fraction is reduced to $\sim 97\%$ and the parallel efficiency drops to $\sim 70\%$. This is

Table 4.12 Parallel performance for the ACS $\epsilon = 10^{-3}$ case (Vacuum Boundaries)

Number of Proc.	Parallel Speedup	Parallel Efficiency	Parallel Fraction
2	1.97	0.99	0.99
4	3.86	0.97	0.99
8	7.35	0.92	0.99
16	11.78	0.73	0.97

Table 4.13 Parallel performance for the ACS $\epsilon = 10^{-3}$ case (Reflective Boundary)

Number of Proc.	Parallel Speedup	Parallel Efficiency	Parallel Fraction
2	1.98	0.99	0.99
4	3.80	0.95	0.98
8	7.18	0.90	0.98
16	9.63	0.60	0.96

especially true in the reflective case for ACS with $\epsilon = 10^{-3}$. At this tolerance, most of the calculations are being done at S_6, which means means that only 3 ordinates are assigned to each processor, while almost the same amount of data as the other ACS calculations must still be communicated (ϕ_l^k and ϵ_{quad} remain the same size). This leads the parallel fraction to be only 96% and the parallel efficiency on 16 processors drops to 60%.

In addition to the parallel fraction issue, there is also the effect of load imbalance. This is seen in the vacuum S_{10} TITAN calculation on 16 processors, as Table 4.8 indicates a fractional load distribution. The effect is also limiting the reflective TITAN speed due to the fractional loads shown in Table 4.9. Note that TITAN-ACS does not have this load distribution issue as it can always use the decomposition from Table 4.8. Despite the additional parallel communication, all ACS calculations showed significant speedups with minimal added errors.

4.2 VENUS-2 Dosimetry Benchmark

In the previous section, the ACS method was tested on a very simple box problem. Now the intention is to compare it on a more realistic, complicated problem.

4.2.1 Problem Description

The VENUS-2 MOX-fueled reactor dosimetry benchmark[51] was issued by the OECD/NEA based on the VENUS reactor owned and operated by SCK•CEN in Belgium. The objective was to validate and compare nuclear data and transport codes for MOX-fueled dosimetry calculations. Our goal is to compare the ACS method to the standard source iteration in TITAN. As such, the actual benchmark performance and accuracy is not the issue - just the difference between the two methods.

The VENUS-2 reactor core is small, approximately 80cm by 80cm. Vertically, the fuel length is 50cm, with reflector regions above and below. Surrounding the core are water and steel regions to simulate the pressure vessel of a full-sized reactor. A full description is omitted for brevity, but the full description can be seen in the benchmark description[51].

The measured data in the benchmark are the reaction rates using $^{58}\mathrm{Ni}(n,p)$, $^{115}\mathrm{In}(n,n)$, $^{103}\mathrm{Rh}(n,n)$, $^{64}\mathrm{Zn}(n,p)$, $^{237}\mathrm{Np}(n,f)$, and $^{27}\mathrm{Al}(n,\alpha)$ dosimeters at 34 locations at the core mid-plane, which correspond to locations in both steel and water areas. These dosimeters are all sensitive only to fast neutrons ($> 0.1MeV$).

4.2.2 Reference Results

Using the benchmark information, a model was generated for TITAN, with a range of 70 cm to 168 cm on the Z axis and 0 to 70 cm on the X-Y plane. The mesh and materials for the core level are shown in Figure 4.33. The 34 detector locations are overlaid over the mesh

distribution. Locations 1-19 indicate regions surrounded by water, while locations 20-34 indicate regions surrounded by steel. The fission source distribution used for the model is shown in Figure 4.34. Again, vertical diagrams are omitted for brevity.

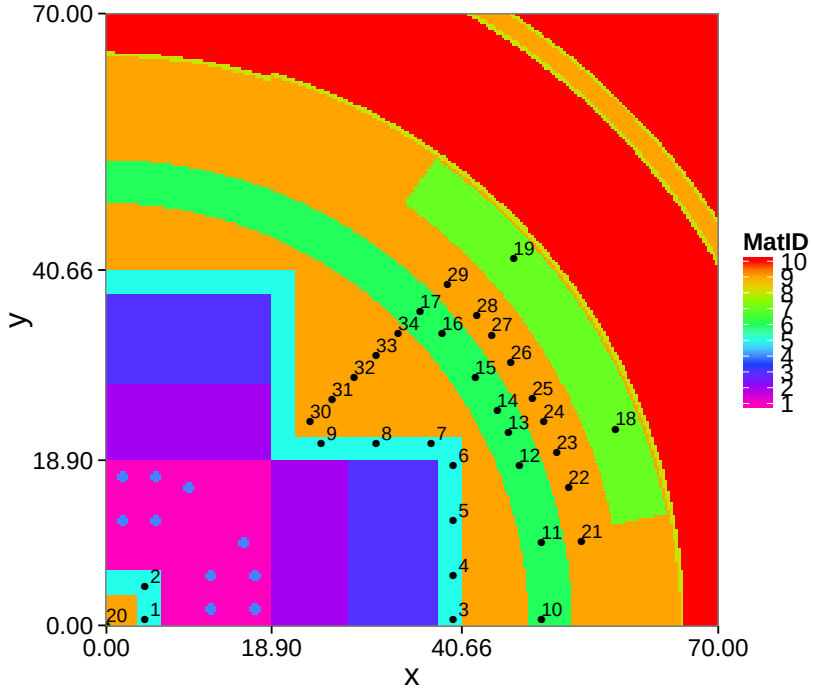


Figure 4.33 VENUS-2 mesh and dosimeter Layout

Multi-group cross sections were derived from the 47-group BUGLE-96[52] shielding cross-section library, based on the ENDF/B-VI data set[53]. The fuel region was homogenized by volume fraction of fuel, clad and water. The dosimeter cross sections were also obtained from BUGLE, although $^{64}\mathrm{Zn}(n,p)$ data is not available so it was ignored in this analysis. Only the first 26 energy groups from BUGLE were used, which corresponds to 0.11 to 20 MeV, since below those energies the dosimeter cross sections are negligible. The BUGLE energy group structure for the first 26 groups is shown in Table 4.14. A plot of the dosimeter cross section as a function of BUGLE energy group is shown in Figure 4.35.

The mesh size used was 0.315 cm in the X/Y directions and 0.63 cm in the Z direction, for a total of 557,583 spatial meshes. The DD method is used for spatial differencing. For angular quadrature, Legendre-Chebyschev (Pn-Tn) quadrature sets of order 4 to 16 (corresponding

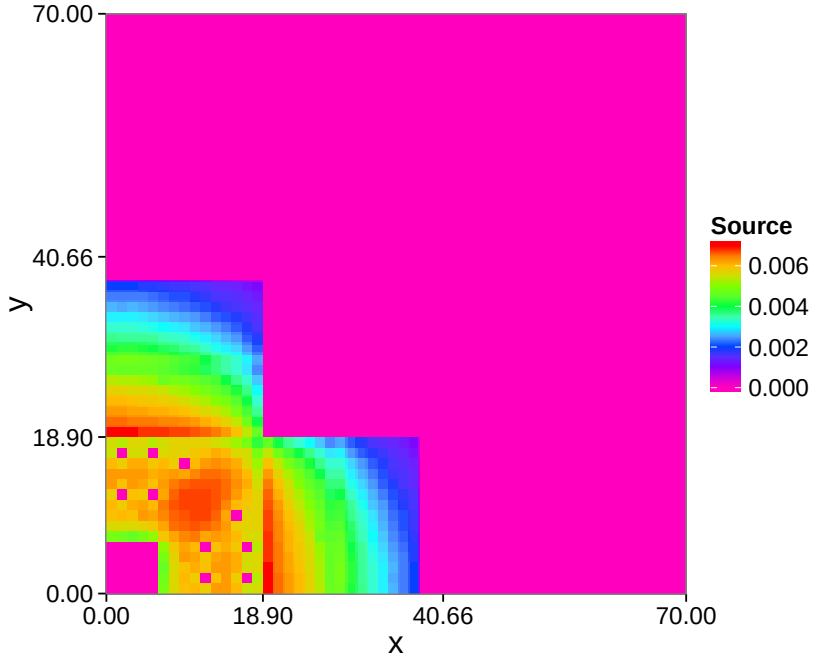


Figure 4.34 VENUS-2 source distribution

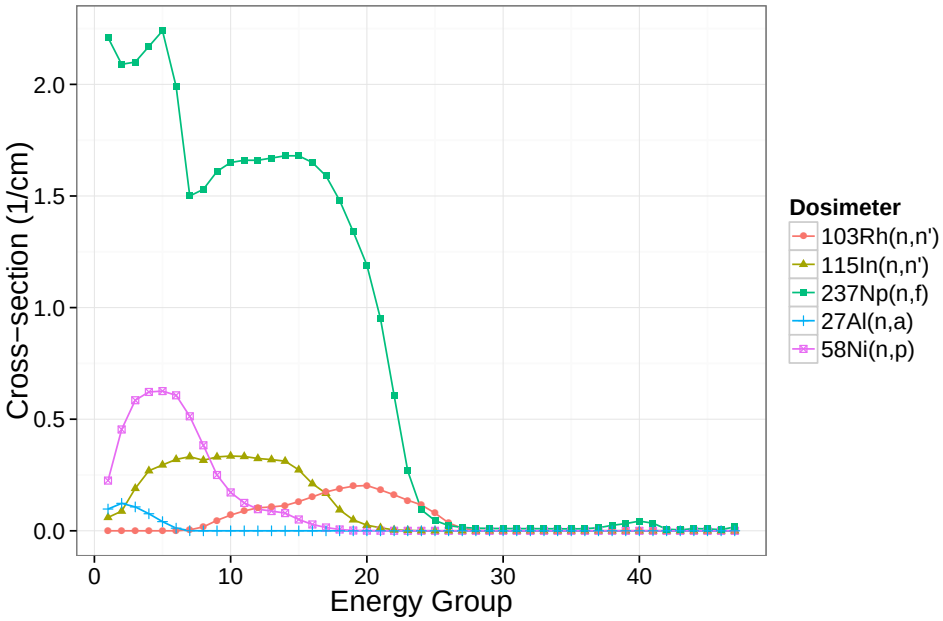


Figure 4.35 VENUS-2 dosimeter cross-sections using the BUGLE energy-group structure

Table 4.14 BUGLE-96 Energy Group Structure for the First 26 Groups

Group	E low (MeV)	E high (MeV)
1	14.20	19.60
2	12.20	14.20
3	10.00	12.20
4	8.61	10.00
5	7.41	8.61
6	6.07	7.41
7	4.97	6.07
8	3.68	4.97
9	3.01	3.68
10	2.73	3.01
11	2.47	2.73
12	2.37	2.47
13	2.35	2.37
14	2.23	2.35
15	1.92	2.23
16	1.65	1.92
17	1.35	1.65
18	1.00	1.35
19	0.82	1.00
20	0.74	0.82
21	0.61	0.74
22	0.50	0.61
23	0.37	0.50
24	0.30	0.37
25	0.18	0.30
26	0.11	0.18

to 24 - 288 discrete ordinates) were used. The flux convergence tolerance was set at 5×10^{-4} for all cases.

Reference 26-group flux results were obtained using the standard TITAN code with a S_{16} quadrature set. Fluxes for energy groups 7 (4.97-6.03 MeV) and 18 (1.0 - 1.35 MeV) are shown in Figures 4.36 and 4.37, respectively. Other energy groups look similar, and plots of these are omitted for brevity. As expected, the flux is highest in the core, dips in the center and in the Pyrex pins, and drops quickly outside of the core.

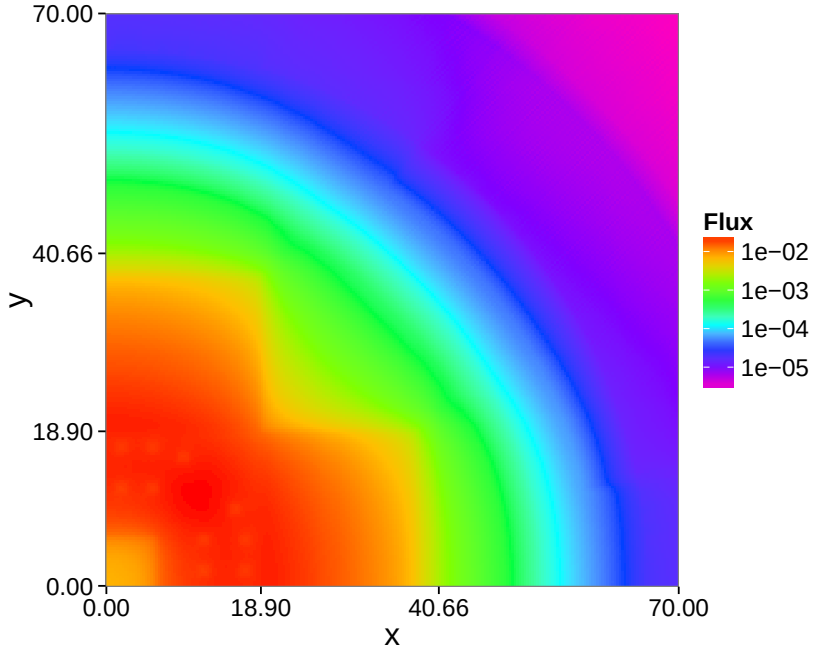


Figure 4.36 VENUS-2 reference flux distribution, group 7 (4.97-6.03 MeV)

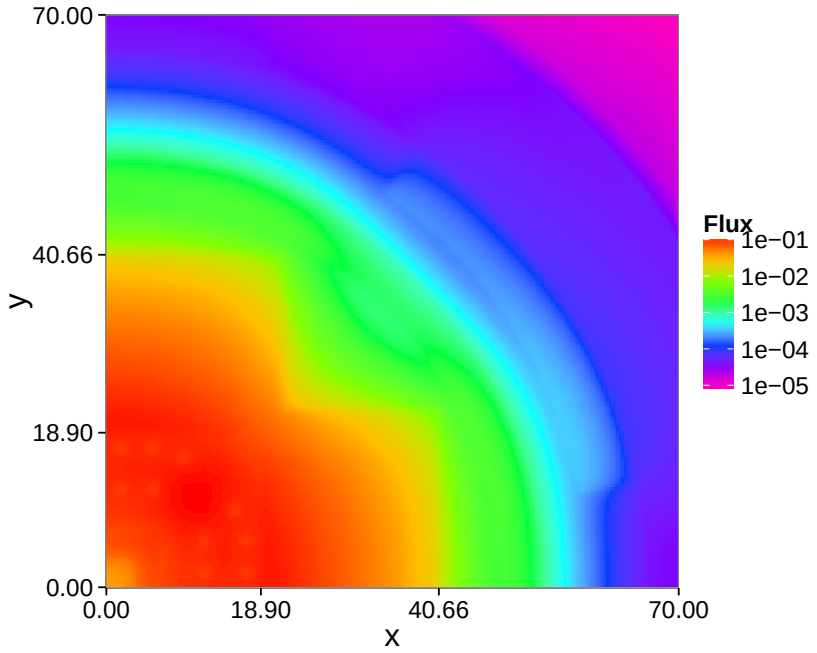


Figure 4.37 VENUS-2 reference flux distribution, group 18 (1.00-1.35 MeV)

Using these fluxes, coupled with the dosimeter cross sections, reaction rates were obtained for every dosimeter location. Results for the $^{27}\mathrm{Al}(n,\alpha)$ reaction are shown in Figure 4.38 for all locations. Results for $^{237}\mathrm{Np}(n,f)$ are shown in Figure 4.39. These two were chosen because $^{27}\mathrm{Al}(n,\alpha)$ is responsive only to the highest energies, while $^{237}\mathrm{Np}(n,f)$ is sensitive to lower energies.

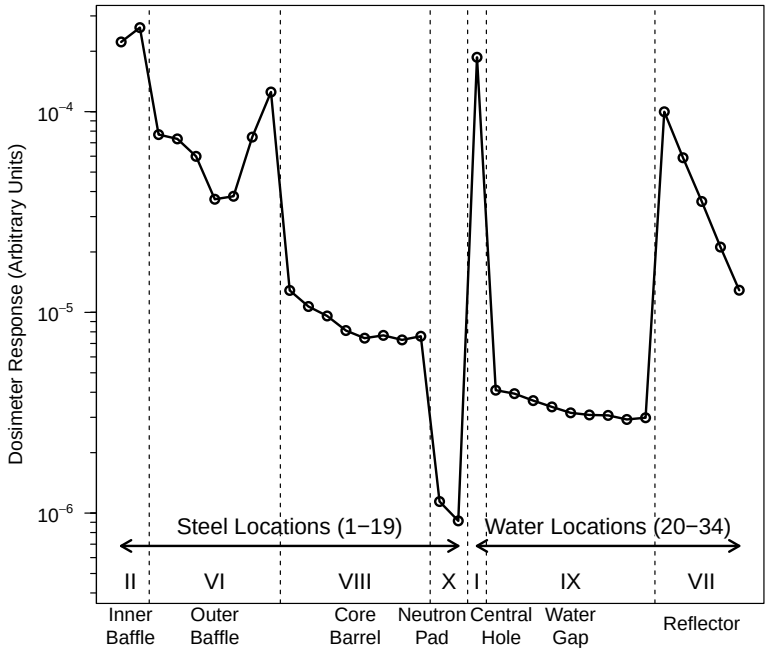


Figure 4.38 $^{27}\mathrm{Al}(n,\alpha)$ reaction rates at all locations

Calculations were also performed using the standard TITAN code for lower quadrature orders (S4-S10). Relative errors compared to the S_{16} case for S_4 and S_{10} are given in Figure 4.40 for $^{27}\mathrm{Al}(n,\alpha)$ and Figure 4.41 for $^{237}\mathrm{Np}(n,f)$. As expected, the error decreases with quadrature order, and is highest in areas far from the source and dominated by streaming and high angular dependency. Further, the Al errors are higher than the Np errors, since the Al response is dictated by higher energies which are more streaming-dominated than lower energies which have more scatter.

To compare between cases, we consider the RMS relative error of all dosimeter locations and types. The maximum error is also considered. These errors, along with computation times

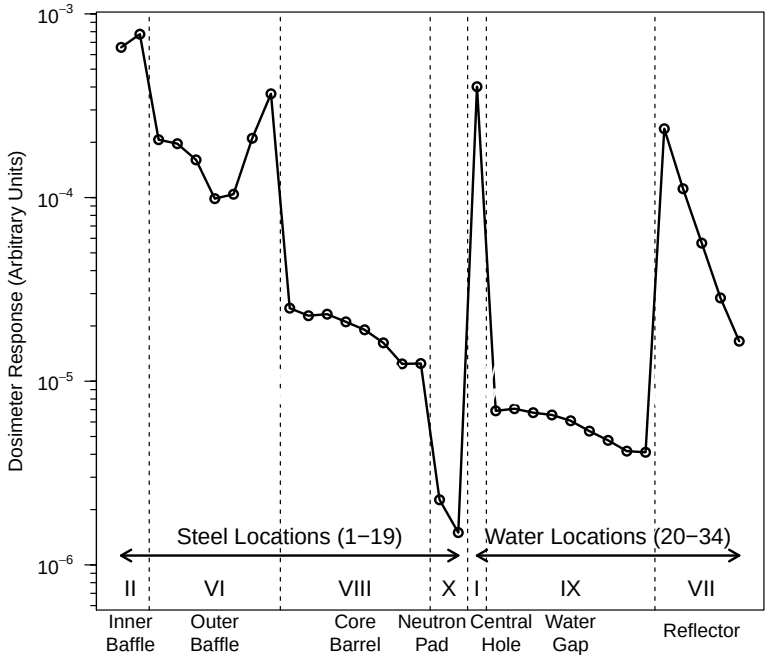


Figure 4.39 $^{237}\mathrm{Np}(n, f)$ reaction rates at all locations

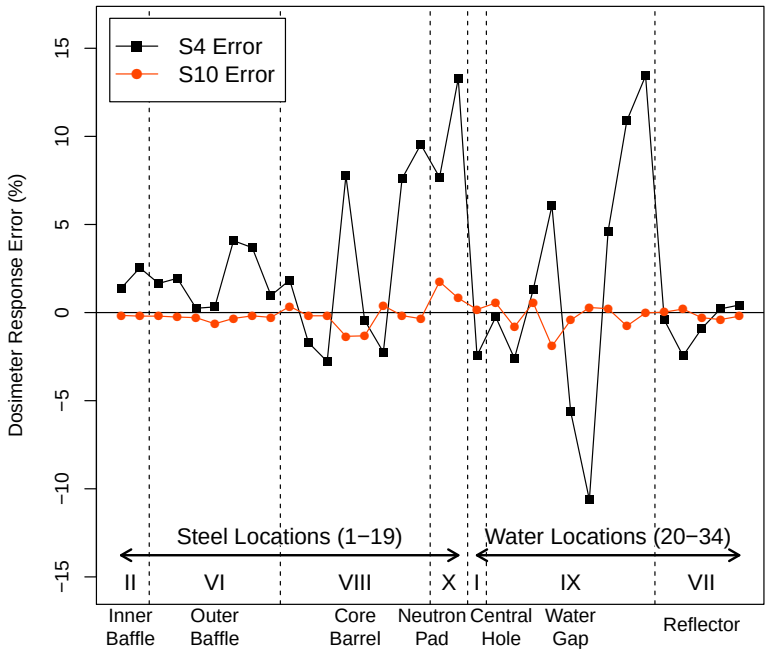


Figure 4.40 $^{27}\mathrm{Al}(n, \alpha)$ reaction rates errors by quadrature order

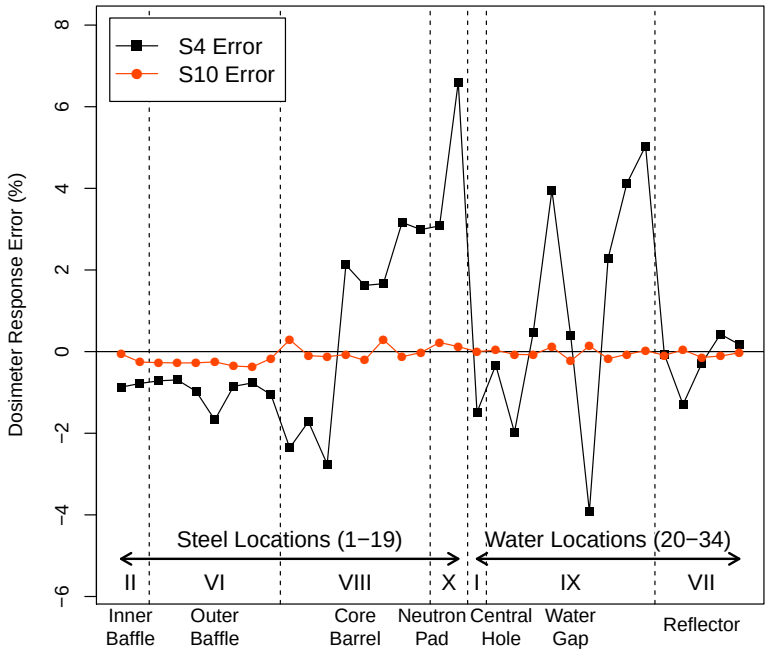


Figure 4.41 $^{237}\mathrm{Np}(n, f)$ reaction rates errors by quadrature order

for the TITAN calculations are given in Table 4.15. As expected, the errors consistently decrease while the computation time increases with the quadrature order. The "acceptable" quadrature level would depend on the required accuracy of the computation.

Table 4.15 Source Iteration Error and Computation Time for VENUS problem

Quad. Order	CPU Time (min)	Inner Iterations	RMS Error	Max Error
S_{16}	1100.2	355	-	-
S_{10}	452.5	355	0.37%	1.91%
S_8	303.9	356	0.56%	2.88%
S_6	187.3	363	0.93%	4.22%
S_4	96.3	367	3.53%	13.43%

4.2.3 ACS Results for VENUS Problem

These calculations were repeated using the ACS algorithm using several quadrature combinations from S_{10} to S_4 and and tolerances ϵ from 10^{-3} to 10^{-1}. Results from the ACS calculations including computation times, RMS errors, and equal-error speedups are given in Table 4.16.

Table 4.16 ACS Results and Speedup for VENUS problem

Quad. Order	ACS Tol. ϵ	Time (min)	# Iterations at: Quad. 1	Quad. 2	Total	RMS Error %	MAX Error %	Equal-Error Speedup
S_{10}/S_4	10^{-3}	296.3	211	182	393	0.43%	2.35%	1.362
S_{10}/S_4	10^{-2}	188.1	80	310	390	0.59%	2.38%	1.576
S_{10}/S_4	10^{-1}	115.9	26	366	392	2.03%	6.63%	1.284
S_8/S_4	10^{-3}	200.0	243	154	397	0.60%	3.57%	1.464
S_8/S_4	10^{-2}	130.6	110	285	395	1.06%	3.58%	1.397
S_8/S_4	10^{-1}	103.2	30	361	391	2.09%	6.68%	1.422
S_6/S_4	10^{-3}	144.2	234	158	392	0.98%	4.79%	1.287
S_6/S_4	10^{-2}	113.9	111	281	392	1.04%	4.78%	1.610
S_6/S_4	10^{-1}	95.1	28	365	393	1.89%	5.49%	1.614

With increasing ϵ, more iterations are performed at the lower quadrature, and consequently the computation time decreases while the error increases. With a reasonable level of the adaptive tolerance, the error goes up very little, while the computation time decreases by a significant amount. An equal-error speedup is obtained for all ACS cases, up to a factor of 1.6.

The speedup and RMS error of more combinations of quadrature and adaptive tolerance are summarized in Figure 4.42. All cases offered a more efficient solution than the alternative of simply uniformly changing the quadrature. It is also noted that regardless of chosen parameters (quadrature and tolerance), ACS performs better, so *a priori* knowledge of some optimal parameters is not required to obtain a speedup.

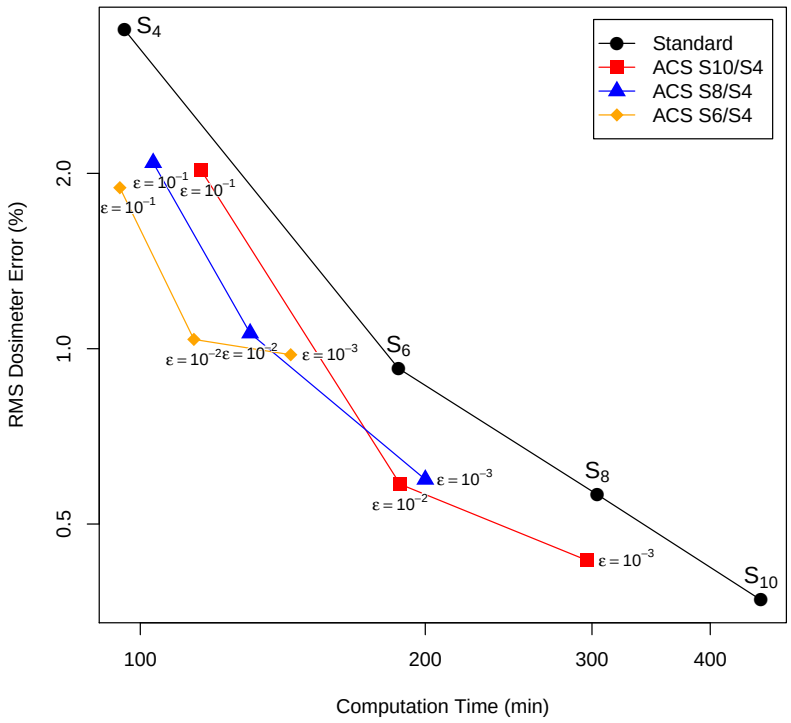


Figure 4.42 Efficiency of ACS compared to Source Iteration on the VENUS problem

4.2.4 ACS Iteration Details

To get some insight on the details of an ACS calculation, one case is examined in more detail. As an example, the details of the ACS calculation for group 7 (4.97-6.07 MeV) with S_{10}/S_4 quadrature and $\epsilon = 10^{-3}$ are examined. Figure 4.43 shows the error estimates and other details during the within-group iterations for this calculation. Note that the iterative change is defined as the relative change in flux (i.e., $\phi^{(i)}/\phi_t^{(i)}$) that is used to calculate the iterative error.

Several things are noted. First, calculations start using S_{10}, using approximately 8 seconds per iteration. The quadrature error, ϵ_{quad}, decreases over several iterations until reaching a constant value. The iterative error, ϵ_{iter}, decreases continuously after an initial period, along with ϵ_{total}, the combined error. The total error, ϵ_{total}, reaches the tolerance at iteration

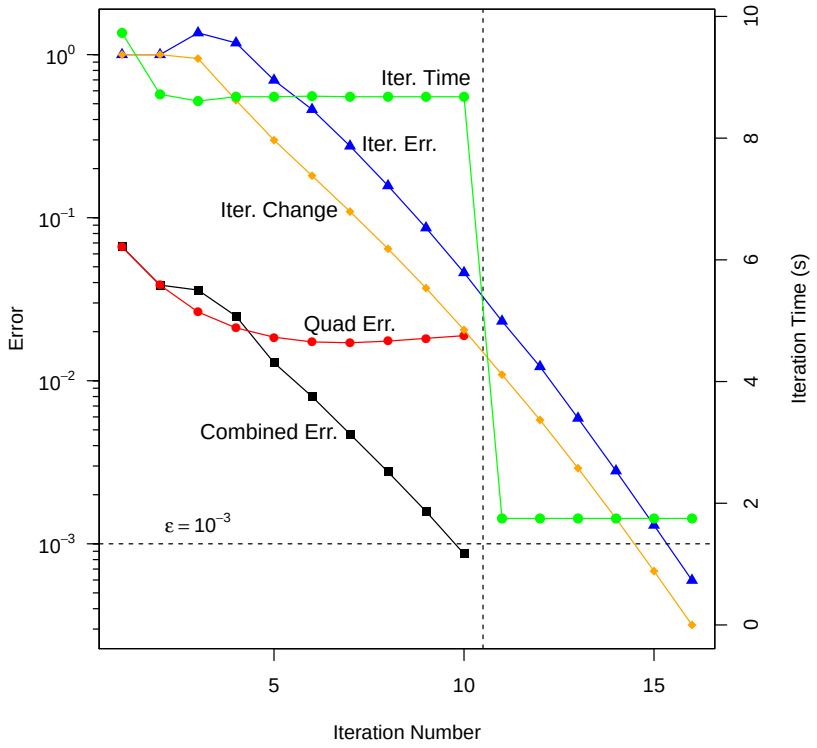


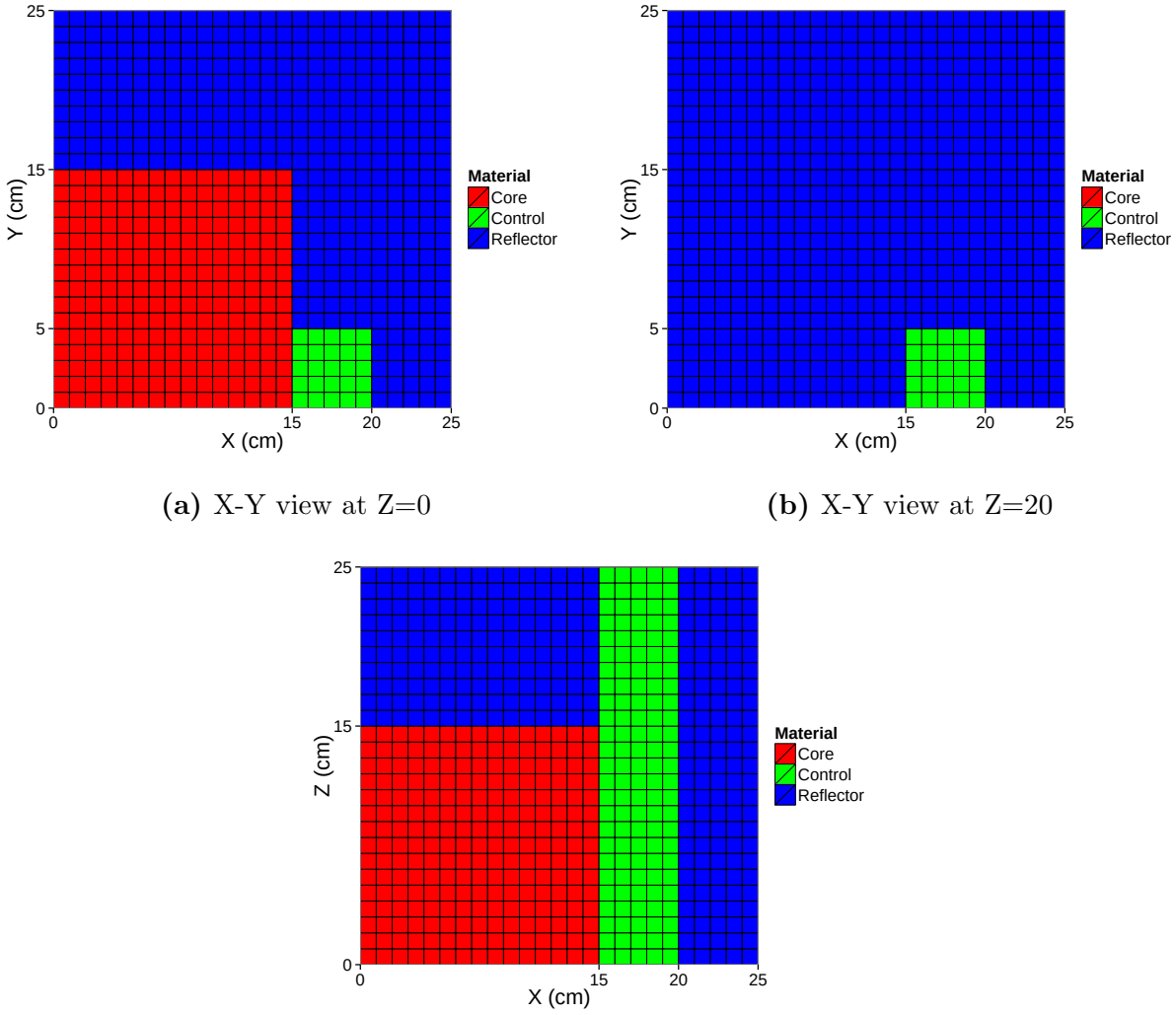
Figure 4.43 ACS iteration details for group 7 (5-6 MeV) in the VENUS problem

number 10 at which point the quadrature is changed to S_4. Once at S_4, the iteration time drop to below 2 seconds. At this point there are no more estimates of the quadrature error since there is no further quadrature to change to. The iterations are stopped when the iterative change reaches the flux tolerance (i.e., 5×10^{-4}).

4.3 Takeda Criticality Benchmark

The first criticality problem to be tested is a small LWR benchmark proposed by Takeda and Ikeda and is now part of the OECD/NEA 3-D Neutron Transport Benchmarks [54]. The model represents a homogenized version of the Kyoto University Critical Assembly (KUCA), which is a very small research reactor. There are several variations of the benchmark, but here, only the model with the control element fully inserted was used. The model mesh

distribution is shown in Figure 4.44. Two X-Y slices are shown, one inside the core at $Z = 0$ and one above the core at $Z = 20$. Also shown is an X-Z slice at $Y = 0$ that shows the control element sticking up above the core. The mesh size is a uniform 1.0 cm.



(a) X-Y view at Z=0

(b) X-Y view at Z=20

(c) X-Z view at Y=0

Figure 4.44 Takeda material mesh distributions

Two-group cross sections for the problem are given in Table 4.17. Only P_0 scattering is used, but transport-corrected cross sections are used to account for anisotropic scattering.

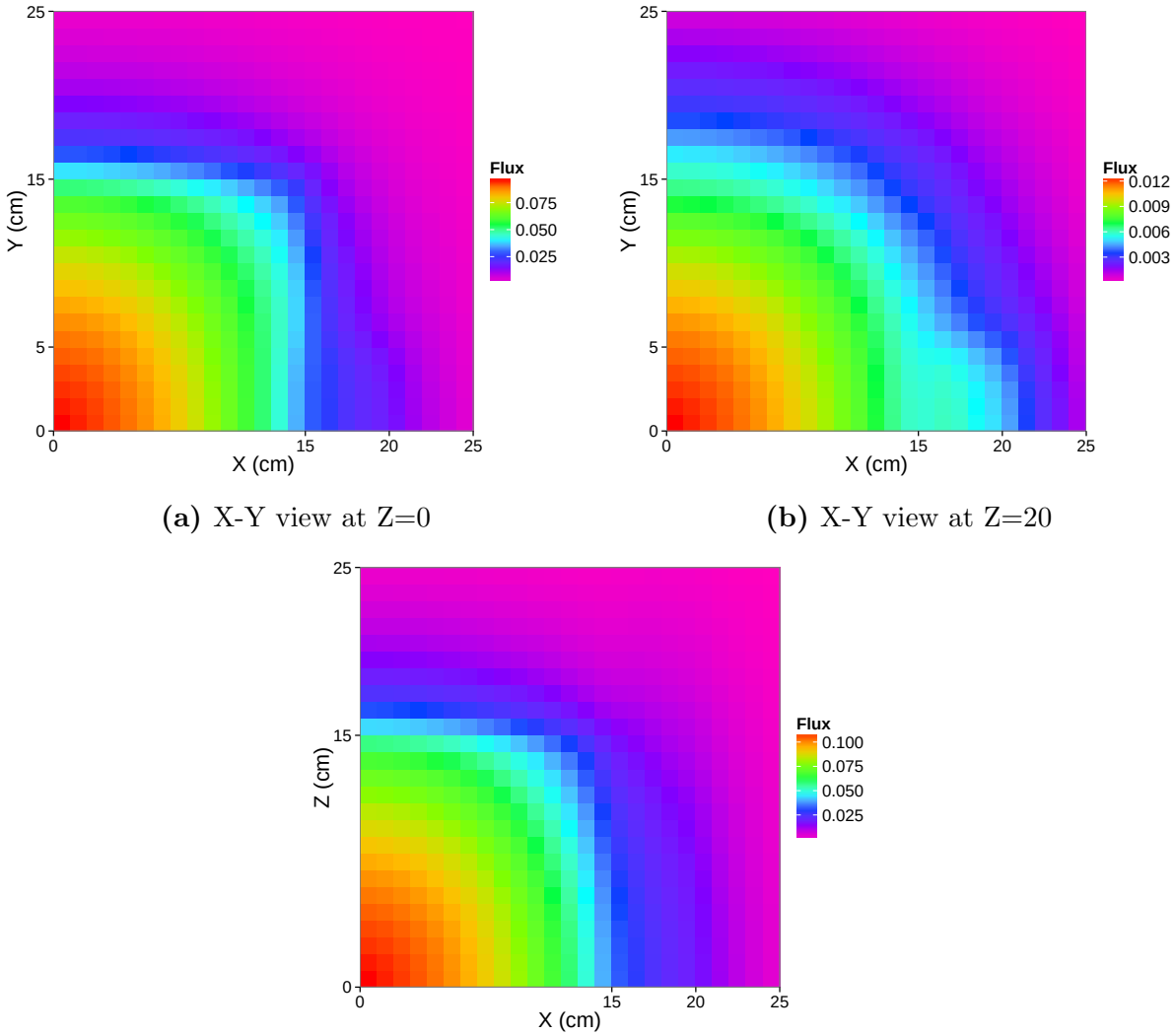
Table 4.17 Takeda Problem Cross Sections (cm^{-1})

Material	Group	σ_a	$\nu\sigma_f$	σ_t	$\sigma_{s,g\to g}$	$\sigma_{s,1\to 2}$
Core	1	8.53E-03	9.09E-03	2.24E-01	1.92E-01	2.28E-02
	2	1.58E-01	2.90E-01	1.04E+00	8.80E-01	
Reflector	1	4.16E-04	0	2.50E-01	1.93E-01	5.65E-02
	2	2.03E-02	0	1.64E+00	1.62E+00	
Control	1	1.74E-02	0	8.52E-02	6.77E-02	6.45E-05
	2	1.82E-01	0	2.17E-01	3.52E-02	

As a reference, TITAN was used to calculate the eigenvalue for the system. The mesh size is fixed at 1.0 cm. The angular quadrature was varied between S_{10} and S_4. The outer iteration tolerance was set at 1×10^{-6}, with an inner tolerance of 1×10^{-5}. Step differencing was used to minimize the difference between source iteration and ACS for comparison purposes. Reference flux values at S_{10} as well as material mesh distributions are shown in Figure 4.45 for the fast group and Figure 4.46 for the thermal group.

These flux plots show that as expected, the fast flux is highest in the core and drops in the moderator. The thermal flux is highest in the moderator and drops significantly in the core and especially the control region.

Next, the standard TITAN code was compared while varying two parameters - the quadrature order and the maximum number of inner iterations per outer iteration. The maximum number of inner iterations is important because when using the source iteration, it is not required to completely converge on the within-group scatter (i.e., the inner iterations) before updating the fission source and performing a new outer iteration. The scattering source is saved between outer iterations and can significantly speed up the process. The comparisons between the k-eigenvalues and mean relative flux errors are shown in Table 4.18. For the difference between eigenvalues, the percent millirho (pcm) unit is used, where 1 $pcm = 0.00001\Delta k/k$.



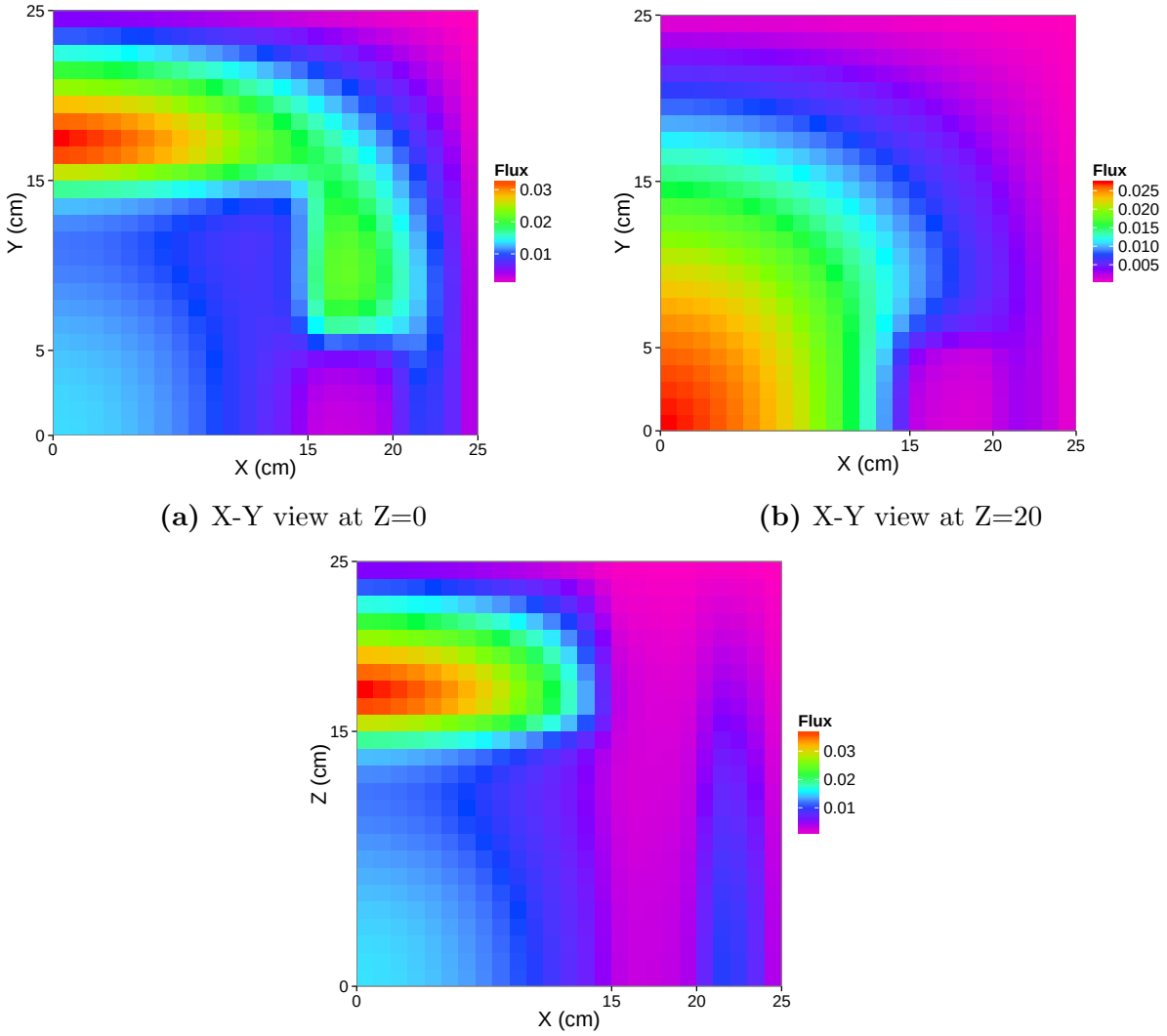
(a) X-Y view at Z=0

(b) X-Y view at Z=20

(c) X-Z view at Y=0

Figure 4.45 Takeda reference model fast flux (group 1)

As expected, with lowered quadrature order, the eigenvalue and flux errors all increase, while the computation time decreases. The error for group 1 is higher as it is more streaming-dominant. The benefits of using a low number of inner iterations per outer are very apparent. Approximately the same number of outer iterations are used, but the total number of inner iterations is decreased significantly - almost a factor of 3.



(a) X-Y view at Z=0

(b) X-Y view at Z=20

(c) X-Z view at Y=0

Figure 4.46 Takeda reference model thermal flux (group 2)

A comparison between the flowcharts for ACS and standard TITAN for the eigenvalue problem (Figure 3.6 and Figure 3.7) shows that ACS requires a reset of the source between outer iterations to only include the fission source and not retain the scatter information. This is a problem for ACS, since the inner iterations must be completely converged within each outer iteration to obtain a good result. In order to slightly compensate for this, another adaptive scheme was adopted for ACS. For this, the maximum number of inner iterations per outer starts at 5, and doubles after each outer iteration until the max is achieved (1000). This

Table 4.18 Standard TITAN Results for Takeda Benchmark

Quad.	Max Inn.	k	diff (pcm)	Grp 1 RMSE	Grp 2 RMSE	CPU Time (s)	Total Out.	Total Inn.	Time /inner
S10	1000	0.91744	-	-	-	271.2	15	1572	0.1725
S8	1000	0.91720	-25.9	2.40E-03	1.20E-03	178.0	15	1564	0.1138
S6	1000	0.91666	-85.5	7.71E-03	3.66E-03	106.6	16	1553	0.0686
S4	1000	0.91549	-212.4	2.04E-02	8.53E-03	53.6	16	1533	0.0350
S10	20	0.91744	-	-	-	102.7	16	599	0.1715
S8	20	0.91720	-26.1	2.40E-03	1.20E-03	68.5	16	598	0.1145
S6	20	0.91666	-85.6	7.71E-03	3.66E-03	41.2	16	598	0.0689
S4	20	0.91549	-212.7	2.04E-02	8.52E-03	19.7	15	572	0.0344

allows the fission source distribution to converge slightly before "wasting" too many inner iterations. The results for the ACS on this problem are shown in Table 4.19. These tests were performed for various adaptive tolerances and using the S_{10}/S_4 quadrature combination. The same outer tolerance as for SI was used.

Table 4.19 ACS Results for Takeda Benchmark

Quad.	Adapt Inners?	ϵ	k	diff (pcm)	Grp 1 Flux Err	Grp 2 Flux err	CPU Time (s)	Total Out.	Total Inn.	Time /inner
S10/S4	No	10^{-5}	0.91743	-1.2	3.41E-05	2.03E-04	896.5	16	6633	0.1352
S10/S4	Yes	10^{-5}	0.91743	-1.1	5.10E-05	2.71E-04	606.6	17	5113	0.1186
S10/S4	Yes	10^{-4}	0.91743	-1.5	2.06E-04	7.72E-04	445.2	17	5113	0.0871
S10/S4	Yes	10^{-3}	0.91760	17.3	1.50E-03	8.69E-03	235.3	17	5103	0.0461
S10/S4	Yes	10^{-2}	0.91685	-65.0	4.19E-03	9.85E-03	212.5	17	5103	0.0416

The first thing to note is the difference between the adaptive inner iteration limit - the total number of iterations is reduced by almost 25% with very little effect on error. Overall, ACS can produce very accurate results compared to standard TITAN, but for this case, it is slower for almost every case. Although the average time per iteration is low, a huge number of inner iterations are required (approximately 5000 for ACS compared to 600 for SI).

The ACS method can obtain the correct answer, but is not very efficient for this problem. This could be partly due to the fact that the quadrature requirements are fairly low for this problem - it is homogenized and highly scattering. Perhaps it could be more efficient

on different types of eigenvalue problems that are dominated by streaming. However the inability to save the scattering source is a huge drawback.

4.4 PWR Pin-cell Criticality Benchmark

In the previous section, it was determined that the ACS method can have trouble obtaining fast results with eigenvalue problems, at least on homogenized models with relatively low quadrature requirements. The main factor in slowing the ACS method is due to the inability to save the scattering source between outer iterations. Here, we examine a problem without spatial homogenization - a simple PWR fuel pin-cell, as shown in Figure 4.47.

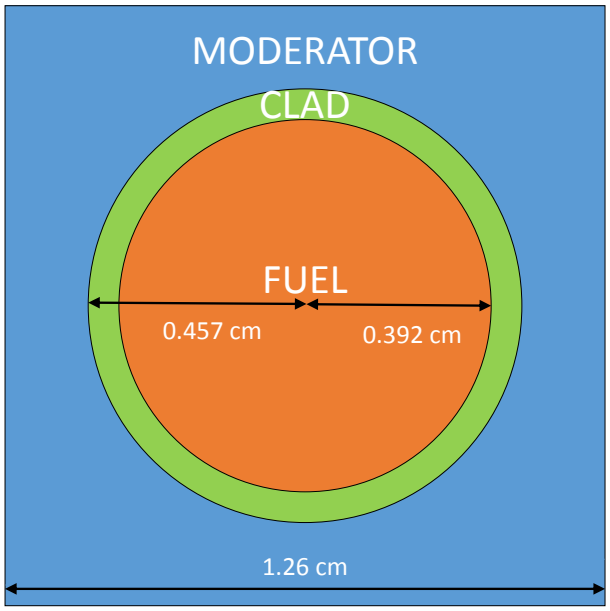


Figure 4.47 PWR pin-cell model description

Three-group cross sections were obtained for this problem from a simple SCALE/TRITON [29] model. These cross sections are given in Table 4.20.

First, a reference eigenvalue calculation was performed using the standard TITAN code at S_{20}. Material and flux distributions for this case are shown in Figure 4.48. For all cases, the

Table 4.20 PWR Pin Cross Sections (cm^{-1})

Material	Group	σ_a	$\nu\sigma_f$	σ_t	$\sigma_{s,g\to g}$	$\sigma_{s,g\to g+1}$
Fuel	1	1.25E-2	2.97E-2	2.72E-1	2.10E-1	5.02E-2
	2	4.17E-2	2.64E-2	4.92E-1	4.49E-1	1.18E-3
	3	3.52E-1	6.62E-1	7.66E-1	4.14E-1	-
Clad	1	3.53E-4	0	2.00E-1	1.77E-1	2.26E-2
	2	1.87E-3	0	3.41E-1	3.38E-1	3.91E-4
	3	4.18E-3	0	2.75E-1	2.71E-1	-
Moderator	1	3.46E-4	0	1.71E-1	9.37E-2	7.68E-02
	2	2.55E-4	0	7.46E-1	7.09E-1	3.62E-02
	3	8.34E-3	0	1.75E+0	1.74E+0	-

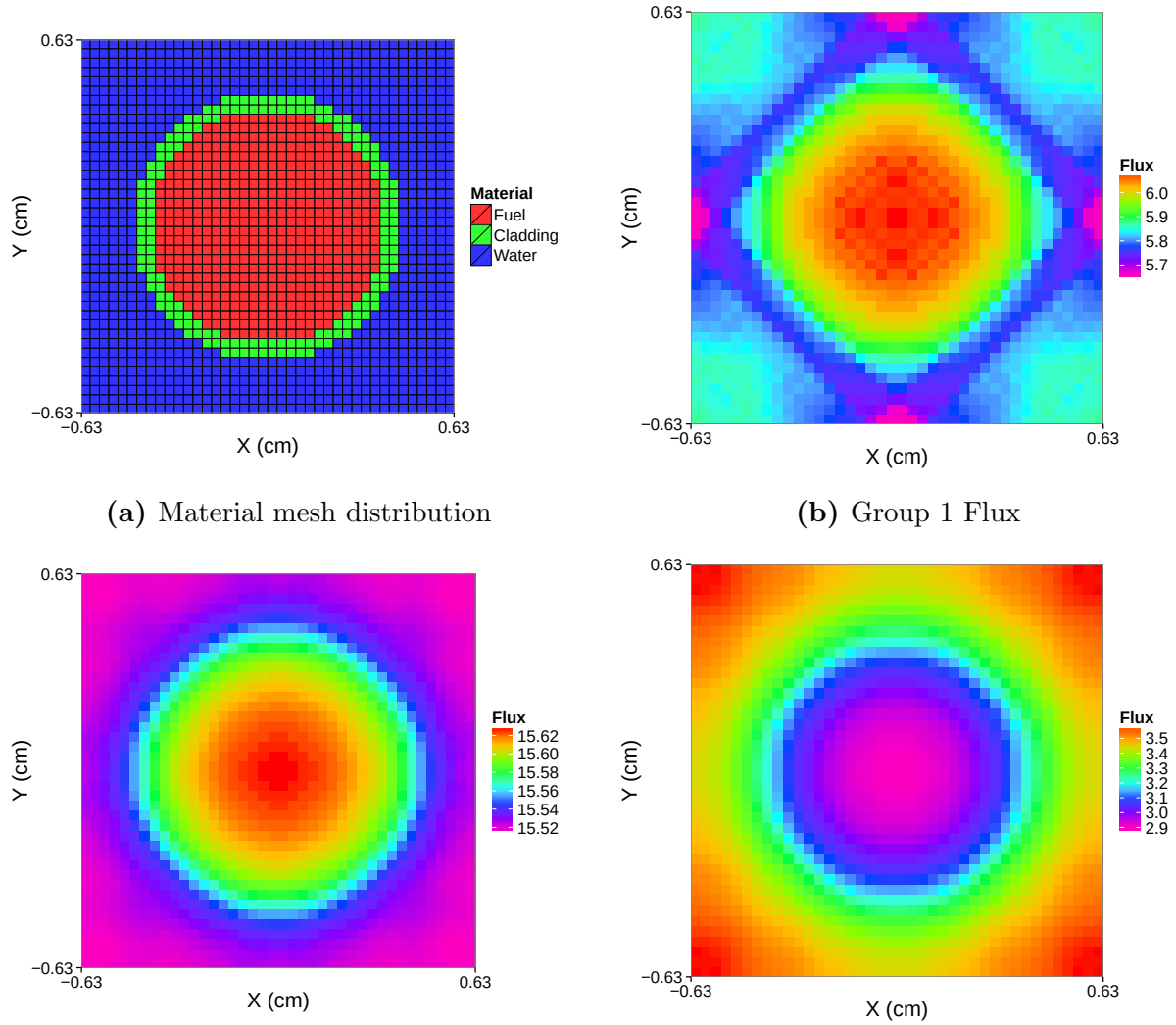
outer (k) tolerance was 10^{-7} with a inner (flux) tolerance of 10^{-6} and 20 maximum inner iterations per outer.

These calculations were repeated at several other quadrature levels. Detailed results are given in Table 4.21. As expected, the error goes up and the time goes down as the quadrature order decreases.

Table 4.21 SI Results for PWR Pin-cell calculation

Quad. Order	k	Error (pcm)	Time (s)	Outer Iterations	Inner Iterations	Time/inner (s)
S_{20}	1.38656	-	701.4	21	1260	0.557
S_{12}	1.38680	17	267.9	21	1260	0.213
S_{10}	1.38706	36	191.6	21	1260	0.152
S_8	1.38753	70	127.6	21	1260	0.101
S_6	1.38836	130	76.9	21	1260	0.061
S_4	1.38918	189	39.2	21	1260	0.031

Next, the calculations were performed using ACS with S_4 as the second quadrature. The adaptive tolerance was set at 10^{-3}. These results are shown in Table 4.22. For some parameters, ACS provides a more efficient solution (notably, the S_{10}/S_4 and S_{12}/S_4 cases). This is better seen in a plot of error vs. computation time, as in Figure 4.49. Although ACS can



(a) Material mesh distribution

(b) Group 1 Flux

(c) Group 2 Flux

(d) Group 3 Flux

Figure 4.48 Material meshing and reference flux for pin-cell problem

sometimes perform better than a standard source iteration for eigenvalue problems, it is not nearly as efficient as for shielding problems.

Table 4.22 ACS Results for PWR Pin-cell calculation

Quad. Order	Adaptive Tolerance	k	Error (pcm)	Time (s)	Outer Iterations	Inner Iterations	Time/inner (s)
S_{12}/S_4	10^{-3}	1.38698	30	174.4	9	3300	0.053
S_{10}/S_4	10^{-3}	1.38723	48	139.6	9	3300	0.042
S_8/S_4	10^{-3}	1.38769	81	115.7	9	3300	0.035
S_6/S_4	10^{-3}	1.38854	143	99.1	9	3300	0.030

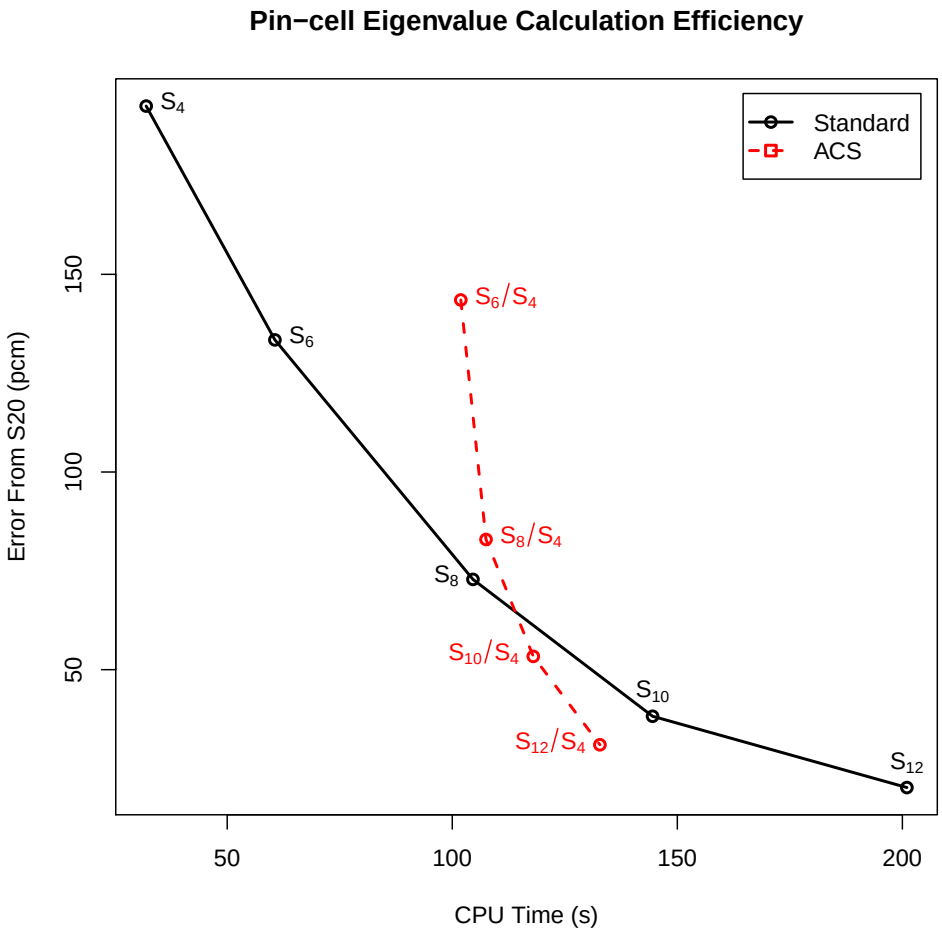


Figure 4.49 Computational Efficiency of ACS for the PWR Pin problem

Chapter 5

Conclusions and Future Work

In all types of calculations, there is a trade-off between speed and accuracy. More accurate solutions are always obtainable by taking more computational effort. In the development of new methods, the important thing is the efficiency - that is, what is the speed for a given level of accuracy, or vice versa. In this dissertation, a method was developed to improve the efficiency of discrete ordinates calculations by optimizing the angular quadrature. By separating the flux components by collision order, it is possible to use high-order quadrature initially when it is most needed, and shift to lower order quadrature for the remaining calculations. This quadrature adaptation is performed based on an intelligent automated estimate of the error and a user-adjusted tolerance. Although the method requires user-defined quadrature orders and adaptive tolerance, the choices of these do not greatly affect the efficiency of the adaptive collision source method as opposed to the alternative of using a uniform quadrature set.

This methodology was developed and incorporated into the 3-D, multi-group, parallel, hybrid transport code TITAN and tested on a variety of problems. The 3-D, multi-group implementation means that ACS is practically applicable to real-world problems and is not just a theoretical method. For shielding problems that were tested (a simple box-in-a-box problem and the VENUS-2 dosimetry benchmark), the ACS method proved to be very efficient.

Depending on the problem specifics and the parameters chosen, ACS can perform between 1.5-4 times faster than the standard source iteration. Equivalently, an accuracy up to 4 times better is obtainable in the same amount of computation time. Further, the method appears to be very stable. There were no convergence problems noted on any problem that was given to it.

The ACS method does not require any special knowledge beyond that of being able to run a standard discrete ordinates code. The two arbitrary parameters, additional quadrature order(s) and adaptive tolerance, do not need to be carefully chosen. If the user picks reasonable values (a second quadrature approximately half the order of the first, and adaptive tolerance in the range of 0.001-0.1), then a good speedup should be obtainable.

For eigenvalue problems, ACS can sometimes be more efficient, but is much more problem and parameter dependent. As a result ACS is only recommended for shielding problems, but can provide an excellent speedup on these problems.

5.1 Future Work

Much work has completed been to investigate the method and to optimize the performance. However, there are still several issues that can be investigated for future work:

- Investigate the performance of the method on other problems, e.g., medical physics applications
- In the method, the iterative flux error is estimated as a global quantity. This error could be calculated on a mesh-wise basis and used as a correction term for the final result.
- Devise a way to intelligently select the second (or third, or higher) quadrature set(s) based on some adaptive procedure. Results from the box problem indicate that having

a second quadrature order of about 1/2 of the primary quadrature is approximately the most efficient, but this might not be true for all problems.

- Allow spatially dependent quadrature adaptation. This would require a re-work of the projection of quadrature sets between spatial regions.
- Investigate alternative error estimates (e.g., error in current as opposed to scalar flux)
- Implement ACS with a different angular discretization method. The ACS method in theory is not limited to discrete ordinates calculations - any method that uses a source iteration should be adaptable to the ACS methodology. This includes, but is not limited to, angular finite elements or the method of characteristics.

List of References

[1] G. I. Bell and S. Glasstone. *Nuclear Reactor Theory*. Van Nostrand Reinhold Company (1970).

[2] N. Metropolis and S. Ulam. "The monte carlo method." *Journal of the American statistical association*, **44(247)**: pp. 335–341 (1949).

[3] G. Sjoden and A. Haghighat. "PENTRAN - a 3-D cartesian parallel SN code with angular, energy, and spatial decomposition." In: *Proceedings of the Joint International Conference on Mathematical Methods and Supercomputing in Nuclear Applications, Saratoga Springs, NY, Oct. 6-10,*, volume 2, (pp. 1267–1276) (1997).

[4] C. Yi and A. Haghighat. "A three-dimensional block-oriented hybrid discrete ordinates and characteristics method." *Nuclear Science and Engineering*, **164(3)**: pp. 221–247 (2010).

[5] W. A. Rhoades and R. L. Childs. *The TORT three-dimensional discrete ordinates neutron/photon transport code. Technical report*, Oak Ridge National Lab., TN (USA) (1987).

[6] T. A. Wareing, J. M. McGhee, and J. E. Morel. "ATTILA: A 3-D unstructured tetrahedral-mesh sn code." In: *3-D Deterministic Radiation Transport Computer Programs: Features, Applications and Perspectives: 2-3 December 1996, OECD Château de*

la Muette, Paris, France, (p. 49). Organisation for Economic Co-operation and Development (1997).

[7] S. Chandrasekhar. *Radiative transfer*. New York: Dover (1950).

[8] B. Carlson. *Transport Theory: Discrete Ordinates Quadrature over the Unit Sphere. Technical report*, Los Alamos Scientific Lab., N. Mex. (1970).

[9] K. D. Lathrop. "Remedies for ray effects." *Nuclear Science and Engineering*, **45**: pp. 255–268 (1971).

[10] E. Lewis and W. Miller. *Computational methods of neutron transport*. John Wiley and Sons, Inc., New York, NY (1984).

[11] K. Lathrop. "Ray effects in discrete ordinates equations." *Nuclear Science and Engineering*, **32**: pp. 357–269 (1968).

[12] C. D. Ahrens. "Highly efficient, exact quadratures for three-dimensional discrete ordinates transport calculations." *Nuclear Science and Engineering*, **170(1)**: pp. 98–101 (2012).

[13] K. L. Manalo. *Detailed Analysis of Phase Space Effects in Fuel Burnup/depletion for PWR Assembly & Full Core Models Using Large-scale Parallel Computation*. Ph.D. thesis, Georgia Institute of Technology (2013).

[14] J. J. Jarrell. *An Adaptive Angular Discretization Method for Neutral-Particle Transport in Three-Dimensional Geometries*. Ph.D. thesis, Texas A&M University (2010).

[15] J. J. Jarrell and M. L. Adams. "Discrete-ordinates quadrature sets based on linear discontinuous finite elements." In: *International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering (M&C 2011), Rio de Janeiro, RJ, Brazil* (2011).

[16] G. Longoni and A. Haghighat. "Development of new quadrature sets with the 'ordinate splitting' technique." In: *Proceedings of the 2001 ANS International Meeting on Mathematical Methods for Nuclear Applications (M&C),Salt Lake City, Utah, USA, Sept. 2001* (2001).

[17] G. Longoni and A. Haghighat. "Development and application of the regional angular refinement technique and its application to non-conventional problems." In: *Proceedings of PHYSOR 2002, Seoul, Korea, October 2002* (2002).

[18] G. Longoni and A. Haghighat. "Simulation of a CT-scan device with PENTRAN using the new Pn-Tn quadrature set and an angular regional refinement technique." In: *Proceedings of the 12th Biennial RPSD Topical Meeting, Santa Fe, NM, April* (2002).

[19] G. Longoni. *Advanced Quadrature Sets, Acceleration, and Preconditioning Techniques for the Discrete Ordinates Method in Parallel Computing Environments*. Ph.D. thesis, University of Florida (2004).

[20] G. Sjoden and A. Haghighat. "PENTRAN: A three-dimensional scalable transport code with complete phase-space decomposition." In: *Transactions of the American Nuclear Society*, volume 74 (1996).

[21] P. N. Brown, B. Chang, and C. Clouse. *Locally Refined Quadrature Rules for SN Transport. Technical report*, Report UCRL-JRNL-220755, Lawrence Livermore National Laboratory (2006).

[22] J. C. Stone and M. L. Adams. "Progress on adaptive discrete-ordinates algorithms and strategies." In: *Nuclear Mathematical and Computational Sciences Meeting, Gatlinburg, TN* (2003).

[23] J. Stone. *Adaptive Discrete-ordinates Algorithms and Strategies*. Ph.D. thesis, Texas A&M University (2007).

[24] R. Alcouffe. "A first collision source method for coupling monte carlo and discrete ordinates for localized source problems." In: *Monte-Carlo Methods and Applications in Neutronics, Photonics and Statistical Physics*, volume 240, (pp. 352–366). Springer Berlin / Heidelberg. ISBN 978-3-540-16070-0 (1985).

[25] R. Alcouffe, R. O'Dell, and F. Brinkley Jr. "A first-collision source method that satisfies discrete S n transport balance." *Nuclear Science and Engineering*, **105(2)**: pp. 198–203 (1990).

[26] T. Wareing, J. Morel, and D. Parsons. "A first collision source method for ATTILA, an unstructured tetrahedral mesh discrete ordinates code." *Proceedings of the 1998 American Nuclear Society Radiation Protection and Shielding Division Topical Conference, Nashville TN*, **1**: pp. 376–382 (1998).

[27] R. Lillie. "GRTUNCL3D: A discontinuous mesh three-dimensional first collision source code." In: *Proceedings of the 1998 American Nuclear Society Radiation Protection and Shielding Topical Conference, Nashville, TN, April 19-23*, (pp. 1–5) ((1998)).

[28] M. Abramowitz and I. A. Stegun. *Handbook of mathematical functions*. New York: Dover (1972).

[29] S. M. Bowman. "SCALE 6: comprehensive nuclear safety analysis code system." *Nuclear Technology*, **174(2)**: pp. 126–148 (2011).

[30] A. Alpan. *An advanced methodology for generating multigroup cross sections for shielding calculations*. PhD dissertation, The Pennsylvania State University (2003).

[31] A. Alpan and A. Haghigat. "Development of the CPXSD methodology for generation of fine-group libraries for shielding applications." *Nuclear Science and Engineering*, **149(1)**: pp. 51–64 (2005).

[32] W. H. Reed. "Spherical harmonic solutions of the neutron transport equation from discrete ordinate codes." *Nuclear Science and Engineering*, **49**: p. 10 (1972).

[33] J. Fletcher. "The solution of the multigroup neutron transport equation using spherical harmonics." *Nuclear Science and Engineering*, **84(1)**: pp. 33–46 (1983).

[34] L. L. Briggs, W. F. Miller, and L. L. Lewis. "Ray effect mitigation in discrete ordinate-like angular finite element approximations in neutron transport." *Nuclear Science and Engineering*, **57**: p. 205 (1975).

[35] B. G. Petrović and A. Haghighat. "Analysis of inherent oscillations in multidimensional Sn solutions of the neutron transport equation." *Nuclear Science and Engineering*, **124(1)**: pp. 31–62 (1996).

[36] B. G. Petrović and A. Haghigat. "New directional theta-weighted (DTW) differencing scheme and reduction of estimated pressure vessel fluence uncertainty." In: *Proceedings of the 9th International Symposium on Reactor Dosimetry, Prag*, (pp. 746–753) (1996).

[37] K. Lathrop. "Spatial differencing of the transport equation: positivity vs. accuracy." *Journal of computational physics*, **4(4)**: pp. 475–498 (1969).

[38] R. Alcouffe *et al.* "Computational efficiency of numerical methods for the multigroup, discrete-ordinates neutron transport equations: the slab geometry case." *Nuclear Science and Engineering*, **71(2)**: pp. 111–127 (1979).

[39] W. H. Reed *et al. TRIPLET: a two-dimensional, multigroup, triangular mesh, planar geometry, explicit transport code. Technical report*, Los Alamos Scientific Lab., N. Mex.(USA) (1973).

[40] G. E. Sjoden and A. Haghighat. "The exponential directional weighted (EDW) sn differencing scheme in 3-d cartesian geometry." In: *Proceedings of the joint international comference on mathematical methods and supercomputing in nuclear applications* (1997).

[41] G. E. Sjoden. "An efficient exponential directional iterative differencing scheme for three-dimensional sn computations in xyz geometry." *Nuclear Science and Engineering*, **155(2)**: pp. 179–189 (2007).

[42] W. J. Walters and A. Haghighat. "Development of the adaptive collision source (ACS) method for discrete ordinates." In: *International Conference on Mathematics and Computational Methods Applied to Nuclear Science & Engineering (M&C 2013), Sun Valley, Idaho, USA, May 5-9* (2013).

[43] W. J. Walters and A. Haghighat. "Performance of the adaptive collision source (ACS) method for discrete ordinates in parallel environments." In: *SNA+ MC 2013-Joint International Conference on Supercomputing in Nuclear Applications+ Monte Carlo, Paris, France, Oct 27-31*, (p. 04106). EDP Sciences (2013).

[44] W. J. Walters and A. Haghighat. "ACS algorithm in discrete ordinates for pressure vessel dosimetry." In: *Proc. 15th International Symposium on Reactor Dosimetry, Aix en Provence, France, May 18-23* (2014).

[45] C. Yi and A. Haghighat. "A hybrid block-oriented discrete ordinates and characteristics method algorithm for solving linear boltzmann equation." In: *Proc. Int. Conf. M&C*, volume 77 (2007).

[46] C. Yi. *Hybrid discrete ordinates and characteristics method for solving the linear Boltzmann equation.* Ph.D. thesis, University of Florida (2007).

[47] W. Oberkampf and C. Roy. *Verification and validation in scientific computing.* Cambridge University Press, Cambridge (2010).

[48] B. Noble and J. W. Daniel. *Applied linear algebra*, volume 3. Prentice-Hall New Jersey (1988).

[49] W. Gropp *et al.* "A high-performance, portable implementation of the mpi message passing interface standard." *Parallel computing*, **22(6)**: pp. 789–828 (1996).

[50] G. M. Amdahl. "Validity of the single processor approach to achieving large scale computing capabilities." In: *Proceedings of the April 18-20, 1967, spring joint computer conference*, (pp. 483–485). ACM (1967).

[51] C. Y. Han *et al. VENUS-2 MOX-Fuelled Reactor Dosimetry Calculations Benchmark Specification. Technical report*, OECD/NEA, NEA/NSC/DOC(2004)6 (2004).

[52] J. E. White *et al. BUGLE-96: a revised multigroup cross section library for LWR applications based on ENDF/B-VI release 3. Technical report*, Oak Ridge National Lab. 37831-6363 (1996).

[53] M. Chadwick *et al.* "ENDF/B-VII: Next generation evaluated nuclear data library for nuclear science and technology." *Nuclear data sheets*, **107(12)**: pp. 2931–3060 (2006).

[54] T. Takeda and H. Ikeda. "3-D neutron transport benchmarks." *Journal of Nuclear Science and Technology*, **28(7)**: pp. 656–669 (1991).