Final Report: 1-D Discrete Ordinates Code

NE515: Radiation Interaction and Transport

Liam Pohlmann[1]

[1]University of New Mexico, Department of Nuclear Engineering

Due: 11 December 2024

Contents

List of Figures

Preface

The code for this project was developed and uploaded to a GitHub repository for improved code readability and version control. The hope is that the results of this work may go on to help other students in the industry who are also getting started learning about radiation transport.

Part 1: *Premise and Theory*

The purpose of this report is to outline the steps taken and results generated for the solution of the linear Boltzmann transport equation on a 1-D slab subject to varying conditions. In the application of radiation transport, the general 1-D, time-independent, monoenergetic transport equation can be written as:

$$\mu \frac{\partial \psi}{\partial z} + \Sigma_t(z)\psi(z,\mu) = \left(\frac{\Sigma_s + \nu\Sigma_f}{4\pi}\right)\phi(z) + S(z) \tag{1}$$

were $\mu = \cos\left(\vec{\Omega}' \cdot \vec{\Omega}\right)$, the cosine of the scattering angle, Σ_t is the total macroscopic cross-section, Σ_s is the zero$^{\text{th}}$-order macroscopic scattering cross-section, $\psi(z,\mu)$ is the angular neutron flux, and $S(z)$ is a generalized, isotropic volumetric source. Additionally, the variable of interest, $\phi(z)$, is the scalar neutron flux, defined as:

$$\phi(z) = \int_{4\pi} \psi(z,\vec{\Omega})\mathrm{d}\vec{\Omega}. \tag{2}$$

The above is subject to boundary conditions on the edges of the slab, namely $\psi(z=0,\mu)$ and $\psi(z=L,\mu)$ for a slab of length L. Equation (1) is known as an *integro-differential equation*, as it is an equation that contains a function, its derivative, and its integral. To solve this, one must discretize both the derivatives and integrals in a set of algebraic relations, then solve.

First, the domain is discretized uniformly using a finite volume approach in the spacial axis. That is, each volume (in 1-D geometry, this is just a line segment) holds the average function value as numerically computed at its *center*:

$$\bar{\psi}_{n,i} = \frac{1}{\Delta z}\int_{z_{i-1/2}}^{z_{i+1/2}} \psi_n(z)\mathrm{d}z \tag{3}$$

where $\bar{\psi}_{n,i}$ is the *cell-averaged* angular flux (and will be henceforth written as $\psi_{n,i}$) at

$z = \Delta z \times (i + 1/2)$ and $\mu = \mu_n \in [\mu_1, \mu_2, \ldots, \mu_N]$, Δz_i is the cell width (the index i gets dropped when each cell is equidistant), and $i \in [1, 2, \ldots, I]$. This implies that functional relations must be made not only for the function values themselves, but also of the numerical "fluxes" at the boundaries of each cell.

Though discretization of the slab spatially is straightforward, the choice of discrete angular values is an opportunity to improve numerical accuracy while also reducing computational effort. Gaussian quadrature is a technique using in numerical integration in which the error is minimized by selecting particular weights and multiplying them by function values evaluated at specially-selected points. The generalized case can be written as [**?**]:

$$\int_{-1}^{1} f(x)\mathrm{d}x = \sum_{k=1}^{K-1} w_k f(x_k) \tag{4}$$

The values of the x_k's (the "nodes") turn out to be zeros of Legendre polynomials of order K on the standard interval $[-1, 1]$. Legendre polynomials are a family of equations that share a common set of favorable properties, such as orthogonality, and can be generated using Rodrigues' formula [**?**]:

$$P_K(x) = \frac{1}{2^K} \sum_{k=0}^{K} \binom{n}{k}^2 \left(\frac{x-1}{2}\right)^k. \tag{5}$$

The weights for Gauss quadrature (the c_k's) can be calculated with [**?**]:

$$c_k = \frac{2}{(1 - x_k^2)\left[P'(x_k)\right]^2} \tag{6}$$

Instead of directly applyingEquations (5) and (6), a Gaussian quadrature library for `C++` was implemented for numerical integration.

By assuming azimuthal independence, we rewrite Equation (2) as:

$$\phi(z) = \int_{4\pi} \psi(z, \vec{\Omega})\mathrm{d}\vec{\Omega} = \int_0^{2\pi} \mathrm{d}\vec{\Omega} \int_{-1}^{1} \psi(z, \mu)\mathrm{d}\mu. \tag{7}$$

Using the previous notation, Equation (1) can be written in terms of discrete scattering angles, or *discrete ordinates*, as:

$$\mu_n \frac{\partial \psi_n(z)}{\partial z} + \Sigma_t(z)\psi_n(z) = \frac{\Sigma_s}{2} \sum_{k=1}^{N} w_k \psi_k(z) + S(z) \tag{8}$$

Letting:

$$q^{(j-1)}(z) = \frac{\Sigma_s}{2} \sum_{k=1}^{N} w_k \psi_k^{(j)}(z) + S(z) \tag{9}$$

and:

$$q_{n,i} = \bar{q}(z) = \frac{1}{\Delta z_i} \int_{z_{i-1/2}}^{z_{i+1/2}} q_n(z) \mathrm{d}z \tag{10}$$

a source iteration is created with:

$$\mu_n \frac{\mathrm{d}\psi_n^{(j)}(z)}{\mathrm{d}z} + \Sigma_t(z)\psi_n^{(j)}(z) = q^{(j-1)}(z) \tag{11}$$

By integrating Equation (11) over an arbitrary cell, we find:

$$\mu_n \left(\psi_{n,i+1/2}^{(j)} - \psi_{n,i-1/2}^{(j)} \right) + \Sigma_{t,i} \Delta z_i \psi_{n,i}^{(j)} = \Delta z_i q_{n,i}^{(j-1)} \tag{12}$$

To iterate the source, first the solution is swept in one direction along z with the flow of neutrons, then is swept in the opposite direction to solve for the neutrons flowing the other way. We define the following differencing closure:

$$\psi_{n,i} = (1-\alpha)\psi_{n,i+1/2} + \alpha\psi_{n,i-1/2}, \quad \mu_n > 0 \tag{13a}$$

$$\psi_{n,i} = \alpha\psi_{n,i+1/2}^{+}(1-\alpha)\psi_{n,i-1/2}, \quad \mu_n < 0 \tag{13b}$$

corresponding to forward and backward particle flow, respectively. When $\alpha = 1/2$, Equation (13) is known as *diamond differencing*, while when $\alpha = 0$, Equation (13) is known as *step differencing*. These correspond to quadratic and linear convergence, respectively. Thus, the following two algorithms were implemented:

$$\psi_{n,i}^{(j)} = \left(\frac{1}{1 + \frac{\Sigma_{t,i}\Delta z_i}{2\mu_n}} \right) \left(\psi_{n,i-1/2}^{(j)} + \frac{\Delta z_i q_{n,i}^{(j-1)}}{2\mu_n} \right), \quad n = 1, 2, \ldots, \frac{N}{2} \tag{14a}$$

$$\psi_{n,i}^{(j)} = \left(\frac{1}{1 + \frac{\Sigma_{t,i}\Delta z_i}{2|\mu_n|}} \right) \left(\psi_{n,i+1/2}^{(j)} + \frac{\Delta z_i q_{n,i}^{(j-1)}}{2|\mu_n|} \right), \quad n = \frac{N}{2} + 1, \frac{N}{2} + 2, \ldots, N \tag{14b}$$

along with:

$$\psi_{n,i+1/2} = 2\psi_{n,i} - \psi_{n,i-1/2} \tag{15}$$

1.1: Numerical Considerations

Since negative flux is non-physical but numerically possible, particularly if the mesh is not fine enough for larger α. When such an event occurs, the code was implemented with a "negative flux fixup", which changes from diamond differencing to step differencing when the angular flux dips below zero.

In addition, all boundary source terms were normalized to 1 for convenience. If $\Gamma_{n,1/2} = \Gamma_{n,L}$ and $\Gamma_{n,I+1/2} = \Gamma_{n,L}$ are boundary flux terms, the following algorthm was implemented:

$$\int_{4\pi} \Gamma_{L/R,n} \mathrm{d}\vec{\Omega} \approx 2\pi \sum_{n=0}^{N/2} w_n \mu_n \Gamma_{L/R,n} \equiv \gamma_n \tag{16}$$

Then: $\Gamma_{L/R,n} \to \frac{\Gamma_{L/R,n}}{\gamma_N}$.

Part 2: Code Verification: Method of Manufactured Solutions

To verify the above numerical, the Method of Manufacture Solutions was implemented. That is, a solution was specified of the separable form where C is a constant:

$$\psi^m(z,\mu) = Cf(z)g(\mu) \tag{17}$$

and substituted into Equation (1). Assuming no fission and solving for the manufactured volumetric source:

$$Q^m(z,\mu) = \mu\frac{\partial\psi}{\partial z} + \Sigma_t\psi(z,\mu) - \frac{\Sigma_s}{4\pi}\phi(z) \tag{18}$$

By choosing a shifted Legendre polynomial that is *strictly positive* in μ and a simple quadratic profile in z with zeros on the boundaries $[0, L]$, the following function was selected:

$$\begin{aligned} \psi_m(z,\mu) &= \frac{1}{\pi L^2} z(L-z)\left[P_3(\mu) + 1\right] \\ &= \frac{1}{2\pi L^2} z(L-z)\left[5\mu^3 - 3\mu + 2\right] \end{aligned} \tag{19}$$

Substituting Equation (19) into Equation (18), we get a manufactured source in terms of $P_3(\mu)$ as:

$$\begin{aligned} Q^m(z,\mu) = &\frac{\mu}{\pi L^2}\left[P_3(\mu)+1\right](L-2z) + \\ &+ \frac{\Sigma_t}{\pi L^2} z\,(L-z)\left[P_3(\mu)+1\right] + \\ &- \frac{\Sigma_s}{\pi L^2} z\,(L-z) \end{aligned} \tag{20}$$

When Equation (19) is integrated over the solid angle, the analytical scalar is given as:

$$\phi^m(z) = \frac{4}{L^2} z\,(L-z) \tag{21}$$

Similar to the previous section, we write a discrete-ordinates transport form of Equation (18) as:

$$\mu_n \frac{\mathrm{d}\psi_n^m}{\mathrm{d}z} + \Sigma_t \psi_n^m(z) = \frac{\Sigma_s}{4\pi}\phi^m(z) + S_{n,i} \tag{22}$$

where we define:

$$S_{n,i} = \frac{1}{\Delta_z}\int_{z_{i-1/2}}^{z_{i+1/2}} Q^m(\mu_n, z)\mathrm{d}z. \tag{23}$$

Equation (23) requires the use of numerical integration again, and to maintain the quadratic convergence from the diamond differencing, a second-order numerical quadrature scheme must also be implemented. One of the simplest choices is the Trapezoid Rule [**?**], which when applied to Equation (23) gives:

$$S_{n,i} \approx \frac{1}{2}\left(Q^m(\mu_n, z_{i+1/2}) + Q^m(\mu_n, z_{i-1/2}\right) \tag{24}$$

When implemented over various number of cells, the numerical solution converged near-quadratically to the manufactured solution. A plot of the numerical solution can be found in Figure 1. A convergence study was also conducted and can be found in Figure 2.

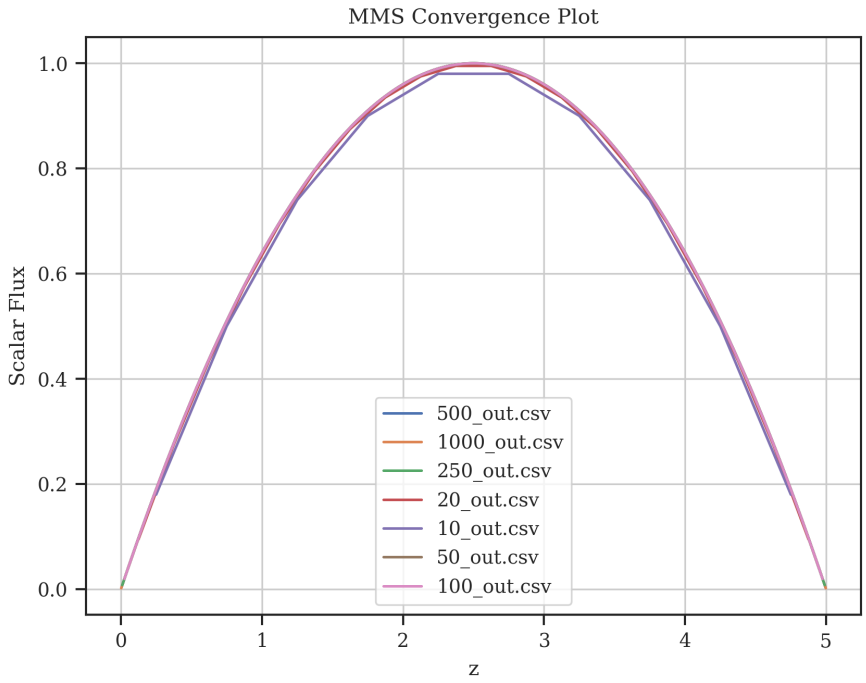


Figure 1: Numerical solution plotted against analytical manufactured solution.

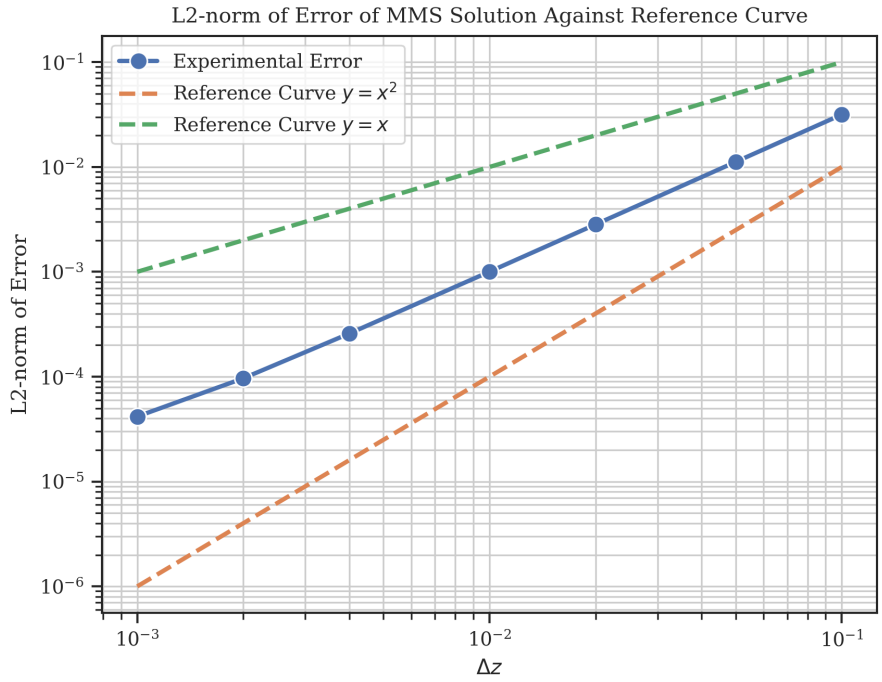


Figure 2: Convergence study of MMS solution using the discrete L2-norm.

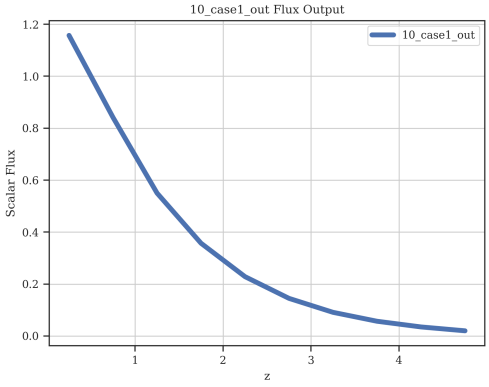
Part 3: *Single-Section 1-D Slab*

The following cases were run with the given parameters and boundary conditions. Cases 1–5 had their conditions hard-coded in because they were based on the code titled `SN1`, while case 5 was based off of `SN2`, which allowed for a variable input using a text file. For all cases in this section, the following parameters were implemented:

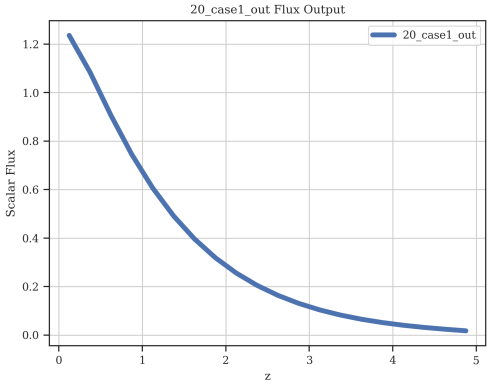
- Isotropic scattering, $\Sigma_{s,l} = 0$ for $l \geq 1$.
- Gaussian quadrature order, $N = 8$.
- Total cross-section, $\Sigma_t = 1.0$.
- No fission, $\Sigma_f = 0.0$.
- Source iteration tolerance, $\epsilon = 10^{-6}$.

3.1: Cases 1 & 2

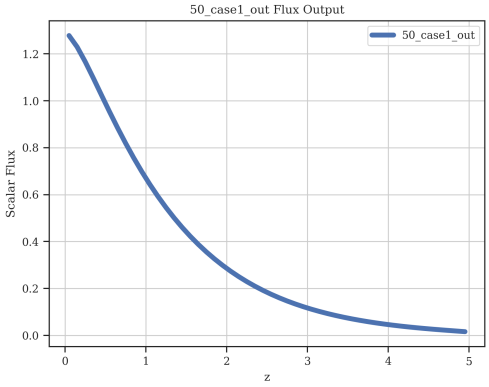
- Incident beam on left boundary moving along the most forward direction.
- Vacuum boundary on right boundary.
- No volumetric source.
- $\Sigma_s = 0.5$.
- $L = 5.0$.
- $I = [10, 20, 50, 100]$.
- $\alpha = 0.5$ (Case 1) and $\alpha = 0.0$ (Case 2).



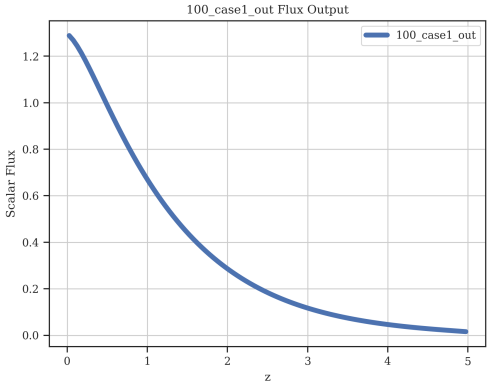
(**a**) $I = 10$ Cells



(**b**) $I = 20$ Cells

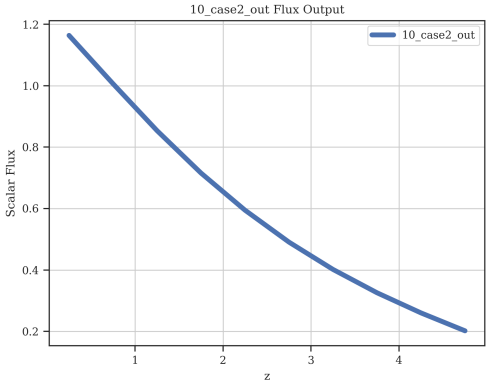


(**c**) $I = 50$ Cells

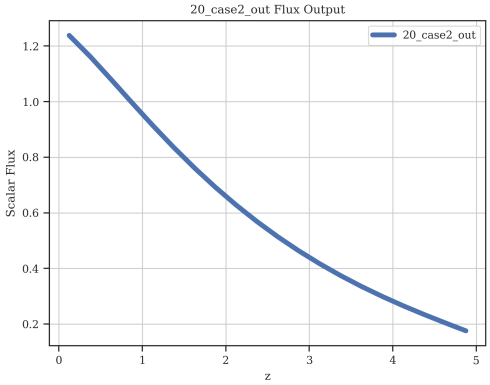


(**d**) $I = 100$ Cells

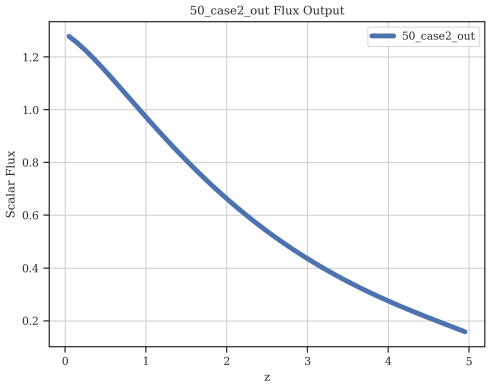
Figure 3: Case 1 generated plots.



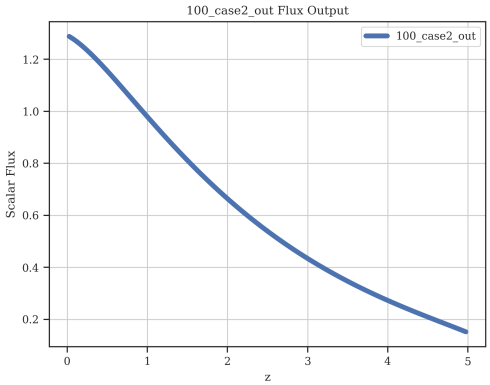
(**a**) $I = 10$ Cells



(**b**) $I = 20$ Cells



(**c**) $I = 50$ Cells

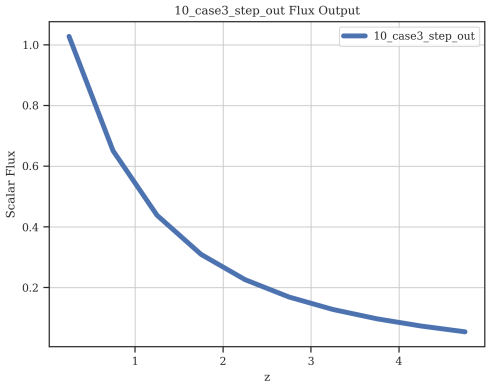


(**d**) $I = 100$ Cells

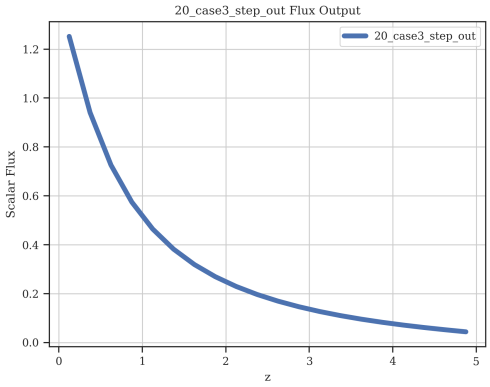
Figure 4: Case 2 generated plots.

3.2: Case 3

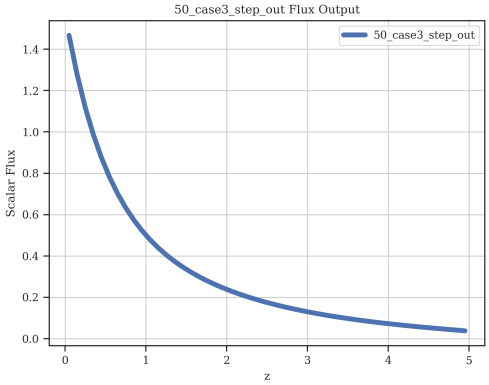
- *Isotropic* incident beam on left boundary, normalized to unit incidence current.
- Vacuum boundary on right boundary.
- No volumetric source.
- $\Sigma_s = 0.5$.
- $L = 5.0$.
- $I = [10, 20, 50, 100]$.
- $\alpha = 0.0, 0.5$.



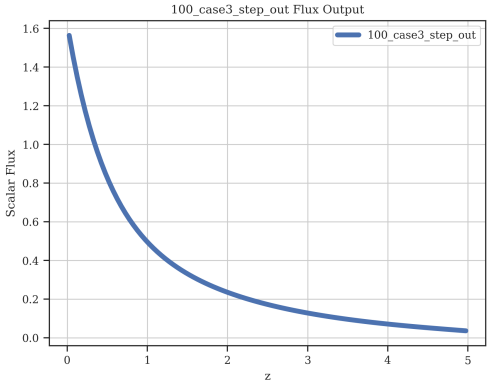
(**a**) $I = 10$ Cells



(**b**) $I = 20$ Cells

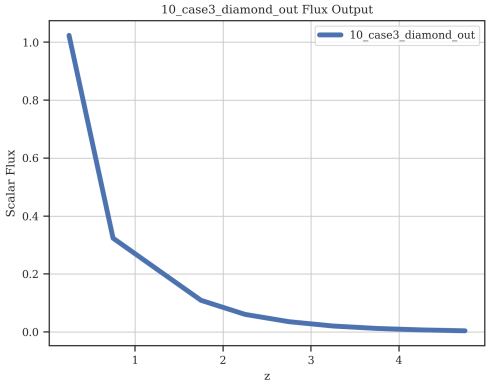


(**c**) $I = 50$ Cells

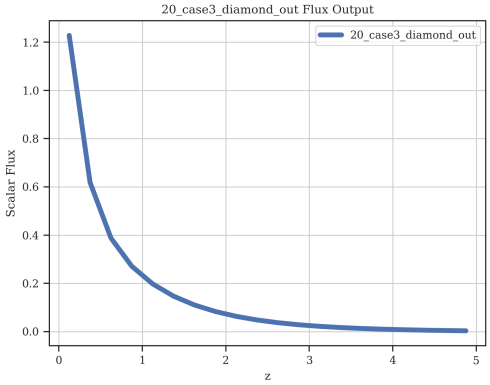


(**d**) $I = 100$ Cells

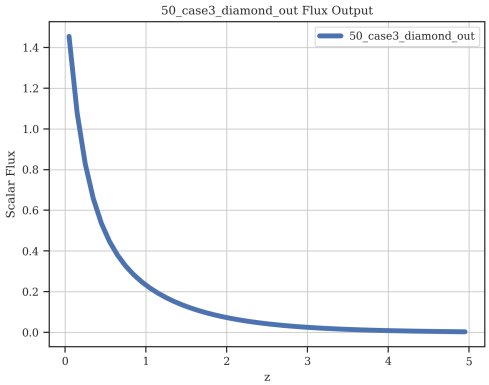
Figure 5: Case 3 generated plots with $\alpha = 0.0$ (step differencing).



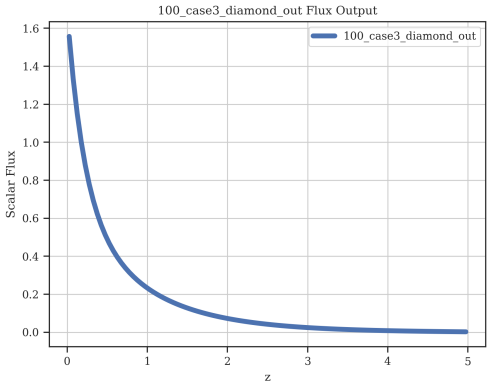
(**a**) $I = 10$ Cells



(**b**) $I = 20$ Cells



(**c**) $I = 50$ Cells



(**d**) $I = 100$ Cells

Figure 6: Case 3 generated plots with $\alpha = 0.5$ (step differencing).

3.3: Case 4

- Incident beam on left boundary moving along the most forward direction.
- Vacuum boundary on right boundary.
- No volumetric source.
- $\Sigma_s = 0.99$.
- $L = 100.0$.
- $I = 100$.
- $\alpha = 0.5$.

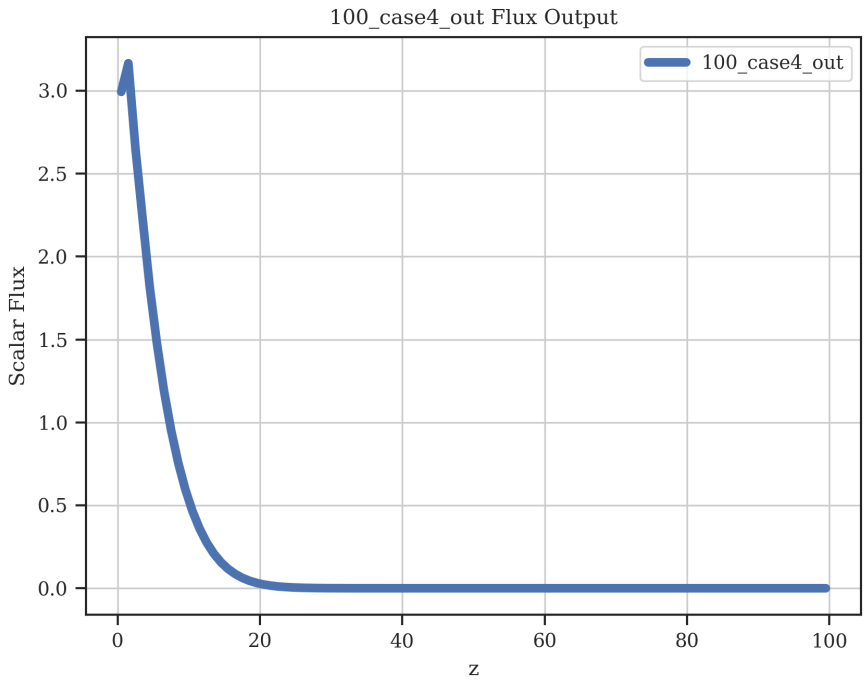
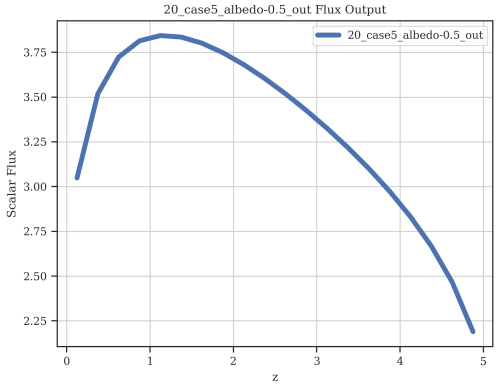


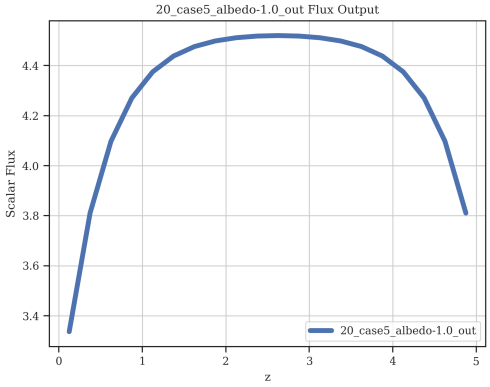
Figure 7: Case 4 Generated Plot

3.4: Case 5

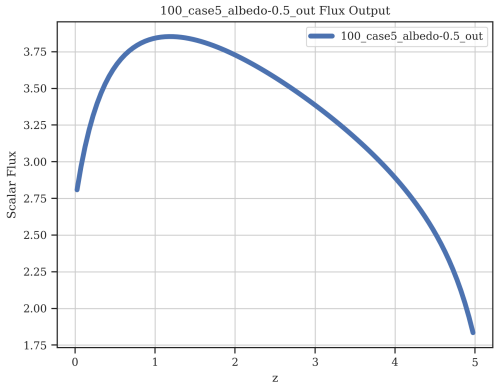
- Incident beam on left boundary.
- Albedo (reflected) condition on right boundary.
- $\Sigma_s = 1.0$
- $L = 5.0$
- $I = 20$
- Albedo coefficient, $\gamma = 0.5, 1.0$



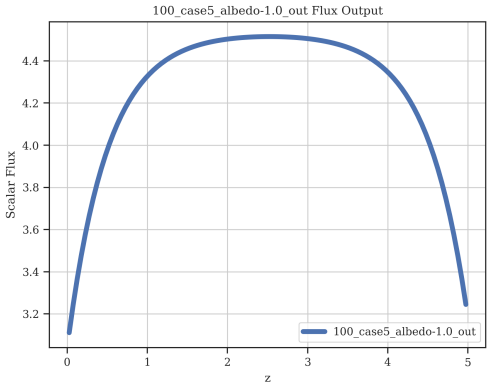
(a) $I = 20$ Cells, $\gamma = 0.5$



(b) $I = 20$ Cells, $\gamma = 1.0$



(c) $I = 100$ Cells, $\gamma = 0.5$

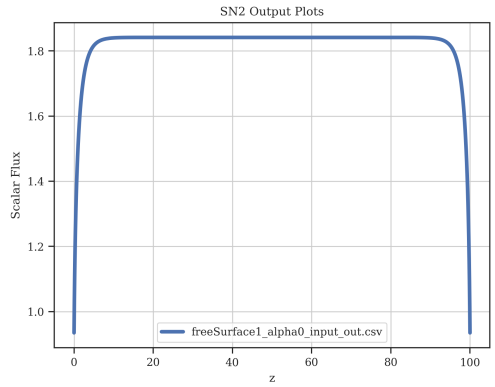


(d) $I = 20$ Cells, $\gamma = 1.0$

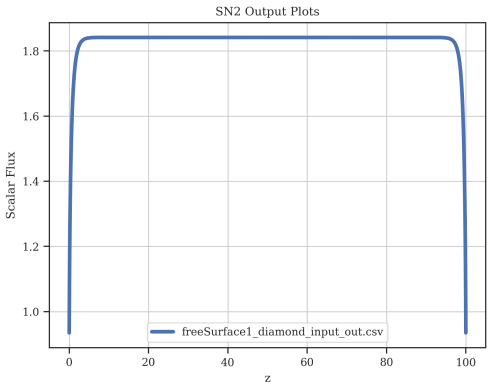
Figure 8: Case 5 generated plots with both albedo conditions: $\gamma = 0.5, 1.0$.

3.5: Case 6

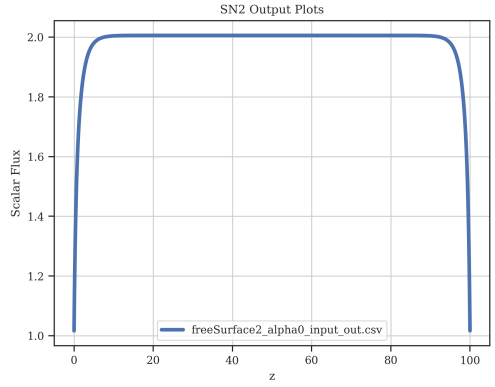
- Spatially uniform isotropic source, normalized such that the total number of neutrons in the slab is unity.
- Vacuum boundary conditions
- $\Sigma_s = 0.9, 0.99$
- $L = 100$
- $I = 100$
- $\alpha = 0.0, 0.5$



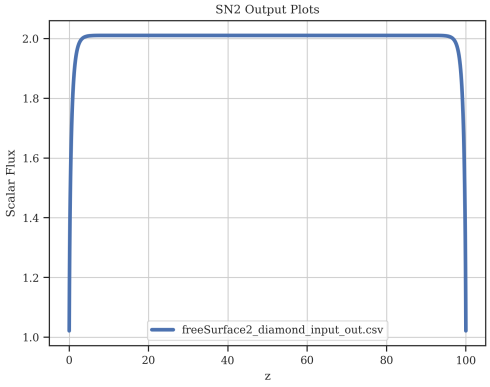
(a) $\Sigma_s = 0.9$, $\alpha = 0.0$



(b) $\Sigma_s = 0.9$, $\alpha = 0.5$



(c) $\Sigma_s = 0.99$, $\alpha = 0.0$



(d) $\Sigma_s = 0.99$, $\alpha = 0.5$

Figure 9: Case 6 generated plots with both scattering conditions, $\Sigma_s = 0.9, 0.99$, and both step and diamond differencing ($\alpha = 0.0, 0.5$).

Region No.	**1**	**2**	**3**	**4**	**5**
Σ_t	50.0	5.0	0.0	1.0	1.0
Σ_s/Σ_t	0.0	0.0	0.0	0.9	0.9
Source Strength	50.0	0.0	0.0	1.0	0.0
Length	2.0	1.0	2.0	1.0	2.0

Table 1: Reed Problem Parameters

Part 4: *Reed Problem*

As a final, and arguably most important aside from the MMS solution, demonstration of the written code, the so-called Reed problem was implemented. First, the basic code `SN1` was generalized to allow for arbitrary number of regions in a slab, with each region being able to handle different physical parameters and cell sizes. This application is meant to test the codes ability to handle sudden, drastic changes in material properties. The parameters for this can be found in Table 1.

To improve numerical accuracy, $N = 32$ quadrature nodes were implemented, along with a total of 900 cells. The input file which was used to generate the numerical results can be found in Appendix A. As can be seen in Figure 10, the code performed very well, matching the expected result provided.

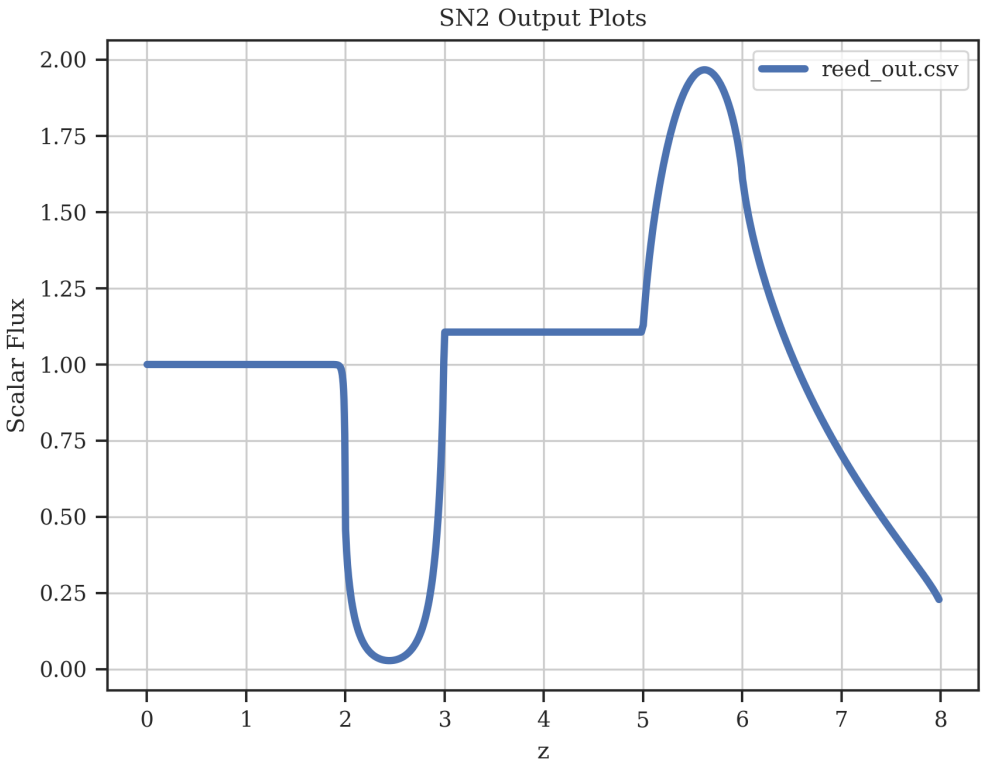


Figure 10: Generated plot from Reed problem application.

Appendix A: Reed Input File

```
// ------------------------------------------------
// SN2 Input File
// ------------------------------------------------
//
// ------------------------------------------------
// Computation Settings
// ------------------------------------------------
// tolerance    max_iterations  quadrature_nodes    alpha
    1e-6        100             32                  0.5
// ------------------------------------------------
// Number of regions
// ------------------------------------------------
    5
// ------------------------------------------------
// Length of regions
// ------------------------------------------------
    2   1   2   1   2
// ------------------------------------------------
// Number of cells in each region
// ------------------------------------------------
    500 100 100 100 100
// ------------------------------------------------
// Total cross section for each region
// ------------------------------------------------
    50  5   0   1   1
// ------------------------------------------------
// Scattering cross section for each region
// ------------------------------------------------
    0   0   0   0.9 0.9
// ------------------------------------------------
// Source strengths (constants)
// ------------------------------------------------
    50  0   0   1   0
// ------------------------------------------------
// Albedo coefficient (LH boundary)
// ------------------------------------------------
    1
```