# Therefore:

Simple Conner Balance Implemented on a One Cell Inversion Scheme with Numba Acceleration [4]

Jackson P. Morgan

Computational Particle Transport (NSE 659), Oregon State University

Prof. Todd S. Palmer

June 10th 2022

## Contents

**1 Introduction** **2**
1.1 Iteration Methods (SI v OCI) . . . 2
1.2 Simple Corner Balance Discretization Scheme . . . 3
1.3 Parallelization in SI v OCI . . . 4
1.4 Therefore . . . 4

**2 Derivation of SCB In OCI in 1D** **5**

**3 Test Problem Validation** **6**

**4 Spectral Radii Analysis and Conclusions** **6**

**5 Future Work** **7**

**6 Appendix** **9**

# 1 Introduction

Numerical solutions to the neutron transport equation have proven to be difficult and computationally intensive. While the equation exhibits the *curse of dimensionality*, having seven independent variables, these can commonly be discretized (e.g. multi-group for energy, where now the problem is split into multiple mono-energetic transport problems) in a given order to arrive at a transport solution. The farthest under the hood discretization is space, where many different methods of solving the inherent non-linearity (due to scalar flux in the scattering term) have been derived.

Often when experimenting and analyzing these methods we set up the *simplest problem in the world* a mono-energetic, steady state, axially symmetric, problem [2]

$$\vec{\Omega} \cdot \nabla \Psi(\vec{r}, \vec{\Omega}) + \sigma(\vec{r}) = S(\vec{r}, \vec{\Omega}) + \frac{1}{4\pi} \sum_{l=0}^{\infty} \sigma_{s,l} \sum_{n=-1}^{l} \phi_{l,n}(\vec{r}) Y_{l,n}(\vec{\Omega}) \tag{1}$$

where $\vec{\Omega}$ is the three-component direction vector, $\Psi$ is the angular flux, $\sigma$ is the total cross-section of the material, $\sigma_s$ is the scattering cross-section, $\vec{r}$ the three component position vector, $\phi_{l,n} = \int_{4\pi} Y^*_{l,n}(\vec{\Omega})\psi(\vec{r}, \vec{\Omega})$ is the scalar flux, and $Y_{l,n}$ is a spherical harmonic. We make further simplifications by assuming a slab wall geometry and homogeneous media

$$\mu \frac{\partial}{\partial x} \Psi(x, \mu) + \sigma \Psi(x, \mu) = S(x, \mu) + \sum_{l=0}^{L} \frac{2l + l}{2} \sigma_{s,l} \phi_l(x) P_l(\mu) \tag{2}$$

where $P_l(\mu)$ is a Legendre polynomial, and $\mu$ is the cosine of the azimuthal angel. Next we discretize in angle with with discrete ordinates $(S_n)$

$$\phi = \int_{-1}^{1} \Psi d\mu = \sum_{m=0}^{M} w_m \psi_m \tag{3}$$

where $w_m$ is the weight from quadrature which includes the negative and positive angles. Assuming isotropic scattering the Legendre polynomial drops out. Notice how $\psi$ is now indexed with the angular index $m$. Finally we arrive at our *über simplest equation in the world* that we will be implement using a spatial discretization

$$\mu_m \frac{\partial \psi_m(x)}{\partial x} + \sigma \psi_m(x) = S + \frac{\sigma_s}{2} \phi(x) \tag{4}$$

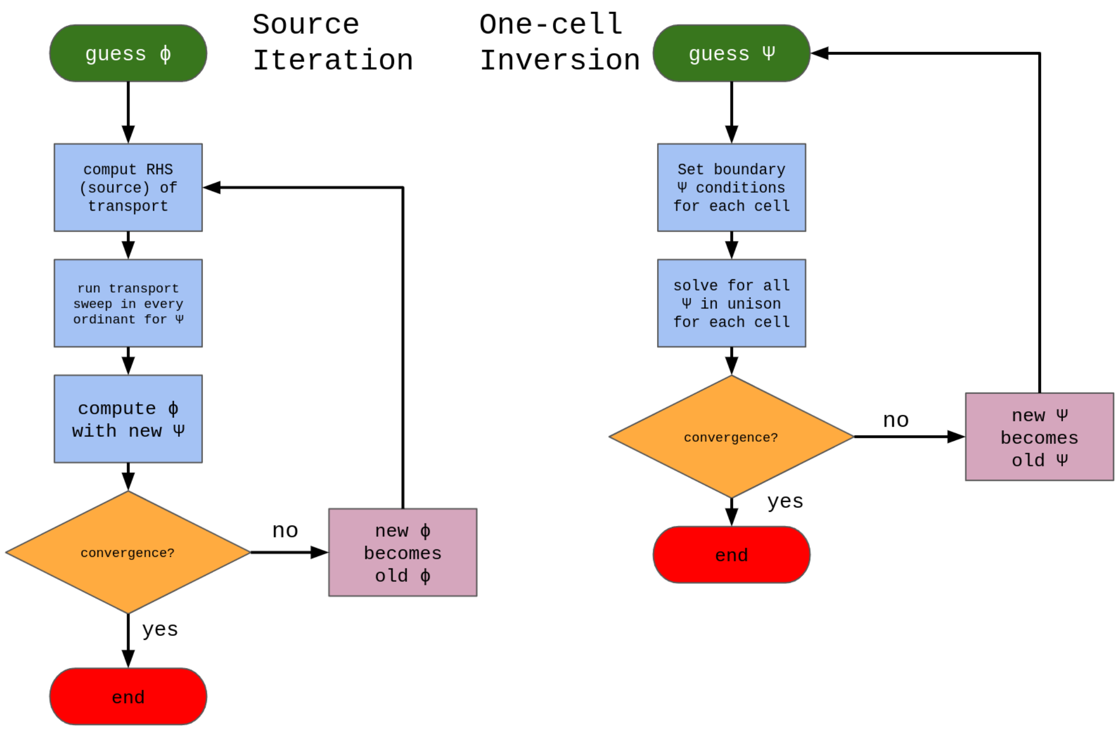
where $\phi = \sum_{m=0}^{m} w_m \psi_m(x)$ and the index m denotes quadrature values. We are now ready to hit some slabs!

## 1.1 Iteration Methods (SI v OCI)

Unfortunately equation 4 is nonlinear as the scalar flux on the RHS is based on the angular flux on the LHS. From your ethereal grey matter the phrase ***iterate till convergence*** should be being echoing throughout the foggy moor of your mind. We will set one term to lag behind while we calculate the others, test for convergence then do it again if required.

A commonly used method of iteration is the Source Iteration, or SI (AKA Richardson or fixed point) where the scattering source scalar flux is let to lag behind (initially being guessed) while the next angular flux across all angles and in every cell are found from conducting transport sweeps. This lowest possible operation (sweeping across the cells) is inherently serial as the solution in a given angle in a given cell is based on the outgoing flux from it's predecessor.

Cell inversions (a simple form of domain decomposition) lets the cellular incident angular fluxes on the left and right lag behind while the within cell angular flux, and scattering source term are all



Source Iteration
guess ϕ
comput RHS (source) of transport
run transport sweep in every ordinant for Ψ
compute ϕ with new Ψ
convergence?
no
new ϕ becomes old ϕ
yes
end
One-cell Inversion
guess Ψ
Set boundary Ψ conditions for each cell
solve for all Ψ in unison for each cell
convergence?
no
new Ψ becomes old Ψ
yes
end

Figure 1: Flow charts showing Source Iterations (SI) at left and One-Cell Inversion at right.

computed in unison through a linear algebra solve. This algorithm becomes embarrassingly parallel over the cell space. We explore the simplest cell inversion, the so call One-Cell Inversion or OCI [1]. Note that this OCI is a form of the parallel block Jacobi scheme.

Figure 1 shows the flow chart of both SI and OCI algorithms. In SI the term being iterated on is the scalar flux $\phi$ while in OCI it is the angular flux $\psi$.

## 1.2 Simple Corner Balance Discretization Scheme

Simple corner balance [3] (SCB) is a finite volume method that is the same as the linear discontinuous method in 1D. We start with two half cell balance equations (we have fewer reservations about methods based on conservation!) for the left

$$\mu_m[\psi_{m,i} - \psi_{m,i-1/2}] + \sigma\frac{\Delta x}{2}\psi_{m,i,L} = \frac{\Delta x}{2}Q_{i,L} \tag{5}$$

and the right

$$\mu_m[\psi_{m,i+1/2} - \psi_{m,i}] + \sigma\frac{\Delta x}{2}\psi_{m,i,R} = \frac{\Delta x}{2}Q_{i,R} \tag{6}$$

where $i$ is the spatial index, and $\Delta x$ is the cell width. We now assume that the midpoint angular flux ($\psi_{m,i}$) is

$$\psi_{m,i} = \frac{\psi_{m,i,L} + \psi_{m,i,R}}{2} \tag{7}$$

the average between the the left and right half cell average angular fluxes. We then apply upstream closure to produce a system of linear equations that can be solved (in 1D SI this is done directly) to get the within cell (left and right) angular fluxes. Note that in SI we are iterating on both the material and scattering source. This means that the scalar flux from the last iteration is used in the Q term.

$$Q_i = \sigma_s \phi_i + S \tag{8}$$

## 1.3 Parallelization in SI v OCI

Say you and a few of your friends are deadbeat undergraduates who have to pay for their masochistic educational habits with a job (oh to be a business major with scholarships). You've been told that before you can go home and study for that three hour long thermo midterm tomorrow, you all need to sweep a *non-square rectangular* (length ¿¿ width) floor of a given size. "Aha!" you exclaim, you know from your days collecting firewood with your six siblings that, "Many hands make light of large loads," and when work is in effect *parallelized* the work will get done faster!

But now a problem; how to efficiently divide the work? You want every broom you have operating at the same time to take advantage of your numbers (same broom, at same speed, with no penalty for moving lanes) so how do you line your friends up to sweep:

1. length wise; or
2. width wise

Width-wise of course. The more brooms the better! Marx would be proud! Figure 2 shows the difference between OCI and SI.

That is the idea behind the parallelization advantage of Cell Inversions. The width is governed by the number of angles in quadrature while the length is the number of cells. Almost always there will be far more cells in a given computation than angels and being able to compute multiple cells at once will allow our modern many-core machines (the brooms) to be more efficiently used.

Now there is no such thing as a free lunch. While individual iterations of OCI might be faster when parallelized, depending on problem parameters it might require significantly more iterations till convergence than SI. This presents interesting question about which method will be done faster in wall time. Who cares if a given method takes an order of magnitude more iterations to converge if the time to finish an iteration is two magnitudes less time than the competitor? Food for thought.

## 1.4 Therefore

Therefore is a simple Python based code, produced to experiment with these algorithms in 1D. As OCI's primary point of interest is that it is embarrassingly parallel, optimization and palatalization using the Numba package was implemented. This allowed for rapid methods development and easy analysis of large batch's of test runs required for things like spectral radii analysis (which is done in later sections).

OCI was parallelized over the number of cells whereas SI was parallelized over the number of angles using Numba's native threading abilities. All that was required for parallelization in both methods was two or three compilation flags and one numba command. This further demonstrates Python's ability to be used in a highly accelerated and parallelized manor for rapid methods exploration and comparison.

If this testbed is taken forward this acceleration and parallelization structure will be as well.

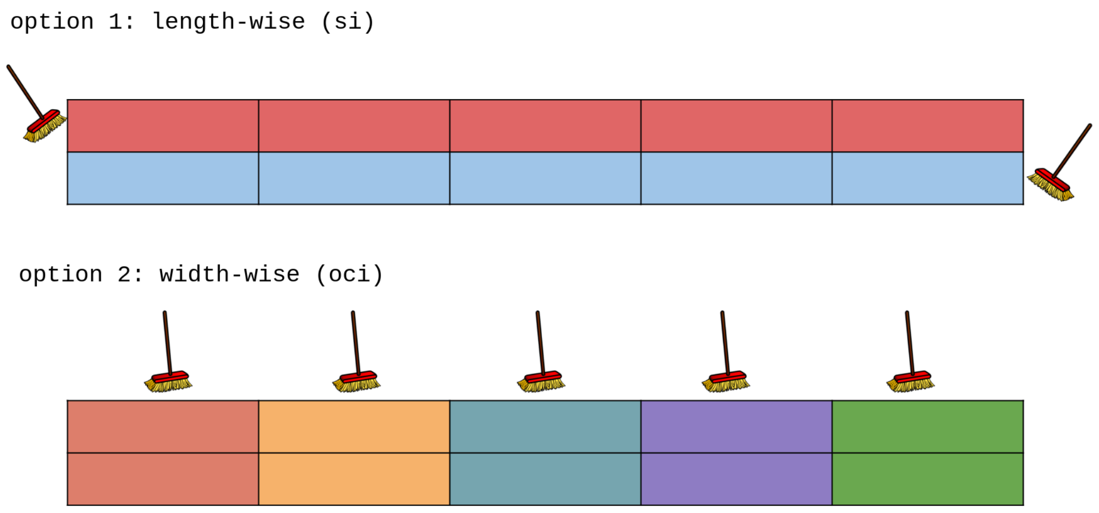


Figure 2: Shows sweep method

## 2 Derivation of SCB In OCI in 1D

**Important:** Start by recognizing that the upstream closure term means we will have different sets of equations as *upstream* can be different depending on angles from quadrature. First we have SCB with expanded RHS Q, and scalar flux $\phi_{i,L/R}$

$$\mu_m[\psi_{m,i} - \psi_{m,i-1/2}] + \sigma\frac{\Delta x}{2}\psi_{m,i,L} = \frac{\Delta x}{2}(\sigma_s\sum_{m=0}^{M} w_m\psi_{m,i,L} + S_{i,L}) \tag{9}$$

$$\mu_m[\psi_{m,i+1/2} - \psi_{m,i}] + \sigma\frac{\Delta x}{2}\psi_{m,i,R} = \frac{\Delta x}{2}(\sigma_s\sum_{m=0}^{M} w_m\psi_{m,i,R} + S_{i,R}) \tag{10}$$

Then using mid-cell average closure (equation 7), upstream closure, and OCI (lagging the incident angular fluxes on the cell) we form two sets of two equations for the negative ($\mu < 0$)

$$\mu_n[\frac{\psi_{n,i,L}^{l+1/2} + \psi_{n,i,R}^{l+1/2}}{2} - \psi_{n,i,L}^{l+1/2}] + \frac{\sigma\Delta x}{2}\psi_{p,i,L}^{l+1/2} - \sum_{m=0}^{M} w_m\psi_{m,i,L}^{l+1/2} = \frac{\Delta x}{2}S_{i,L} \tag{11}$$

$$\mu_n[\psi_{n,i+1/2}^{l} - \frac{\psi_{n,i,L}^{l+1/2} + \psi_{n,i,R}^{l+1/2}}{2} + \frac{\sigma\Delta x}{2}\psi_{n,i,R}^{l+1/2}] - \sum_{m=0}^{M} w_m\psi_{m,i,R}^{l+1/2} = \frac{\Delta x}{2}S_{i,R} \tag{12}$$

and positive ($\mu > 0$) ordinates

$$\mu_p[\frac{\psi_{p,i,L}^{l+1/2} + \psi_{p,i,R}^{l+1/2}}{2} - \psi_{p,i-1/2}^{l}] + \frac{\sigma\Delta x}{2}\psi_{p,i,L}^{l+1/2} - \sum_{m=0}^{M} w_m\psi_{m,i,L}^{l+1/2} = \frac{\Delta x}{2}S_{i,L} \tag{13}$$

$$\mu_p[\psi_{p,i,R}^{l+1/2} - \frac{\psi_{p,i,L}^{l+1/2} + \psi_{p,i,R}^{l+1/2}}{2} + \frac{\sigma\Delta x}{2}\psi_{p,i,R}^{l+1/2}] - \sum_{m=0}^{M} w_m\psi_{m,i,R}^{l+1/2} = \frac{\Delta x}{2}S_{i,R} \tag{14}$$

where $p$ is the index over all positive ordinates, $n$ is the index over all negative ordinates, $m$ is the index over the total number of ordinates, $i$ is the index over cell space, and $l$ is the index of the iteration step in OCI. This can then be set up as a 2Mx2M $\mathbf{A}$ matrix, 2M $\vec{x}$ and $\vec{B}$ vector in the form $\mathbf{A}\vec{x} = \vec{B}$.

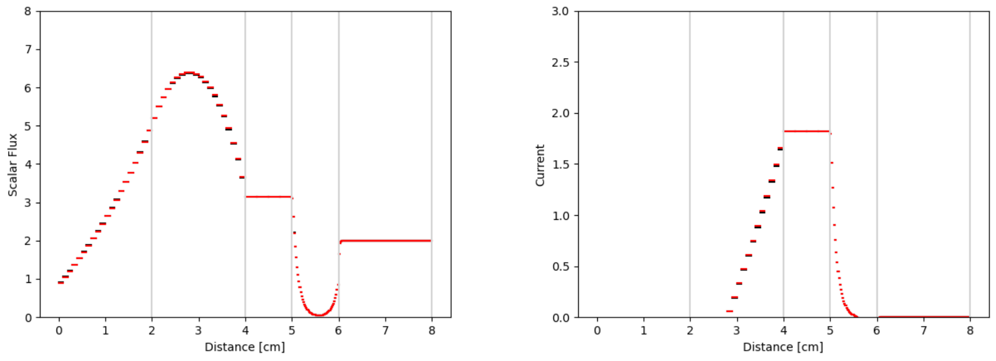


Figure 3: Solution to Reed's problem found with both SI (black) and OCI (red)

Consider SCB OCI in $S_2$. In every cell a 4X4 **A**-matrix system is set up

$$\begin{bmatrix} \frac{-\mu_1}{2} - w_1\frac{\Delta x\sigma_s}{2} & \frac{-\mu_1}{2} + \frac{\sigma\Delta x}{2} & -w_2\frac{\Delta x\sigma_s}{2} & 0 \\ -\frac{\mu_1}{2} + \frac{\sigma\Delta x}{2} & \frac{\mu_1}{2} - w_1\frac{\Delta x\sigma_s}{2} & 0 & -w_2\frac{\Delta x\sigma_s}{2} \\ -w_1\frac{\Delta x\sigma_s}{2} & 0 & \frac{\mu_2}{2} + \frac{\sigma\Delta x}{2} - w_2\frac{\Delta x\sigma_s}{2} & \frac{\mu_2}{2} \\ 0 & -w_1\frac{\Delta x\sigma_s}{2} & -\frac{\mu_2}{2} & -\frac{\mu_1}{2} + \frac{\sigma\Delta x}{2} - w_2\frac{\Delta x\sigma_s}{2} \end{bmatrix} *$$

$$\begin{bmatrix} \psi_{1,i,L}^{l+1/2} \\ \psi_{1,i,R}^{l+1/2} \\ \psi_{2,i,L}^{l+1/2} \\ \psi_{2,i,R}^{l+1/2} \end{bmatrix} = \begin{bmatrix} \frac{\Delta x}{2}S_L - \mu_1\psi_{1,i+1/2}^{l} \\ \frac{\Delta x}{2}S_R \\ \frac{\Delta x}{2}S_L + \mu_2\psi_{2,i-1/2}^{l} \\ \frac{\Delta x}{2}S_R \end{bmatrix}$$

where $\mu_1 \approx -0.58$ and $\mu_2 \approx 0.58$. This system can now be solved for a single cell in a single iteration step with any linear algebra solver with pivoting.

## 3 Test Problem Validation

A series of test problems with analytical solutions was used to validate the implementation in *Therefore*. These examples can be found in a Jupiter note book in the directory of the implementation. Figure 3 shows the most difficult problem used in the test suite (*Reed's problem*) which was found after 270 iterations via OCI. SI used 59 iterations to converge.

## 4 Spectral Radii Analysis and Conclusions

We evaluated the spectral radii

$$\rho = \frac{||\Psi^{l+1} - \Psi^{l}||}{||\Psi^{l} - \Psi^{l-1}||} \tag{15}$$

of both SI and OCI. The problem used was a source free pure absorber with isotropic incident fluxes on both the left and right boundaries. The scattering ratio was allowed to very between 0 and .999 and the optical thickness between 0.1 and 20 mfp.

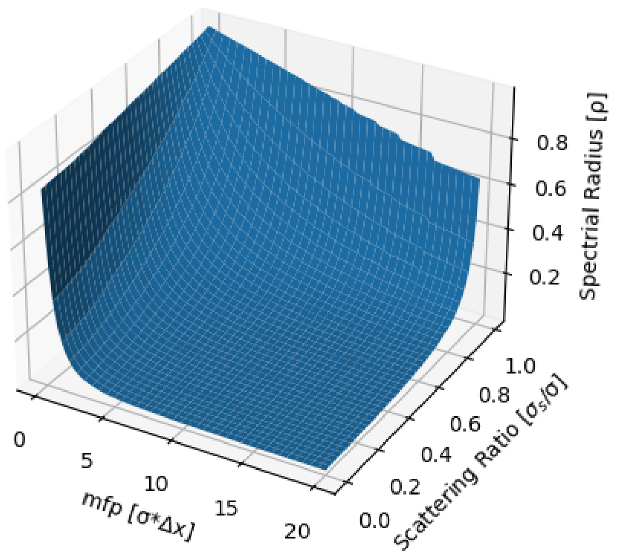


Figure 4: Spectral radius of One-cell Inversions over a range of cellular mean free paths and scattering ratios

Figure 5 shows the shelf-like behavior of SI where it is independent of cellular optical thickness. This makes senses as the algorithm allows radiation to be propagated through an iteration step where as OCI does not.

Figure 4 shows a much more interesting picture of the spectral radius depending asymptotically on both the optical thickness and the scattering ratio. As radiation crossing cell bounds can only be communicated between iterations it makes perfect since that the thickness of these cells would have an impact.

Where there is curvature there is possibility. Time dependence adds a scattering term into the absorption side ($\sigma$ turns into $\hat{\sigma}$ while $\sigma_s$ stays constant). The time dependent cross section depends on time step. Looking at figure 4 if we had a problem with a very high spectral radius due to optically thin cells, or due to a high scattering ratio we could move the effective spectral radius down by making the total cross section ($\sigma$) larger with time dependence. If the relations where known we could develop some kind of a tuning parameter to take maximal advantage of this.

As both methods where implemented and parallelized with Numba where possible, we can do a comparison of the wall time required for the simulations used to generate figures 4 and 5. SI took 91.5s to complete and OCI took 69.2s, **about 25% percent faster**! Note that this is only for $S_2$ and as more angles are added the linear algebra problem in OCI will become larger and SI will probably be unaffected until above $S_{16}$ on my 16 core machine.

## 5 Future Work

As of turning this in OCI is statically implemented in $S_2$. This is relatively easy to fix and will be done in the next day or two. Further exploration with the transient tuning idea as well as OCI's performance in 2D when compared with SI and it's ability to run on heterogeneous architectures may be pursued in the future.

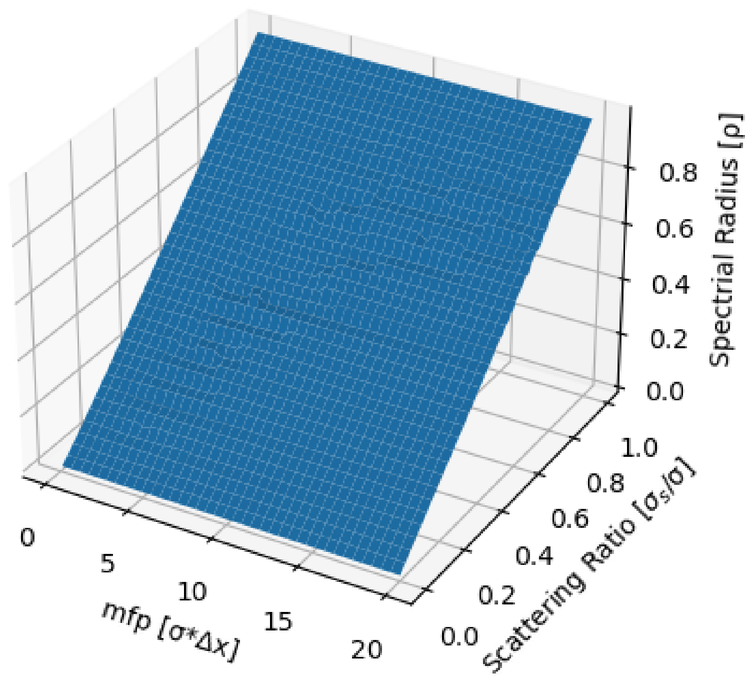


Figure 5: Spectral radius of Source Iterations over a range of cellular mean free paths and scattering ratios

| Variable | Description |
|---|---|
| $\psi$ | A singular angular flux element (m,i,L/R) |
| $\Psi$ | Total angular flux vector of size M (i, L/R) |
| $\phi$ | Scalar flux (i, L/R) |
| S | Material source term (i, L/R) |
| Q | RHS of transport equation, scattering for SI (i, L/R) |
| $\Delta x$ | Spatial discretization distance |
| $\sigma$ | total cross section |
| $\sigma_s$ | scattering cross section |

| Index | Description |
|---|---|
| m | angle in quadrature (upto M) |
| i | x-direction spatial index |
| n | negative ordinate index |
| p | negative ordinate index |
| L | Left hand side of sub cell discretization |
| R | Left hand side of sub cell discretization |

Table 1: Variables used and their descriptions

## References

[1] Kang-Seog Kim. *Coarse mesh and one-cell block inversion based diffusion synthetic acceleration.* PhD thesis, Oregon State University, 2000.

[2] E. E. Lewis and W. F. Miller. *Computational Methods of Neutron Transport.* Wiley, New York, NY, 1984.

[3] Adams Marvin and Todd S Palmer. NSE 654 Course Notes, 2022.

[4] Jackson P. Morgan. Therefore: Simple corner balance implemented on a one block inversion scheme, June 2022.

## 6 Appendix