Computing Neutron Balance Equations by Largest Eigenvalue Calculation with Power Iteration Method

Jack Gerrity

May 3, 2024

NPRE 247
Computer Project 2

May 1, 2024

1 Two Group Neutron Balance through an Infinite Medium

1.1 Theory

Through an infinite medium, no neutrons may be leaked from the system and therefore the system may be thought of as steady state ($\frac{dn}{dt} = 0$). Because of this, the neutron losses are equal to the neutrons produced.

$$Loss = Gain \tag{1}$$

For a two-group calculation, two neutron energy ranges will be considered: the fast group, and the slow group neutrons. Neutrons are lost in two ways (not considering leakage). The first way is neutrons are lost to the system is if they are captured (absorbed) by particles in the system. This absorption can be calculated by multiplying the macroscopic absorption cross section Σ_a the flux of particles of traveling through that energy group by ϕ. Another component causing neutron loss involves neutrons scattering to other energy levels. This occurs when a neutron interacts with a nucleus but is not absorbed and instead loses (or gains) energy and joins another energy group. Since this is only a two-group system, neutrons are only scattered to the other group and therefore, the scattering component can be calculated by multiplying that cross section $\Sigma_{s_{g\to g'}}$ by the flux of particles in that energy group ϕ. With the same idea, one way neutrons are "produced" in a group is when they are scattered from another energy range into it. For the two group system, there is only one other group that neutrons can scatter from and the number of neutrons that scatter from it is that cross section $\Sigma_{s_{g'\to g}}$ times the flux of particles in the other group ϕ. Lastly, the major way that neutrons enter a group is from fission. In order to calculate this, fission from all energy ranges is considered and the probability that fission-generated neutrons end up in the desired energy range is multiplied by that value. The sum involves the fission cross section Σ_f from every group times their respective average number of neutrons produced per fission reaction ν. That is then multiplied by the probability that a fission ends up in that energy range $\frac{\chi_g}{k}$. Taking all into consideration, the neutrons lost equals neutrons gained equation becomes:

$$Absorption + Outscattering = Fission + Inscattering \tag{2}$$

$$\Sigma_{a_g}\phi_g + \Sigma_{s_{g\to g'}}\phi_g = \frac{\chi_g}{k}(\nu\Sigma_{f_g}\phi_g + \nu\Sigma_{f_{g'}}\phi_{g'}) + \Sigma_{s_{g'\to g}}\phi_{g'} \tag{3}$$

This equation is generalized with the left side of the equation dealing with neutrons lost from group g and the right side of the equation dealing with neutrons gained in group g. In a two group system, this can be made into two equations about neutron balance in the fast and thermal range respectively. Lets call the fast group, group 1 and the slow group, group 2. For the fast group the equation looks like

$$\Sigma_{a_1}\phi_1 + \Sigma_{s_{1\to 2}}\phi_1 = \frac{\chi_1}{k}(\nu\Sigma_{f_1}\phi_1 + \nu\Sigma_{f_2}\phi_2) + \Sigma_{s_{2\to 1}}\phi_2 \tag{4}$$

Similarly, the equation for the slow group looks like this:

$$\Sigma_{a_2}\phi_2 + \Sigma_{s_{2\to 1}}\phi_2 = \frac{\chi_2}{k}(\nu\Sigma_{f_2}\phi_2 + \nu\Sigma_{f_1}\phi_1) + \Sigma_{s_{1\to 2}}\phi_1 \tag{5}$$

The most effective way to solve this system of equations is to format them together as a matrix equation. Absorption is represented as an absorption matrix times a flux vector.

$$Absorption\ Matrix = \begin{bmatrix} \Sigma_{a_1} & 0 \\ 0 & \Sigma_{a_2} \end{bmatrix} \tag{6}$$

$$Flux\ Vector = \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} \tag{7}$$

The fission matrix is constructed by putting the fast $\nu\Sigma_f$ in each element of the first column and the slow $\nu\Sigma_f$ in each element of the second. The χ value for the fast group was placed in each element of the first row and the χ element of the slow group was placed in each element of the second row. For each element, the χ value is multiplied by the $\nu\Sigma_f$ value. This is also multiplied by the flux vector

$$Fission\ Matrix = \begin{bmatrix} \chi_1\nu\Sigma_{f_1} & \chi_1\nu\Sigma_{f_2} \\ \chi_2\nu\Sigma_{f_1} & \chi_2\nu\Sigma_{f_2} \end{bmatrix} \tag{8}$$

For the outscattering matrix, flux is multiplied by all entries and therefore want everything times the first flux on the first column, and everything times the second flux in the second column. For a two-group system, this means putting the scattering from the fast group to the slow group on the top left entry and the scattering from the slow group to the fast group on the bottom right entry. For the inscattering matrix, similarly, flux will be multiplied should be distributed accordingly. In the two-group case, this means having the scattering from the fast to the slow group in the bottom left and the slow to fast scattering in the top right.

$$Outscattering\ Matrix = \begin{bmatrix} \Sigma_{s_{1\to 2}} & 0 \\ 0 & \Sigma_{s_{2\to 1}} \end{bmatrix} \tag{9}$$

$$Inscattering\ Matrix = \begin{bmatrix} 0 & \Sigma_{s_{2\to 1}} \\ \Sigma_{s_{1\to 2}} & 0 \end{bmatrix} \tag{10}$$

All this may be put together into one large matrix equation to describe *Equations 4 & 5.*

$$\begin{bmatrix} \Sigma_{a_1} & 0 \\ 0 & \Sigma_{a_2} \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} + \begin{bmatrix} \Sigma_{s_{1\to 2}} & 0 \\ 0 & \Sigma_{s_{2\to 1}} \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} = \frac{1}{k} \begin{bmatrix} \chi_1\nu\Sigma_{f_1} & \chi_1\nu\Sigma_{f_2} \\ \chi_2\nu\Sigma_{f_1} & \chi_2\nu\Sigma_{f_2} \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} + \begin{bmatrix} 0 & \Sigma_{s_{2\to 1}} \\ \Sigma_{s_{1\to 2}} & 0 \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} \tag{11}$$

The inscattering part of the equation is moved over to the other side and then factor out the flux vector to combine this matrix into one. This matrix is called the migration matrix and it takes into account absorption, outscattering, and inscattering.

$$\left(\begin{bmatrix} \Sigma_{a_1} & 0 \\ 0 & \Sigma_{a_2} \end{bmatrix} + \begin{bmatrix} \Sigma_{s_{1\to 2}} & 0 \\ 0 & \Sigma_{s_{2\to 1}} \end{bmatrix} - \begin{bmatrix} 0 & \Sigma_{s_{2\to 1}} \\ \Sigma_{s_{1\to 2}} & 0 \end{bmatrix}\right) \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} = \frac{1}{k} \begin{bmatrix} \chi_1\nu\Sigma_{f_1} & \chi_1\nu\Sigma_{f_2} \\ \chi_2\nu\Sigma_{f_1} & \chi_2\nu\Sigma_{f_2} \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} \tag{12}$$

$$\begin{bmatrix} \Sigma_{a_1} + \Sigma_{s_{1\to 2}} & -\Sigma_{s_{2\to 1}} \\ -\Sigma_{s_{1\to 2}} & \Sigma_{a_2} + \Sigma_{s_{2\to 1}} \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} = \frac{1}{k} \begin{bmatrix} \chi_1\nu\Sigma_{f_1} & \chi_1\nu\Sigma_{f_2} \\ \chi_2\nu\Sigma_{f_1} & \chi_2\nu\Sigma_{f_2} \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} \tag{13}$$

$$Migration\ Matrix = \begin{bmatrix} \Sigma_{a_1} + \Sigma_{s_{1\to 2}} & -\Sigma_{s_{2\to 1}} \\ -\Sigma_{s_{1\to 2}} & \Sigma_{a_2} + \Sigma_{s_{2\to 1}} \end{bmatrix} = A + S_{out} - S_{in} \tag{14}$$

Both sides of the equation are multiplied by k and in the same motion move the migration matrix over to the other side of the equation by taking an inverse in order to change the form of the equation to resemble an eigenvalue problem.

$$k \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} = \begin{bmatrix} \Sigma_{a_1} + \Sigma_{s_{1\to 2}} & -\Sigma_{s_{2\to 1}} \\ -\Sigma_{s_{1\to 2}} & \Sigma_{a_2} + \Sigma_{s_{2\to 1}} \end{bmatrix}^{-1} \begin{bmatrix} \chi_1\nu\Sigma_{f_1} & \chi_1\nu\Sigma_{f_2} \\ \chi_2\nu\Sigma_{f_1} & \chi_2\nu\Sigma_{f_2} \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} \tag{15}$$

Manipulating the equation to this form allows the equation to be formatted as an eigenvalue problem where the flux vector $\begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix}$ is an eigenvector to the inverse migration matrix times the fission matrix and k is an eigenvalue. The eigenvalues and eigenvectors are able to be found which will dictate in what direction the inverse migration time fission matrix will pull the flux (eigenvectors) and by how much (eigenvalues). By finding the largest eigenvalue, it is possible to determine in which direction (the respective eigenvector) the flux will trend towards. This is the steady-state flux. To do this, a hand computation as well as show a computation using power iteration method, a useful method for very large matrices, will be performed.

1.2 Largest Eigenvalue Calculation and Power Iteration Method

As mentioned above, calculating the eigenvalues (and their eigenvectors) allows the determination of where the flux vector will end up after a large amount of time. For a two-group system, this is a fairly simple; however, for larger systems, computational methods are necessary to more quickly and efficiently find the eigenvalue and it's eigenvector.

To start the two group hand calculation, the first step is to find the eigenvalues of the inverted migration matrix times the fission matrix. The first step in this process is inverting the migration matrix. Since it is a two-by-two matrix (in the two-group case), the values along the diagonal are flipped, the two other values are made negative, and the whole matrix is divided by the determinant.

$$det\left(\begin{bmatrix} \Sigma_{a_1} + \Sigma_{s_{1\to 2}} & -\Sigma_{s_{2\to 1}} \\ -\Sigma_{s_{1\to 2}} & \Sigma_{a_2} + \Sigma_{s_{2\to 1}} \end{bmatrix}\right) = (\Sigma_{a_1} + \Sigma_{s_{1\to 2}})(\Sigma_{a_2} + \Sigma_{s_{2\to 1}}) - (-\Sigma_{s_{2\to 1}})(-\Sigma_{s_{1\to 2}}) \tag{16}$$

$$= \Sigma_{a_1}\Sigma_{a_2} + \Sigma_{a_1}\Sigma_{s_{1\to 2}} + \Sigma_{a_2}\Sigma_{s_{1\to 2}} \tag{17}$$

$$Migration\ Matrix^{-1} = \frac{\begin{bmatrix} \Sigma_{a_2} + \Sigma_{s_{2\to1}} & \Sigma_{s_{2\to1}} \\ \Sigma_{s_{1\to2}} & \Sigma_{a_1} + \Sigma_{s_{1\to2}} \end{bmatrix}}{\Sigma_{a_1}\Sigma_{a_2} + \Sigma_{a_1}\Sigma_{s_{1\to2}} + \Sigma_{a_2}\Sigma_{s_{1\to2}}} \tag{18}$$

$$= \begin{bmatrix} \frac{\Sigma_{a_2}+\Sigma_{s_{2\to1}}}{\Sigma_{a_1}\Sigma_{a_2}+\Sigma_{a_1}\Sigma_{s_{1\to2}}+\Sigma_{a_2}\Sigma_{s_{1\to2}}} & \frac{\Sigma_{s_{2\to1}}}{\Sigma_{a_1}\Sigma_{a_2}+\Sigma_{a_1}\Sigma_{s_{1\to2}}+\Sigma_{a_2}\Sigma_{s_{1\to2}}} \\ \frac{\Sigma_{s_{1\to2}}}{\Sigma_{a_1}\Sigma_{a_2}+\Sigma_{a_1}\Sigma_{s_{1\to2}}+\Sigma_{a_2}\Sigma_{s_{1\to2}}} & \frac{\Sigma_{a_1}+\Sigma_{s_{1\to2}}}{\Sigma_{a_1}\Sigma_{a_2}+\Sigma_{a_1}\Sigma_{s_{1\to2}}+\Sigma_{a_2}\Sigma_{s_{1\to2}}} \end{bmatrix} \tag{19}$$

With the inverted Mitigation Matrix, the B-matrix is defined as the inverted mitigation matrix times the fission matrix. This is the matrix from which the eigenvalues and their corresponding eigenvectors will be calculated.

$$B\ Matrix = \frac{\begin{bmatrix} \Sigma_{a_2} + \Sigma_{s_{2\to1}} & \Sigma_{s_{2\to1}} \\ \Sigma_{s_{1\to2}} & \Sigma_{a_1} + \Sigma_{s_{1\to2}} \end{bmatrix}}{\Sigma_{a_1}\Sigma_{a_2} + \Sigma_{a_1}\Sigma_{s_{1\to2}} + \Sigma_{a_2}\Sigma_{s_{1\to2}}} \begin{bmatrix} \chi_1\nu\Sigma_{f_1} & \chi_1\nu\Sigma_{f_2} \\ \chi_2\nu\Sigma_{f_1} & \chi_2\nu\Sigma_{f_2} \end{bmatrix} \tag{20}$$

$$= \frac{\begin{bmatrix} (\Sigma_{a_2} + \Sigma_{s_{2\to1}})(\chi_1\nu\Sigma_{f_1}) + (\Sigma_{s_{2\to1}})(\chi_2\nu\Sigma_{f_1}) & (\Sigma_{s_{2\to1}})(\chi_1\nu\Sigma_{f_1}) + (\Sigma_{a_1} + \Sigma_{s_{1\to2}})(\chi_2\nu\Sigma_{f_1}) \\ (\Sigma_{s_{1\to2}})(\chi_1\nu\Sigma_{f_2}) + (\Sigma_{a_2} + \Sigma_{s_{2\to1}})(\chi_2\nu\Sigma_{f_2}) & (\Sigma_{a_1} + \Sigma_{s_{1\to2}})(\chi_2\nu\Sigma_{f_2}) + (\Sigma_{s_{1\to2}})(\chi_1\nu\Sigma_{f_2}) \end{bmatrix}}{\Sigma_{a_1}\Sigma_{a_2} + \Sigma_{a_1}\Sigma_{s_{1\to2}} + \Sigma_{a_2}\Sigma_{s_{1\to2}}} \tag{21}$$

To find the eigenvalues, the fact that $B\vec{\phi} = k\vec{\phi}$ can be leveraged by moving the eigenvalue term to the other side and factoring out the $\vec{\phi}$. After this, the determinant of the new matrix is set equal to 0 in order to solve for k.

$$det(B - kI) = 0 \tag{22}$$

$$\begin{aligned} 0 = {} & ((\Sigma_{a_2} + \Sigma_{s_{2\to1}})(\chi_1\nu\Sigma_{f_1}) + (\Sigma_{s_{2\to1}})(\chi_2\nu\Sigma_{f_1}) - k)((\Sigma_{a_1} + \Sigma_{s_{1\to2}})(\chi_2\nu\Sigma_{f_2}) + (\Sigma_{s_{1\to2}})(\chi_1\nu\Sigma_{f_2}) - k) \\ & -((\Sigma_{s_{2\to1}})(\chi_1\nu\Sigma_{f_1}) + (\Sigma_{a_1} + \Sigma_{s_{1\to2}})(\chi_2\nu\Sigma_{f_1}))((\Sigma_{s_{1\to2}})(\chi_1\nu\Sigma_{f_2}) + (\Sigma_{a_2} + \Sigma_{s_{2\to1}})(\chi_2\nu\Sigma_{f_2})) \end{aligned} \tag{23}$$

This results in a quadratic for which k may be solved. The larger of the eigenvalues may then be subtracted from the diagonal and flux values that complete the equation may then be solved for.

Solving by hand often becomes unrealistic especially when solving for more than just two-groups. Because of this, there are several computational methods available to avoid this difficult calculation. The first method is NumPy's eig function which returns all of the eigenvalues and an array containing their respective eigenvectors. This built-in function does all of the calculations from the above method and often streamlines a calculation. However, as matrices get larger, this function takes an exponentially longer time to run. Due to this fact, it is not the most efficient way to perform this calculation.

An alternative method is power iteration. Power iteration works by normalizing the matrix (in this case, the B matrix) and then multiplying a starting flux by that matrix repeatedly so that the results approach the steady-state vector (in this case, the steady-state flux). This is done by choosing an initial flux and then multiplying that flux by the B matrix (and normalizing that value). In code, it is:

$$\vec{\phi}_{i+1} = \frac{B\vec{\phi}_i}{||B\vec{\phi}_i||} \tag{24}$$

Once the flux vectors are computed, the corresponding k values are also able to be calculated.

$$k_{i+1} = \frac{(B\vec{\phi}_i)^T\vec{\phi}_{i+1}}{\vec{\phi}_{i+1}^T\vec{\phi}_{i+1}} \tag{25}$$

After many iterations, the flux vector will approach the steady-state flux vector (the eigenvector with the largest eigenvalue) and k will approach the largest eigenvalue.

Table 1: Two Group Sample Data

Group	Σ_a	$\nu\Sigma_f$	χ
1	0.0092	0.0046	1.0000
2	0.0932	0.1139	0.0000

Table 2: Two Group Sample Scattering Data

To Row ↓ From Column →	1	2
1	1.0000	0.0000
2	0.0202	2.0000

$$\vec{\phi}_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \; k_0 = 1 \tag{26}$$

1.3 Two Group Solution using Sample Data

Each of the above processes is solved computationally. Input data is shown in *Tables 1 & 2* as well as *Equation 26*. With this data, plugging in numbers allows the calculation of matrices needed for the calculation. The fission matrix is created by plugging in data from the Group Sample Data Table into *Equation 8*. The absorption matrix is created by plugging in data from the Group Sample Data Table into *Equation 6*. The outscattering matrix is created by plugging in data from the Scattering Data table into *Equation 9* and the inscattering matrix is created by plugging that data into *Equation 10*.

$$Fission\ Matrix = \begin{bmatrix} 0.0046 & 0.1139 \\ 0 & 0 \end{bmatrix} \tag{27}$$

$$Absorbtion\ Matrix = \begin{bmatrix} 0.0092 & 0 \\ 0. & 0.0932 \end{bmatrix} \tag{28}$$

$$Outscattering\ Matrix = \begin{bmatrix} 0.0202 & 0 \\ 0 & 0 \end{bmatrix} \tag{29}$$

$$Inscattering\ Matrix = \begin{bmatrix} 0 & 0 \\ 0.0202 & 0 \end{bmatrix} \tag{30}$$

With these matrices, the migration matrix is able to be computed by subtracting the inscattering matrix from the sum of the absorption matrix and the outscattering matrix.

$$Migration\ Matrix = A + S_{out} - S_{in} = \begin{bmatrix} 0.0294 & 0 \\ -0.0202 & 0.0932 \end{bmatrix} \tag{31}$$

From the migration matrix, the B matrix is found by multiplying the inverse of that matrix by the fission matrix.

$$B\ Matrix = (A + S_{Out} - S_{in})^{-1}F = \begin{bmatrix} 0.15646259 & 3.87414966 \\ 0.03391142 & 0.83967621 \end{bmatrix} \tag{32}$$

These new values are necessary to finish the hand calculation from earlier. Plugging the values into *Equation 23* and simplifying yields:

$$0 = k^2 - 0.9961387988671864k \tag{33}$$

$$k = 0\ \&\ 0.9961387988671864 \tag{34}$$

Selecting the larger k value (the non-zero one), $S - kI\vec{\phi} = 0$ can be performed to solve for $\vec{\phi}$. In doing this, $\vec{\phi}$, the corresponding eigenvector to that k value, $\begin{bmatrix} 0.97730867 \\ 0.21182012 \end{bmatrix}$.

$$\vec{\phi} = \begin{bmatrix} 0.97730867 \\ 0.21182012 \end{bmatrix} \tag{35}$$

Along with the hand calculation, computational methods for generating the eigenvalue and vector may also be performed. The numpy.linalg.eig function was used to find the eigenvalues and eigenvectors.

$$k = 9.96138799 * 10^{-01} \ \& \ -1.11022302 * 10^{-16} \approx 0 \tag{36}$$

$$\vec{\phi} = \begin{bmatrix} -0.97730867 \\ -0.21182012 \end{bmatrix} \ \& \ \begin{bmatrix} -0.99918547 \\ 0.04035341 \end{bmatrix} \tag{37}$$

From these, the largest eigenvalue and it's corresponding eigenvector are selected and match up with the hand calculation. Lastly, the power iteration method was employed to find the largest eigenvalue and it's eigenvector. Initial Values from *Equation 26* were used to start the iteration and *Equations 24 & 25* were used to iterate through many iterations. The progress through the iterations is shown below.

$$Before\ Iteration:\ k = 1\ \vec{\phi} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{38}$$

$$After\ 1\ Iteration:\ k = 0.9961388\ \&\ \vec{\phi} = \begin{bmatrix} 0.97730867 \\ 0.21182012 \end{bmatrix} \tag{39}$$

$$After\ 2\ Iterations:\ k = 0.9961388\ \&\ \vec{\phi} = \begin{bmatrix} 0.97730867 \\ 0.21182012 \end{bmatrix} \tag{40}$$

Being a two-group calculation, the flux and k values converged very quickly and finished with values that match up with both the hand calculation and the numpy calculation.

2 Eight Group Neutron Balance through an Infinite Medium

2.1 Neutron Balance with Many Groups

With many groups, the original premise outlined in *Equations 1 & 2* are still valid. The difference lies in that there are more groups, so there are more absorption values, more fission and χ values, and more scattering cross sections. For a system with n groups, *Equation 3* changes to accommodate these new values. The loss from absorption only pertains to the specific group the equation references and therefore will not actually change. The outscattering component of the equation now not only considers scattering to the other one group, but must consider scattering to every other group in the system (not including itself). Since all neutrons are scattering out of the same group, the flux component is constant throughout this summation. Inscattering abides by a similar principle in that with many groups, there many more energy ranges that neutrons can scatter from which all need to be considered. In this case, flux does change since for each group, there is scattering to every other group each a part of different equation. The summation will therefore include the flux inside of it. For the fission component, the total fission cross section is still summed over and multiplied by the probability that neutrons are produced in the right energy range. This means that χ remains in the same energy level while the total fission is summed over. The flux is also located in the summation because each fission occurs from particles in it's respective energy level.

$$\Sigma_{a_g}\phi_g + \phi_g(\sum_{i=1}^{g-1}\Sigma_{s_{g\to i}} + \sum_{i=g+1}^{n}\Sigma_{s_{g\to i}}) = \frac{\chi_g}{k}(\sum_{i=1}^{n}\nu\Sigma_{f_i}\phi_i) + (\sum_{i=1}^{g-1}\Sigma_{s_{i\to g}}\phi_i + \sum_{i=g+1}^{n}\Sigma_{s_{i\to g}}\phi_i) \tag{41}$$

Equation 41 demonstrates what the equation for group g in a system with n number of groups would look like. Some considerations in this equation is that for groups $g = 1$ and $g = n$, the $\sum_{i=1}^{g-1}$ and $\sum_{i=g+1}^{n}$ respectively need to be ignored to account for over/under indexing.

Each component of this equation can be transformed into a matrix representing the component for the entire system. The flux vector will continue to contain all fluxes of the different energy levels and the other matrices will be built around that.

$$Flux\ Vector = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_n \end{bmatrix} \tag{42}$$

The absorption matrix is very similar to the one in *Equation 6* with the addition of more groups.

$$Absorbtion\ Matrix = \begin{bmatrix} \Sigma_{a_1} & 0 & \dots & 0 \\ 0 & \Sigma_{a_2} & & \\ \vdots & & \ddots & \\ 0 & & & \Sigma_{a_n} \end{bmatrix} \tag{43}$$

The fission matrix is similar to it's two-group counterpart in that each row corresponds to a different χ and each column to a different $\nu\Sigma_f$. The only real difference is there are more values for each group.

$$Fission\ Matrix = \begin{bmatrix} \chi_1\nu\Sigma_{f_1} & \chi_1\nu\Sigma_{f_2} & \dots & \chi_1\nu\Sigma_{f_n} \\ \chi_2\nu\Sigma_{f_1} & \chi_2\nu\Sigma_{f_2} & & \\ \vdots & & \ddots & \\ \chi_n\nu\Sigma_{f_1} & & & \chi_n\nu\Sigma_{f_n} \end{bmatrix} \tag{44}$$

The scattering matrices are a little more complicated. The outscattering matrix has one flux per equation in the system and therefore is represented with a diagonal matrix. A large difference when generalizing to a many-group system is that each cross section includes scattering to every other group other than itself. To represent this, a sum is taken over these cross sections (again, for $g = 1$ and $g = n$, ignore $\sum_{i=1}^{g-1}$ and $\sum_{i=g+1}^{n}$ respectively).

$$Outscattering\ Matrix = \begin{bmatrix} \sum_{i=2}^{n} \Sigma_{s_{1\to i}} & 0 & \dots & 0 \\ 0 & \Sigma_{s_{2\to 1}} + \sum_{i=3}^{n} \Sigma_{s_{g\to i}} & & \\ \vdots & & \ddots & \\ 0 & & & \sum_{i=1}^{n} \Sigma_{s_{n\to i}} \end{bmatrix} \tag{45}$$

For the inscattering matrix, every cross section for scattering to the energy group is taken into account in every equation. The major difference is that each one subbed into an equation is multiplied by a different flux value. Because of that, instead of summing as was done into the outscattering matrix, values are placed in different indices corresponding to where they are scattered to and which flux they are multiplied by. In other words, each row corresponds to the cross sections scattering to that number and the columns correspond to where the neutrons are scattering to. One final step is removing the cross sections where neutrons are scattered to the same energy level as these do not remove neutrons from the energy level. In practice, this means making the diagonal zeros.

$$Inscattering\ Matrix = \begin{bmatrix} 0 & \Sigma_{s_{2\to 1}} & \dots & \Sigma_{s_{n\to 1}} \\ \Sigma_{s_{1\to 2}} & 0 & & \\ \vdots & & \ddots & \\ \Sigma_{s_{1\to n}} & & & 0 \end{bmatrix} \tag{46}$$

Putting all of these together yields the enormous *Equation 47*.

$$\begin{bmatrix} \Sigma_{a_1} & 0 & \dots & 0 \\ 0 & \Sigma_{a_2} & & \\ \vdots & & \ddots & \\ 0 & & & \Sigma_{a_n} \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_n \end{bmatrix} + \begin{bmatrix} \sum_{i=2}^{n} \Sigma_{s_{1\to i}} & 0 & \dots & 0 \\ 0 & \Sigma_{s_{2\to 1}} + \sum_{i=3}^{n} \Sigma_{s_{g\to i}} & & \\ \vdots & & \ddots & \\ 0 & & & \sum_{i=1}^{n} \Sigma_{s_{n\to i}} \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_n \end{bmatrix} = \tag{47}$$

$$\begin{bmatrix} \chi_1\nu\Sigma_{f_1} & \chi_1\nu\Sigma_{f_2} & \dots & \chi_1\nu\Sigma_{f_n} \\ \chi_2\nu\Sigma_{f_1} & \chi_2\nu\Sigma_{f_2} & & \\ \vdots & & \ddots & \\ \chi_n\nu\Sigma_{f_1} & & & \chi_n\nu\Sigma_{f_n} \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_n \end{bmatrix} + \begin{bmatrix} 0 & \Sigma_{s_{2\to 1}} & \dots & \Sigma_{s_{n\to 1}} \\ \Sigma_{s_{1\to 2}} & 0 & & \\ \vdots & & \ddots & \\ \Sigma_{s_{1\to n}} & & & 0 \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_n \end{bmatrix}$$

Solving this equation can be done with the same methods described in *Section 1.* Although, this paper will not discuss hand calculations of very large matrices, an example of an eight-group system is performed using the aforementioned computational methods.

Table 3: Eight Group Sample Data

Group	Σ_a	$\nu\Sigma_f$	χ
1	0.0056	0.0134	0.3507
2	0.0029	0.0056	0.4105
3	0.0025	0.0011	0.2388
4	0.0133	0.0067	0.0000
5	0.0473	0.0220	0.0000
6	0.0180	0.0222	0.0000
7	0.0558	0.0897	0.0000
8	0.1798	0.2141	0.0000

Table 4: Eight Group Sample Scattering Data

To Row ↓ From Column →	1	2	3	4	5	6	7	8
1	0.1179	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
2	0.0530	0.1949	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
3	0.0301	0.1159	0.5868	0.0000	0.0000	0.0000	0.0000	0.0000
4	0.0001	0.0005	0.0769	0.8234	0.0000	0.0000	0.0000	0.0000
5	0.0000	0.0000	0.0019	0.1961	0.8180	0.0000	0.0000	0.0000
6	0.0000	0.0000	0.0000	0.0050	0.1737	0.6902	0.0023	0.0000
7	0.0000	0.0000	0.0000	0.0007	0.0246	0.2707	0.8626	0.0275
8	0.0000	0.0000	0.0000	0.0001	0.0073	0.0550	0.3589	1.9761

2.2 Applying Neutron Balance to an Eight-Group Example

The process outlined will be demonstrated with an eight group example. Both computational methods discussed will be used to solve for a final flux vector as well as the largest eigenvalue k.

The first step in this process is creating all of the matrices needed.

$$Absorbtion\ Matrix = \begin{bmatrix} 0.0056 & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0.0029 & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0.0025 & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0.0133 & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0.0473 & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0.018 & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0.0558 & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0.1798 \end{bmatrix} \tag{48}$$

$$Fission\ Matrix = \begin{bmatrix} 0.00469938 & 0.00196392 & 0.00038577 & 0.00234969 & 0.0077154 & 0.00778554 & 0.03145779 & 0.07508487 \\ 0.0055007 & 0.0022988 & 0.00045155 & 0.00275035 & 0.009031 & 0.0091131 & 0.03682185 & 0.08788805 \\ 0.00319992 & 0.00133728 & 0.00026268 & 0.00159996 & 0.0052536 & 0.00530136 & 0.02142036 & 0.05112708 \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \end{bmatrix} \tag{49}$$

$$Outscattering\ Matrix = \begin{bmatrix} 0.0832 & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0.1164 & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0.0788 & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0.2019 & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0.2056 & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0.3257 & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0.3612 & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0.0275 \end{bmatrix} \tag{50}$$

$$Inscattering\ Matrix = \begin{bmatrix} 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.053 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.0301 & 0.1159 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.0001 & 0.0005 & 0.0769 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.0019 & 0.1961 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.0050 & 0.1737 & 0.000 & 0.0023 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.0007 & 0.0246 & 0.2707 & 0.000 & 0.0275 \\ 0.000 & 0.000 & 0.000 & 0.0001 & 0.0073 & 0.0552 & 0.3589 & 0.000 \end{bmatrix} \tag{51}$$

The matrices are combined to create the migration matrix $A + S_{out} - S_{in}$ and B matrix $M = (A + S_{out} - S_{in})F$.

$$Migration\ Matrix = \begin{bmatrix} 0.08880 & 0.00000 & 0.00000 & 0.00000 & 0.00000 & 0.00000 & 0.00000 & 0.00000 \\ -0.05300 & 0.1193 & 0.00000 & 0.00000 & 0.00000 & 0.00000 & 0.00000 & 0.00000 \\ -0.03010 & -0.1159 & 0.08130 & 0.00000 & 0.00000 & 0.00000 & 0.00000 & 0.00000 \\ -0.00010 & -0.00050 & -0.07690 & 0.2152 & 0.00000 & 0.00000 & 0.00000 & 0.00000 \\ 0.00000 & 0.00000 & -0.00190 & -0.1961 & 0.2529 & 0.00000 & 0.00000 & 0.00000 \\ 0.00000 & 0.00000 & 0.00000 & -0.00500 & -0.1737 & 0.3437 & -0.00230 & 0.00000 \\ 0.00000 & 0.00000 & 0.00000 & -0.00070 & -0.02460 & -0.2707 & 0.4170 & -0.02750 \\ 0.00000 & 0.00000 & 0.00000 & -0.00010 & -0.00730 & -0.05500 & -0.3589 & 0.2073 \end{bmatrix} \tag{52}$$

$$B\ Matrix = \begin{bmatrix} 0.0529 & 0.0221 & 0.0043 & 0.0265 & 0.0869 & 0.0877 & 0.3542 & 0.8456 \\ 0.0696 & 0.0291 & 0.0057 & 0.0348 & 0.1143 & 0.1153 & 0.4660 & 1.1123 \\ 0.1582 & 0.0661 & 0.0130 & 0.0791 & 0.2597 & 0.2621 & 1.0590 & 2.5277 \\ 0.0567 & 0.0237 & 0.0047 & 0.0284 & 0.0931 & 0.0940 & 0.3797 & 0.9062 \\ 0.0452 & 0.0189 & 0.0037 & 0.0226 & 0.0742 & 0.0748 & 0.3024 & 0.7217 \\ 0.0238 & 0.0099 & 0.0020 & 0.0119 & 0.0391 & 0.0394 & 0.1593 & 0.3802 \\ 0.0211 & 0.0088 & 0.0017 & 0.0106 & 0.0347 & 0.0350 & 0.1415 & 0.3378 \\ 0.0445 & 0.0186 & 0.0037 & 0.0223 & 0.0731 & 0.0738 & 0.2981 & 0.7116 \end{bmatrix} \tag{53}$$

At this point, computational methods are applied to the matrix equation. The first method applied is the numpy.linalg.eig function which calculates all eigenvalues and their respective eigenvectors.

$$k = 0, 1.09, 0, 0, 0, 0, 0, 0 \tag{54}$$

$$\vec{\phi} = \begin{bmatrix} -0.98273382 \\ 0.06596855 \\ 0.14990534 \\ 0.05374405 \\ 0.04279964 \\ 0.0225461 \\ 0.02003412 \\ 0.04220015 \end{bmatrix}, \begin{bmatrix} -0.26158725 \\ -0.3441239 \\ -0.78197886 \\ -0.280355 \\ -0.22326364 \\ -0.1176114 \\ -0.10450765 \\ -0.22013644 \end{bmatrix}, \begin{bmatrix} 0.399973 - 0.21180175i \\ -0.12928709 - 0.19583017i \\ -0.78623794 \\ 0.17629812 - 0.10921823i \\ -0.24539138 + 0.0347829i \\ -0.12852035 + 0.00928834i \\ -0.00773372 - 0.00770253i \\ 0.01865255 + 0.02048597i \end{bmatrix}, \begin{bmatrix} 0.399973 + 0.21180175i \\ -0.12928709 + 0.19583017i \\ -0.78623794 \\ 0.17629812 + 0.10921823i \\ -0.24539138 - 0.0347829i \\ -0.12852035 - 0.00928834i \\ -0.00773372 + 0.00770253i \\ 0.01865255 - 0.02048597i \end{bmatrix}, \tag{55}$$

$$\begin{bmatrix} -0.7430386 \\ 0.28121498 \\ -0.3052725 + 0.41576903i \\ -0.17907207 + 0.20099353i \\ -0.01974902 + 0.04655395i \\ -0.04475517 + 0.0393373i \\ 0.00649408 - 0.07496547i \\ 0.050271 + 0.01082343i \end{bmatrix}, \begin{bmatrix} -0.7430386 \\ 0.28121498 \\ -0.3052725 - 0.41576903i \\ -0.17907207 - 0.20099353i \\ -0.01974902 - 0.04655395i \\ -0.04475517 - 0.0393373i \\ 0.00649408 + 0.07496547i \\ 0.050271 - 0.01082343i \end{bmatrix}, \begin{bmatrix} -0.02135173 - 0.19441034i \\ 0.0286494 - 0.0079735i \\ 0.9718174 \\ -0.0721313 - 0.03745027i \\ -0.01176025 - 0.00764278i \\ 0.03637053 + 0.08896724i \\ -0.00809824 - 0.01022412i \\ -0.00131869 - 0.01494329i \end{bmatrix}, \begin{bmatrix} -0.02135173 + 0.19441034i \\ 0.0286494 + 0.0079735i \\ 0.9718174 \\ -0.0721313 + 0.03745027i \\ -0.01176025 + 0.00764278i \\ 0.03637053 - 0.08896724i \\ -0.00809824 + 0.01022412i \\ -0.00131869 + 0.01494329i \end{bmatrix}$$

There is only one non-zero eigenvalue. Taking that value's eigenvector and multiplying it by -1 preserves it as an eigenvector. That eigenvector and eigenvalue are the solution.

$$k = 1.09\,\vec{\phi} = \begin{bmatrix} 0.26158725 \\ 0.3441239 \\ 0.78197886 \\ 0.280355 \\ 0.22326364 \\ 0.1176114 \\ 0.10450765 \\ 0.22013644 \end{bmatrix} \tag{56}$$

The power iteration method is also performed using the method discussed earlier and summarized in *Equations 24 & 25.*

$$Before\ Iteration : k = 1\,\vec{\phi} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{57}$$

$$After\ One\ Iteration : k = 1.090031\,\vec{\phi} = \begin{bmatrix} 0.26158725 \\ 0.3441239 \\ 0.78197886 \\ 0.280355 \\ 0.22326364 \\ 0.1176114 \\ 0.10450765 \\ 0.22013644 \end{bmatrix} \tag{58}$$

$$After\ Two\ Iterations : k = 1.090031\,\vec{\phi} = \begin{bmatrix} 0.26158725 \\ 0.3441239 \\ 0.78197886 \\ 0.280355 \\ 0.22326364 \\ 0.1176114 \\ 0.10450765 \\ 0.22013644 \end{bmatrix} \tag{59}$$

As shown above, the vector converged very quickly and after just a couple iterations, the final flux vector is calculated.