CP 2 - Neutron Balance
NPRE 247

Nalin Gadihoke

03/30/2018

Contents

Project was written in Latex and programming was done in Python.

1 Theory

For this project, a program was written to read cross section data, set up Migration and Fission matrices which were then used to setup the equation $B\phi = k\phi$. The Eigenvectors and values were then calculated for the neutron balance problem.

Table 1: List of Variables

Symbol	**Variable**
L	loss of neutrons through Leakage
A	loss of neutrons through absorption
S_o	scattering out of neutrons
S_i	scattering in of neutrons
F	neutrons produced in fission
χ	fission spectrum
ϕ	neutron flux
ν	neutrons per fission
$\sum_a$	absorption cross section
$\sum_{s,g\leftarrow g'}$	scattering cross section from g' to g
g	energy group
G	total number of groups
k	eigenvalue

1.1 Neutron Balance in Infinite Space

Neutron balance refers to the equilibrium in the gain of neutrons and loss of neutrons. This is shown by the following equation:

$$L + A + S_o = F + S_i \tag{1}$$

Since the reaction rate is equal to the cross-section multiplied by the flux and since in infinite space we will consider no neutron diffusion taking place, $L = 0$ and (1) can be written in terms of $\sum$ and ϕ as

$$(\sum_a + \sum_{s,1\to g})\phi = (\sum_{s,g\to 1} + \frac{1}{k}\chi\nu\sum_f)\phi \tag{2}$$

For a multi-group problem we evaluate these variables as matrices with the same naming convention as before (a matrix is denoted with lines above the variable name corresponding to dimensions). Hence, (2) takes the form of

$$\bar{\bar{A}}\bar{\phi} + \bar{\bar{S}}_{out}\bar{\phi} - \bar{\bar{S}}_{in}\bar{\phi} - \frac{1}{k}\bar{\bar{f}}\bar{\phi} = 0$$

rearranging, we get

$$k(\bar{\bar{A}} + \bar{\bar{S}}_{out} - \bar{\bar{S}}_{in})\bar{\phi} - \bar{\bar{f}}\bar{\phi} = 0 \tag{3}$$

Let the migration matrix be

$$\bar{\bar{M}} = \bar{\bar{A}} + \bar{\bar{S}}_{out} - \bar{\bar{S}}_{in} \tag{4}$$

Therefore,

$$k\bar{\phi} = \bar{\bar{M}}^{-1}\bar{\bar{f}}\bar{\phi}$$

Let matrix B be

$$\bar{\bar{B}} = \bar{\bar{M}}^{-1}\bar{\bar{f}}$$

Therefore, we have our final equation:

$$k\bar{\phi} = \bar{\bar{B}}\bar{\phi} \tag{5}$$

1.2 Multi-group Migration and Fission Matrices

Now we define our three matrices according to inputs (provided in input file) and (3)

$$\bar{\bar{A}} = \begin{bmatrix} \sum_{a1} & 0 & 0 \\ 0 & \sum_{ai} & 0 \\ 0 & 0 & \sum_{aG} \end{bmatrix}$$

$$\bar{\bar{S}}_{out} = \begin{bmatrix} \sum_i^G(\sum_{s,g=1\to i}) & 0 & 0 \\ 0 & \dots & 0 \\ 0 & 0 & \sum_i^G(\sum_{s,g=G\to i}) \end{bmatrix}$$

$$\bar{\bar{S}}_{in} = \begin{bmatrix} 0 & \sum_{s,g=i\to 1} & \sum_{s,g=G\to 1} \\ \dots & \dots & \dots \\ \sum_{s,g=1\to G} & \sum_{s,g=i\to G} & 0 \end{bmatrix}$$

$$\bar{\bar{f}} = \begin{bmatrix} \chi_i\nu_i\sum_{fi} & \dots & \chi_i\nu_i\sum_{fG} \\ \dots & \dots & \dots \\ \chi_G\nu_G\sum_{fi} & \dots & \chi_G\nu_G\sum_{fG} \end{bmatrix}$$

$$\bar{\phi} = \begin{bmatrix} \phi_{g=1} \\ \dots \\ \phi_{g=G} \end{bmatrix}$$

The migration matrix is constructed according to (4)

$$\bar{\bar{M}} = \begin{bmatrix} \sum_{a1} & 0 & 0 \\ 0 & \sum_{ai} & 0 \\ 0 & 0 & \sum_{aG} \end{bmatrix} + \begin{bmatrix} 0 & \sum_{s,g=i\to 1} & \sum_{s,g=G\to 1} \\ \cdots & \cdots & \cdots \\ \sum_{s,g=1\to G} & \sum_{s,g=i\to G} & 0 \end{bmatrix} - \begin{bmatrix} \chi_i \nu_i \sum_{fi} & \cdots & \chi_i \nu_i \sum_{fG} \\ \cdots & \cdots & \cdots \\ \chi_G \nu_G \sum_{fi} & \cdots & \chi_G \nu_G \sum_{fG} \end{bmatrix}$$

2 2G Eigenvalue/Eigenvector Problem

2.1 Inputs

Inputs were provided in a table format and can be found in the input file in the programming part of the project.

2.2 The Matrices

All necessary matrices are made according to outlines in section 1.2

$$\bar{\bar{A}} = \begin{bmatrix} 0.0092 & 0 \\ 0 & 0.0932 \end{bmatrix} \bar{\bar{S}}_{out} = \begin{bmatrix} 0.0202 & 0 \\ 0 & 0 \end{bmatrix} \bar{\bar{S}}_{in} = \begin{bmatrix} 0 & 0 \\ 0.0202 & 0 \end{bmatrix}$$

$$\bar{\bar{M}} = \begin{bmatrix} 0.0294 & 0 \\ -0.0202 & 0.0932 \end{bmatrix} \bar{\bar{f}} = \begin{bmatrix} 0.0046 & 0.1139 \\ 0 & 0 \end{bmatrix} \bar{\bar{B}} = \bar{\bar{M}}^{-1}\bar{\bar{f}} = \begin{bmatrix} 0.1564 & 3.8741 \\ 0.0339 & 0.8396 \end{bmatrix}$$

2.3 Analytical Solution

Assuming I_G is an identity matrix, we can rewrite (5)

$$(k\bar{\bar{I}}_G - \bar{\bar{B}})\bar{\phi} = 0$$

From the previous subsection, we have matrix B

$$\bar{\bar{B}} = \begin{bmatrix} 0.1564 & 3.8741 \\ 0.0339 & 0.8396 \end{bmatrix}$$

Therefore, we deduce that the equation will have a solution only when

$$determinant(k\bar{\bar{I}}_G - \bar{\bar{B}}) = (k - 0.1564)(k - 0.8396) + (3.8741 * 0.0339) = 0$$

This simplifies to the following,

$$k^2 - 0.99613k - 1.110 * 10^{-16} = 0$$

giving us roots $k_1 = 0.99613$ and $k_2 = 0$. These can be solved independently; for the first root the following steps were followed

$$(\begin{bmatrix} 0.99613 & 0 \\ 0 & 0.99613 \end{bmatrix} - \begin{bmatrix} 0.1564 & 3.8741 \\ 0.0339 & 0.8396 \end{bmatrix})\bar{\phi} = \begin{bmatrix} 0.8396 & -3.8741 \\ -0.0339 & 0.1564 \end{bmatrix}\bar{\phi} = 0$$

$$\begin{bmatrix} 0.8396 & -3.8741 \\ 0.8057 & -3.7177 \end{bmatrix} \begin{bmatrix} \phi_{1'} \\ \phi_{2'} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\phi_{1'}(0.8057 + 0.8396) - \phi_{2'}(3.8741 + 3.7177) = 0$$

Therefore we get

$$\phi_{1'} = 4.6142\phi_{2'}, \bar{\phi_1} = \begin{bmatrix} -0.9773 \\ -0.2118 \end{bmatrix} (\frac{neutrons}{cm^2 s}), k_1 = 0.99613 \tag{6}$$

And k_2 is solved as follows

$$(\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} 0.1564 & 3.8741 \\ 0.0339 & 0.8396 \end{bmatrix})\bar{\phi} = \begin{bmatrix} -0.15646 & -3.8741 \\ -0.0339 & -0.8396 \end{bmatrix} \bar{\phi} = 0$$

$$\begin{bmatrix} -0.1564 & -3.8741 \\ -0.1903 & -4.7137 \end{bmatrix} \begin{bmatrix} \phi_{1'} \\ \phi_{2'} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\phi_{1'}(0.1564 + 0.1903) + \phi_{2'}(3.8741 + 4.7137) = 0$$

$$\phi_{1'} = -24.77\phi_{2'}, \bar{\phi_2} = \begin{bmatrix} -0.9991 \\ 0.04035 \end{bmatrix} (\frac{neutrons}{cm^2 s}), k_2 = 0 \tag{7}$$

2.4 Solution using the python 'eig' function

To make use of a computer to verify the results, the 'eig' function in python was used, which accepts an array and returns the Eigen vectors as column vectors and Eigen values as a single vector. These were the data structures computed

$$vectors = \begin{bmatrix} -0.99918547 & -0.97730867 \\ 0.04035341 & -0.21182012 \end{bmatrix}, values = \begin{bmatrix} 0 & 0 \\ 0 & 0.996138799 \end{bmatrix}$$

Written out separately, the match the values analytically calculated in (7) and (8)

$$\phi_{1'} = 4.6142\phi_{2'}, \bar{\phi_1} = \begin{bmatrix} -0.9773 \\ -0.2118 \end{bmatrix} (\frac{neutrons}{cm^2 s}), k_1 = 0.99613$$

$$\phi_{1'} = -24.77\phi_{2'}, \bar{\phi_2} = \begin{bmatrix} -0.9991 \\ 0.04035 \end{bmatrix} (\frac{neutrons}{cm^2 s}), k_2 = 0$$

2.5 Power Iteration

Next, we used power iteration on the Eigen values and vectors with given initial starting values as 1 for both the vector and value. The largest of both were calculated using the formula given to us

$$\bar{\phi}_{i+1} = \frac{\bar{\bar{B}}\bar{\phi}_i}{\left\|\bar{\bar{B}}\bar{\phi}_i\right\|_2} \quad k_{i+1} = \frac{(\bar{\bar{B}}\bar{\phi}_i)^T\bar{\phi}_{i+1}}{\bar{\phi}_{i+1}^T\bar{\phi}_{i+1}} \tag{8}$$

This was implemented via a for loop and the maximum values were found to be as follows in the second iteration

$$\bar{\phi}_{i=2} = \begin{bmatrix}0.97730867\\0.21182012\end{bmatrix}(\frac{neutrons}{cm^2 s}), k_{i=2} = 0.9961388$$

3 8G Eigenvalue/Eigenvector Problem

3.1 Inputs

Inputs were provided in a table format and can be found in the input file in the programming part of the project.

3.2 The Matrices

In this section, the matrices became too complicated to do by hand so functions were implemented to populate the arrays accordingly. Here are the migration and fission matrices

$$\bar{\bar{M}} = \begin{bmatrix} 0.0888 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.053 & 0.1193 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.0301 & -0.1159 & 0.0813 & 0 & 0 & 0 & 0 & 0 \\ -0.0001 & -0.0005 & -0.0769 & 0.2152 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0.0019 & -0.1961 & 0.2529 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.005 & -0.1737 & 0.3437 & -0.0023 & 0 \\ 0 & 0 & 0 & -0.0007 & -0.0246 & -0.2707 & 0.417 & -0.0275 \\ 0 & 0 & 0 & -0.0001 & -0.0073 & -0.055 & -0.3589 & 0.2073 \end{bmatrix}$$

$$\bar{\bar{f}} = \begin{bmatrix} 0.004699 & 0.001963 & 0.0003857 & 0.002349 & 0.007715 & 0.007785 & 0.03145 & 0.07508 \\ 0.005500 & 0.002298 & 0.0004515 & 0.002750 & 0.009031 & 0.009113 & 0.03682 & 0.08788 \\ 0.003199 & 0.001337 & 0.0002626 & 0.001599 & 0.005253 & 0.005301 & 0.02142 & 0.05112 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

3.3 Solution using the python 'eig' function

Section 2.4 goes over how the following matrices were calculated. The matrix vectors holds the Eigen Vectors in each of its columns

$$vectors = \begin{bmatrix} -0.9827 & -0.2615 & -0.3405 & -0.2160 & -0.2160 & 0.6949 & -0.3218 & -0.3218 \\ 0.06596 & -0.3441 & -0.6733 & 0.7237 & 0.7237 & -0.04546 & -0.1006 & -0.1006 \\ 0.1499 & -0.7819 & -0.6305 & -0.3516 & -0.3516 & -0.3168 & -0.6704 & -0.6704 \\ 0.05374 & -0.28035 & -0.03253 & 0.24800 & 0.24800 & 0.6409 & 0.2277 & 0.2277 \\ 0.04279 & -0.2232 & -0.1093 & -0.1374 & -0.1374 & 0.0287 & 0.01155 & 0.01155 \\ 0.02254 & -0.1176 & -0.07239 & -0.04970 & -0.04970 & 0.04220 & -0.05812 & -0.05812 \\ 0.02003 & -0.1045 & 0.1207 & 0.02256 & 0.02256 & -0.02836 & 0.01663 & 0.01663 \\ 0.042200 & -0.2201 & 0.01135 & -0.001538 & -0.001538 & -0.01606 & 0.01695 & 0.01695 \end{bmatrix}$$

$$values = \begin{bmatrix} 0 & 1.090031 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

3.4 Power Iteration

Equation (8) was used here too and the following vector and value were obtained after the second iteration. The output file contains values from the second iteration onwards as described in the problem.

$$\bar{\phi}_{i=2} = \begin{bmatrix} 0.26158725 \\ 0.3441239 \\ 0.78197886 \\ 0.280355 \\ 0.22326364 \\ 0.1176114 \\ 0.10450765 \\ 0.22013644 \end{bmatrix} (\frac{neutrons}{cm^2 s}), k_{i=2} = 1.090031$$

4 Results

This computer project gave us a good insight into eigenvalues and how to solve for them in a neutron diffusion problem.

For the 2G case, analytical values calculated by hand matched those produced by the 'eig' function which matched the largest values calculated using the power iteration method demonstrating different ways to go about solving a similar problem. For the 8G case, the power iteration method converged on the value calculated by the 'eig' function further demonstrating the method's validity.

5 References

1. www.sharelatex.com 2. www.stackoverflow.com