Two group, Heterogeneous Diffusion Equation Solver

5 Dec 2021

Zach Condon

Introduction **3**

Mathematics **4**
One Group Finite Volumes 4
Two Group Finite Volumes 5
Heterogeneous Material 7
Known Errors 7

Algorithms **9**
Inputs and Outputs, Scenario 1 10
Inputs and Outputs, Scenario 2 10
Inputs and Outputs, Scenario 3 11
Inputs and Outputs, Scenario 4 12
Inputs and Outputs, Scenario 5 13

Code Use **16**

Test Problems and Results **17**
Test for Scenario 1 17
Test for Scenario 2 17
Test for Scenario 3 18
Test of Scenario 4 19
Test of Scenario 5 20

Conclusions **22**

Sources **23**

Appendix A - Gauss Seidel **24**

Introduction

The code accompanying this paper solves a two-group neutron diffusion problem using finite volumes. Starting from the diffusion differential equation, some mathematical approximations are made to generate a system that solves for approximate values of neutron flux within a heterogeneous system. Following the discussion of those approximations, a few scenarios will be shown that compare the approximate flux compared to both the analytical solution for the differential diffusion equation as well as a direct solution of the matrix developed during the finite volumes process.

Mathematics

The basis of this project is the diffusion equation:

(1) $$-\frac{d}{dx}D(x)\frac{d\phi(x)}{dx} + \Sigma_a(x)\phi(x) = S(x)$$

where D is the diffusion coefficient and is equal to:

$$D = 1/3\Sigma_t$$

and Σ_t is the total cross section.

The diffusion equation is an approximation for the neutron transport equation assuming that there is a low absorption cross-section, isotropic scattering, no large sources, and no neutron-neutron interactions. In simple cases, like a single-group, homogeneous material, the diffusion equation can be analytically solved with fairly low effort. As more complications are introduced, the analytical solution can become quite unwieldy.

One Group Finite Volumes

To circumnavigate the need to analytically solve the diffusion equation, the finite volumes method can be used to derive an approximate solution for the neutron flux. In this method, the heterogeneous material was divided into cells. Each cell was assigned its own properties (shown in Figure 1 by the green lines). The division between the cells is where the flux was calculated (shown in Figure 1 by the purple lines).

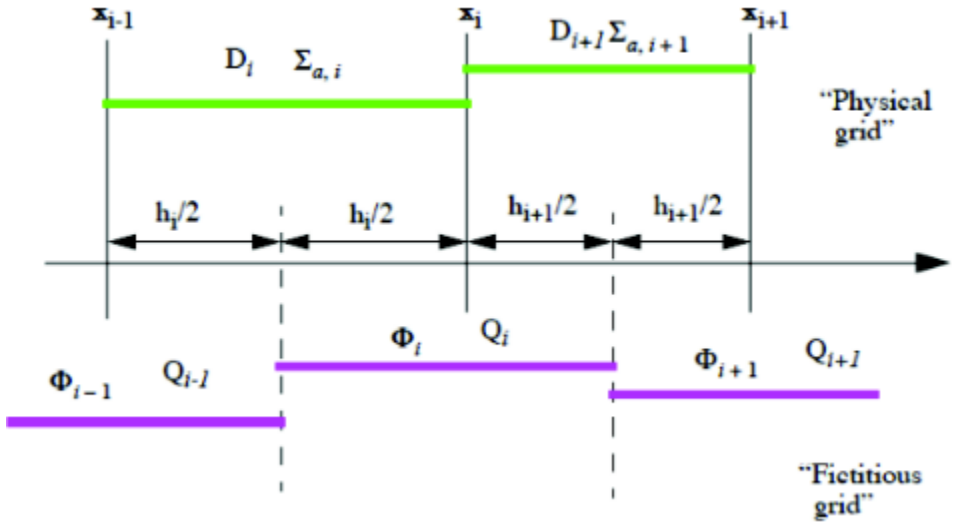

Figure 1. Representation of the cells definition the scenario use the finite volume method

Using the grid represented in Figure 1, the diffusion equation can be expanded to formulate an approximation for the analytical solution. The purple lines demonstrate the assumptions that the flux and source terms are constant over the half-width volumes to either side of the boundary between cells. Similarly, the green lines demonstrate that the diffusion and absorption cross-section terms are constant within a cell.

The first step in the approximation is integrating the diffusion equation over each cell $(x_i - h_i/2) \leq x \leq (x_i + h_{i+1}/2)$:

$$(2) \qquad \int_{(x_i - h_i/2)}^{(x_i + h_{i+1}/2)} \left(-\frac{d}{dx} D(x) \frac{d\phi(x)}{dx}\right) dx + \int_{(x_i - h_i/2)}^{(x_i + h_{i+1}/2)} \Sigma_a(x)\phi(x)dx = \int_{(x_i - h_i/2)}^{(x_i + h_{i+1}/2)} S(x)dx$$

With each term being approximated as:

$$(3) \qquad \int_{(x_i - h_i/2)}^{(x_i + h_{i+1}/2)} S(x)dx \approx S_i\left(\frac{h_i + h_{i+1}}{2}\right)$$

$$(4) \qquad \int_{(x_i - h_i/2)}^{(x_i + h_{i+1}/2)} \Sigma_a(x)\phi(x)dx \approx \left(\frac{\Sigma_{a,i} h_i + \Sigma_{a,i+1} h_{i+1}}{2}\right)\phi_i$$

$$(5) \qquad \int_{(x_i - h_i/2)}^{(x_i + h_{i+1}/2)} \left(-\frac{d}{dx} D(x) \frac{d\phi(x)}{dx}\right) dx = -\left[D(x)\frac{d\phi(x)}{dx}\right]_{x_i + h_i/2}^{x_i + h_{i+1}/2}$$

Simplifying equation 5 even more:

$$(6) \qquad -\left[D(x)\frac{d\phi(x)}{dx}\right]_{x_i + h_i/2}^{x_i + h_{i+1}/2} \approx -D_{i+1}\left(\frac{\phi_{i+1} - \phi_i}{h_{i+1}}\right) + D_i\left(\frac{\phi_i - \phi_{i-1}}{h_i}\right)$$

Using equations 3, 4, and 6, equation 2 then becomes:

$$(7) \qquad -D_{i+1}\left(\frac{\phi_{i+1} - \phi_i}{h_{i+1}}\right) + D_i\left(\frac{\phi_i - \phi_{i-1}}{h_i}\right) + \left(\frac{\Sigma_{a,i} h_i + \Sigma_{a,i+1} h_{i+1}}{2}\right)\phi_i = S_i\left(\frac{h_i + h_{i+1}}{2}\right)$$

This equation can be used to solve for the flux at the *i*th cell if the flux to either side of it is known. Since the flux in each cell is unknown, the boundary conditions must be used to provide termination points on either end of the process. Once boundary conditions are established, a Gauss-Seidel iterative solution method will be used to solve the coupled linear equations. Each cell will have a respective equation 7 that goes along with it.

Two Group Finite Volumes

The finite volumes method for two groups is nearly identical to that shown in the previous section. In Group 1 (the fast neutron group), the only difference arises in the second term on the left side of Equation 1. There are now two ways for neutrons to be lost from Group 1, being

absorbed by the material or being down-scattered into Group 2. Thus the only change to Equation 7 would be to modify the Σ_a terms to be Σ_r, the removal cross-section, which is defined as:

(8) $$\Sigma_r = \Sigma_a + \Sigma_{s,1-2}$$

where Σ_a is the absorption cross section and $\Sigma_{s,1-2}$ is the down-scattering cross section from Group 1 to Group 2.

For Group 2, the only modification to Equation 7 is that the source includes the incoming flux that is down-scattered from Group 1. This can be captured in the source term without by renaming the source variable as:

(9) $$S' = S + \Sigma_{s,1-2}\phi_1$$

where ϕ_1 is the Group 1 flux.

The resulting equations for two groups using the finite volumes method are:

Group 1:

(10) $$-D_{1,i+1}\left(\frac{\phi_{1,i+1}-\phi_{1,i}}{h_{i+1}}\right) + D_{1,i}\left(\frac{\phi_{1,i}-\phi_{1,i-1}}{h_i}\right) + \left(\frac{\Sigma_{1,r,i}h_i+\Sigma_{1,r,i+1}h_{i+1}}{2}\right)\phi_{1,i} = S_{1,i}\left(\frac{h_i+h_{i+1}}{2}\right)$$

Group 2:

(11) $$-D_{2,i+1}\left(\frac{\phi_{2,i+1}-\phi_{2,i}}{h_{i+1}}\right) + D_{2,i}\left(\frac{\phi_{2,i}-\phi_{2,i-1}}{h_i}\right) + \left(\frac{\Sigma_{2,a,i}h_i+\Sigma_{2,a,i+1}h_{i+1}}{2}\right)\phi_{2,i} = S_{2,i}\left(\frac{h_i+h_{i+1}}{2}\right) + \left(\frac{\Sigma_{2,s,1-2,,i}h_i+\Sigma_{1,s,1-2,,i+1}h_{i+1}}{2}\right)\phi_{1,i}$$

where the initial numerical subscript indicates the group number. There are no preceding numerical subscripts for the cell widths because the cell widths will be established to be the same for both groups. This will result in some error in the final results, but greatly simplifies the finite volumes method. If more rigorous steps were taken, the cell width would be different for each group because the extrapolated boundary on the vacuum side would be different for each group. I chose to establish the width of the material on the vacuum boundary as equal to the set width + the Group 1 extrapolated distance to say that the flux for both groups is equal to zero.

Heterogeneous Material

After developing the one and two group models listed above, introducing a heterogeneous material is comparatively simple. The way I implemented the finite volumes method is by making an array for each constant in Equations 7, 10, and 11 that contains an element for each cell in the model. For example, equation 12 is an example of the array containing the information for the absorption cross-section. From left to right, the index describes:

1. That this is the absorption cross-section
2. The cell number (starting from the left side material one with *n1* number of cells and ending at *n2* for a total of *n1+n2* number of elements)
3. That this is for group 1
4. That this is for material 1 or material 2

(12) $$\Sigma_{a,g1} = [\Sigma_{a,0,g1,m1}, \Sigma_{a,1,g1,m1}, \dots, \Sigma_{a,n1,g1,m1}, \Sigma_{a,0,g1,m2}, \Sigma_{a,2,g1,m2}, \dots, \Sigma_{a,n2,g1,m2}]$$

The modification occurs when defining each of those constants and is essentially doubling the size of the array (if there are the same number of cells per material). The code has a variable for each material and each group and will build the array such that it is of a length equal to the sum of the number of cells in both materials. Once the arrays are established, the rest of the process of solving for phi is exactly the same.

Known Errors

In coding the single group scenarios, I included the extrapolated distance at the vacuum boundary directly into my cell width for the material with the vacuum boundary using the following equation:

$$cell\ width = (material\ width + extrapolated\ distance)/number\ of\ cells$$

This was done for ease of calculations, but in hindsight, I would have changed this so that the vacuum boundary had its own defined number of cells in my algorithm.

There are two reasons why my method of including the vacuum boundary leads to errors and can be improved upon. The first is that, with my way of defining a source, my code calculates the flux as if there is material and sources within that extrapolated distance. The second source of error is in the two group scenario. I use the extrapolated distance for group 1 to define the cell width, using the above equation, for both groups. This is incorrect in that group 2 would have a different extrapolated distance. At first, I did this to place the points at which I

calculate phi in the same locations and did not realize the error this introduced at the vacuum boundary.

Algorithms

The end result of using the finite volumes method is a linear system of equations. For a one-dimensional system divided into *n*-cells, there are *n+1* cell-edges to calculate the flux. The system will have *n-1* equations if both boundary conditions are vacuum (since the flux at both ends will be known and zero) and *n* equations if one side is vacuum and the other side is a reflective boundary (since the flux at the reflective boundary is not known).

In developing the code for this project, the first step was to use Equation 7 to solve for a homogeneous material with a vacuum boundary on either side. This was then modified to include a reflective boundary on the right side. Next was to simulate a heterogeneous material by placing two materials with different properties next to each other such that there is a vacuum on the left side, the two materials touching each other, then the reflective boundary on the right. Finally, the solution for the project was developed with two materials, two groups, a vacuum boundary on the left and a reflective boundary on the right. Each of the steps listed in this paragraph will be referred to as scenarios from here.

In all cases, the algorithms are nearly identical and follow the steps listed below:

1. Establish the finite-volumes representation of the system by:
    a. Determining the number of cells in the material or, if heterogeneous, materials.
    b. Input the material parameters, such as cross sections and source strength, for each material and neutron group.
    c. Defining an array for each parameter with the appropriate number of cells such that there is an element associated with each cell
2. Define the system of linear equations according to Equation 7 if solving for one group or Equations 10 and 11 for two groups. The system of equations will use the arrays generated in step 1 to keep track of the values needed for each row.
3. Solve the system of equations using a Gauss-Seidel architecture (described more in Appendix A) converging when the difference between each iteration is no more than .00001.
4. Check the solution using either an analytical solution or a direct-solve of the system of equations.

Inputs and Outputs, Scenario 1

Scenario 1 is the one-group, homogeneous case with vacuum boundaries on both sides. It takes the following inputs:

- **n** - The number of finite volumes (cells)
- **w** - [cm] width of the material, that width will be simulated on both sides of the x-axis for a total width of 2*w
- **Sig_ai_frac** - The ratio of absorption cross section to total cross section. This should be between 0 and 1.
- **Sig_ti** - [1/cm], The macroscopic total cross section. This can be any positive value.
- **Si** - [n/cm^3], The source strength per cell. In this scenario, the source is distributed uniformly through the whole material.

The following are the outputs:

- An image file named “scenario1.eps” of a plot of the flux as a function of position
- A csv file named “scenario1.csv” with two rows of data:
    - Position values
    - Corresponding flux values

Inputs and Outputs, Scenario 2

Scenario 2 is the one-group, homogeneous case with a vacuum boundary on the left side and a reflective boundary on the right side. It takes the following inputs:

- **n** - The number of finite volumes (cells)
- **w** - [cm] width of the material, starting from x = 0 to x = -w
- **Sig_ai_frac** - The ratio of absorption cross section to total cross section. This should be between 0 and 1.
- **Sig_ti** - [1/cm], The macroscopic total cross section. This can be any positive value.
- **Si** - [n/cm^3], The source strength per cell. In this scenario, the source is distributed uniformly through the whole material.

The following are the outputs:

- An image file named “scenario2.eps” of a plot of the flux as a function of position
- A csv file named “scenario2.csv” with two rows of data:

    - Position values
    - Corresponding flux values

Inputs and Outputs, Scenario 3

Scenario 3 is the one group, heterogeneous case with a vacuum boundary on the left side and a reflective boundary on the right side. It takes the following inputs:

- **#----- Material 1 -----**
- **n1** - The number of cells in material 1
- **w1** - [cm], the width of material one, located from x=-w2 to x=-w2-w1
- **Sig_ai_frac1** - The ratio of absorption cross section to total cross section. This should be between 0 and 1.
- **Sig_ti_m1** - [1/cm], The macroscopic total cross section. This can be any positive value.
- **Si_m1** - [n/cm^3], The source strength per cell. In this scenario, the source is distributed uniformly through the whole material.

- **#----- Material 2 -----**
- **n2** - The number of cells in material 2
- **w2** - [cm], the width of material two, located from x=0 to x=-w2
- **Sig_ai_frac2** - The ratio of absorption cross section to total cross section. This should be between 0 and 1.
- **Sig_ti_m2** - [1/cm], The macroscopic total cross section. This can be any positive value.
- **Si_m2** - [n/cm^3], The source strength per cell. In this scenario, the source is distributed uniformly through the whole material.

The following are the outputs:

- An image file named “scenario3.eps” of a plot of the flux as a function of position
- A csv file named “scenario3.csv” with two rows of data:
    - Position values
    - Corresponding flux values

Inputs and Outputs, Scenario 4

Scenario 4 implements all requirements for the project. It is the two group, heterogeneous case with a vacuum boundary on the left side and a reflective boundary on the right side. It takes the following inputs:

- **#----- Geometry -----**
- **n1** - The number of cells in material 1
- **n2** - The number of cells in material 2
- **w1** - [cm], The width of material 1
- **w2** - [cm], The width of material 2
- **#----- Material 1 -- Group 1 -----**
- **Sig_ti_m1_g1** - [1/cm], The macroscopic total cross section. This can be any positive value.
- **Sig_ai_m1_g1_frac** - [1/cm], The macroscopic absorption cross section. This should be between 0 and 1. The sum of Sig_ai_m1_g1_frac and Sig_s12_m1_frac should be equal or less than 1.
- **Sig_s12_m1_frac** - [1/cm], The macroscopic scatter cross section from group 1->2. This should be between 0 and 1. The sum of Sig_ai_m1_g1_frac and Sig_s12_m1_frac should be equal or less than 1.
- **Si_m1_g1** - [n/cm^3], The source strength per cell. In this scenario, the source is distributed uniformly through the whole material.
- **#----- Material 2 -- Group 1 -----**
- **Sig_ti_m2_g1** - [1/cm], The macroscopic total cross section. This can be any positive value.
- **Sig_ai_m2_g1_frac** - [1/cm], The macroscopic absorption cross section. This should be between 0 and 1. The sum of Sig_ai_m2_g1_frac and Sig_s12_m2_frac should be equal or less than 1.
- **Sig_s12_m2_frac** - [1/cm], The macroscopic scatter cross section from group 1->2. This should be between 0 and 1. The sum of Sig_ai_m2_g1_frac and Sig_s12_m2_frac should be equal or less than 1.
- **Si_m2_g1** - [n/cm^3], The source strength per cell. In this scenario, the source is distributed uniformly through the whole material.
- **#----- Material 1 -- Group 2 -----**

- **Sig_ti_m1_g2** - [1/cm], The macroscopic total cross section. This can be any positive value.
- **Sig_ai_m1_g2_frac** - [1/cm], The macroscopic absorption cross section. This should be between 0 and 1.
- **Si_m1_g2** - [n/cm^3], The source strength per cell. In this scenario, the source is distributed uniformly through the whole material.
- **#----- Material 2 -- Group 2 -----**
- **Sig_ti_m2_g2** - [1/cm], The macroscopic total cross section. This can be any positive value.
- **Sig_ai_m2_g2_frac** - [1/cm], The macroscopic absorption cross section. This should be between 0 and 1.
- **Si_m2_g2** - [n/cm^3], The source strength per cell. In this scenario, the source is distributed uniformly through the whole material.

The following are the outputs:

- An image file named “scenario4.eps” of a plot of the both flux groups as a function of position
- A csv file named “scenario4.csv” with three rows of data:
    - Position values
    - Corresponding group 1 flux values
    - Corresponding group 2 flux values

Inputs and Outputs, Scenario 5

Scenario 5 is exactly the same as Scenario 4 with the exception of the source definition. I wanted to be able to test placing sources individually in various cells.

- **#----- Geometry -----**
- **n1** - The number of cells in material 1
- **n2** - The number of cells in material 2
- **w1** - [cm], The width of material 1
- **w2** - [cm], The width of material 2
- **#----- Sources -----**
    - First an array is defined for each group and material.
- **S_m1_g1 =** np.zeros(n1)

- **S_m2_g1** = np.zeros(n2)
- **S_m1_g2** = np.zeros(n1)
- **S_m2_g2** = np.zeros(n2)
    - Then that array can be populated in any cell. The index for each definition must be within the number of cells defined for that material. These arrays are later concatenated for use in the solver.
    - For example: S_m2_g1[12] = 1 puts a group 1 source of strength 1 in the 12th cell in material 2. This can be done for any and up to all cells.
- **S_m2_g1[12]** = 1
- **#----- Material 1 -- Group 1 -----**
- **Sig_ti_m1_g1** - [1/cm], The macroscopic total cross section. This can be any positive value.
- **Sig_ai_m1_g1_frac** - [1/cm], The macroscopic absorption cross section. This should be between 0 and 1. The sum of Sig_ai_m1_g1_frac and Sig_s12_m1_frac should be equal or less than 1.
- **Sig_s12_m1_frac** - [1/cm], The macroscopic scatter cross section from group 1->2. This should be between 0 and 1. The sum of Sig_ai_m1_g1_frac and Sig_s12_m1_frac should be equal or less than 1.
- **#----- Material 2 -- Group 1 -----**
- **Sig_ti_m2_g1** - [1/cm], The macroscopic total cross section. This can be any positive value.
- **Sig_ai_m2_g1_frac** - [1/cm], The macroscopic absorption cross section. This should be between 0 and 1. The sum of Sig_ai_m2_g1_frac and Sig_s12_m2_frac should be equal or less than 1.
- **Sig_s12_m2_frac** - [1/cm], The macroscopic scatter cross section from group 1->2. This should be between 0 and 1. The sum of Sig_ai_m2_g1_frac and Sig_s12_m2_frac should be equal or less than 1.
- **#----- Material 1 -- Group 2 -----**
- **Sig_ti_m1_g2** - [1/cm], The macroscopic total cross section. This can be any positive value.
- **Sig_ai_m1_g2_frac** - [1/cm], The macroscopic absorption cross section. This should be between 0 and 1.
- **#----- Material 2 -- Group 2 -----**

- **Sig_ti_m2_g2** - [1/cm], The macroscopic total cross section. This can be any positive value.
- **Sig_ai_m2_g2_frac** - [1/cm], The macroscopic absorption cross section. This should be between 0 and 1.

The following are the outputs:

- An image file named “scenario5.eps” of a plot of the both flux groups as a function of position
- A csv file named “scenario5.csv” with three rows of data:
    - Position values
    - Corresponding group 1 flux values
    - Corresponding group 2 flux values

Code Use

Hopefully at this stage the code is fairly straight-forward to manipulate when sticking to this guide. The only parameters that should be changed are those described in Step 1a and 1b from above. All of these parameters are in the “Condon_6708Project.py” file. The four scenarios described in the second paragraph of the Algorithms section are programmed in the same order, with the accompanying files containing the functions needed to perform the calculations.

Each scenario file will take the inputs that can be changed in the “Condon_6708Project.py” file and will calculate the flux using the finite volumes method as well as some calculating either the analytical solution or direct solution. All solutions calculated are then shown in a plot. The only value that could be changed without causing issues (other than less correct solutions) would be the convergence limit for the only while statement in each file.

Test Problems and Results

Test for Scenario 1

Inputs:

- **n** = 60
- **w** = 10
- **Sig_ai_frac** = 0.1
- **Sig_ti** = 1
- **Si** = 1

Output:

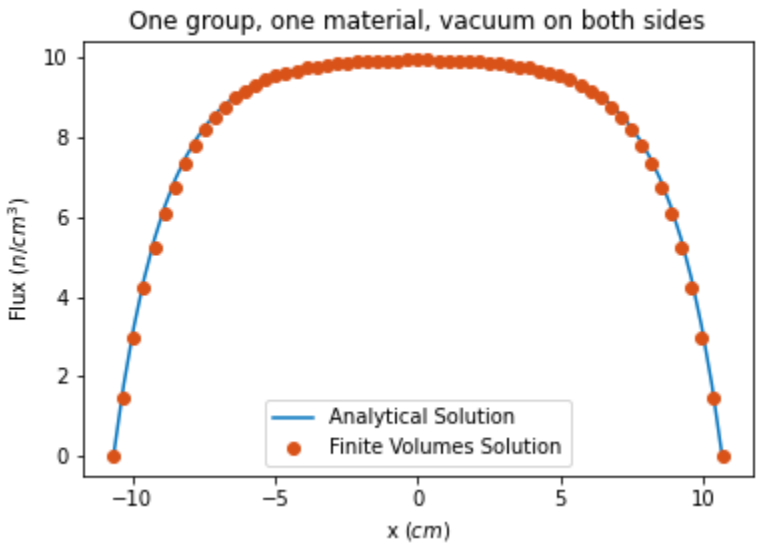

Figure 2. Scenario 1 solution

Test for Scenario 2

Inputs:

- **n** = 60
- **w** = 10
- **Sig_ai_frac** = 0.1
- **Sig_ti** = 1
- **Si** = 1

Outputs:

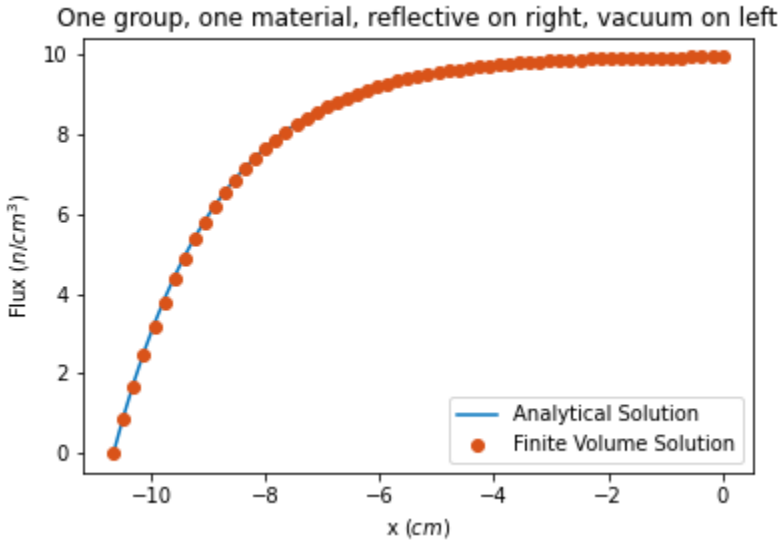

Figure 3. Scenario 2 solution

Note that this has exactly the same inputs as Scenario 1, but is only for the region from x=0 to x=-w-extrapolated distance. For this region, the plots are the same although they are found through different boundary conditions.

Test for Scenario 3

Inputs:

- **#----- Material 1 -----**
- **n1** = 20
- **w1** = 5
- **Sig_ai_frac1** = 0.01
- **Sig_ti_m1** = 1
- **Si_m1** = 1
- **#----- Material 2 -----**
- **n2** = 20
- **w2** = 5
- **Sig_ai_frac2** = 0.2
- **Sig_ti_m2** = 3
- **Si_m2** = 2

Outputs:

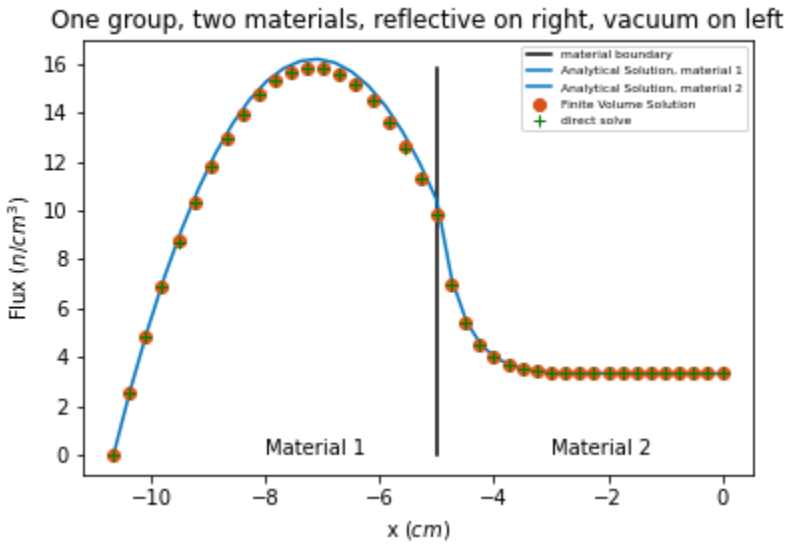

Figure 4. Scenario 3 solution

The results from my code deviate from the analytical solution of the diffusion equation close to the boundary between the two materials. I believe this is caused by my inputs starting to be outside of the diffusion limit. With my inputs, the absorption cross-section changes by about a factor of 60.

Test of Scenario 4

Inputs:

- **#----- Geometry -----**
- **n1** = 20
- **n2** = 20
- **w1** = 5
- **w2** = 5
- **#----- Material 1 -- Group 1 -----**
- **Sig_ti_m1_g1** = 1
- **Sig_ai_m1_g1_frac** = 0.01
- **Sig_s12_m1_frac** = 0.2
- **Si_m1_g1** = 2
- **#----- Material 2 -- Group 1 -----**
- **Sig_ti_m2_g1** = 1

- **Sig_ai_m2_g1_frac** = 0.002
- **Sig_s12_m2_frac** = 0.4
- **Si_m2_g1** = 1
- **#----- Material 1 -- Group 2 -----**
- **Sig_ti_m1_g2** = 1
- **Sig_ai_m1_g2_frac** = 0.2
- **Si_m1_g2** = 0
- **#----- Material 2 -- Group 2 -----**
- **Sig_ti_m2_g2** = 4
- **Sig_ai_m2_g2_frac** = 0.1
- **Si_m2_g2** = 1

Output:

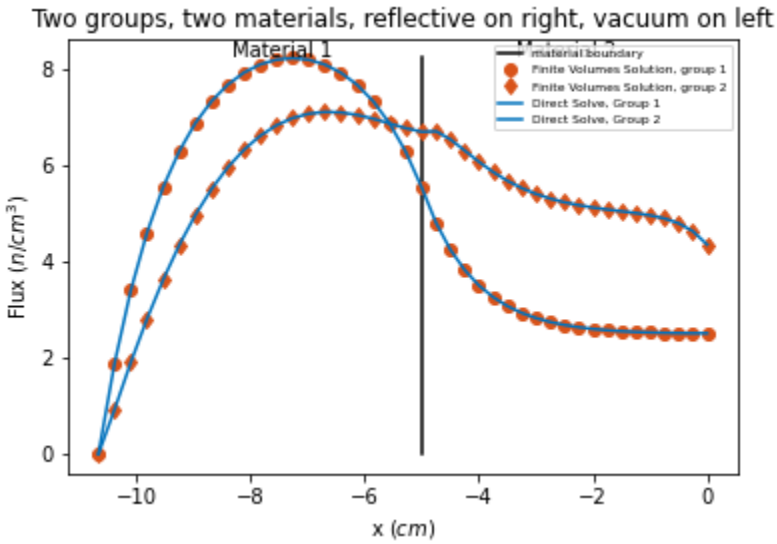

Figure 5. Scenario 4 solution

Test of Scenario 5

Inputs:

- **#----- Geometry -----**
- **n1** = 20
- **n2** = 20
- **w1** = 5
- **w2** = 5

- **#----- Sources -----**
- **S_m2_g1[12]** = 1
- **S_m1_g2[4]** = 1
- **#----- Material 1 -- Group 1 -----**
- **Sig_ti_m1_g1** = 1
- **Sig_ai_m1_g1_frac** = 0.001
- **Sig_s12_m1_frac** = 0.1
- **#----- Material 2 -- Group 1 -----**
- **Sig_ti_m2_g1** = 1
- **Sig_ai_m2_g1_frac** = 0.002
- **Sig_s12_m2_frac** = 0.1
- **#----- Material 1 -- Group 2 -----**
- **Sig_ti_m1_g2** = 1
- **Sig_ai_m1_g2_frac** = 0.2
- **#----- Material 2 -- Group 2 -----**
- **Sig_ti_m2_g2** = 1
- **Sig_ai_m2_g2_frac** = 0.1

Output:

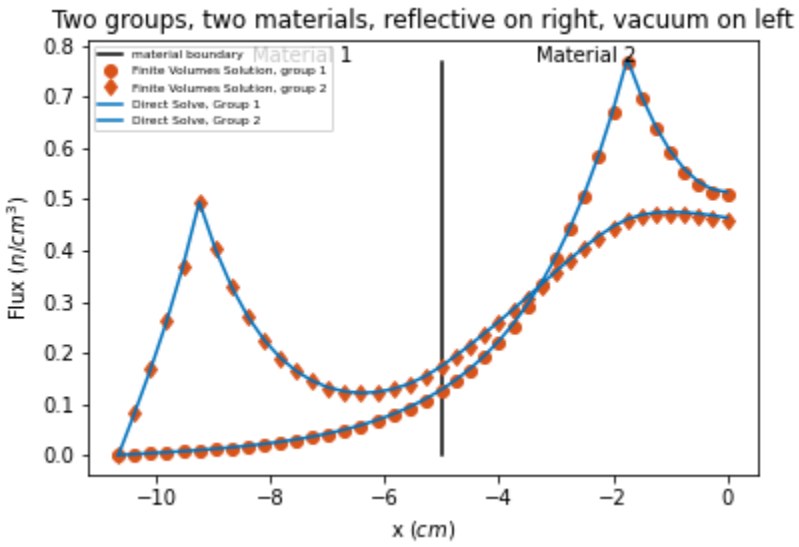

Figure 6. Scenario 5 solution

Conclusions

In this project, a code was developed to solve a two group, heterogeneous neutron diffusion equation. The finite volumes method was used and compared to an analytical solution as well as a direct solution of the matrices created from discretizing the diffusion equation. The algorithm from the finite volumes calculated solutions that agreed well with two other solutions methods. A known error was realized in the determination of the flux through the extrapolated distance at the vacuum boundary and is a topic that could be improved upon if this code were to be refined more.

Sources

Class Notes

Appendix A - Gauss Seidel

Below is my gaussSeidel function. In the Test section above, each plot has the result of the gaussSeidel function as well as a direct-solve of my matrices. In each case, the results of the gaussSeidel function match up with the direct-solve.

```
def gaussSeidel(guess,F,S,i,F_size):
    first_sum = 0
    for j in range(0,i):
        first_sum = first_sum + F[i][j]*guess[j]
    second_sum = 0
    for j in range(i+1,F_size):
        second_sum = second_sum + F[i][j]*guess[j]
    return (1/F[i][i])*(S[i]-first_sum-second_sum)
```