## NE 501 Computational Project

Due Date: April 19, 2016

Name: William Dawn

Instructor: Dr. Anistratov

Teaching Assistant: David Anderson

**Abstract**

This computational project set out to calculate the multiplication factor and fundamental mode for a general 1D geometry input by the user with associated cross sections. This was accomplished by iteratively solving the P1 equations. The code was tested for three Test A, Test B, and Test C of increasing complexity. Test A is sufficiently simple that an analytic solution exists. This allowed for the comparison of the iterative method and analytical method. Both the analytical method and iterative method returned the same answer to seven significant digits implying zero error (for this particular test). For Test A, = 1.147588. Test B introduced a heterogeneous geometry without an analytical solution. For Test B, = 1.006953. Test C introduced a relatively complex geometry and for Test C, = 1.087319. Most interesting about Test C was the spatial distribution of neutron flux because it reflected the distribution of moderator and fuel. Overall, these calculations are sufficiently accurate and the code is general for any 1D slab geometry. This assignment allowed for experience in creating a computational method to solve engineering calculations.

## Table of Contents

[**Introduction and Theory 1**](#_Toc448685408)

[**Procedure and Methods 6**](#_Toc448685409)

[**Results 8**](#_Toc448685410)

[**Discussion 21**](#_Toc448685411)

[**Conclusions 23**](#_Toc448685412)

[**References 24**](#_Toc448685413)

# Introduction and Theory

The underlying goal for this code is to calculate the fundamental eigenvalue (neutron multiplication factor) and its associated eigenfunction for a 1D slab using the multi-group diffusion (P1) equations. This is accomplished by using the finite-volume discretization method and the power iteration method in which the eigenfunctions are solved at every point in a spatial mesh and the calculation process is iterated until a solution converges. For this process, MATLAB is chosen to perform these calculations. However, none of these processes are dependent on MATLAB and can be performed in any computer language (such as FORTRAN). MATLAB is chosen mainly for its ability to simplify matrix math (used in below calculations) and its extremely simple debugging tools. At points, MATLAB syntax may be used but this none of the calculations below are specific to a particular language.

In the most broad sense, the underlying principle of these calculations is to make an initial “guess” for  and *k*, calculate the spatial distribution of flux and current at all spatial mesh points for all energy groups. The answer is of course wrong and then a new guess for the inputs are used, and the process begins again and repeats until the solutions converge. The discussion will begin with this iterative process. The iteration process is based on reference [2] “Notes on Numerical Methods for Solving k-Eigenvalue Problems for the Multigroup Neutron Diffusion Equations.” In the below discussion, *s* will be used to indicate the iteration index. Below,  will be used in calculations rather than  because it will allow the solution to converge more quickly but the same solution would be calculated if  were used.

The process to determine the geometry for the general case is worth mentioning. The user input describes the dimensions of each fuel pin type present and the material name present at each coordinate based on the user input cross section file. The user then supplies a “pin map” which describes where each type of fuel pin is located. Note, “fuel pin” is used but the pin is not required to contain fissionable material (e.g. guide tube). The code then builds an overall list of coordinates and the associated material at each coordinate. Then, for each point in the user specified spatial mesh, the code determines what material is present at that mesh point and populates a variable with the cross sections to be used at each spatial point. Once there is an array containing all cross sections for each point on the spatial mesh, calculations may begin.

To begin the calculation, it must be determined if the solution has converged. These convergence criteria are specified as



where  represents the “infinite norm” and  are the tolerance of the convergence criteria. It is possible to specify different criteria for both  and *k* but for these calculations, =1E-05 for both. Once it is determined that the convergence criteria are not met and the iteration begins, the neutron sources are to be calculated. These sources are sources from upscattering, downscattering, and fission. These sources are calculated inside of a loop, iterating over all energy groups *G*









where *i* is the index for the spatial mesh, and the required cross sections are the cross sections a the particular point in the spatial mesh

Once the overall source  is known, the P1 equations can be solved. This will be discussed after the iteration process. With the solution to the P1 equations , *J*, and  are known for the particular iteration. These newly calculated values are used to update the guess for *k* according to



It is at this point that any of these function can be normalized. For these calculations, normalization is performed according to



and this general formula can be used for any of the calculated functions. Equation is used to normalize  and . Finally, equation is used to calculate the error terms for the iteration and then these error terms are used at the beginning of the iteration. Then, the process continues until convergence.

The next part of the discussion will focus on the solution to the P1 equations. In the iteration loop, this occurs immediately after the calculation of the neutron sources *Q*. The theory for these calculations begins with the neutron transport equation, which is immediately simplified to the multigroup P1 equations





This can then be solved for given boundary conditions. In this problem, all boundary conditions are reflective such that



Equations and have been discretized and indexed by *s* so that the previously describe iterative solution process can be used. The P1 equations can be further simplified by using the definition of *Q* coming from equations – .





With the same boundary conditions specified in .

When these P1 equations have been discretized according to energy group and space as well, they can be easily simplified as a system of linear equations. This is how the P1 equations are solved within the code. The matrix equation is written simply as



*A* can be calculated before the iteration process begins because it only depends on geometry and materials/cross sections. *A* is composed of mostly zero-elements except for the terms specified below. *A* is a (3N+2)x(3N+2)xG 3D matrix.







The next terms specify reflective boundary conditions.





All other elements in the *A* matrix are zero.

The *f* “vector” is (3N+2)xG. *f* and *y* are vectors because they are solved individually for each group in which case they are a vector. *f* describes the neutron sources and it all zero elements except for the following.



*y* will also be a “vector” of dimension (3N+2)xG solved for each group. This vector is solved using matrix math according to



*y* contains elements from , *J*, and  in the form



Once equation is solved and *y* is known, this can be parsed to obtain the desired function.

Now the iteration process has been described and the methodology to solve the P1 equations has been determined. One other feature implemented into this code is programmatic “testing” or “bug-traps” used to catch errors with input or calculation methodologies. For example, input is checked to ensure that the summation of  for fissionable materials is equal to 1 within a tolerance. Also, the input is verified to ensure that the correct number of materials are specified for each pin. If *k* ever becomes negative, an error is thrown and the code exits. If the number of iterations increases past 100, the code also exits because experience suggests that even for complex problems (such as Test C) iterations should be much less than 100.

This theory is founded in the multigroup diffusion theory and as long as the mesh is sufficiently small, the results should be accurate. This iterative procedure does rely on convergence so it is impossible to say that this answer is exact. For simplified cases, the results from the iterative calculation can be compared to an analytical solution. This is true for solutions to the infinite homogenous diffusion equation. The production and loss terms of the diffusion equation can be simplified as matrices and then written as



Which allows for a solution of the form



Where the fundamental mode is the largest eigenvalue of matrix *A* and the fundamental mode of  is the associated eigenvector of *A*. This allows for an analysis of error in an especially simple case.

This theory for calculating the spatial distribution of neutrons can also be verified qualitatively by observing the spatial distribution and comparing this with the expected distribution based on different materials present. The algorithm described as sufficiently rigorous and sufficiently accurate for engineering calculations. In addition to its accuracy, the methodology described will also allow for the observation of macroscopic effects of different materials and different geometries. Though this problem has been reduced to 1D for simplification, this methodology will calculate accurate results and remains sufficiently general to understand neutron effects and interactions.

# Procedure and Methods

The overall procedure to calculate the fundamental eigenvalue and eigenfunction is rather straightforward. Since this process requires iteration and a solution over a large number of spatial regions it is greatly simplified by performing the calculation with the use of a computer algorithm. The general form of this algorithm is specified in reference [2] “Notes on Numerical Methods for Solving k-Eigenvalue Problems for the Multigroup Neutron Diffusion Equation.” The beginning of the calculation is the geometric processing. The code takes input of the types of fuel-pin present, the dimensions of the pins, the material in each pin, and the location of the different pins. This “pin-map” containing the location of the pins is used to generate an overall description of the materials across the length of the problem. The spatial mesh is then defined. For each point in the spatial mesh, the code then determines what material is present and populates that region with the associated cross sections. This results in a cross section defined at each point for the spatial mesh.

Next, the A matrix is defined. This matrix will allow for the solution of the P1 equations. Since this matrix is dependent on only materials properties, it can be calculated before the iterative process begins. The form of matrix A is given by equations – . The matrix is set to all zero values and then populated. This matrix also contains information regarding the boundary conditions of the problem.

At this point, all setup is completed and the iteration process can begin. This iterative process will calculate a new eigenvalue and associated eigenfunction for each iteration. These eigenvalues and eigenfunctions will converge to the true solution to the problem given sufficient iteration. It is therefore important to specify an initial condition so that this can be used in the first round of iteration. The algorithm implemented is sufficiently rigorous so any positive (and non-zero) initial guess for  and *k* will eventually converge to the correct value.

The iteration begins by checking if the convergence criteria are satisfied. These are specified when the fractional change in both  and *k* from one generation to the next is less than a specified tolerance (for these calculations, 1E-05). Next, the neutron sources are calculated. These sources include fission, upscattering, and downscattering. The fission source and upscattering depend on the previous iteration’s  and the downscattering depends on the current . These sources are then combined and used to calculate the f vector which is zeros everywhere except where specified by equation .

Using the generated A matrix and f vector, the solution vector, y, is calculated using matrix math according to equation . This solution vector must then be parsed in order to determine the individual values for , , and *J*. These values are then used to update the value iteration’s “guess” for *k* and then the process begins again with checking for convergence criteria.

Once the convergence criteria are satisfied, the code will exit and the resulting fundamental solution is known. The solution for , , *J*, and *k* are the values from the last iteration because this is the converged solution. This data can then be plotted or used in subsequent calculations as necessary.

# Results

By testing the code across a variety of geometries, it was demonstrated that the code is both general and robust. General because the code can be used to calculate for any given geometry given proper input and robust because the solution always converged and converged quickly. The three geometries used covered increasingly complex calculations and produced results that make qualitative sense. The first geometry used has an analytic solution so it was possible to compare the iterative (estimation) to the correct analytical solution. The process of these three test showed that the iterative process implemented produces accurate and intuitive results.

To demonstrate this code, three tests were performed: Test A, Test B, and Test C with increasingly complex geometries. Test A was an single material with reflective boundary conditions which is the same as an infinite and homogenous geometry. Using the iterative code, = 1.147588 and the analytical solution yielded 1.147588 which represents zero error to seven significant digits. The iterative solution converged in 4 power iterations. This shows that the methodology is accurate and precise for simple cases for which analytical solutions exist. The comparison between analytical and iterative solutions is also performed by comparing . Table 1 shows the analytical solution to .

Table 1. Analytical solution to .

|  |  |
| --- | --- |
| Group |  |
| 1 | 7.174683E-02 |
| 2 | 8.896544E-01 |
| 3 | 3.790249E-02 |
| 4 | 6.790403E-04 |
| 5 | 1.700471E-05 |
| 6 | 2.120942E-07 |
| 7 | 3.738284E-09 |

 does not change spatially since the reactor is infinite and homogenous. This is also shown by the spatial solution in the iterative solution. The iterative solution is presented in Table 2.

Table 2. Iterative solution to .

|  |  |  |
| --- | --- | --- |
| Group | x=0cm | x=0.54cm |
| 1 | 7.174683E-02 | 7.174683E-02 |
| 2 | 8.896544E-01 | 8.896544E-01 |
| 3 | 3.790249E-02 | 3.790249E-02 |
| 4 | 6.790408E-04 | 6.790408E-04 |
| 5 | 1.700486E-05 | 1.700486E-05 |
| 6 | 2.120989E-07 | 2.120989E-07 |
| 7 | 3.738368E-09 | 3.738368E-09 |

Table 2 and Table 1 do disagree slightly. The maximum difference is 0.00225% which is even better than acceptable for any engineering calculation. The main purpose for Test A was to provide a simple test for the code for which a known solution exists to ensure that the methodology was sound. Test A successfully proved that the methodology was accurate and that the iterative solution will converge to the analytical solution. To better visualize the neutron energy spectrum for Test A,  is plotted at x=0.54cm as a function of group. This spectrum is shown in Figure 1.

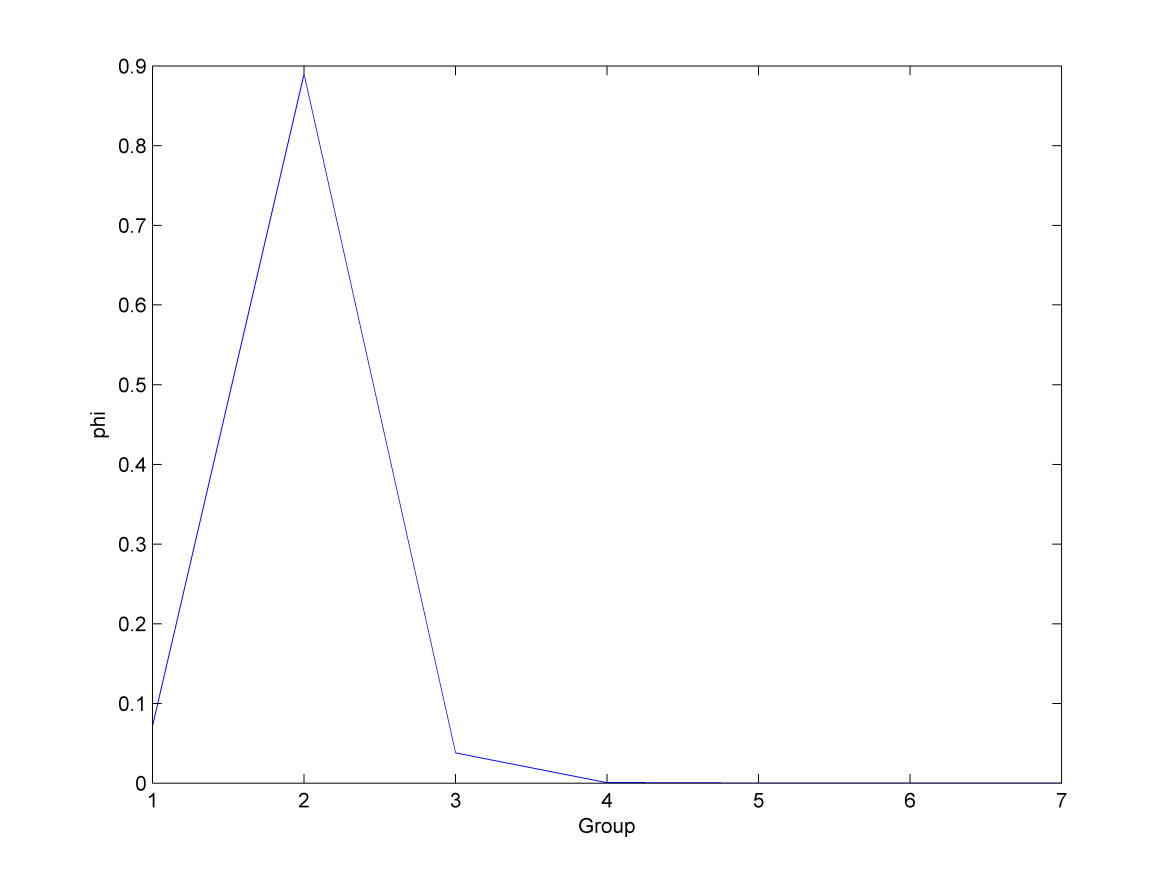


Figure 1. Neutron energy spectrum for Test A.

Figure 1 shows that many of the neutrons remain in the higher energy groups, especially group 2. This makes sense because in this problem, there is no moderator present to allow for the thermalization of neutrons.

Once Test A is completed to show that the code is accurate and calculating correct eigenvalues, Test B is performed. Test B includes a non-homogenous geometry which will result in a spatially variable flux. It also tests the code’s ability to handle differing (heterogeneous) cross sections. Since Test B is more complex, it does not have an analytical solution. The solution for Test B converged in 5 iterations to a = 1.006953. In Table 3, is tabulated at different positions in the test for the seven groups to again show the neutron energy spectrum.

Table 3.  at various positions.

|  |  |  |  |
| --- | --- | --- | --- |
| **Group** | **x=0.045cm** | **x=0.09cm** | **x=0.63cm** |
| 1 | 1.566735E-01 | 1.566822E-01 | 1.569403E-01 |
| 2 | 6.164838E-01 | 6.165625E-01 | 6.183212E-01 |
| 3 | 1.815390E-01 | 1.814749E-01 | 1.799888E-01 |
| 4 | 3.372542E-02 | 3.368190E-02 | 3.248145E-02 |
| 5 | 9.950350E-03 | 9.925545E-03 | 9.440337E-03 |
| 6 | 2.153910E-03 | 2.128663E-03 | 1.629803E-03 |
| 7 | 7.052522E-04 | 6.879568E-04 | 4.949522E-04 |

Table 3 can be difficult to interpret so the data is plotted in Figure 2.

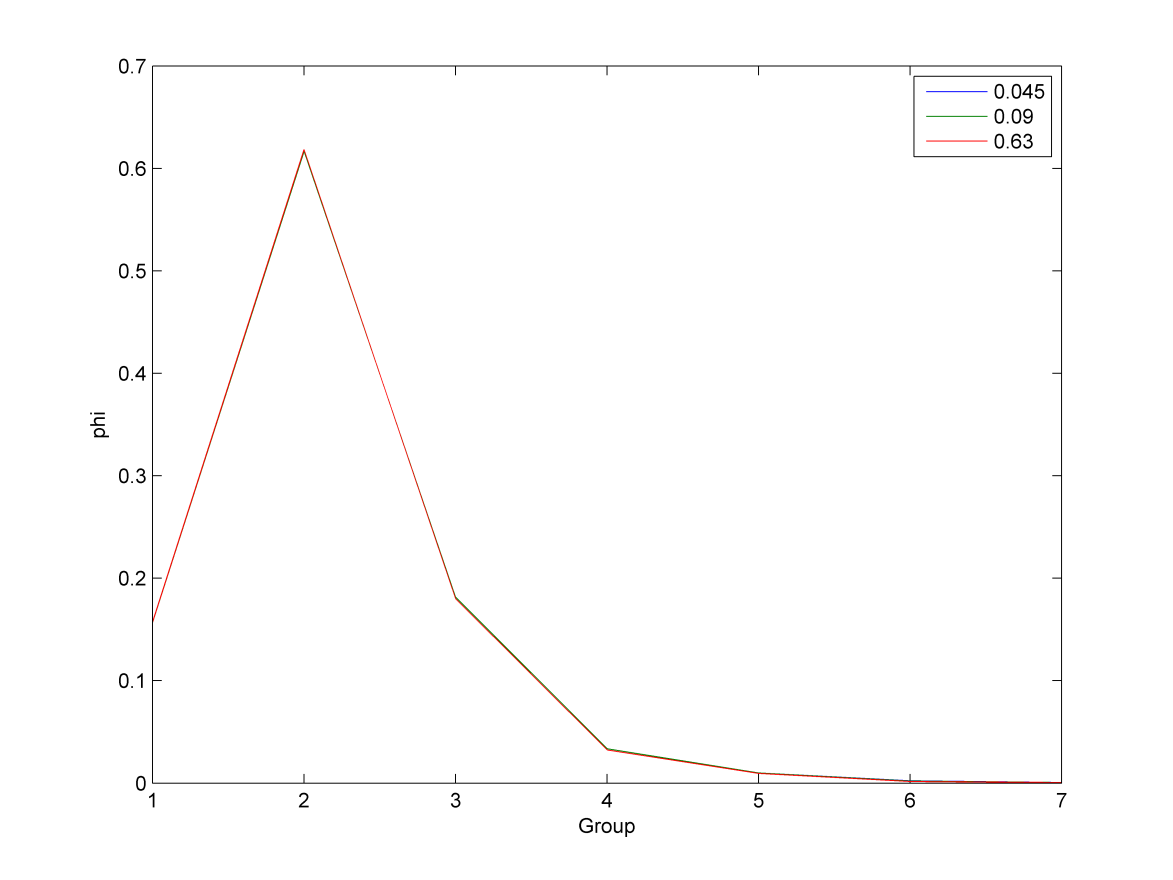


Figure 2. Neutron energy spectrum for Test B.

Figure 2 shows a similar shape to Figure 1. This is logical because similar materials are present but Test B includes a small amount of moderator which accounts for the minimal thermalization observed in Figure 2. Similar tabulation and plotting is done for  for Test B in Table 4.

Table 4.  at various positions.

|  |  |  |  |
| --- | --- | --- | --- |
| **Group** | **x=0.045cm** | **x=0.09cm** | **x=0.63cm** |
| 1 | -2.385245E-18 | -8.088688E-04 | -1.277576E-04 |
| 2 | -1.040834E-17 | -2.824565E-03 | -4.660102E-04 |
| 3 | 1.301043E-17 | 1.607956E-03 | 2.674027E-04 |
| 4 | 5.529431E-18 | 1.103234E-03 | 1.817633E-04 |
| 5 | 9.757820E-19 | 5.118168E-04 | 8.403519E-05 |
| 6 | -2.663072E-18 | 2.981603E-04 | 4.553749E-05 |
| 7 | -2.625802E-20 | 9.667574E-05 | 1.463993E-05 |

Table 4 is then plotted to visualize more easily in Figure 3.

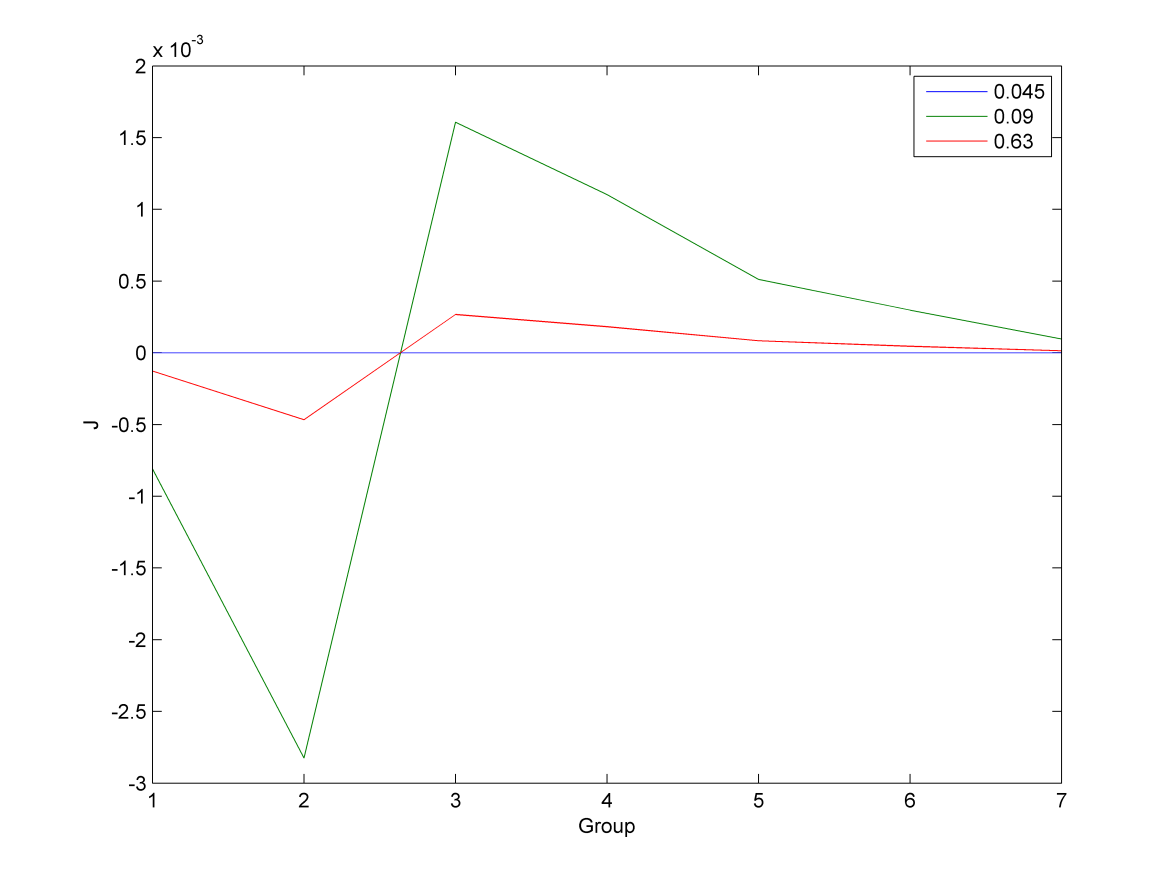


Figure 3.  as a function of energy group for Test B.

Figure 3 shows that the net current at the center of the moderator (x=0.045cm) is zero meaning neutrons are moving equally in each direction. The current obviously changes as a function of position and the distribution is different at different coordinates.

The neutron spatial distribution is then summarized for Test B. The cell averaged flux is plotted at the center of each cell for seven energy groups in Figure 4.

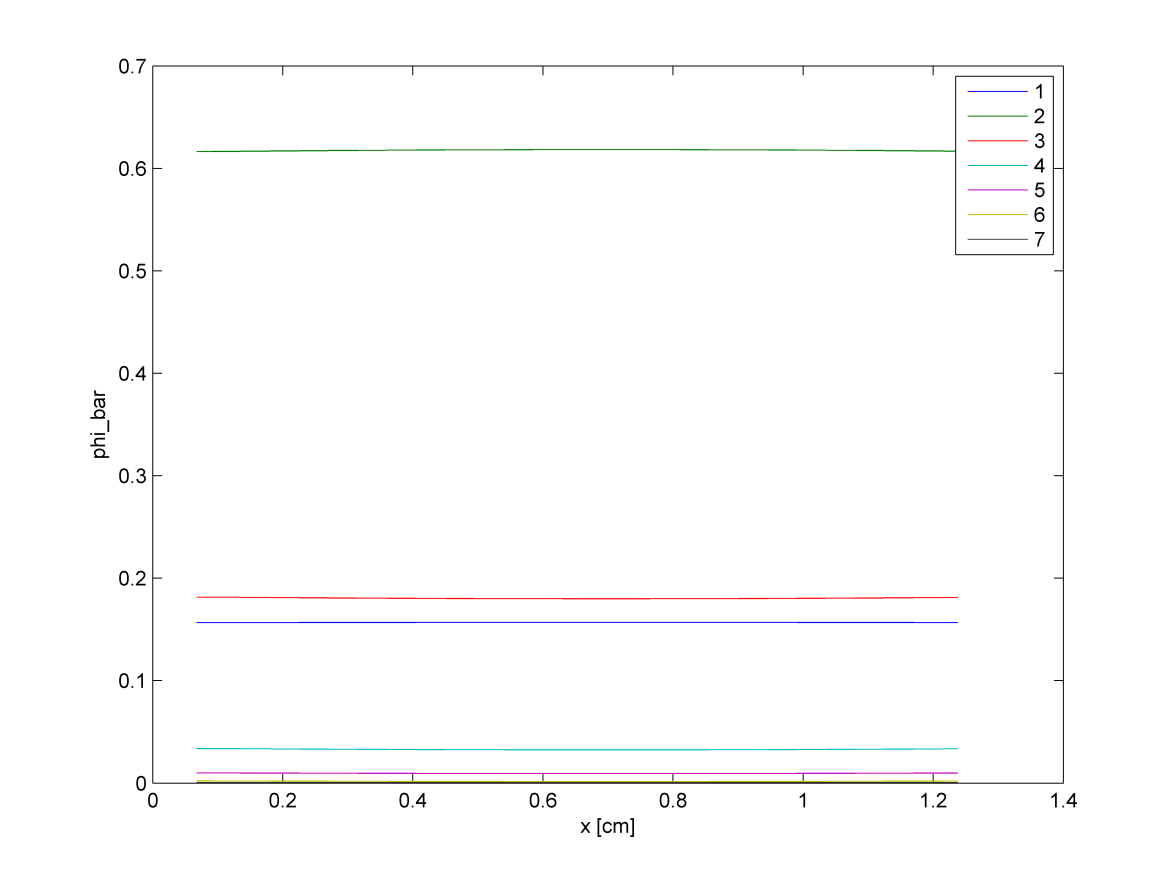


Figure 4.  for energy groups as a function of position.

Figure 4 gives the impression that flux is constant spatially. However, this is a result of the scale of the plot. The overall  is best summarized by the sum across all groups. This sum of  is shown in Figure 5.

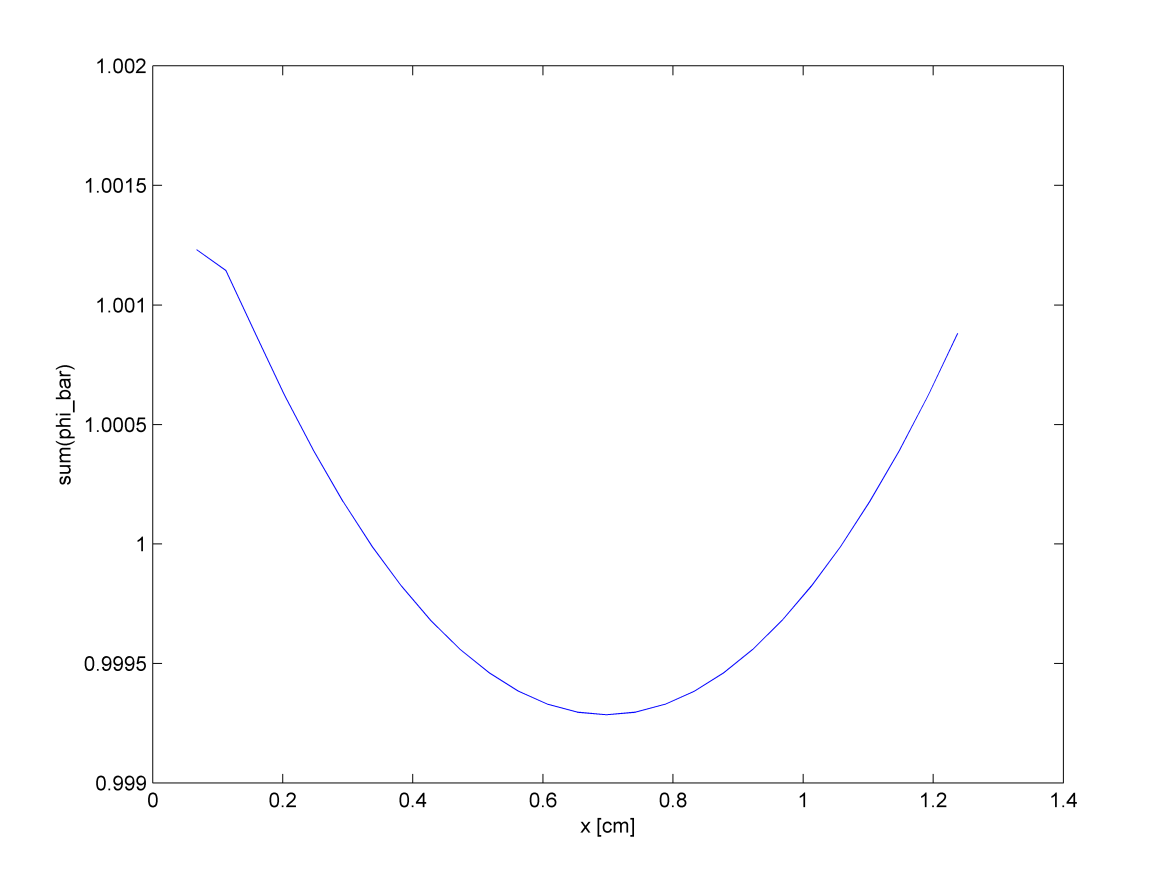


Figure 5.  as a function of positon.

Figure 5 shows the expected neutron distribution shape for Test B. Flux decreases in the center of the fuel element and increases on the edges of the pin where there is moderator to thermalize neutrons. Current is also plotted as a function of position to show how net leakage change as a function of position. This is presented in Figure 6.

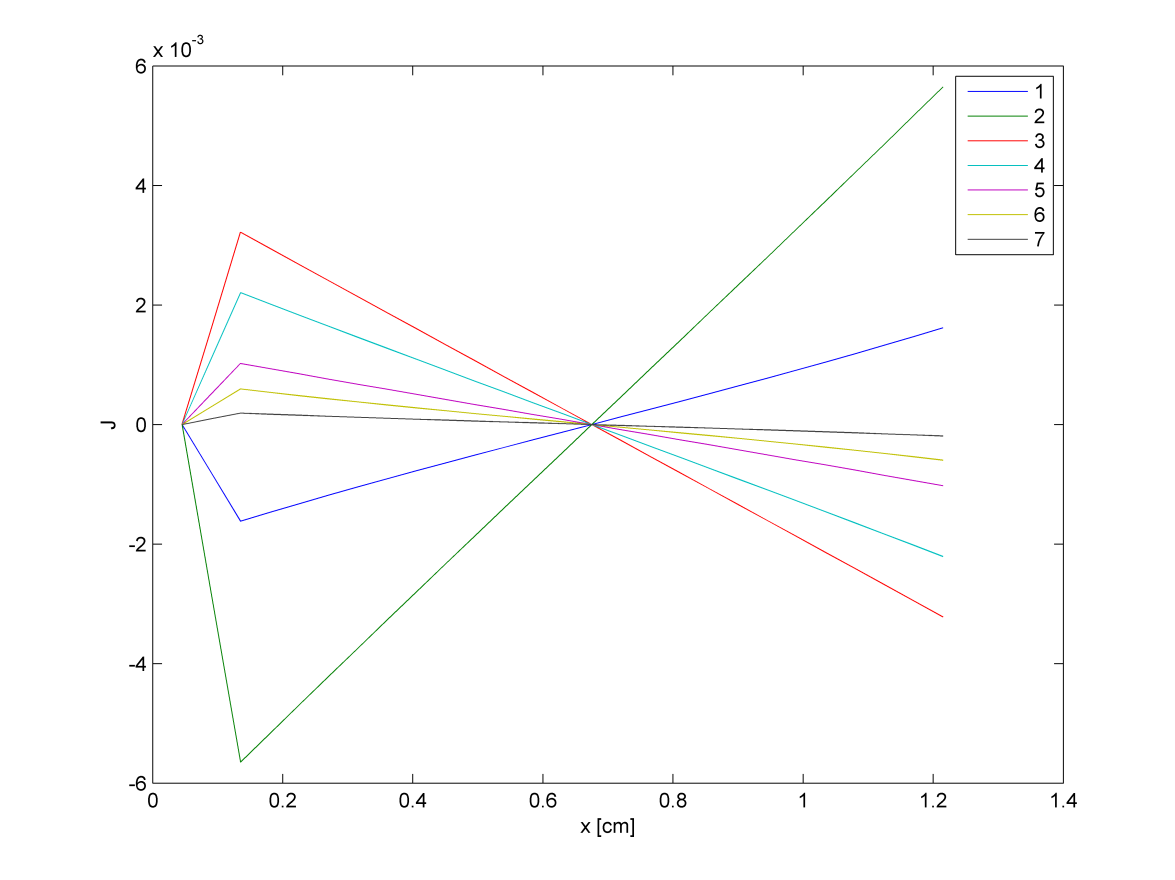


Figure 6. Net leakage as a function of position.

Figure 6 is consistent with Figure 5 because, according to Fick’s Law, the current should be proportional to the derivative of the flux. Figure 6 shows a net leakage of zero in the center of the fuel region which is expected. This result indicates that all neutrons are leaking from the fuel into the moderator equally in both left and right directions.

These results properly summarize Test B. Test C can now be presented. Test C was a significantly more complex geometry with five different pins present and two assemblies, which contained 17 pins each. However, the code was designed to be robust and, as such, solved the problem in a similar matter. By introducing a more complex geometry, interesting neutron interactions can be observed. Test C converged to a = 1.087319 in 37 iterations. This is significantly more iterations than the other cases but is expected for a more intricate geometry with 714 spatial regions compared to 28 and 10 in Tests B and A respectively. To measure the neutron energy spectrum  is again measured at various positions as a function of energy group. This data for  is tabulated in Table 5.

Table 5.  at various positions.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Group** | **x=9.45cm** | **x=21.42cm** | **x=30.87cm** | **x=42.84cm** | **x=53.55cm** |
| 1 | 3.447640E-01 | 2.554205E-01 | 1.777167E-01 | 6.982264E-02 | 5.727333E-03 |
| 2 | 9.521590E-01 | 7.481293E-01 | 4.914375E-01 | 1.723676E-01 | 8.602984E-03 |
| 3 | 3.908567E-01 | 2.816346E-01 | 1.911913E-01 | 7.668070E-02 | 5.458724E-03 |
| 4 | 1.218971E-01 | 7.518243E-02 | 5.327070E-02 | 2.703725E-02 | 2.715955E-03 |
| 5 | 9.114329E-02 | 4.791689E-02 | 2.467883E-02 | 1.664213E-02 | 2.290910E-03 |
| 6 | 7.662726E-02 | 2.745439E-02 | 7.456900E-03 | 1.978072E-02 | 1.400967E-02 |
| 7 | 6.209029E-02 | 1.615300E-02 | 4.623816E-03 | 3.225539E-02 | 4.701293E-02 |

Table 5 shows that energy distribution certainly changes as a function of position. This data is better visualized in Figure 7.

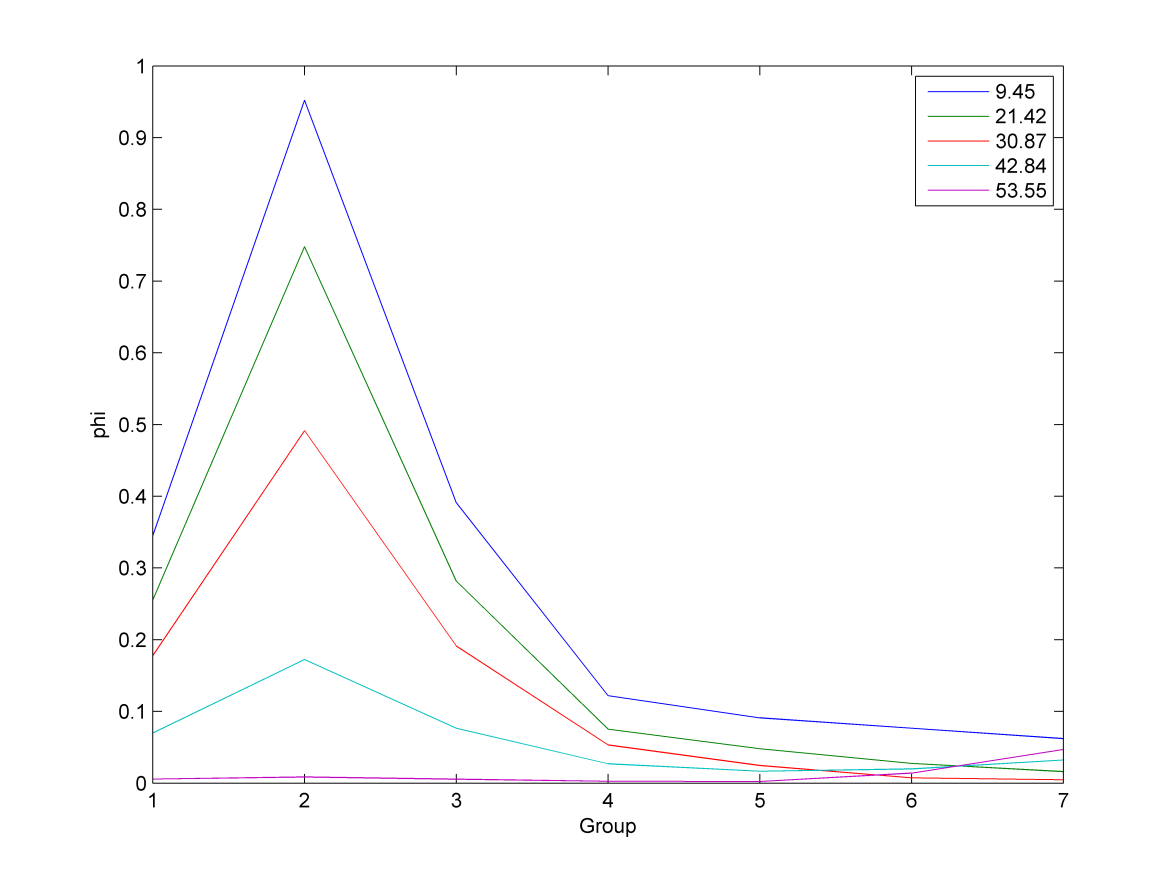


Figure 7. Neutron energy spectrum for Test C.

Figure 7 shows a neutron energy spectrum similar to Test A and Test B. One interesting point in Figure 7 is the bottom line (x=53.55cm) has a relatively large amount of thermal neutrons compared to fast neutrons. Compared to the other coordinates, this coordinate is in a large block of moderator allowing for good thermalization. The energy spectrum is also represented according to  as was done for Test B. The energy spectrum for  is presented in Table 6.

Table 6.  at various positions.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Group** | **x=9.45cm** | **x=21.42cm** | **x=30.87cm** | **x=42.84cm** | **x=53.55cm** |
| 1 | 1.462302E-02 | 1.594080E-02 | 1.939076E-02 | 3.426545E-02 | 2.775341E-03 |
| 2 | 2.204292E-02 | 2.670642E-02 | 2.883292E-02 | 4.358566E-02 | 1.757303E-03 |
| 3 | 1.936095E-03 | 5.285112E-03 | 5.977746E-03 | 5.256948E-03 | 7.923336E-04 |
| 4 | -1.510439E-03 | 5.539275E-04 | 3.415069E-04 | -1.321376E-03 | 3.999459E-04 |
| 5 | 1.108896E-05 | 5.863667E-03 | -2.586018E-04 | -2.166080E-03 | 2.739522E-04 |
| 6 | -1.432325E-03 | 4.517427E-03 | -7.370250E-04 | -7.560009E-03 | 8.577855E-04 |
| 7 | -3.833421E-03 | 1.034333E-03 | -7.145035E-04 | -1.284801E-02 | 1.343109E-03 |

Table 6 clearly shows the large variation in net leakage at different positions with some positive and some negative values. This data in Table 6 is summarized in Figure 8.

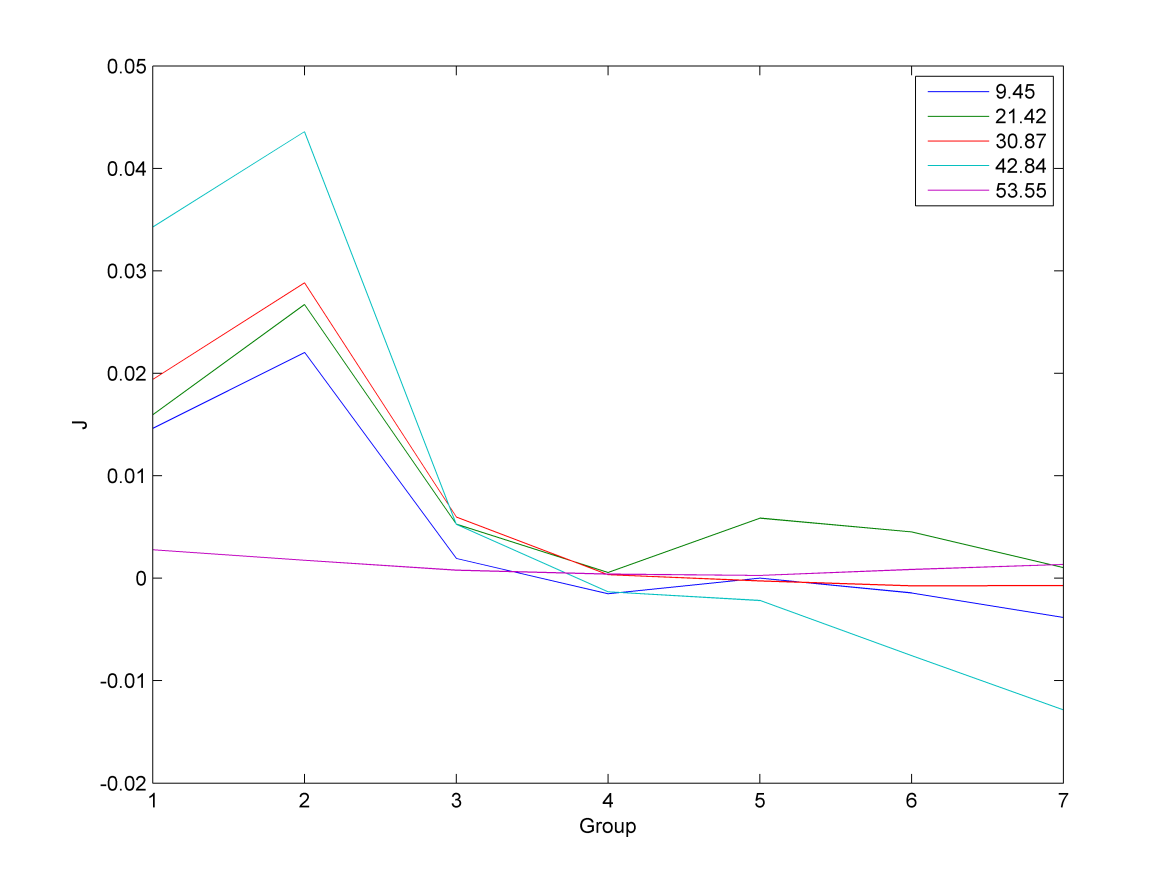


Figure 8.  as a function of energy group for Test C.

As in Figure 7, x=53.55cm is an anomaly because all other lines plotted are in a region with a significant amount of fuel. Figure 7 and Figure 8 agree that in regions with a large amount of fuel, there are many fast neutrons (due to their production from fission). In regions with a large amount of moderator, there are significantly fewer fast neutrons and more thermal neutrons.

Due to the complex geometry of Test C, the spatial distribution of neutrons is especially interesting. First, the cell averaged scalar flux is plotted at the cell centers for the seven different neutron energy groups. This is displayed in Figure 9.

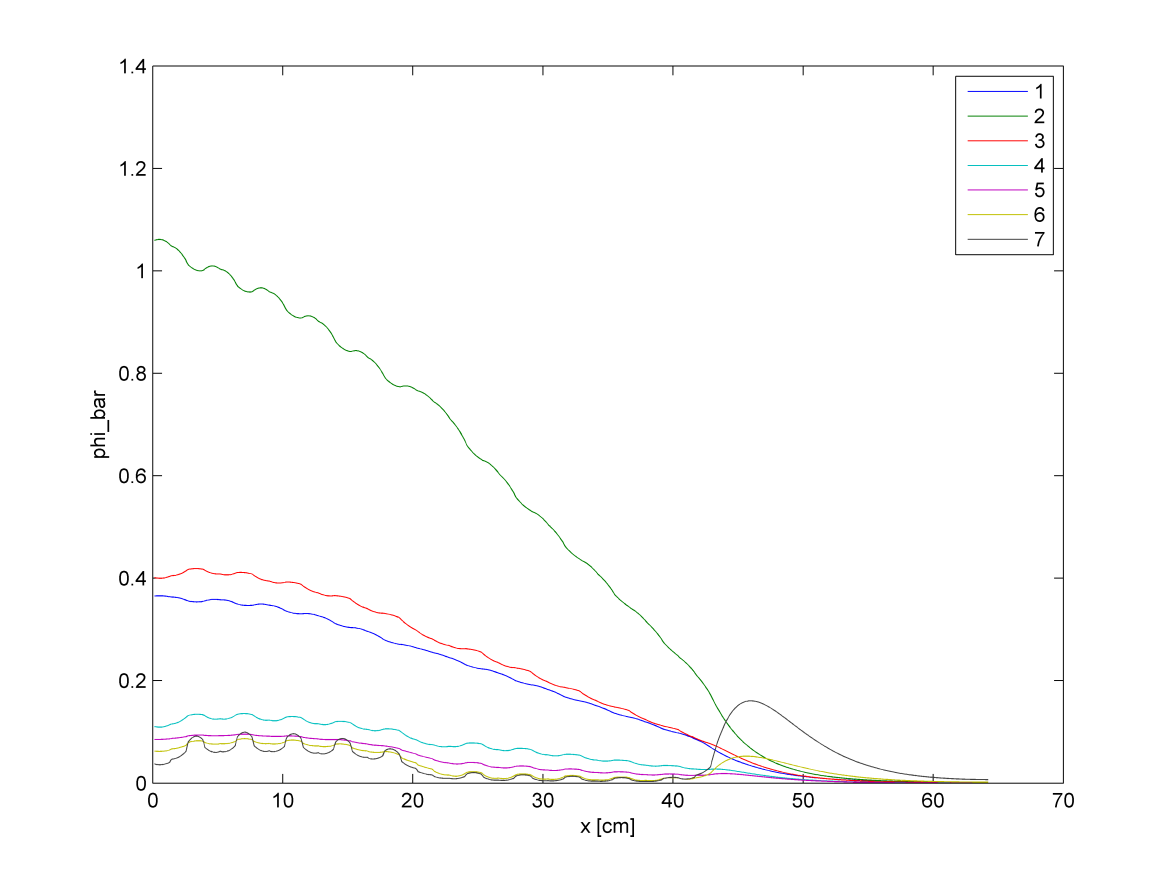


Figure 9.  for energy groups as a function of position.

Figure 9 shows a number of phenomena, first to be addressed are the spikes in the region around 0 – 20cm which is a region with UO2 and guide tubes. The lower peaks occur in guide tubes where moderator is present. That is, thermal flux peaks outside of the fuel, in regions (like guide tubes) with high moderator concentration. The faster fluxes peak inside of the fuel where the fuel is present. The relative decrease in fluxes in the MOX region around 20 – 40cm shows the relatively poor neutron production in MOX compared to UO2 for this particular geometry. A final notable feature is the thermal flux peak around 45cm. This is a “water-wing” where a large region of moderator from 40 – 60cm allows for the thermalization of fast neutrons. Similar trends are observed when all of the energy groups are summed. This summation across energy groups is shown in Figure 10.

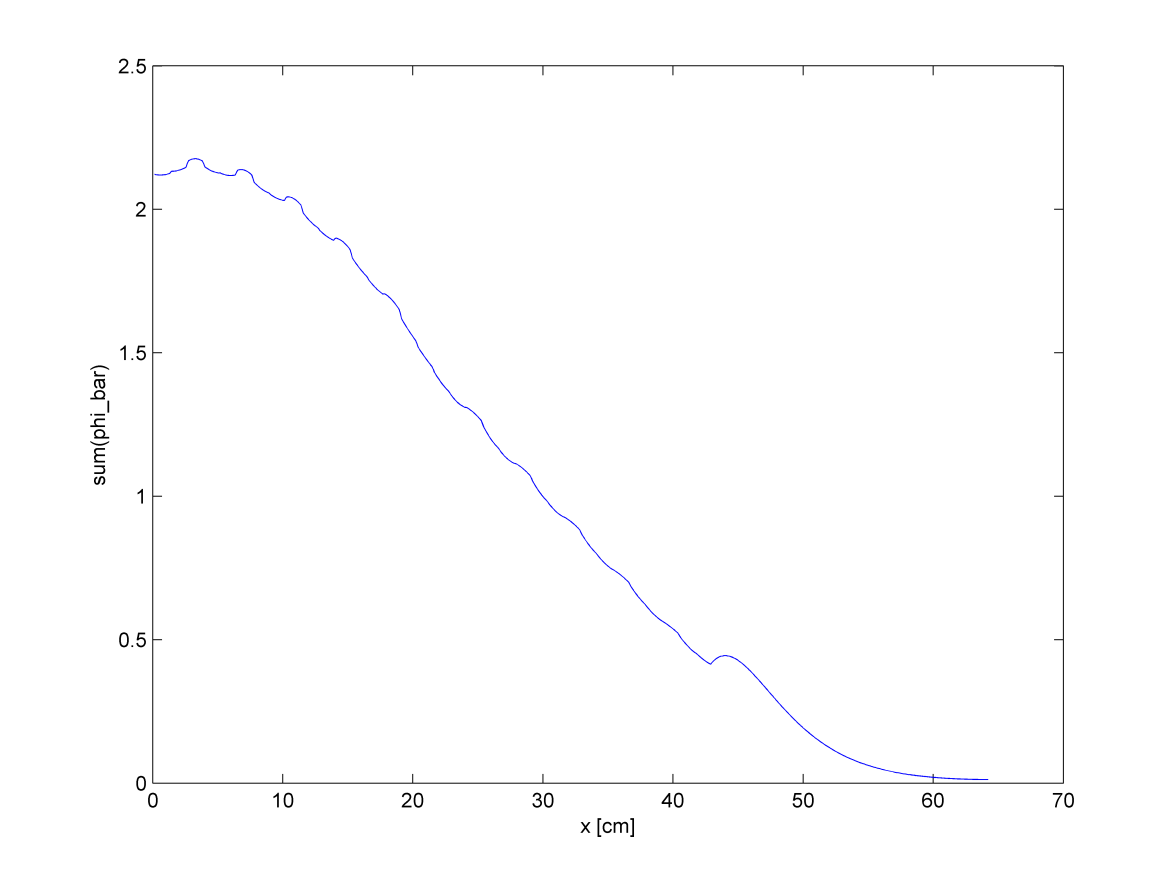


Figure 10.  as a function of positon.

The combined average scalar flux in Figure 10 shows the same general trends as Figure 9. Notable are the peaks in the guide tube and the fuel elements and the reflector peak due to the moderator. The net leakage (current) is next plotted as a function of position in Figure 11.

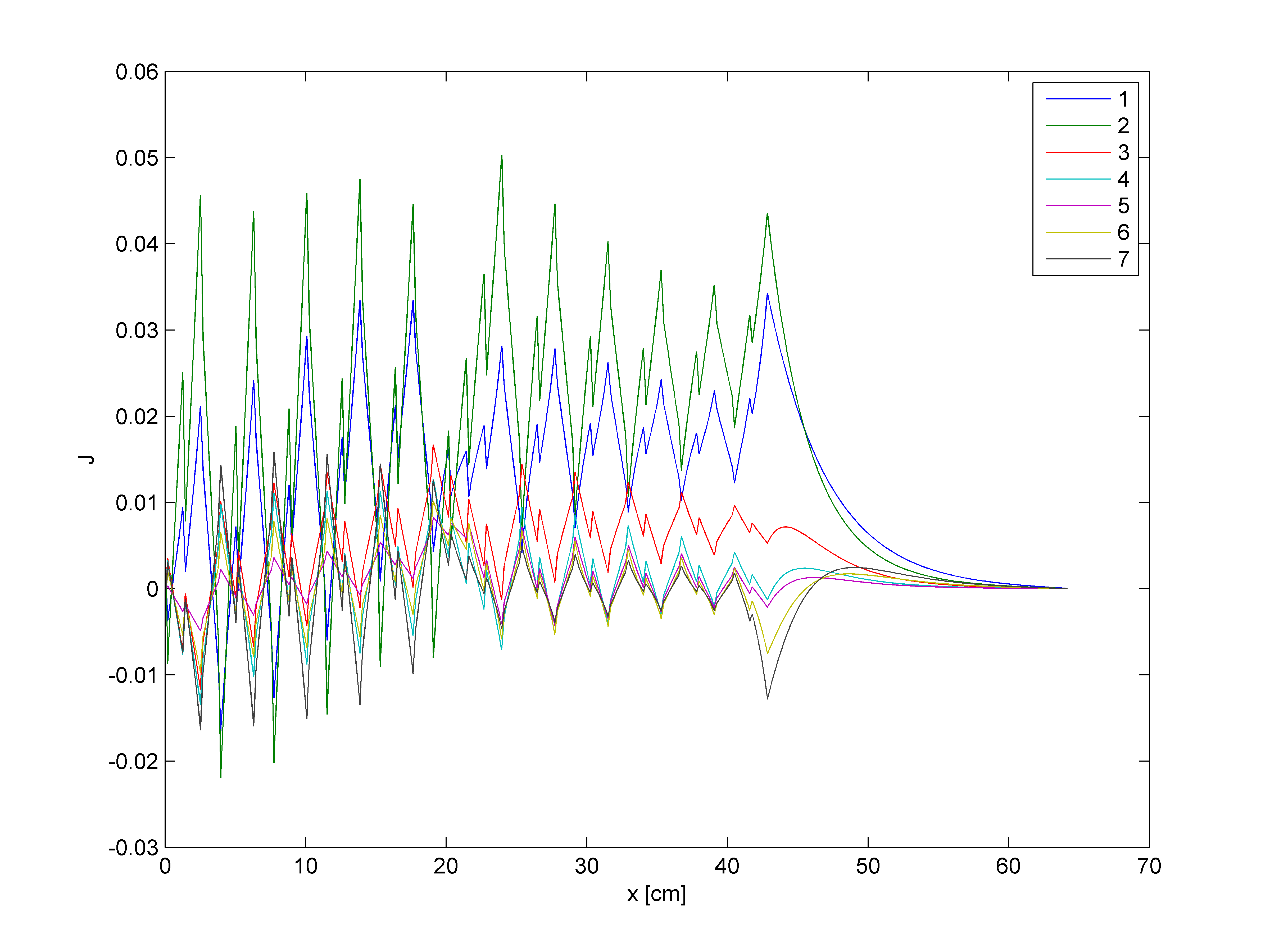


Figure 11. Net leakage as a function of position.

Figure 11 shows the interesting pattern of net leakage as in relationship to position. The leakages for fast and thermal fluxes differ significantly across the length of the reactor. Leakage rates peak in the fuel elements because they are moving in a particular direction as they diffuse. Specifically, the neutrons move towards the nearest moderator or guide tube. In the moderator and guide tubes, the neutrons are moving approximately equally in both directions as they move from moderator to fuel to continue to fission chain reaction. In the large moderator slab, leakage approaches zero because neutrons are moving equally in both directions due to the reflective boundary at the end of the slab.

Test C may seem the most interesting but Test B was an important stepping stone for a single fuel pin and Test A ensured that the results produced by the iterative method were accurate. By using this testing methodology, it has been demonstrated that the code developed produces accurate an intuitive results for any 1D slab geometry given proper input and cross section data.

# Discussion

Tests A, B, and C have demonstrated the generality with which this code was written. Given any general 1D slab geometry and appropriate cross sections, this code is capable of solving the P1 equations. It may take more time or more iterations for more complex geometries but these tests have thoroughly demonstrated the capability of this code to produce accurate results for the neutron multiplication factor and fundamental mode for scalar neutron flux.

For Test B and Test C it is necessary to use an iterative method rather than an iterative method to solve for the fundamental mode of the solution rather than an analytical solution because an analytical solution does not exist. The nature of using an iterative method requires a convergence criteria which means that the solution is not exact, by definition. The convergence tolerance can be adjusted to provide a more or less accurate. However, at a certain point the result also depends on the precision of the computer being used. No matter the criteria, there is still error in the result. The study of error quantification is not discussed in this report because it is relatively complex. All of these points considered, the comparison between the analytical solution and the iterative solution for Test A showed no error out to seven significant digits (error of less than 0.01pcm). This leads to the conclusion that though error is not numerically quantified, the iterative method is sufficiently accurate for any engineering calculation.

Also, this problem is solved numerically so it is necessary to discretize both space (into a spatial mesh) and energy (into energy groups). Though this problem was not solved continuously, the plots generated use sufficiently small step sizes to give the appearance of a continuous function. These plots allow for an understanding of the relationship between flux, space, and energy though these dimensions are discretized. However, because these values are discretized it is important to ensure that the proper cross sections are used to ensure accurate results. This code is undoubtedly sensitive to incorrect cross sections. This is one of the reasons why a significant amount of time and resources are dedicated to measuring accurate cross sections for use in engineering calculations.

This undertaking showed the importance of “clean” and efficient code that was easy to read, understand, and debug. Without fail, a code will have bugs after its first execution so if the code is written efficiently, it should be simple to determine where the error occurred and remedy the problem. Writing code in a uniform manner also allows others to compare codes and understand the methodology implemented. Though this is a major coding undertaking, by breaking the code into discrete functions or sections it becomes more intelligible and also makes it easier to identify the source of errors.

Overall, the results support the assertion that the code would remain general for any 1D slab geometry and could produce accurate and precise results for the multiplication factor and fundamental mode. This project shed light on the intricacies of coding and hinted at the importance of proper cross sections. One of the remaining unknowns from this experiment is the quantification of error. It is known that the error is sufficiently small due to the comparison performed in Test A but based on these calculations, it is not possible to assign a standard deviation or describe the probability distribution based on the results of this code.

# Conclusions

The goal of this experiment to produce a code to iteratively calculate the multiplication factor and fundamental mode for a general 1D slab was successful. Unlike a more traditional laboratory experiment, this coding project required a different set of problem solving skills. These included writing code efficiently, debugging code, and translating a calculation procedure into a series of machine instructions. The iterative calculation of the P1 equations required an understanding of the equations themselves as well as the matrix math operations required in the actual calculations.

This code showed the importance of a simple neutron diffusion code. It is generally possible to estimate the neutron distribution for a general geometry but complex geometries make it more difficult. Even for experienced engineers, it is possible to overlook a particular neutron interaction. Therefore, a code that can simply and accurately generate a spatial neutron distribution ensures that the neutron distribution across space and energy is properly understood and can be accounted for in design processes.

The nature of this computational project differs from the typical experiments and assignments for students. The creation of this code forced students to become familiar with scripting and/or coding languages and understand how they can be used for calculations. One of the biggest impacts of this exercise was a better understanding of the debugging process. The ability to quickly and easily repair buggy code allows for the creation of more complex computational algorithms in the future. However, the ability to debug engineering calculations comes with experience. This exercise did produce a method for the accurate calculation of the distribution of neutrons in a 1D slab but also required understanding of the underlying physical principles and forced experience with engineering computational caclulations.

# References

1. E. E. Lewis, M. A. Smith, N. Tsoulfanidis, G. Palmiotti, T. A. Taiwo & R. N. Blomquist. “Benchmark specification for Deterministic 2-D/3-D MOX fuel assembly transport calculations without spatial homogenization (C5G7),” Expert Group on 3-D Radiation Transport Benchmarks, NEA/NSC/DOC(2001)4
2. D. Anistratov, “Notes on Numerical Methods for Solving k-Eigenvalue Problems for the Multigroup Neutron Diffusion Equations,” 2016.
3. D. Anistratov, “Compuational Project,” 2016.
4. D. Anistratov, “On Solving k-eigenvalue Problem for the Multigroup Diffusion Equations in Infinite Homogeneous Medium,” 2016.