Homework 1

Simon Bolding
NUEN 629

September 17, 2015

NUEN 629, Homework 1

Due Date Sept. 17

1 Ayzman

(10 points) Henyey and Greenstein (1941) introduced a function which, by the variation of one parameter, $-1 \le h \le 1$, ranges from backscattering through isotropic scattering to forward scattering. In our notation we can write this as

$$K(\mu_0, v' \to v) = \frac{1}{2}\frac{1-h^2}{(1+h^2-2h\mu_0)^{3/2}}\delta(v'-v).$$

Verify that this is a properly normalized $f(\mu_0)$ and compute $K_l(v' \to v)$ for $l = 0,1,2$ as a function of h.

2 Bolding

(20 points) In an elastic scatter between a neutron and a nucleus, the scattering angle in the center of mass system is related to the energy change as

$$\frac{E}{E'} = \frac{1}{2}\left((1+\alpha)+(1-\alpha)\cos\theta_{\mathrm{c}}\right),$$

where E is the energy after scattering and E' is the initial energy of the neutron and

$$\alpha = \frac{(A-1)^2}{(A+1)^2}.$$

The scattered angle in the center-of-mass system is related the lab-frame scattered angle as

$$\tan\theta_{\mathrm{L}} = \frac{\sin\theta_{\mathrm{c}}}{A^{-1}+\cos\theta_{\mathrm{c}}}.$$

Also, the distribution of scattered energy is given by

$$P(E' \to E) = \begin{cases} \frac{1}{(1-\alpha)E'}, & \alpha E' \le E \le E' \\ 0 & \text{otherwise} \end{cases}.$$

Derive an expression for $K(\mu_0, E' \to E)$, where μ_0 is $\cos\theta_{\mathrm{L}}$. What is the distribution in angle of neutrons of energy in the range $[0.05\,\mathrm{MeV}, 10\,\mathrm{MeV}]$ to energies below 1 eV if the scatter is hydrogen?

3

(70 points) Consider an infinite square lattice of infinitely tall cylindrical UO2 fuel pins in water. A quarter of a pin cell looks for a square lattice is shown in Fig. 1 and an infinite hex lattice in Fig. 2. The cross-section data for each is given in Table 1. The neutron transport equation for this problem is given simply by

$$\hat{\Omega}\cdot\nabla\psi(x,y,\hat{\Omega}) + \Sigma_{\mathrm{t}}\psi(x,y,\hat{\Omega}) = \frac{1}{4\pi}\Sigma_{\mathrm{s}}\phi(x,y) + \frac{Q}{4\pi}.$$

You may choose whichever lattice you wish – square or hex. For one or the other, perform the following:

Table 1: Data for Test Problem

	Fuel	Moderator
Σ_{t} (cm^{-1})	0.1414	0.08
Σ_{s} (cm^{-1})	0	0
Q (n/cm^3·s)	1	0

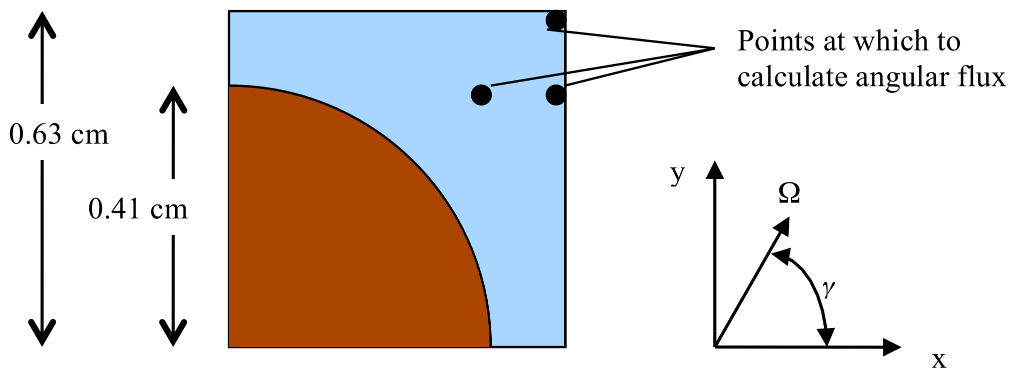


Figure 1: Quarter of a pin cell of infinite square lattice problem. The azimuthal angle φ is written as γ in the figure.

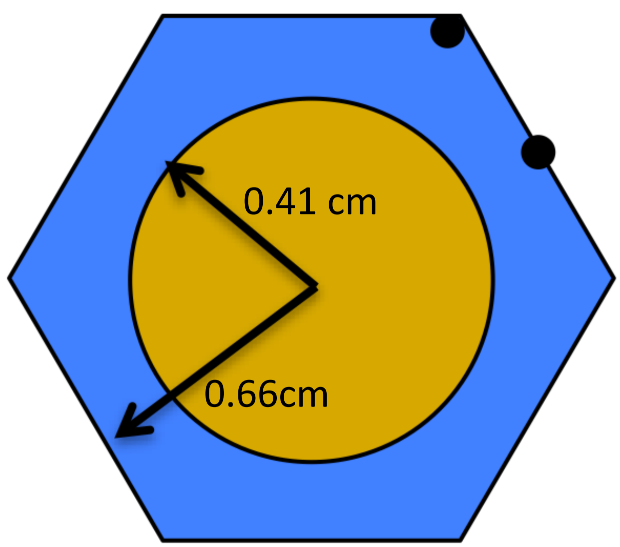


Figure 2: Pin cell of infinite hex lattice problem.

1. Calculate the angular flux as a function of the azimuthal angle, φ, at the spatial points indicated in the figure (two points for the hex lattice; three for the square). Use two different polar angles: $\pi/2$, which means the neutrons are traveling in the x-y plane, and $\pi/8$. Use the dimensions and cross sections from Table 1. Note that to simplify the problem we have abolished scattering. In the highest energy group of a fine-group set, there is very little within-group scattering, so it does not change the problem very much to ignore scattering. We have also assumed that the neutrons are born uniformly in the fuel – a flat radial profile. This isn't precisely true, but again, the simplification does not change the character of the solution that we wish to study. You will need to write a simple computer program for this in whatever language you'd like. You will need to trace rays and compute points of intersection. When you reach the boundary of a pin cell you use a periodic boundary condition to translate the ray across the cell, and then you continue. You will need a strategy to know how far a ray must be traced before you say "enough." Your code should accept as input:
   (a) the number of values of the azimuthal angle, φ at which to calculate the angular flux;
   (b) the precision to which to calculate the angular flux at a given spatial point and given φ. (This tells you when you can say "enough." You can say "enough" when you've traced through τ mean free paths, where $\exp(-\tau) =$ the requested precision.)
2. Convince me that your code calculates the angular flux correctly.
3. Plot the angular flux as a function of φ for each of the two polar angles, for each of the three spatial points (two if hex lattice). Use enough φ values to convince yourself that you have resolved all the significant bumps and wiggles in the angular flux. Discuss your plots, and in particular compare them against what was shown in the notes for square pins. Do the circles make things smoother? Be prepared to present your solutions to the class, and (see part above) be prepared to argue that they are correct.

Problem 1:

Henyey and Greenstein (1941) introduced a function which, by the variation of one parameter, $1 \leq h \leq 1$, ranges from backscattering through isotropic scattering to forward scattering. In our notation we can write this as

$$K(\mu_0, v' \to v) = \frac{1}{2} \frac{1 - h^2}{(1 + h^2 - 2h\mu_0)^{3/2}} \delta(v' - v). \tag{1}$$

Verify that this is a properly normalized $f(\mu_0)$ and compute $K_l(v' \to v)$ for $l = 0, 1, 2$ as a function of h.

Solution:

Solution:

- First, expand scattering kernel in Legendre Polynomials.
- Begin w/ Legendre generating fn:

$$\frac{1}{\sqrt{1-2xh+h^2}} = \sum_{n=0}^{\infty} P_n(x) h^n \qquad (1) \text{ [wiki]}$$

- Take a derivative w.r.t. x

$$\frac{h}{(1-2xh+h^2)^{3/2}} = \sum_{n=0}^{\infty} P_n'(x) h^n \qquad (2)$$

- Multiply both sides by $\frac{(1-h^2)}{2h}$:

$$K(x) = \frac{1}{2} \frac{(1-h^2)}{(1-2xh+h^2)^{3/2}} = \frac{1}{2}\sum_{n=0}^{\infty} P_n'(x) h^{n-1}(1-h^2) \qquad (3)$$

- (3) is our scattering kernel. Now we need to eliminate the derivative in terms of $P_n(x)$. Rewrite sum by letting $n \to n+1$

$$\frac{1}{2}\sum_{n=0}^{\infty} P_n'(x) h^{n-1}(1-h^2) = \frac{1}{2}\sum_{n=0}^{\infty} P_{n+1}'(x) h^{n}(1-h^2) \qquad (4)$$

- Note that the sum still starts at zero because $P_0'(x)=0$, so we can trivially add this term. From Abromowitz:

$$P_{n+1}'(x) = \sum_{m \text{ even}}^{n} (2m+1) P_m(x) \qquad (5)$$

- (5)→(4) and write as two sums:

$$K(x) = \frac{1}{2}\sum_{n=0}^{\infty} \sum_{\substack{m \\ \text{even}}}^{n} (2m+1) P_m(x) h^n - \frac{1}{2}\sum_{n=0}^{\infty} \sum_{\substack{m \\ \text{even}}}^{n} (2m+1) P_m(x) h^{n+2} \qquad (6)$$

* Note: Here orthogonality is defined as $\int P_m P_n = \frac{\delta_{mn} 2}{2n+1}$ *

• Shift the second infinite sum by $n' = n+2$

$$K = \frac{1}{2} \sum_{n=0}^{\infty} \sum_{\substack{m \\ \text{even}}}^{n} (2m+1) P_m(x) h^n - \frac{1}{2} \sum_{n'=2}^{\infty} \sum_{\substack{m \\ \text{even}}}^{n'-2} (2m+1) P_m(x) h^{n'}$$

• Combine the two series, writing out $n=0,1$ & letting $n' \to n$

$$K = \frac{1}{2} \sum_{n=0}^{1} (2n+1) P_n(x) h^n + \frac{1}{2} \sum_{n=2}^{\infty} \left(\sum_{\substack{m \\ \text{even}}}^{n} (2m+1) P_m(x) - \sum_{\substack{m \\ \text{even}}}^{n-2} (2m+1) P_m(x) \right) h^n$$

• For the inner sums all terms but n cancel, adding back energy:

$$\boxed{K(\mu_0, \nu' \to \nu) = \frac{1}{2} \sum_{n=0}^{\infty} (2n+1) P_n(\mu_0) h^n \delta(\nu' - \nu)} \qquad (7)$$

• Taking legendre moments:

$$K_\ell(\nu' \to \nu) = \frac{\delta(\nu' - \nu)}{2} \sum_{n=0}^{\infty} (2n+1) h^n \int_{-1}^{1} d\mu_0 \, P_n(\mu_0) P_\ell(\mu_0)$$

$$K_\ell(\nu' \to \nu) = \delta(\nu' - \nu) \sum_{n=0}^{\infty} \left(\frac{2n+1}{2} \right) h^n \frac{\delta_{n\ell} 2}{(2\ell+1)}$$

$$\boxed{K_\ell(\nu' \to \nu) = h^\ell}$$

• Noting $\int_0^\infty d\nu \int_{-1}^{1} d\mu_0 (\cdot) = \int_0^\infty d\nu \int_{-1}^{1} P_0(\mu_0) (\cdot)$; $\int_0^\infty d\nu \int_{-1}^{1} d(\mu_0) K(\mu_0, \nu' \to \nu) = h^0 = 1$,

So normalization is correct.

$$\boxed{\begin{aligned} K_0 &= 1\, \delta(\nu - \nu') \\ K_1 &= h\, \delta(\nu - \nu') \\ K_2 &= h^2 \delta(\nu - \nu') \end{aligned}}$$

Problem 2:

In an elastic scatter between a neutron and a nucleus, the scattering angle in the center of mass system is related to the energy change as

$$\frac{E}{E'} = \frac{1}{2}\left((1+\alpha) + (1-\alpha)\cos\theta_c\right) \tag{2}$$

where E is the energy after scattering and E' is the initial energy of the neutron and

$$\alpha = \frac{(A-1)^2}{(A+1)^2}. \tag{3}$$

The scattered angle in the center-of-mass system is related the lab-frame scattered angle as

$$\tan\theta_L = \frac{A\sin\theta_c}{1 + A\cos\theta_c} \tag{4}$$

Also, the distribution of scattered energy is given by

$$P(E' \to E) = \begin{cases} \frac{1}{(1-\alpha)E'} & E'\alpha \le E \le E' \\ 0 & \text{otherwise} \end{cases}. \tag{5}$$

Derive an expression for $K(\mu_0, E' \to E)$, where μ_0 is $\cos\theta_L$. What is the distribution in angle of neutrons of energy in the range [0.05 MeV, 10 MeV] to energies below 1 eV if the scatter is with hydrogen?

Solution:

Scattering Kernel Derivation

Due to Eq. (2), for a fixed A, a given value of E and E' fully define μ_c; the lab frame cosine of the scattering angle μ_0 is also fully defined through Eq. (4). As a result, the shape of the doubly differential scattering cross section is fully defined by the probability density function (PDF) $P(E' \to E)$. Thus, it is possible to write the scattering cross section in the COM frame as [1]

$$\Sigma_s(\mu_0, E' \to E) = \Sigma_s(E')P(E' \to E)\delta(\mu_c - f_\mu(E, E')) \tag{6}$$

where $f_\mu(E, E')$ is the value of μ_c that satisfies Eq. (2) for a given E, i.e.,

$$f_\mu(E, E') = \frac{2(\frac{E}{E'}) - (1+\alpha)}{(1-\alpha)} \tag{7}$$

Because we are interested in the scattering kernel as a function of the lab frame cosine μ_0, we define the scattering cross section in an equivalent form

$$\Sigma_s(\mu_0, E' \to E) = \Sigma_s(E')P(\mu_0)\delta(E - f_E(\mu_c(\mu_0), E')) \tag{8}$$

where $P(\mu_0)$ is a PDF for μ_0 given a certain value of E', f_E is defined as

$$f_E(\mu_C, E') = \frac{E'}{2}\left((1+\alpha) + (1-\alpha)\mu_c\right), \tag{9}$$

and μ_c as a function of μ_0 will be derived later in Eq. (21). The scattering kernel is defined as

$$K(\mu_0, E' \to E) = \frac{\Sigma_s(E' \to E, \mu_0)}{\int\limits_0^\infty \mathrm{d}E \int\limits_{-1}^{1} \mathrm{d}\mu_0\, \Sigma_s(E' \to E, \mu_0)} \tag{10}$$

The denominator is evaluated as

$$\int\limits_{-1}^{1} \mathrm{d}\mu_0 \int\limits_0^\infty \mathrm{d}E\ \Sigma_s(E')P(\mu_0)\delta(E - f_E(\mu_c(\mu_0), E)) = \Sigma_s(E') \int\limits_{-1}^{1} \mathrm{d}\mu_0 P(\mu_0) = \Sigma_s(E') \tag{11}$$

where the first equality is true because the argument of the delta function is zero for the value of μ_0 and E' that satisfy f, which in this case gives the μ_0 that is the integration variable of the outer integral. The scattering Kernel is then just

$$K(\mu_0, E' \to E) = P(\mu_0)\delta(E - f_E(\mu_C(\mu_0), E)). \tag{12}$$

We now need to transform the PDF $P(E' \to E)$ into a density function $P(\mu_0)$. From Eq. (2), there is a one-to-one relationship between E and $\mu_c = \cos(\Theta_c)$ in the range of $E \in [\alpha E', E']$, thus

$$P(E' \to E)\mathrm{d}E = P(\mu_c)\mathrm{d}\mu_c \tag{13}$$

or

$$P(\mu_c) = P(E' \to E)\frac{\mathrm{d}E}{\mathrm{d}\mu_c}. \tag{14}$$

Multiplication of Eq. (2) by E', followed by differentiation, yields

$$\frac{\mathrm{d}E}{\mathrm{d}\mu_c} = \frac{1}{2}(1-\alpha)E' \tag{15}$$

Evaluating μ_c for E at the limits $\alpha E'$ and E' gives the support for $P(\mu_c)$, defined for $\mu_c \in [-1, 1]$. Substitution of the above equation and Eq. (5) into Eq. (14) gives the PDF in the COM frame

$$P(\mu_c) = \frac{1}{(1-\alpha)E'}\left(\frac{1}{2}(1-\alpha)E'\right) = \frac{1}{2}, \quad \mu_c \in [-1, 1] \tag{16}$$

We must now transform to the lab frame scattering cosine μ_0. First, we solve Eq. (4) for μ_0 in terms of μ_c as follows:

$$\tan^2\theta_L = \left(\frac{A\sin\theta_c}{1 + A\cos\theta_c}\right)^2 \tag{17}$$

$$\sec^2\theta_L - 1 = \left(\frac{A\sin\theta_c}{1 + A\cos\theta_c}\right)^2 \tag{18}$$

$$\mu_0^{-2} = \frac{A^2(\sin^2\theta_c + \cos^2\theta_c) + 1 + 2A\mu_c}{(1 + A\mu_c)^2} \tag{19}$$

$$\mu_0 = \frac{1 + A\mu_c}{\sqrt{1 + 2\mu_c A + A^2}}. \tag{20}$$

Solution of the above equation for μ_c in terms of μ_L gives

$$\mu_c = -\frac{1}{A}(1-\mu_0^2) + \mu_0\sqrt{1 - \frac{1}{A^2}(1-\mu_0^2)}\,. \tag{21}$$

Eq. (20) demonstrates a one-to-one relationship between μ_0 and μ_C. As before,

$$P(\mu_0) = P\left(\mu_C(\mu_0)\right)\frac{\mathrm{d}\mu_c}{\mathrm{d}\mu_0}. \tag{22}$$

Differentiation of Eq. (21) with respect to μ_0 and algebraic manipulation ultimately yields

$$\frac{\mathrm{d}\mu_c}{\mathrm{d}\mu_0} = \frac{2\mu_0}{A} + \frac{1 - \frac{1}{A^2}(1-2\mu_0^2)}{\sqrt{1 - \frac{1}{A^2}(1-\mu_0^2)}}. \tag{23}$$

Substitution of the above equation and Eq. (16) into Eq. (22) gives an expression for $P(\mu_0)$. The final expression for the scattering kernal is, for $A > 1$

$$\boxed{K(\mu_0, E' \to E) = \begin{cases} \left[\dfrac{\mu_0}{A} + \dfrac{1 - \frac{1}{A^2}(1-2\mu_0^2)}{2\sqrt{1 - \frac{1}{A^2}(1-\mu_0^2)}}\right]\delta(E - f_E\left(\mu_c(\mu_0), E'\right)), & \mu_0 \in [-1,1],\ E \in [\alpha E', E'] \\ 0, & \text{otherwise} \end{cases}}. \tag{24}$$

where the support is from evaluation of Eq. (20) at $\mu_c = -1, 1$. The case of $A = 1$ must be treated separately. This can be seen, for instance, because evaluation of Eq. (20) at $\mu_c = -1$ results in an indeterminant $0/0$. Evaluation of Eq. (21) for $A = 1$ gives a non-indeterminant expression for μ_0 as

$$\mu_0 = \sqrt{\frac{1 + \mu_c}{2}} \tag{25}$$

Thus, the support becomes $\mu_0 \in [0, 1]$. The kernel also simplifies significantly at $A = 1$. The final scattering kernel, for the case of $A = 1$, is

$$\boxed{K(\mu_0, E' \to E) = \begin{cases} 2\mu_0\,\delta(E - f_E\left(\mu_c(\mu_0), E'\right)), & \mu_0 \in [0,1],\ E \in [0, E'] \\ 0, & \text{otherwise} \end{cases}}. \tag{26}$$

which is a PDF normalized over μ_0 and E. It is noted the support of E is implicitly defined by the value of E' and the support of μ_0, and the units of the delta function are per unit energy.

Plots for A=1

To plot the desired distributions, we need to know the equivalent μ_0 corresponding to a given E and E'. Evaluation of Eq. (2) at $A = 1$ gives

$$\frac{E}{E'} = \frac{1+\mu_c}{2}. \tag{27}$$

Then, using Eq. (25), μ_0 in terms of E and E' is

$$\mu_0 = \sqrt{\frac{E}{E'}}. \tag{28}$$

For a given $E' = E_i$, we can get the distribution in angle $P(\mu_0)$ by integrating the kernel over the range of desired outgoing energies $E \in [0, E_{\max}]$. The kernel is a joint PDF in E and μ_0 (for scattering into dE about E and dμ_0 about μ_0), whereas E', as we have defined the scattering kernel, is just a parameter of the distribution. Thus, performing the integration for a particular E' gives

$$P(\mu_0, 0 \le E \le E_{\max}; E' = E_i) = \int_0^{E_{\max}} \mathrm{d}E \; 2\mu_0 \delta(E - f_E(\mu_0, E_i)) \tag{29}$$

The delta function argument is now only zero for values of μ_0 and E' that satisfy $0 \le E \le E_{\max}$; the integration essentially restricts the support of μ_0. So the desired distribution becomes, using Eq. (28),

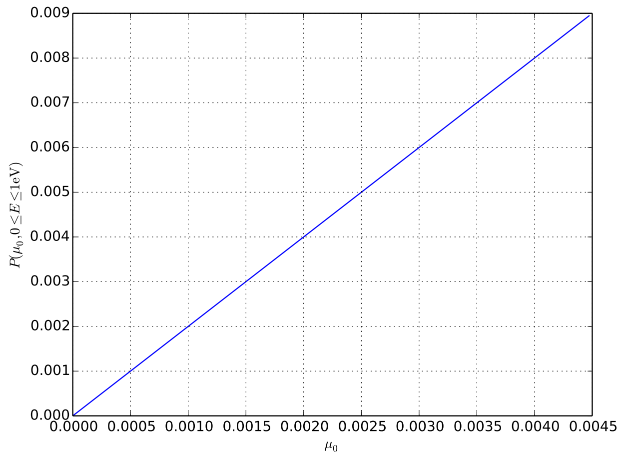
$$P(\mu_0, 0 \le E \le E_{\max}) = 2\mu_0, \quad 0 \le \mu_0 \le \sqrt{\frac{E_{\max}}{E'}}. \tag{30}$$

Physically, this result suggests that if E' is too much larger than $E_{\max}$, it cannot undergo a small deflection scatter and achieve an energy below $E_{\max}$.

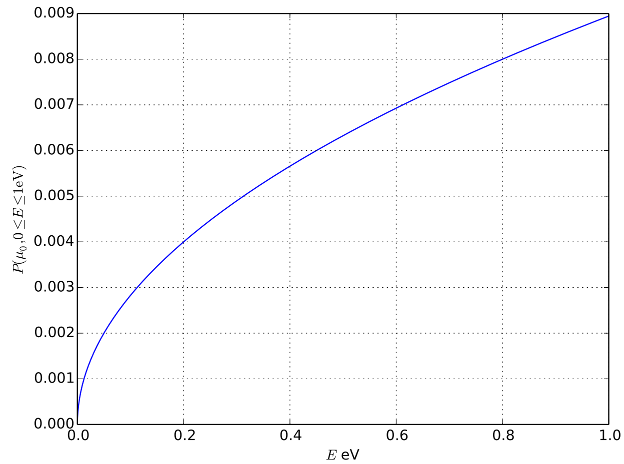
A plot of $P(\mu_0, 0 \le E \le 1 \text{ eV})$ vs μ_0 and vs $E(\mu_0)$ are given below. Plots are shown for the limiting cases of $E' = 10$ MeV and $E' = 0.05$ MeV. The shape of the distribution in μ_0 is linear for all energies, however the range of the distribution differs (the scales are different on figures). The magnitudes of the plots versus energy demonstrate that neutrons with a lower E' are more likely to scatter below 1 eV, as expected

References

[1] W.L. Dunn and J.K. Shultis, *Exploring Monte Carlo Methods*, 2012.

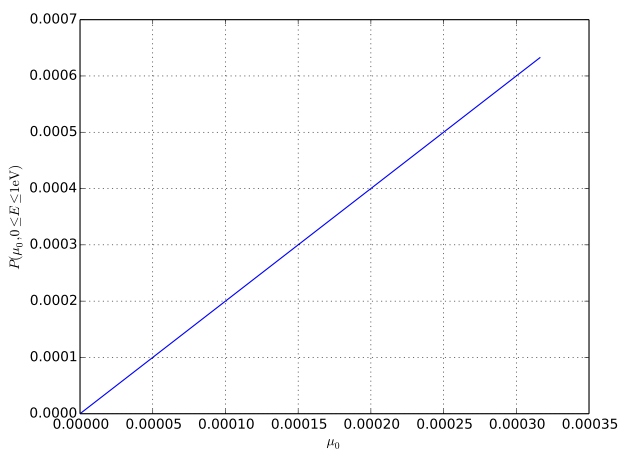


(a) $P(\mu_0)$ vs μ_0

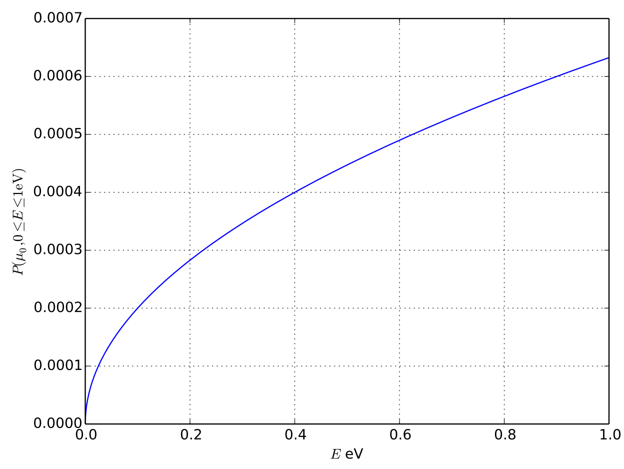


(b) $P(\mu_0)$ vs E

Figure 1: Angular ditributions for $E' = 0.05$ MeV



(a) $P(\mu_0)$ vs μ_0



(b) $P(\mu_0)$ vs E

Figure 2: Angular distributions for $E' = 10$ MeV

Problem 3:

The problem details are given on the second page.

Solution:

Description of code

The angular flux ψ is computed by tracing characteristics as discussed in class. To compute points of intersection, the ray and surfaces of intersection are written in parametric form. The position of a particle in the projected $x-y$ plane is denoted $\mathbf{r} = x\hat{\mathbf{i}} + y\hat{\mathbf{j}}$. Since we want to trace upstream, the parametric equation for the particle position is given by

$$\mathbf{r} = (x_{i-1} - \Omega_x s)\hat{\mathbf{i}} + (y_{i-1} - \Omega_y s)\hat{\mathbf{j}} \tag{31}$$

where $\mathbf{r}_{i-1}$ is the previous location, s is a parameter that corresponds to the signed distance the ray has traversed, and

$$\Omega_x = \sin(\theta)\cos(\phi) \tag{32}$$

$$\Omega_y = \sin(\theta)\sin(\phi). \tag{33}$$

The parametric equation for each of the surfaces in the problem as a function of x and y are given in Table 1. These equations are then evaluated with $x = x_{i-1} - \Omega_x s$ and $y = y_{i-1} - \Omega_y s$ and solved for s algebraically. This produces a collection of values $\{s_m\}$, where s_m represent the signed distance to the m-th surface (excluding surfaces where a solution does not exist). The smallest positive value of s_m corresponds to the next point of intersection. If we were already at a surface, then that equation will give $s_m = 0$. Care is taken to exclude this solution, accounting for potential roundoff.

Table 1: Parametric equations for surfaces in problem.

Surface	$f(x,y) = 0$
Fuel	$x^2 + y^2 - R_{\text{fuel}}^2 = 0$
Left Boundary	$x - x_{\text{min}} = 0$
Right Boundary	$x - x_{\text{max}} = 0$
Bottom Boundary	$y - y_{\text{min}} = 0$
Top Boundary	$y - y_{\text{max}} = 0$

The current position of the ray is updated, and the number of mean free paths traveled $\tau_i = s_i \Sigma_t(x,y)$ along the i-th path is computed. The total number of MFP traveled up to the latest point s_i is accumulated as $\tau_{\text{tot},i} = \sum_{k=1}^{i} \tau_k$.

Because the transport equation is linear, we can consider the contribution from each fuel element to the angular flux separately. If the i-th path traced to point $\mathbf{r}_i$ was across a fuel element, then a contribution is made to the flux. If the path of length s_i crossed the j-th fuel element, the contribution to the flux from that fuel element is computed as

$$\psi_j = \frac{Qe^{-\tau_{\mathrm{tot},i-1}}}{4\pi\Sigma_{t,F}} \left(1 - e^{-\Sigma_{t,F} s_i}\right) \tag{34}$$

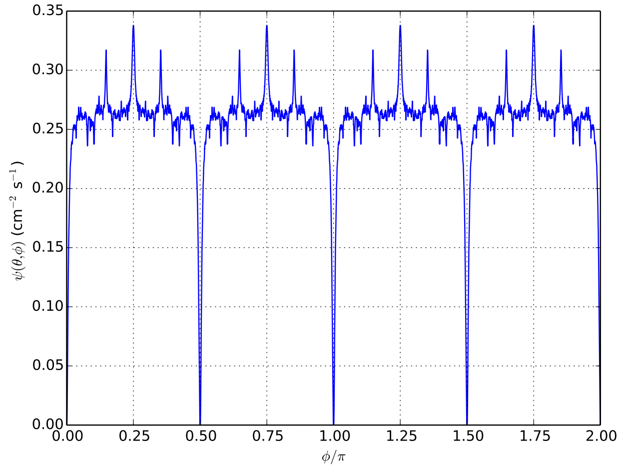
where $\tau_{\mathrm{tot},i-1}$ does not include $s_i\Sigma_{t,F}$ because that attenuation was accounted for in derivation of the term in parenthesis. The ray tracing is then continued from this point until $\tau_{\mathrm{tot,i}} > \tau_{\mathrm{max}}$.

Finally, after computing potential contributions to ψ, if the ray hits a boundary, the corresponding coordinate is translated to the opposing boundary. For example, if the right boundary is hit at point $\mathbf{r} = x_{\mathrm{max}}\hat{\mathbf{i}} + y_1\hat{\mathbf{j}}$, the new position is $\mathbf{r} = x_{\mathrm{min}}\hat{\mathbf{i}} + y_1\hat{\mathbf{j}}$. Tracing then continues as before, by computing new distances to intersections, with $\hat{\Omega}$ unchanged. Care is taken to handle roundoff issues and corners.

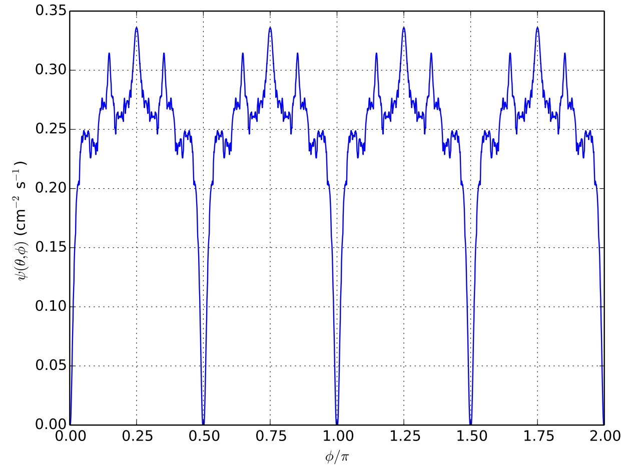
The final solution for ψ, at the location and direction of interest, will be

$$\psi(\mathbf{r}, \hat{\Omega}) = \sum_{j}^{N_{\mathrm{fuel}}} \psi_j \tag{35}$$

where N_{fuel} is the total number of fuel elements crossed during the ray tracing. The process outlined above is repeated for all equally spaced $\phi \in [0, 2\pi)$, for the polar angle and position of interest.



(a) $\theta = \pi/2$



(b) $\theta = \pi/8$

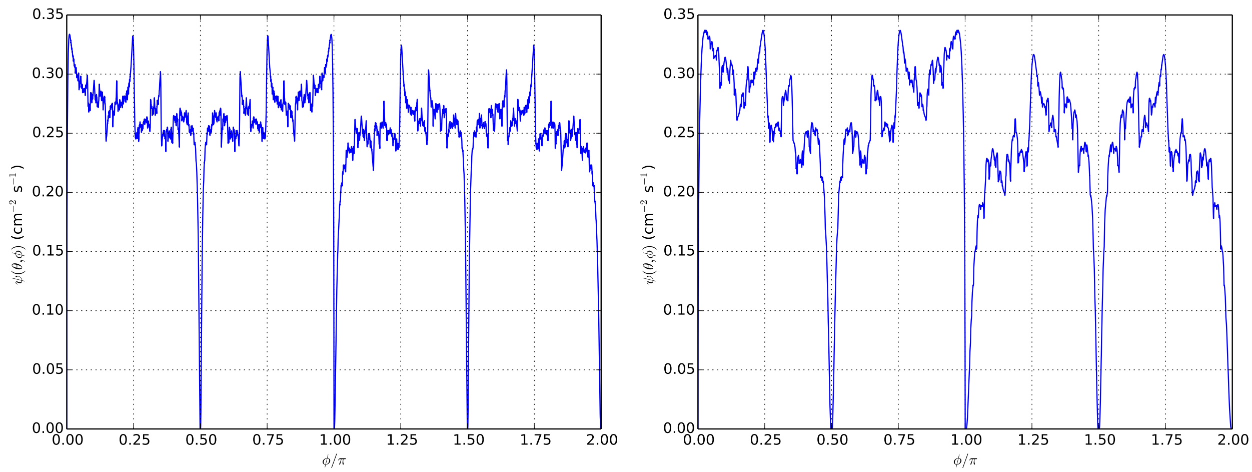
Figure 3: Angular flux results at $x = 0.63$ cm $y = 0.63$ cm for 8000 azimuthal angles.

Results

To verify the code works, the algorithm was modified to handle a source in the moderator by evaluating Eq. (34) at every point, with Q and Σ_t the same in all regions of the problem. This was found to produce the expected answer of $\psi = Q/4\pi\Sigma_t$ for all angles and positions tested. It is noted that a bug was found later after running this test, so this is not the most rigorous verification test.

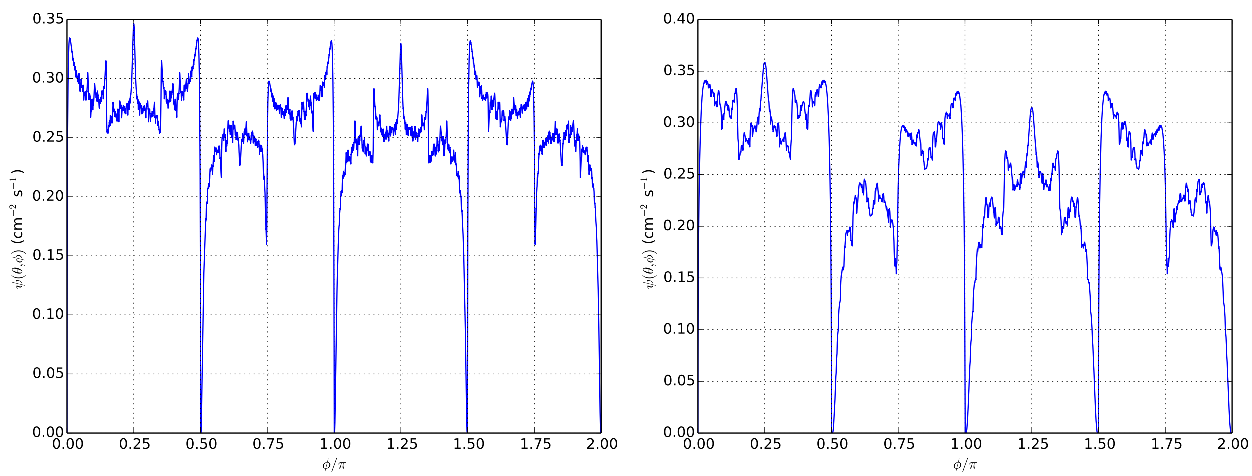
Figures are given below for each of the desired points and directions The results were found to have physically expected values. Comparing the plots at $\theta = \pi/2$ and $\theta = \pi/8$, the latter results are more stretched with slightly higher peaks and lower troughs. This is expected as the fuel element will appear more elliptic to the point of interest. For the case of (0.63,0.63), symmetry in the 4 quadrants appeared as expected. The largest values of the flux, with the same magnitude, were found at $\phi = \pi/4, 3\pi/4, 5\pi/4$, and $7\pi/4$; these are the points where the rays cross the entirety of the fuel circumference. The values of 0 at factors of $\pi/2$ were also seen as expected. This is the only angle where no fuel tracks to the spatial location. At slightly different angles, a fuel pin is hit at some point due to the limited attenuation by the moderator. In the other plots, the expected relative magnitudes of the flux in the different quadrants were seen as expected. For instance, at (0.41,0.41), in $[0, \pi/2]$ there is a similar shape as from $[\pi, 3\pi/2]$, but in larger magnitude. The extra edges near 0 and $\pi/2$ occur because the ray sees most of the next fuel element past the one it is nearest to, however this does not occur in $[\pi, 3\pi/2]$. As an additional verification, the code was compared against other students' codes who developed algorithms independently and were found to agree. Although this is not proof the solution method is correct, it indicates that the code is implemented as intended.

It was found that at 8000 values of ϕ there was no visible difference in the results. A plot of one case for 8000 and 800 directions can be seen below in Fig. 6. There are still a few points that are not picked up at 800 directions.



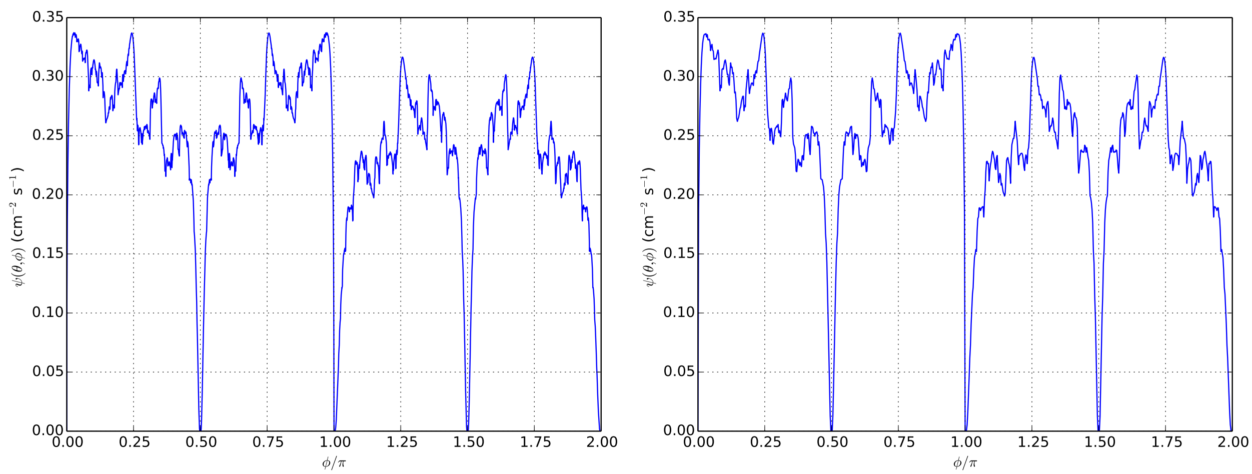
(a) $\theta = \pi/2$ (b) $\theta = \pi/8$

Figure 4: Angular flux results at $x = 0.63$ cm $y = 0.41$ cm for 8000 azimuthal angles.



(a) $\theta = \pi/2$ (b) $\theta = \pi/8$

Figure 5: Angular flux results at $x = 0.41$ cm $y = 0.41$ cm for 8000 azimuthal angles.



(a) 8000 azimuthal angles

(b) 800 azimuthal angles.

Figure 6: Angular flux results at $\theta = \pi/8$, $x = 0.63$ cm, and $y = 0.41$ cm.

```python
import numpy as np
import re
from math import *
from scipy.optimize import fsolve
import matplotlib.pyplot as plt

def main(n_azimuth, polar_ang, tol=1.e-12):

    # Define geometry parameters based on origin is center of fuel pin
    x_left  = -0.63
    x_right =  0.63
    y_top   =  0.63
    y_bot   = -0.63
    radius  =  0.41

    x_start = 0.62999999
    y_start = 0.41

    #Must be 'definitely' inside or algorithm will fail
    if abs(abs(x_start)-x_right) < 1.E-10 or abs(abs(y_start)-y_top) < 1.E-10:
        raise IOError("Must start at a point inside boundary or problems will occur")

    #cross sections
    sigma_f = 0.1414
    sigma_m = 0.08
    Q_f_tot = 1.
    q_f     = Q_f_tot/(4.*pi)
    q_mod   = 0.0

    debug_mode = False
    if debug_mode:
        sigma_m = sigma_f
        q_mod = q_f


    phi_list = np.linspace(0.,2.*pi,num=n_azimuth+1) #add one to get endpoints
    psi_list = []

    #loop over azimuthal angles
    for phi in phi_list:

        #Pick the point of interest and trace upstream from it
        x_prev = x_start
        y_prev = y_start

        #Pick direction
        theta = polar_ang
        xcos =  sin(theta)*cos(phi)
        ycos =  sin(theta)*sin(phi)

        print "Tracing phi (Omega) (max mfp)", phi, -1.*log(tol)

        #Num of mfp we've traveled, and angular flux contribution to this point
        psi = 0.0
        n_mfp = 0.0
        max_mfp = -1.*log(tol)

        #We are ray tracing upstream, so flip cosines
        xcos *= -1.
        ycos *= -1.

        #s is parametric length of vector
        #f_circ can be used to check if circle hit or not
        f_circ = lambda s: (x_prev + xcos*s)**2 + (y_prev + ycos*s)**2 - radius**2

        while (n_mfp < max_mfp):

            #Calculate all boundary intersections
```

```
            if xcos == 0.:
                s_left = -99
                s_right = -99
            else:
                s_left  = (x_left - x_prev)/xcos
                s_right = (x_right - x_prev)/xcos

            if ycos == 0.:
                s_top = -99
                s_bot = -99
            else:
                s_top   = (y_top  - y_prev)/ycos
                s_bot   = (y_bot - y_prev)/ycos

            #Roots of parametric equation for circle
            A = xcos**2 + ycos**2
            B = 2.*xcos*x_prev + 2.*ycos*y_prev
            C = x_prev**2 + y_prev**2 - radius**2
            det = B*B - 4.*A*C

            #Determine if we hit the circle, if so this overrides the boundary
            if (det > 0): #We hit the circle

                #Roots of quadratic eq
                s_circ1 = (-1.*B + sqrt(det))/(2.*A)
                s_circ2 = (-1.*B - sqrt(det))/(2.*A)

            else: #We didnt hit the circle, so we must have left

                s_circ1 = -99
                s_circ2 = -99

            #Find the min s that is positive, this is the face we hit
            intersects = [s_left, s_right, s_top, s_bot, s_circ1,s_circ2]
            s_min = min(i for i in intersects if i > 1.E-15 ) #ignore very small roots
            face_id = intersects.index(s_min)
            face_map = ["left","right","top","bot","circ1","circ2"]
            face = face_map[face_id]

            #Check if center of path is in fuel, in this case we were in the fuel
            r_cent = sqrt((x_prev + xcos*s_min*0.5)**2 + (y_prev + ycos*s_min*0.5)**2)
            if (r_cent < radius): #in the fuel:

                #Contribution to psi from this source is based on flux leaving fuel and how
                #many mfp it traveled to get to this point
                mfp_fuel = s_min*sigma_f
                psi     += q_f/(sigma_f)*(1.-exp(-1.*mfp_fuel))*exp(-1.*n_mfp)
                n_mfp   += mfp_fuel

            else: #just traveling in moderator

                n_mfp += s_min*sigma_m #We had to have been in moderator

                #For debugging we have a moderator source that we add in. It will contribute
                #however much is at previous point and how far it has had to attenuate
                if debug_mode:
                    psi    += q_mod/(sigma_f)*(1.-exp(-1.*s_min*sigma_m))*exp(-1.*(n_mfp - s_min*sigma_m))

            #Move to the new coordinates
            x_prev = x_prev + xcos*s_min
            y_prev = y_prev + ycos*s_min

            #Determine if we hit a boundary. Either we hit a circle or boundary
            if not re.search("circ",face):

                #Move to opposite face
                if face == "left":
                    x_prev = x_right
```

```python
                elif face == "right":
                    x_prev = x_left
                elif face == "bot":
                    y_prev = y_top
                elif face == "top":
                    y_prev = y_bot
                else:
                    raise ValueError("Something wrong in faces")

                #Check if we are in a corner, based on symmetry, requires attention
                if x_left == y_bot: #Check for symmetry
                    if abs(abs(x_prev) - abs(y_prev)) < 1.E-13*abs(x_left):

                        #flip the face we haven't flipped yet
                        if face == "left" or face == "right":
                            y_prev *= -1.
                        else:
                            x_prev *= -1.


        #Done tracing for this phi
        if debug_mode:
            print "Desired solution: ", q_mod/sigma_f, q_mod, q_f
            print "error: ", (q_mod/sigma_f - psi)
            print "tol:   ", psi*tol

        psi_list.append(psi)

    #plot angular flux as a function of azimuthal angle
    plot_phi = [i/pi for i in phi_list]
    plt.plot(plot_phi,psi_list)
    plt.xlabel("$\phi/\pi$")
    ax = plt.gca()
    ax.set_xticks([0,0.25,0.5,0.75,1.0,1.25,1.5,1.75,2.0])
    plt.ylabel(r"$\psi(\theta,\phi)$ (cm$^{-2}$ s$^{-1}$) ")
    plt.grid()
    plt.savefig("plot.pdf",bbox_inches='tight')


if __name__ == "__main__":
    main(800,pi/8.,tol=2.06115362e-13)
```