Problem 1

We are given

$$\left[\hat{\Omega}\cdot\nabla + \Sigma_t(E)\right]\psi(\vec{r},E,\hat{\Omega}) = \int_{4\pi} d\hat{\Omega}' \int_0^\infty dE' \Sigma_s(E',\hat{\Omega}'\to E,\hat{\Omega})\psi(\vec{r},E',\hat{\Omega}')$$
$$+ \frac{1}{k}\frac{\chi(E)}{4\pi}\int_0^\infty dE'\nu(E')\Sigma_f(E')\int_{4\pi} d\hat{\Omega}'\psi(\vec{r},E',\hat{\Omega}').$$

If we have isotropic scattering (and no cross section spatial dependence), then $\Sigma_s(E',\hat{\Omega}'\to E,\hat{\Omega}) = \Sigma_s(E'\to E)$.

$$\left[\hat{\Omega}\cdot\nabla + \Sigma_t(E)\right]\psi(\vec{r},E,\hat{\Omega}) = \int_0^\infty dE' \Sigma_s(E'\to E)\int_{4\pi} d\hat{\Omega}'\psi(\vec{r},E',\hat{\Omega}')$$
$$+ \frac{1}{k}\frac{\chi(E)}{4\pi}\int_0^\infty dE'\nu(E')\Sigma_f(E')\int_{4\pi} d\hat{\Omega}'\psi(\vec{r},E',\hat{\Omega}').$$

Now, we let $\psi(\vec{r},E,\hat{\Omega}) = \psi_0(E,\hat{\Omega})\exp(i\hat{\Omega}\cdot\vec{B})$. Noting that $\hat{\Omega}\cdot\vec{B} = \mu|\vec{B}|$, the equation for ψ can be substituted into our equation above to yield

$$\left[\hat{\Omega}\cdot\nabla + \Sigma_t(E)\right]\psi_0(E,\hat{\Omega})\exp(i\mu|\vec{B}|) = \int_0^\infty dE' \Sigma_s(E'\to E)\int_{4\pi} d\hat{\Omega}'\psi_0(E',\hat{\Omega}')\exp(i\mu|\vec{B}|)$$
$$+ \frac{1}{k}\frac{\chi(E)}{4\pi}\int_0^\infty dE'\nu(E')\Sigma_f(E')\int_{4\pi} d\hat{\Omega}'\psi_0(E',\hat{\Omega}')\exp(i\mu|\vec{B}|).$$

Rearranging to solve for k,

$$\left[\hat{\Omega}\cdot\nabla + \Sigma_t(E)\right]\psi_0(E,\hat{\Omega})\exp(i\mu|\vec{B}|) - \int_0^\infty dE' \Sigma_s(E'\to E)\int_{4\pi} d\hat{\Omega}'\psi_0(E',\hat{\Omega}')\exp(i\mu|\vec{B}|)$$
$$= \frac{1}{k}\frac{\chi(E)}{4\pi}\int_0^\infty dE'\nu(E')\Sigma_f(E')\int_{4\pi} d\hat{\Omega}'\psi_0(E',\hat{\Omega}')\exp(i\mu|\vec{B}|).$$

$$k = \frac{\frac{\chi(E)}{4\pi}\int_0^\infty dE'\nu(E')\Sigma_f(E')\int_{4\pi} d\hat{\Omega}'\psi_0(E',\hat{\Omega}')\exp(i\mu|\vec{B}|)}{\left[\hat{\Omega}\cdot\nabla + \Sigma_t(E)\right]\psi_0(E,\hat{\Omega})\exp(i\mu|\vec{B}|) - \int_0^\infty dE'\Sigma_s(E'\to E)\int_{4\pi} d\hat{\Omega}'\psi_0(E',\hat{\Omega}')\exp(i\mu|\vec{B}|)}$$

...

Problem 2

The one-group diffusion equation can be derived from the neutron transport equation using Fick's Law ($\vec{J}(\vec{r}) = -D(\vec{r})\nabla\phi$).

If we have no external source, the one-group transport equation is

$$\frac{1}{v}\frac{\partial\phi(\vec{r},\hat{\Omega},t)}{\partial t} + \Omega\cdot\nabla\phi(\vec{r},\hat{\Omega},t) + \Sigma_t\psi(\vec{r},\hat{\Omega},t) = \int_{4\pi} d\hat{\Omega}'\Sigma_s(\hat{\Omega}'\to\hat{\Omega})\psi(\vec{r},\hat{\Omega}',t) + \frac{\nu\Sigma_f}{4\pi}\int_{4\pi} d\hat{\Omega}'\psi(\vec{r},\hat{\Omega}',t)$$

This would then be integrated over angle, under the assumption that the angular flux is only weakly dependent on angle. Additionally, we assume that the scattering is azimuthally symmetric, and so $\int_{4\pi} d\hat{\Omega}\,\Sigma_s(\hat{\Omega}'\to\hat{\Omega}) = $

$\int_{4\pi} d\hat{\Omega}\, \Sigma_s(\hat{\Omega}' \cdot \hat{\Omega}) = \Sigma_s.$

$$\int_{4\pi} d\hat{\Omega} \left[\frac{1}{v} \frac{\partial \psi(\vec{r}, \hat{\Omega}, t)}{\partial t} + \hat{\Omega} \cdot \nabla \psi(\vec{r}, \hat{\Omega}, t) + \Sigma_t \psi(\vec{r}, \hat{\Omega}, t) \right] = \int_{4\pi} d\hat{\Omega} \left[\int_{4\pi} d\hat{\Omega}' \Sigma_s(\hat{\Omega}' \to \hat{\Omega}) \psi(\vec{r}, \hat{\Omega}', t) \right.$$

$$\left. + \frac{\nu \Sigma_f}{4\pi} \int_{4\pi} d\hat{\Omega}' \psi(\vec{r}, \hat{\Omega}', t) \right]$$

$$\frac{1}{v} \frac{\partial \phi(\vec{r}, t)}{\partial t} + \nabla \cdot \vec{J}(\vec{r}, t) + \Sigma_t \phi(\vec{r}, t) = \Sigma_s \phi(\vec{r}, t) + \nu \Sigma_f \phi(\vec{r}, t)$$

$$\frac{1}{v} \frac{\partial \phi(\vec{r}, t)}{\partial t} - D(\vec{r}) \nabla \phi(\vec{r}, t) + \Sigma_t \phi(\vec{r}, t) = \Sigma_s \phi(\vec{r}, t) + \nu \Sigma_f \phi(\vec{r}, t)$$

Now, we can assume our reactor is approximately steady state, so the time dependence vanishes, and we note that since we have only one group, the only scattering reactions which contribute to our neutron gain are (n, xn) reactions. The equation reduces to

$$-D(\vec{r}) \nabla \phi(\vec{r}) + \Sigma_t \phi(\vec{r}) = \sum_{n=2}^{\infty} x \Sigma_{(n,xn)} \phi(\vec{r}) + \nu \Sigma_f \phi(\vec{r})$$

$$-D(\vec{r}) \nabla \phi(\vec{r}) + \Sigma_t \phi(\vec{r}) - \sum_{n=2}^{\infty} x \Sigma_{(n,xn)} \phi(\vec{r}) = \nu \Sigma_f \phi(\vec{r})$$

For small deviations of k_{eff}, this can be

$$-D(\vec{r}) \nabla \phi(\vec{r}) + \Sigma_t \phi(\vec{r}) - \sum_{n=2}^{\infty} x \Sigma_{(n,xn)} \phi(\vec{r}) = \frac{1}{k} \nu \Sigma_f \phi(\vec{r})$$

$$\boxed{ k = \frac{\nu \Sigma_f \phi(\vec{r})}{-D(\vec{r}) \nabla \phi(\vec{r}) + \Sigma_t \phi(\vec{r}) - \sum_{n=2}^{\infty} x \Sigma_{(n,xn)} \phi(\vec{r})} }$$

Problem 3

Define G as the ratio between total power generated, P, to fusion power generated, P_F.

$$G \equiv \frac{P}{P_F}$$

As a ratio, this is also equivalent to the ratio between total energy created, E, to fusion energy created, E_F.

$$G = \frac{E}{E_F}$$

Since we are told that energy contributions from non-fission reactions are neglible. This suggests that the total energy is simply the sum of the fission energy produced, E_f, and fusion energy produced, E_F.

$$G = \frac{E_F + E_f}{E_F} = 1 + \frac{E_f}{E_F}$$

Each DT fusion reaction produces 17.6 MeV of energy and one neutron for triggering fission in the blanket, so $E_F = 17.6 n_0$ MeV, where n_0 is the number of neutrons produced from fusion. With an 80% probability of triggering an $(n, 2n)$ reaction in beryllium, the average fusion event results in approximately 1.8 neutrons eventually reaching the blanket. If we again take the fact that even negative energy-contributions from non-fission reactions are neglible, as well as the fact that no fusion neutrons leak from the system, then we can assume that each fusion neutron produces at least one fission event or elastically scatters. Mathematically,

$$\Sigma_f + \Sigma_s = \Sigma_t,$$

and the probability of a "first strike" fission event is

$$P(F) = \frac{\Sigma_f}{\Sigma_t}.$$

From this, we can determine the number of neutrons in the first generation after interaction in the blanket, n_1 as the number of neutrons produced in those "first strike" fissions plus the number of neutrons remaining after scattering

$$n_1 = 1.8(\nu P(f) + P(s))n_0 = 1.8 \left(\nu \frac{\Sigma_f}{\Sigma_t} + \frac{\Sigma_s}{\Sigma_t} \right) n_0.$$

Some of these neutrons may now leak out of the system, however, so in general the number of neutrons in generation t, given as n_t, will obey the relation

$$n_t = k_{\text{eff}} n_{t-1},$$

where k_{eff} now also accounts for leakage. The total fission energy produced by a single generation of fusion events and the ensuing multiplication and fission (assuming the energy produced in a single fission event to be about 200 MeV) will be

$$E_f = 200 \left(\frac{\Sigma_f}{\Sigma_t} \right) \left(1.8 n_0 + \sum_{t=1}^{\infty} n_t \right) \text{MeV}$$

$$E_f = 200 \left(\frac{\Sigma_f}{\Sigma_t} \right) \left(1.8 n_0 + \sum_{t=1}^{\infty} k_{eff} n_{t-1} \right) \text{MeV}$$

$$E_f = 200 \left(\frac{\Sigma_f}{\Sigma_t} \right) \left(1.8 n_0 + 1.8 \sum_{t=1}^{\infty} k_{eff}^{t-1} n_0 \right) \text{MeV}$$

$$E_f = 360 \left(\frac{\Sigma_f}{\Sigma_t} \right) n_0 \left(1 + \sum_{t=1}^{\infty} k_{eff}^{t-1} \right) \text{MeV}$$

$$E_f = 360 \left(\frac{\Sigma_f}{\Sigma_t} \right) n_0 \sum_{t=0}^{\infty} k_{eff}^{t-1} \text{MeV}$$

Altogether, we find

$$G = 1 + \frac{360 \left(\frac{\Sigma_f}{\Sigma_t} \right) n_0 \sum_{t=0}^{\infty} k_{eff}^{t-1} \text{MeV}}{17.6 n_0 \text{MeV}}$$

$$\boxed{G = 1 + 20.5 \left(\frac{\Sigma_f}{\Sigma_t} \right) \sum_{t=0}^{\infty} k_{eff}^{t-1}}$$

Problem 4

a.)

We are given the matrix:

$$\begin{bmatrix} 1 & 1 & -1 & 3 \\ 1 & 2 & -4 & -2 \\ 2 & 1 & 1 & 5 \\ -1 & 0 & -2 & -4 \end{bmatrix}$$

The inverse, $\mathbf{A}^{-1}$ of a square matrix, $\mathbf{A}$, is equal to the adjugate of the matrix, $\mathbf{A}^{\dagger}$ divided by the determinant of $\mathbf{A}$.

$$\mathbf{A}^{-1} = \frac{\mathbf{A}^{\dagger}}{\det \mathbf{A}}$$

The adjugate of a square matrix, $\mathbf{A}^\dagger$, is the transpose of the cofactor matrix, $\mathbf{C_A}$.

$$\mathbf{A}^\dagger = \mathbf{C}_\mathbf{A}^T$$

The cofactor of a square matrix, $\mathbf{C_A}$ is the signed matrix of minors, $\mathbf{M_A}$.

$$\mathbf{C}_{\mathbf{A},ij} = (-1)^{i+j}\,\mathbf{M_A}$$

The minor of matrix element $\mathbf{A}_{ij}$ is the determinant of submatrix formed with the rows and columns other than i and j.

We can use this all together to find the inverse of $\mathbf{A}$. The matrix given, however, has a determinant of zero, and so is <u>not invertible</u>.

(see attached Jupyter notebook for full calculations)

b.)

We are given the matrix:

$$\begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix}$$

and we know that the eigenvalue λ and eigenvector $\vec{v}$ obey the rule

$$\begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix}\vec{v} = \lambda\vec{v}$$

Equivalently,

$$\left(\begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix} - \lambda\mathbb{1}\right)\vec{v} = 0$$

We want the non-trivial solution to this equation, when $\vec{v} \neq \vec{0}$. $\vec{v} = \vec{0}$ when $\left(\begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix} - \lambda\mathbb{1}\right)$ is invertible, so we will instead assert that $\left(\begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix} - \lambda\mathbb{1}\right)$ is not invertible. By definition, this means

$$\det\left(\begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix} - \lambda\mathbb{1}\right) = 0$$

or

$$\det\begin{bmatrix} 3 - \lambda & -1 \\ -1 & 3 - \lambda \end{bmatrix} = 0$$

We can solve this now for lambda:

$$(3 - \lambda)^2 - (-1)^2 = 0$$
$$9 - 6\lambda + \lambda^2 - 1 = 0$$
$$8 - 6\lambda + \lambda^2 = 0$$

and using the quadratic formula we find

$$\lambda = \frac{-(-6) \pm \sqrt{(-6)^2 - 4(8)}}{2}$$

$$\lambda = \frac{6 \pm \sqrt{36 - 32}}{2}$$

$$\lambda = \frac{6 \pm 2}{2}$$

$$\lambda = 3 \pm 1$$

$$\boxed{\lambda = 2, 4}$$

We can use this eigenvalue to solve for $\vec{v}$.

$$\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \vec{v} = 0 \text{ and } \begin{bmatrix} -1 & -1 \\ -1 & -1 \end{bmatrix} \vec{v} = 0$$

This gives the equations

$$\lambda = 2 : \begin{cases} v_1 - v_2 = 0 \\ v_2 - v_1 = 0 \end{cases} \qquad \lambda = 4 : \{ -v_1 - v_2 = 0$$

For $\boxed{\lambda = 2, \ \vec{v} = \begin{bmatrix} v_0 \\ v_0 \end{bmatrix}}$, and for $\boxed{\lambda = 4, \ \vec{v} = \begin{bmatrix} v_0 \\ -v_0 \end{bmatrix}}$.

Problem 5

In problem 4 we stated that the inverse, $\mathbf{A}^{-1}$, of a square matrix, $\mathbf{A}$, is equal to the adjugate of the matrix, $\mathbf{A}^{\dagger}$ divided by the determinant of $\mathbf{A}$.

$$\mathbf{A}^{-1} = \frac{\mathbf{A}^{\dagger}}{\det \mathbf{A}}$$

We can manipulate this expression to find

$$(\det \mathbf{A})\mathbb{1} = \mathbf{A}^{\dagger}\mathbf{A}$$

Since we are looking for a self-adjugate matrix, $\mathbf{A}^{\dagger} = \mathbf{A}$, and

$$(\det \mathbf{A})\mathbb{1} = \mathbf{A}^2.$$

Then, taking the square root of both sides,

$$\left(\sqrt{\det \mathbf{A}} \right) \mathbb{1} = \mathbf{A}.$$

$$\mathbf{A} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} = \begin{bmatrix} \sqrt{\det \mathbf{A}} & 0 & 0 \\ 0 & \sqrt{\det \mathbf{A}} & 0 \\ 0 & 0 & \sqrt{\det \mathbf{A}} \end{bmatrix}.$$

$$a = e = i = \sqrt{\det \mathbf{A}}$$

$$\mathbf{A} = \begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{bmatrix} =$$

and

$$\det \mathbf{A} = a \begin{vmatrix} a & 0 \\ 0 & a \end{vmatrix}$$

$$\det \mathbf{A} = a(a^2)$$

$$\det \mathbf{A} = a^3.$$

We know that $a = \sqrt{\det \mathbf{A}}$, so

$$a = \sqrt{a^3}$$

$$a = a^{\frac{3}{2}}$$

$$a = 1$$

$$\boxed{\mathbf{A} = \mathbb{1}}$$

We can furthermore use the functions defined in the previous problem to show that $\mathbf{A}$ and $\mathbf{A}^\dagger$ are equal when found through the cofactor method.

(see attached Jupyter notebook for full calculations)

Problem 6

From Duderstadt and Hamilton, we can "define the operator $M^\dagger$ adjoint to the operator M as that operator $M^\dagger$ for which

$$(M^\dagger f, g) = (f, Mg),"$$

where parentheses denote the inner product such that

$$(f, g) \equiv \int_{a_i}^{a_f} f^*(\vec{r}) g(\vec{r}) dx$$

for the interval $[a_i, a_f]$, and f^* is the complex conjugate of f. Now, we wish to show that $L \equiv \frac{d^2}{dx^2}$ is self adjoint,

$$L = L^\dagger,$$

when f has a derivative everywhere, is symmetric about the origin, and obeys Dirichlet boundary conditions ($f(\pm a) = 0$), all on the closed interval $[-a, a]$.

If $L = L^\dagger$, then

$$(Lf, g) = (f, Lg)$$

$$= \int_{-a}^{a} f^*(x) Lg(x) \, dx$$

$$= \int_{-a}^{a} f^*(x) \frac{d^2 g(x)}{dx^2} \, dx$$

Using integration by parts once,

$$(Lf, g) = \left[f^*(x) \frac{dg(x)}{dx} \right]_{-a}^{a} - \int_{-a}^{a} \frac{dg(x)}{dx} \frac{df^*(x)}{dx} \, dx$$

and again,

$$(Lf, g) = \left[f^*(x) \frac{dg(x)}{dx} \right]_{-a}^{a} - \left[\frac{dg(x)}{dx} f^*(x) \right]_{-a}^{a} + \int_{-a}^{a} g(x) \frac{d^2 f^*(x)}{dx^2} \, dx$$

gives us an equation that can be manipulated into the form of our original operator definition.

$$(Lf, g) = f^*(a) \frac{dg(x)}{dx} \bigg|_{a} - f^*(-a) \frac{dg(x)}{dx} \bigg|_{-a} - \frac{dg(x)}{dx} \bigg|_{a} f^*(a) + \frac{dg(x)}{dx} \bigg|_{-a} f^*(-a) + \int_{-a}^{a} g(x) \frac{d^2 f^*(x)}{dx^2} \, dx$$

Given our boundary conditions, $f(\pm a) = 0$, we can say that $f^*(\pm a) = 0$. Then,

$$(Lf, g) = \int_{-a}^{a} g(x) \frac{d^2 f^*(x)}{dx^2} \, dx$$
$$= \int_{-a}^{a} \frac{d^2 f^*(x)}{dx^2} g(x) \, dx$$
$$(Lf, g) = (Lf, g)$$

□

Problem 7

The diffusion equation describing this one-dimensional slab is (assuming constant diffusion coefficients and cross sections in each region)

Fuel:

$$-D_F \frac{d^2}{dx^2} \phi(x) + \Sigma_{a,F} \phi(x) = 0$$

Moderator:

$$-D_M \frac{d^2}{dx^2} \phi(x) + \Sigma_{a,M} \phi(x) = S_0$$

Boundary Conditions:

$$\phi_F(a) = \phi_M(a) \qquad \text{(interface condition)}$$
$$\vec{J}_M \left(\pm \frac{a}{2} \pm b \right) = 0 \quad \left(\text{or } \phi \left(\pm \frac{a}{2} \pm \tilde{b} \right) = 0 \right) \qquad \text{(effective vacuum boundary condition)}$$
$$\frac{d}{dx} \phi_F \bigg|_{x=0} = 0 \qquad \text{(symmetry condition)}$$

We can then solve for the flux in each region. In the fuel,

$$\frac{d^2}{dx^2} \phi_F(x) - \frac{1}{L_F^2} \phi(x) = 0$$

where $L_F = \sqrt{\frac{D_F}{\Sigma_{a,F}}}$. The solution to this differential equation is of the form

$$\phi_F(x) = A_1 e^{\frac{x}{L_F}} + A_2 e^{-\frac{x}{L_F}}$$

Using our symmetry condition,

$$\frac{d\phi_F(x)}{dx} = \frac{A_1}{L_F}e^{\frac{x}{L_F}} - \frac{A_2}{L_F}e^{-\frac{x}{L_F}}$$

$$0 = \frac{A_1}{L_F} - \frac{A_2}{L_F}$$

$$A_1 = A_2 = A_F$$

Then $\phi(x)$ becomes

$$\phi_F(x) = A_F\left(e^{\frac{x}{L_F}} + e^{-\frac{x}{L_F}}\right)$$

$$\phi_F(x) = A_F\cosh\left(\frac{x}{L_F}\right)$$

In the moderator,

$$\frac{d^2}{dx^2}\phi(x) - \frac{1}{L_M^2}\phi(x) = -\frac{S_0}{D_M}$$

where $L_M = \sqrt{\frac{D_M}{\Sigma_{a,M}}}$. Like the solution in the fuel, the homogeneous solution to this differential equation is of the form

$$\phi_{\mathrm{h}}(x) = A_3 e^{\frac{x}{L_M}} + A_4 e^{-\frac{x}{L_M}}$$

while the particular solution is

$$\phi_{\mathrm{p}}(x) = \frac{S_0 L_M^2}{D_M}.$$

The general solution in the moderator is then,

$$\phi_M(x) = A_3 e^{\frac{x}{L_M}} + A_4 e^{-\frac{x}{L_M}} + \frac{S_0 L_M^2}{D_M}$$

Imposing our boundary conditions,

$$\phi_M(\frac{a}{2} + \tilde{b}) = A_3 e^{\frac{\frac{a}{2}+\tilde{b}}{L_M}} + A_4 e^{-\frac{\frac{a}{2}+\tilde{b}}{L_M}} + \frac{S_0 L_M^2}{D_M} = 0$$

and

$$\phi_M(-\frac{a}{2} - \tilde{b}) = A_3 e^{-\frac{\frac{a}{2}+\tilde{b}}{L_M}} + A_4 e^{\frac{\frac{a}{2}+\tilde{b}}{L_M}} + \frac{S_0 L_M^2}{D_M} = 0.$$

Then,

$$A_3 e^{\frac{\frac{a}{2}+\tilde{b}}{L_M}} + A_4 e^{-\frac{\frac{a}{2}+\tilde{b}}{L_M}} + \frac{S_0 L_M^2}{D_M} = A_3 e^{-\frac{\frac{a}{2}+\tilde{b}}{L_M}} + A_4 e^{\frac{\frac{a}{2}+\tilde{b}}{L_M}} + \frac{S_0 L_M^2}{D_M}$$

and

$$A_3 = A_4 = A_M$$

so

$$\phi_M(x) = A_M\left(e^{\frac{x}{L_M}} + e^{-\frac{x}{L_M}}\right) + \frac{S_0 L_M^2}{D_M}$$

Again,

$$\phi_M(\frac{a}{2} + \tilde{b}) = 0 = A_M\left(e^{\frac{\frac{a}{2}+\tilde{b}}{L_M}} + e^{-\frac{\frac{a}{2}+\tilde{b}}{L_M}}\right) + \frac{S_0 L_M^2}{D_M}$$

$$A_M = \frac{-S_0 L_M^2}{D_M\left(e^{\frac{\frac{a}{2}+\tilde{b}}{L_M}} + e^{-\frac{\frac{a}{2}+\tilde{b}}{L_M}}\right)}$$

Then,

$$\phi_M(x) = \left(\frac{-S_0 L_M^2}{D_M\left(e^{\frac{\frac{a}{2}+\tilde{b}}{L_M}} + e^{-\frac{\frac{a}{2}+\tilde{b}}{L_M}}\right)}\right)\left(e^{\frac{x}{L_M}} + e^{-\frac{x}{L_M}}\right) + \frac{S_0 L_M^2}{D_M}$$

$$\phi_M(x) = \frac{-S_0 L_M^2 \cosh\left(\frac{x}{L_M}\right)}{D_M \cosh\left(\frac{\frac{a}{2}+\tilde{b}}{L_M}\right)} + \frac{S_0 L_M^2}{D_M}$$

$$\boxed{\phi_M(x) = \frac{S_0 L_M^2}{D_M}\left(1 - \frac{\cosh\left(\frac{x}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+\tilde{b}}{L_M}\right)}\right)}.$$

Taking this back to our equation for $\phi_F(x)$, and using our interface condition,

$$\phi_F(a) = \phi_M(a)$$

$$A_F \cosh\left(\frac{a}{L_F}\right) = \frac{S_0 L_M^2}{D_M}\left(1 - \frac{\cosh\left(\frac{a}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+\tilde{b}}{L_M}\right)}\right)$$

$$A_F = \frac{S_0 L_M^2}{D_M \cosh\left(\frac{a}{L_F}\right)}\left(1 - \frac{\cosh\left(\frac{a}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+\tilde{b}}{L_M}\right)}\right)$$

$$\boxed{\phi_F(x) = \frac{S_0 L_M^2 \cosh\left(\frac{x}{L_F}\right)}{D_M \cosh\left(\frac{a}{L_F}\right)}\left(1 - \frac{\cosh\left(\frac{a}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+\tilde{b}}{L_M}\right)}\right)}.$$

If we define f_s as the average flux in the fuel to the average flux in the cell, we find

$$f_s = \frac{\frac{1}{a}\int_{-a/2}^{a/2}\phi_F(x)\,dx}{\frac{1}{a+2b}\int_{-b}^{b}\phi(x)\,dx}.$$

We can recognize, due to symmetry, that

$$f_s = \frac{\frac{2}{a}\int_{0}^{a/2}\phi_F(x)\,dx}{\frac{2}{a+2b}\int_{0}^{b}\phi(x)\,dx}.$$

Then, we can write the denominator as

$$f_s = \frac{\frac{2}{a}\int_{0}^{a/2}\phi_F(x)\,dx}{\frac{2}{a+2b}\left[\int_{0}^{a/2}\phi_F(x)\,dx + \int_{a/2}^{b}\phi_M(x)\,dx\right]}$$

Substituting our flux expressions,

$$f_s = \frac{\frac{2}{a}\int_{0}^{a/2}\frac{S_0 L_M^2 \cosh\left(\frac{x}{L_F}\right)}{D_M \cosh\left(\frac{a}{L_F}\right)}\left(1 - \frac{\cosh\left(\frac{a}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+\tilde{b}}{L_M}\right)}\right)dx}{\frac{2}{a+2b}\left[\int_{0}^{a/2}\frac{S_0 L_M^2 \cosh\left(\frac{x}{L_F}\right)}{D_M \cosh\left(\frac{a}{L_F}\right)}\left(1 - \frac{\cosh\left(\frac{a}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+\tilde{b}}{L_M}\right)}\right)dx + \int_{a/2}^{b}\frac{S_0 L_M^2}{D_M}\left(1 - \frac{\cosh\left(\frac{x}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+\tilde{b}}{L_M}\right)}\right)dx\right]}$$

$$f_s = \frac{\frac{a+2b}{a\cosh\left(\frac{a}{L_F}\right)}\left(1 - \frac{\cosh\left(\frac{a}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+\tilde{b}}{L_M}\right)}\right)\int_{0}^{a/2}\cosh\left(\frac{x}{L_F}\right)dx}{\left[\frac{1}{\cosh\left(\frac{a}{L_F}\right)}\left(1 - \frac{\cosh\left(\frac{a}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+\tilde{b}}{L_M}\right)}\right)\int_{0}^{a/2}\cosh\left(\frac{x}{L_F}\right)dx + \int_{a/2}^{b}dx - \int_{a/2}^{b}\frac{\cosh\left(\frac{x}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+\tilde{b}}{L_M}\right)}dx\right]}$$

$$f_s = \frac{\frac{a+2b}{a\cosh\left(\frac{a}{L_F}\right)}\left(1 - \frac{\cosh\left(\frac{a}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+b}{L_M}\right)}\right)\left[L_F\sinh\left(\frac{x}{L_F}\right)\right]_0^{a/2}}{\left[\frac{1}{\cosh\left(\frac{a}{L_F}\right)}\left(1 - \frac{\cosh\left(\frac{a}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+b}{L_M}\right)}\right)\left[L_F\sinh\left(\frac{x}{L_F}\right)\right]_0^{a/2} + [x]_{a/2}^b - \left[\frac{L_M}{\cosh\left(\frac{\frac{a}{2}+b}{L_M}\right)}\sinh\left(\frac{x}{L_M}\right)\right]_{a/2}^b\right]}$$

$$\boxed{f_s = \frac{\frac{L_F(a+2b)}{a\cosh\left(\frac{a}{L_F}\right)}\left(1 - \frac{\cosh\left(\frac{a}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+b}{L_M}\right)}\right)\sinh\left(\frac{a}{2L_F}\right)}{\frac{L_F}{\cosh\left(\frac{a}{L_F}\right)}\left(1 - \frac{\cosh\left(\frac{a}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+b}{L_M}\right)}\right)\sinh\left(\frac{a}{2L_F}\right) + b - \frac{a}{2} - \frac{L_M}{\cosh\left(\frac{\frac{a}{2}+b}{L_M}\right)}\sinh\left(\frac{b}{L_M}\right) + \frac{L_M}{\cosh\left(\frac{\frac{a}{2}+b}{L_M}\right)}\sinh\left(\frac{a}{2L_M}\right)}}$$

Problem 8

We are given the point source kernel,

$$G_{pt}(\vec{r}, \vec{r}') = \frac{\exp\left(-\frac{|\vec{r}-\vec{r}'|}{L}\right)}{4\pi D|\vec{r}-\vec{r}'|}$$

and we know that the flux is found using this kernel as

$$\phi(\vec{r}) = \int G_{pt}(\vec{r}, \vec{r}')S_{pt}(\vec{r})d^3\vec{r}' = \int \frac{\exp\left(-\frac{|\vec{r}-\vec{r}'|}{L}\right)}{4\pi D|\vec{r}-\vec{r}'|}S_{pt}(\vec{r}')d^3\vec{r}'$$

Problem 9

a.)

The neutron diffusion equation for a bare slab of nonmultiplying material containing a uniformly distributed neutron source is

$$\frac{d^2}{dx^2}\phi(x) - \frac{1}{L^2}\phi(x) = -\frac{s}{D}$$

where $L = \sqrt{\frac{D}{\Sigma_a}}$, and s is the source per volume at a given location. The homogeneous solutions to this differential equation are of the form

$$\phi_h(x) = A_1 e^{\frac{x}{L}} + A_2 e^{\frac{-x}{L}}$$

and the particular solution is

$$\phi_p(x) = \frac{sL^2}{D}$$

Combining these two solutions into a single general solution gives

$$\phi(x) = A_1 e^{\frac{x}{L}} + A_2 e^{\frac{-x}{L}} + \frac{sL^2}{D}$$

Given a bare slab of width $2a$, the flux boundary conditions are

$$\phi(\pm\tilde{a}) = 0.$$

We can use this to simplify our general solution,

$$\phi(a) = A_1 e^{\frac{\tilde{a}}{L}} + A_2 e^{\frac{-\tilde{a}}{L}} + \frac{sL^2}{D} = 0$$

$$\phi(-a) = A_1 e^{\frac{-\tilde{a}}{L}} + A_2 e^{\frac{\tilde{a}}{L}} + \frac{sL^2}{D} = 0$$

and so we can say that $A_1 = A_2 = A$. Then,

$$0 = A\left(e^{\frac{\tilde{a}}{L}} + e^{\frac{-\tilde{a}}{L}}\right) + \frac{sL^2}{D}$$

$$-\frac{sL^2}{D} = A\left(e^{\frac{\tilde{a}}{L}} + e^{\frac{-\tilde{a}}{L}}\right)$$

$$-\frac{sL^2}{D} = 2A\cosh\left(\frac{\tilde{a}}{L}\right)$$

$$A = -\frac{sL^2}{2D\cosh\left(\frac{\tilde{a}}{L}\right)}$$

Substituting this into our equation for the flux,

$$\phi(x) = \frac{sL^2}{D} - \frac{sL^2}{2D\cosh\left(\frac{\tilde{a}}{L}\right)}e^{x/L} - \frac{sL^2}{2D\cosh\left(\frac{\tilde{a}}{L}\right)}e^{-x/L}$$

$$= \frac{sL^2}{D}\left(1 - \frac{1}{2\cosh\left(\frac{\tilde{a}}{L}\right)}\left(e^{x/L} + e^{-x/L}\right)\right)$$

$$\boxed{\phi(x) = \frac{sL^2}{D}\left(1 - \frac{\cosh\left(\frac{x}{L}\right)}{\cosh\left(\frac{\tilde{a}}{L}\right)}\right)}$$

b.)

If we instead use eigenfunction expansions, we can express the flux as follows.

$$\phi(x) = \sum_{n=1}^{\infty} c_n\psi_n(x),$$

where each ψ_n is an eigenfunction in the expansion. Similarly, we can expand the source, s, in the eigenfunctions as well,

$$s = \sum_{n=1}^{\infty} s_n\psi_n(x).$$

The complete diffusion equation is then

$$\frac{d^2}{dx^2}\left[\sum_{n=1}^{\infty} c_n\psi_n(x)\right] - \frac{1}{L^2}\left[\sum_{n=1}^{\infty} c_n\psi_n(x)\right] = -\frac{1}{D}\sum_{n=1}^{\infty} s_n\psi_n(x).$$

This simplifies to

$$\sum_{n=1}^{\infty} c_n\frac{d^2}{dx^2}\psi_n(x) - \sum_{n=1}^{\infty} c_n\frac{1}{L^2}\psi_n(x) = -\frac{1}{D}\sum_{n=1}^{\infty} s_n\psi_n(x).$$

$$\sum_{n=1}^{\infty} c_n\left[\frac{d^2}{dx^2}\psi_n(x) - \frac{1}{L^2}\psi_n(x)\right] = -\frac{1}{D}\sum_{n=1}^{\infty} s_n\psi_n(x).$$

We can state that for any function ψ_n, the following equation is true for some value B_n.

$$\frac{d^2}{dx^2}\psi_n(x) + B_n^2\psi_n(x) = 0$$

Equivalently,

$$\frac{d^2}{dx^2}\psi_n(x) = -B_n^2\psi_n(x).$$

Substituting this equation into our simplified diffusion equation gives

$$\sum_{n=1}^{\infty} c_n \left[-B_n^2 \psi_n(x) - \frac{1}{L^2} \psi_n(x) \right] = -\frac{1}{D} \sum_{n=1}^{\infty} s_n \psi_n(x).$$

$$\sum_{n=1}^{\infty} c_n \left[B_n^2 + \frac{1}{L^2} \right] \psi_n(x) = \frac{1}{D} \sum_{n=1}^{\infty} s_n \psi_n(x).$$

Now, we multiply this equation by $\psi_m(x)$, an arbitrary specific eigenfunction, and then integrate over x.

$$\int_{-\infty}^{\infty} \left[\sum_{n=1}^{\infty} c_n \left[B_n^2 + \frac{1}{L^2} \right] \psi_n(x) \right] = \int_{-\infty}^{\infty} \left[\frac{1}{D} \sum_{n=1}^{\infty} s_n \psi_n(x) \right].$$

$$\int_{-\infty}^{\infty} \sum_{n=1}^{\infty} c_n \left[B_n^2 + \frac{1}{L^2} \right] \psi_n(x) \psi_m(x) dx = \int_{-\infty}^{\infty} \frac{1}{D} \sum_{n=1}^{\infty} s_n \psi_n(x) \psi_m(x) dx.$$

Exploiting the fact that eigenfunctions are orthogonal in these inner products, gives

$$\int_{-\infty}^{\infty} \psi_n(x) \psi_m(x) = \begin{cases} 1, & \text{if } n = m \\ 0, & \text{if } n \neq m \end{cases}$$

so for each $m \in n$

$$c_m \left(B_m^2 + \frac{1}{L^2} \right) = \frac{s_m}{D}$$

$$c_m = \frac{s_m}{D \left(B_m^2 + \frac{1}{L^2} \right)}$$

or, in general,

$$c_n = \frac{s_n}{D \left(B_n^2 + \frac{1}{L^2} \right)}.$$

Now we can use this in our flux equation,

$$\phi(x) = \sum_{n=1}^{\infty} c_n \psi_n(x),$$

$$\phi(x) = \sum_{n=1}^{\infty} \frac{s_n}{D \left(B_n^2 + \frac{1}{L^2} \right)} \psi_n(x).$$

From our original equation defining B_n^2, we can see that

$$\psi_n(x) = A_1 \cos(B_n x) + A_2 \sin(B_n x)$$

Substituting this into our previous equation for the flux,

$$\phi(x) = \sum_{n=1}^{\infty} \frac{s_n}{D \left(B_n^2 + \frac{1}{L^2} \right)} \left(A_1 \cos(B_n x) + A_2 \sin(B_n x) \right).$$

Where the flux must be symmetrical across the origin, we can deduce $A_2 = 0$. Then, if $A_1 = A$

$$\phi(x) = \sum_{n=1}^{\infty} \frac{A s_n \cos(B_n x)}{D \left(B_n^2 + \frac{1}{L^2} \right)}.$$

We can now use our boundary conditions, that

$$\phi(\pm \tilde{a}) = 0$$

and say

$$\phi(\tilde{a}) = \sum_{n=1}^{\infty} \frac{A s_n \cos(B_n \tilde{a})}{D \left(B_n^2 + \frac{1}{L^2} \right)} = 0$$

$$\phi(-\tilde{a}) = \sum_{n=1}^{\infty} \frac{As_n \cos(-B_n \tilde{a})}{D\left(B_n^2 + \frac{1}{L^2}\right)} = 0.$$

Since $s_n \neq 0$ (trivial), these equations only hold true when the cosine is zero. In other words, when

$$B_n \tilde{a} = \frac{n\pi}{2} \text{ for odd } n$$

$$B_n = \frac{(2n-1)\pi}{2\tilde{a}} \text{ for integers } n \in [1, \infty)$$

$$\phi(x) = \sum_{n=1}^{\infty} \frac{As_n \cos\left(\frac{(2n-1)\pi}{2\tilde{a}} x\right)}{D\left(\left(\frac{(2n-1)\pi}{2\tilde{a}}\right)^2 + \frac{1}{L^2}\right)}.$$

We can also solve for s_n, using the relationship that

$$s_n = \frac{2}{\tilde{a}} \int_{-\tilde{a}}^{\tilde{a}} s\psi_n(x) dx$$

$$s_n = \frac{2sA}{\tilde{a}} \int_{-\tilde{a}}^{\tilde{a}} \cos\left(\frac{(2n-1)\pi}{2\tilde{a}} x\right)$$

$$s_n = \frac{4sA}{(2n-1)\pi} \left[\sin\left(\frac{(2n-1)\pi}{2\tilde{a}} x\right)\right]_{-\tilde{a}}^{\tilde{a}}$$

$$s_n = \frac{4sA}{(2n-1)\pi} \left[\sin\left(\frac{(2n-1)\pi}{2}\right) - \sin\left(\frac{(1-2n)\pi}{2}\right)\right]$$

$$s_n = \frac{8sA}{(2n-1)\pi}(-1)^{n-1}$$

The flux is then,

$$\phi(x) = \frac{8sA^2}{D\pi} \sum_{n=1}^{\infty} \frac{(-1)^{n-1}}{2n-1} \frac{\cos\left(\frac{(2n-1)\pi}{2\tilde{a}} x\right)}{\left(\frac{(2n-1)\pi}{2\tilde{a}}\right)^2 + \frac{1}{L^2}}.$$

...

Problem 10

a.) Sphere

Given the general diffusion equation,

$$-D\nabla^2 \phi(\vec{r}) + \Sigma_a \phi(\vec{r}) = \nu\Sigma_f \phi(\vec{r}),$$

We can use the definition of the Laplacian in spherical polar coordinates

$$\nabla^2 = \frac{1}{r^2} \frac{\partial}{\partial r}\left(r^2 \frac{\partial}{\partial r}\right) + \frac{1}{r^2 \sin^2 \varphi} \frac{\partial^2}{\partial \theta^2} + \frac{1}{r^2 \sin \varphi} \frac{\partial}{\partial \varphi}\left(\sin \varphi \frac{\partial}{\partial \varphi}\right)$$

to give the diffusion equation for spherical polar coordinates as

$$\frac{1}{r^2} \frac{\partial}{\partial r}\left(r^2 \frac{\partial}{\partial r}\phi(\vec{r})\right) + \frac{1}{r^2 \sin^2 \varphi} \frac{\partial^2}{\partial \theta^2}\phi(\vec{r}) + \frac{1}{r^2 \sin \varphi} \frac{\partial}{\partial \varphi}\left(\sin \varphi \frac{\partial}{\partial \varphi}\phi(\vec{r})\right) + B^2 \phi(\vec{r}) = 0,$$

where B is the material buckling,

$$B \equiv \frac{\nu\Sigma_f - \Sigma_a}{D}.$$

Since we have a sphere, dependence of ϕ on φ and θ vanishes, so the equation reduces significantly to

$$\frac{1}{r^2}\frac{d}{dr}\left(r^2\frac{d}{dr}\phi(r)\right) + B^2\phi(r) = 0.$$

Where solutions to this equation are of the form

$$\phi(r) = \frac{A_1}{r}\cos(Br) + \frac{A_2}{r}\sin(Br)$$

Since our flux must be symmetric, $\phi(r) = \phi(-r)$, and so we see that $A_1 = 0$. Letting $A = A_2$,

$$\phi(r) = \frac{A}{r}\sin(Br).$$

Now, if the spherical reactor has a radius R, then a boundary condition is

$$\phi(\tilde{R}) = 0.$$

This gives

$$0 = \frac{A}{r}\sin(B\tilde{R})$$

and we see that

$$B\tilde{R} = n\pi \text{ for } n = 1, 2, 3, ...$$
$$B_g = B = \frac{n\pi}{\tilde{R}} \text{ for } n = 1, 2, 3, ...$$

We note however, that any values for n other than $n = 1$ will result in negative flux, which is unphysical. Then

$$\boxed{B_g = \frac{\pi}{\tilde{R}}}$$

$$\boxed{\phi(r) = \frac{A}{r}\sin\left(\frac{\pi}{\tilde{R}}r\right)}$$

b.) **Infinite Cylinder**

For cylindrical geometries, the diffusion equation takes the form:

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\phi(\vec{r})}{\partial r}\right) + \frac{\partial^2\phi(\vec{r})}{\partial z^2} + B^2\phi(\vec{r}) = 0$$

where again $B^2 = \frac{\nu\Sigma_f - \Sigma_a}{D}$ near criticality. Since our cylinder is infinite we again lose dependence on z in our equations.

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\phi(r)}{\partial r}\right) + B^2\phi(r) = 0$$

Solutions to this equation are are zeroth order Bessel functions of first and second kind, $J_0(r)$ and $Y_0(r)$. Then,

$$\phi(r) = A_1 J_0(Br) + A_2 Y_0(Br)$$

Our boundary conditions demand that the flux is finite at $r = 0$, and so $A_2 = 0$. Letting $A_1 = A$, we have

$$\phi(r) = A J_0(Br).$$

Our boundary conditions force the flux to go to zero at $\tilde{R}$,

$$\phi(\tilde{R}) = 0,$$

and again, since the flux must be nonnegative, only the first zero of J_0 can satisfy this condition, which occurs at $B\tilde{R} = 2.4048$. Our flux is then

$$\phi(r) = A J_0(\frac{2.4048}{\tilde{R}}r).$$

$$\boxed{B = \frac{2.4048}{\tilde{R}}}$$

$$\boxed{\phi(r) = A J_0(\frac{2.4048}{\tilde{R}}r)}$$

c.) Parallelpiped

For rectangular geometries, the diffusion equation is

$$\frac{\partial^2}{\partial x^2}\phi(x,y,z) + \frac{\partial^2}{\partial y^2}\phi(x,y,z) + \frac{\partial^2}{\partial z^2}\phi(x,y,z) + B^2\phi(x,y,z) = 0$$

. If we assume separability in each dimension, so that $\phi(x,y,z) = X(x)Y(y)Z(z)$, then

$$\frac{\partial^2}{\partial x^2}X(x)Y(y)Z(z) + \frac{\partial^2}{\partial y^2}X(x)Y(y)Z(z) + \frac{\partial^2}{\partial z^2}X(x)Y(y)Z(z) + B^2 X(x)Y(y)Z(z) = 0$$

.

$$Y(y)Z(z)\frac{\partial^2}{\partial x^2}X(x) + X(x)Z(z)\frac{\partial^2}{\partial y^2}Y(y) + X(x)Y(y)\frac{\partial^2}{\partial z^2}Z(z) + B^2 X(x)Y(y)Z(z) = 0.$$

and dividing by $\phi(x,y,z) = X(x)Y(y)Z(z)$ gives,

$$\frac{1}{X(x)}\frac{\partial^2}{\partial x^2}X(x) + \frac{1}{Y(y)}\frac{\partial^2}{\partial y^2}Y(y) + \frac{1}{Z(z)}\frac{\partial^2}{\partial z^2}Z(z) + B^2 = 0.$$

Where these three components in x, y, and z can be shown to be independent, we can write this equation as three separate ODEs:

$$\frac{1}{X(x)}\frac{\partial^2}{\partial x^2}X(x) + \alpha^2 = 0 \qquad\qquad \frac{1}{Y(y)}\frac{\partial^2}{\partial y^2}Y(y) + \beta^2 = 0 \qquad\qquad \frac{1}{Z(z)}\frac{\partial^2}{\partial z^2}Z(z) + \gamma^2 = 0$$

where $\alpha^2 + \beta^2 + \gamma^2 = B^2$

...

Problem 11

The "Kord Smith Challenge" was a challenge to simulate the local power in each fuel pin within a reactor assembly, using an axial decomposition of 100 regions and a radial decomposition of 10 regions, and where the statistical standard deviation for the power from each cell would be less than 1%. Simulations must also be able to take into account 100 different nuclides, and must be completed in 1 hour on a single CPU. In 2009, Bill Martin and Jan Eduard Hoogenboom proposed a benchmark problem for reference in completing the challenge.

In 2010, D.J. Kelly, *et al.* presented a solution to the Martin-Hoogenboom benchmark using the MC21 code. Despite solving the problem in the required granularity, the solution fell short of meeting many of the requirements set forth in the original challenge. Two of the major hurdles remaining at the time were the time scales needed (significantly longer than the single hour requirement) and large uncertainties (confidence intervals were much larger than prescribed). Memory concerns were also an issue.

Sources:
Romano, P. Parallel Algorithms for Monte Carlo Particle Transport Simulation on Exascale Computing Architectures. 2013.
Hoogenboom, J. *et al.* Monte Carlo Performance Benchmark for Detailed Power Density Calculation in a Full Size Core. 2011.

NE250_HW03_mnegus-prob4

October 15, 2017

1 NE 250 – Homework 3

1.1 Problem 4

10/20/2017

```
In [1]: import numpy as np
        old_settings = np.seterr(divide='raise')
```

a.) We are given the matrix:

$$\begin{bmatrix} 1 & 1 & -1 & 3 \\ 1 & 2 & -4 & -2 \\ 2 & 1 & 1 & 5 \\ -1 & 0 & -2 & -4 \end{bmatrix}$$

```
In [2]: A = np.array([[1,1,-1,3],[1,2,-4,-2],[2,1,1,5],[-1,0,-2,-4]])
```

The inverse, $\mathbf{A}^{-1}$ of a square matrix, $\mathbf{A}$, is equal to the adjugate of the matrix, $\mathbf{A}^{\dagger}$ divided by the determinant of $\mathbf{A}$.

$$\mathbf{A}^{-1} = \frac{\mathbf{A}^{\dagger}}{\det \mathbf{A}}$$

```
In [3]: def invert(matrix):
            adjugate_matrix = adjugate(matrix)
            det_matrix = determinant(matrix)
            try:
                inverse_matrix = adjugate_matrix/det_matrix
                return inverse_matrix
            except FloatingPointError:
                return 'The matrix has a determinant of zero; it is not invertible.
```

The adjugate of a square matrix, $\mathbf{A}^{\dagger}$, is the transpose of the cofactor matrix, $\mathbf{C_A}$.

$$\mathbf{A}^{\dagger} = \mathbf{C_A}^{T}$$

```
In [4]: def adjugate(matrix):
            cofactor_matrix = cofactor(matrix)
            adjugate_matrix = transpose(cofactor_matrix)
            return adjugate_matrix

        def transpose(matrix):
            transpose_matrix = np.empty_like(matrix)
            for i in range(len(matrix)):
                for j in range(len(matrix[0])):
                    transpose_matrix[j,i] = matrix[i,j]
            return transpose_matrix
```

The cofactor of a square matrix, $\mathbf{C_A}$ is the signed matrix of minors, $\mathbf{M_A}$.

$$\mathbf{C}_{\mathbf{A},ij} = (-1)^{i+j}\,\mathbf{M_A}$$

```
In [5]: def cofactor(matrix):
            minors_matrix = minors(matrix)
            cofactor_matrix = np.copy(minors_matrix)
            for i in range(len(cofactor_matrix)):
                for j in range(len(cofactor_matrix[0])):
                    cofactor_matrix[i,j] *= (-1)**(i+j)
            return cofactor_matrix
```

The matrix of minors of a square matrix, $\mathbf{M_A}$ is quite literally a matrix of the minors of $\mathbf{A}$.

```
In [6]: def minors(matrix):
            minors_matrix = np.empty_like(matrix)
            for i in range(len(minors_matrix)):
                for j in range(len(minors_matrix[0])):
                    minors_matrix[i,j] = minor(matrix,i,j)
            return minors_matrix
```

The minor of matrix element $\mathbf{A}_{ij}$ is the determinant of submatrix formed with the rows and columns other than i and j.

```
In [7]: def minor(matrix,i,j):
            submatrix = np.copy(matrix)
            submatrix = np.delete(submatrix,i,axis=0)
            submatrix = np.delete(submatrix,j,axis=1)
            minor_ij = determinant(submatrix)
            return minor_ij
```

Finally, the determinant of a matrix is either, $ad - bc$ for a 2×2 matrix, or the sum of signed minors in a row, i of square matrix of order > 2 mutiplied by the values of the minor's respective j.

```
In [8]: def determinant(matrix):
            assert len(matrix) == len(matrix[0])
```

```
        if len(matrix) == 2:
            return matrix[0,0]*matrix[1,1]-matrix[0,1]*matrix[1,0]
        else:
            signed_minors = []
            for j in range(len(matrix[0])):
                if (j+2)%2 == 1:
                    sign = -1
                else: sign = 1
                signed_minors.append(matrix[0,j]*sign*minor(matrix,0,j))
            return sum(signed_minors)
```

We can use this all together to find the inverse of **A**.

```
In [9]: print(invert(A))
```

```
The matrix has a determinant of zero; it is not invertible.
```

b.) We are given the matrix:

$$\begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix}$$

and we know that the eigenvalue λ and eigenvector $\vec{v}$ obey the rule

$$\begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix} \vec{v} = \lambda \vec{v}$$

Equivalently,

$$\left(\begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix} - \lambda I\!\!\!/ \right) \vec{v} = 0$$

We want the non-trivial solution to this equation, when $\vec{v} \neq \vec{0}$. $\vec{v} = \vec{0}$ when $\left(\begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix} - \lambda I\!\!\!/ \right)$ is invertible, so we will instead assert that $\left(\begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix} - \lambda I\!\!\!/ \right)$ is not invertible. By definition, this means

$$\det \left(\begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix} - \lambda I\!\!\!/ \right) = 0$$

or

$$\det \begin{bmatrix} 3 - \lambda & -1 \\ -1 & 3 - \lambda \end{bmatrix} = 0$$

We can solve this now for lambda:

$$(3 - \lambda)^2 - (-1)^2 = 0$$
$$9 - 6\lambda + \lambda^2 - 1 = 0$$
$$8 - 6\lambda + \lambda^2 = 0$$

and using the quadratic formula we find

$$\lambda = \frac{-(-6) \pm \sqrt{(-6)^2 - 4(8)}}{2}$$

$$\lambda = \frac{6 \pm \sqrt{36 - 32}}{2}$$

$$\lambda = \frac{6 \pm 2}{2}$$

$$\lambda = 3 \pm 1$$

$$\boxed{\lambda = 2, 4}$$

We can use this eigenvalue to solve for $\vec{v}$.

$$\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \vec{v} = 0 \text{ and } \begin{bmatrix} -1 & -1 \\ -1 & -1 \end{bmatrix} \vec{v} = 0$$

This gives the equations

$$\lambda = 2 : \begin{cases} v_1 - v_2 = 0 \\ v_2 - v_1 = 0 \end{cases} \qquad \lambda = 4 : \begin{cases} -v_1 - v_2 = 0 \end{cases}$$

For $\boxed{\lambda = 2, \ \vec{v} = \begin{bmatrix} v_0 \\ v_0 \end{bmatrix}}$, and for $\boxed{\lambda = 4, \ \vec{v} = \begin{bmatrix} v_0 \\ -v_0 \end{bmatrix}}$.

NE250_HW03_mnegus-prob5

October 15, 2017

1 NE 250 – Homework 3

1.1 Problem 5

10/20/2017

```
In [1]: import numpy as np
```

The inverse, $\mathbf{A}^{-1}$ of a square matrix, $\mathbf{A}$, is equal to the adjugate of the matrix, $\mathbf{A}^{\dagger}$ divided by the determinant of $\mathbf{A}$.

$$\mathbf{A}^{-1} = \frac{\mathbf{A}^{\dagger}}{\det \mathbf{A}}$$

We can manipulate this expression to find

$$(\det \mathbf{A})\mathbb{K} = \mathbf{A}^{\dagger}\mathbf{A}$$

Since we are looking for a self-adjugate matrix, $\mathbf{A}^{\dagger} = \mathbf{A}$, and

$$(\det \mathbf{A})\mathbb{K} = \mathbf{A}^2.$$

Then, taking the square root of both sides,

$$\left(\sqrt{\det \mathbf{A}}\right)\mathbb{K} = \mathbf{A}.$$

$$\mathbf{A} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} = \begin{bmatrix} \sqrt{\det \mathbf{A}} & 0 & 0 \\ 0 & \sqrt{\det \mathbf{A}} & 0 \\ 0 & 0 & \sqrt{\det \mathbf{A}} \end{bmatrix}.$$

$$a = e = i = \sqrt{\det \mathbf{A}}$$

$$\mathbf{A} = \begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{bmatrix} =$$

and

$$\det \mathbf{A} = a \begin{vmatrix} a & 0 \\ 0 & a \end{vmatrix}$$

$$\det \mathbf{A} = a(a^2)$$
$$\det \mathbf{A} = a^3.$$

We know that $a = \sqrt{\det \mathbf{A}}$, so
$$a = \sqrt{a^3}$$
$$a = a^{\frac{3}{2}}$$
$$a = 1$$
$$\boxed{\mathbf{A} = \mathbb{1}}$$

We can test this using the functions defined in problem 4.

```python
In [2]: def adjugate(matrix):
            cofactor_matrix = cofactor(matrix)
            adjugate_matrix = transpose(cofactor_matrix)
            return adjugate_matrix

        def cofactor(matrix):
            minors_matrix = minors(matrix)
            cofactor_matrix = np.copy(minors_matrix)
            for i in range(len(cofactor_matrix)):
                for j in range(len(cofactor_matrix[0])):
                    cofactor_matrix[i,j] *= (-1)**(i+j)
            return cofactor_matrix

        def transpose(matrix):
            transpose_matrix = np.empty_like(matrix)
            for i in range(len(matrix)):
                for j in range(len(matrix[0])):
                    transpose_matrix[j,i] = matrix[i,j]
            return transpose_matrix

        def minors(matrix):
            minors_matrix = np.empty_like(matrix)
            for i in range(len(minors_matrix)):
                for j in range(len(minors_matrix[0])):
                    minors_matrix[i,j] = minor(matrix,i,j)
            return minors_matrix

        def minor(matrix,i,j):
            submatrix = np.copy(matrix)
            submatrix = np.delete(submatrix,i,axis=0)
            submatrix = np.delete(submatrix,j,axis=1)
            minor_ij = determinant(submatrix)
            return minor_ij

        def determinant(matrix):
            assert len(matrix) == len(matrix[0])
```

```python
    if len(matrix) == 2:
        return matrix[0,0]*matrix[1,1]-matrix[0,1]*matrix[1,0]
    else:
        signed_minors = []
        for j in range(len(matrix[0])):
            if (j+2)%2 == 1:
                sign = -1
            else: sign = 1
            signed_minors.append(matrix[0,j]*sign*minor(matrix,0,j))
        return sum(signed_minors)
```

```python
In [3]: A = np.identity(3)
        print('A = \n',A)
        print('Adjugate of A = \n',adjugate(A))
```

```
A =
 [[ 1.  0.  0.]
 [ 0.  1.  0.]
 [ 0.  0.  1.]]
Adjugate of A =
 [[ 1. -0.  0.]
 [-0.  1. -0.]
 [ 0. -0.  1.]]
```