

Problem 1

We are given

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

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

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

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

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

Rearranging to solve for k ,

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

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

...

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

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

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 $\int_{4\pi} d\hat{\Omega} \Sigma_s(\hat{\Omega}' \rightarrow \hat{\Omega}) =$

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

$$\begin{aligned} \int_{4\pi} d\hat{\Omega} \left[\frac{1}{v} \frac{\partial \psi(\vec{r}, \hat{\Omega}, t)}{\partial t} + \hat{\Omega} \cdot \nabla \psi(\vec{r}, \hat{\Omega}, t) + \Sigma_t \psi(\vec{r}, \hat{\Omega}, t) \right] &= \int_{4\pi} d\hat{\Omega} \left[\int_{4\pi} d\hat{\Omega}' \Sigma_s(\hat{\Omega}' \rightarrow \hat{\Omega}) \psi(\vec{r}, \hat{\Omega}', t) \right. \\ &\quad \left. + \frac{\nu \Sigma_f}{4\pi} \int_{4\pi} d\hat{\Omega}' \psi(\vec{r}, \hat{\Omega}', t) \right] \end{aligned}$$

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

$$\frac{1}{v} \frac{\partial \phi(\vec{r}, t)}{\partial t} - D(\vec{r}) \nabla^2 \phi(\vec{r}, t) + \Sigma_t \phi(\vec{r}, t) = \Sigma_s \phi(\vec{r}, t) + \nu \Sigma_f \phi(\vec{r}, t)$$

Problem 3

Problem 5

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, $\mathbf{C}_\mathbf{A}$.

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

The cofactor of a square matrix, $\mathbf{C}_\mathbf{A}$ is the signed matrix of minors, $\mathbf{M}_\mathbf{A}$.

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

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 not invertible.

(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:

$$\begin{aligned} (3 - \lambda)^2 - (-1)^2 &= 0 \\ 9 - 6\lambda + \lambda^2 - 1 &= 0 \\ 8 - 6\lambda + \lambda^2 &= 0 \end{aligned}$$

and using the quadratic formula we find

$$\begin{aligned} \lambda &= \frac{-(-6) \pm \sqrt{(-6)^2 - 4(8)}}{2} \\ \lambda &= \frac{6 \pm \sqrt{36 - 32}}{2} \\ \lambda &= \frac{6 \pm 2}{2} \\ \lambda &= 3 \pm 1 \end{aligned}$$

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

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

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

This gives the equations

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

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

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

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

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

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

Then, taking the square root of both sides,

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

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

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

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

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

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:

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

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

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

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,

$$\begin{aligned} \frac{d\phi_F(x)}{dx} &= \frac{A_1}{L_F} e^{\frac{x}{L_F}} - \frac{A_2}{L_F} e^{-\frac{x}{L_F}} \\ 0 &= \frac{A_1}{L_F} - \frac{A_2}{L_F} \\ A_1 &= A_2 = A_F \end{aligned}$$

Then $\phi(x)$ becomes

$$\begin{aligned} \phi_F(x) &= A_F \left(e^{\frac{x}{L_F}} + e^{-\frac{x}{L_F}} \right) \\ \phi_F(x) &= A_F \cosh \left(\frac{x}{L_F} \right) \end{aligned}$$

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

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

while the particular solution is

$$\phi_p(x) = \frac{S_0 L_M^2}{D_M}.$$

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,

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

and

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

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,

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

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

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,

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

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

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

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

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,

$$\begin{aligned} f_s &= \frac{\frac{2}{a} \int_0^{a/2} \frac{S_0 L_M^2 \cosh\left(\frac{x}{L_F}\right)}{D_M \cosh\left(\frac{a}{L_F}\right)} \left(1 - \frac{\cosh\left(\frac{a}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+b}{L_M}\right)}\right) dx}{\frac{2}{a+2b} \left[\int_0^{a/2} \frac{S_0 L_M^2 \cosh\left(\frac{x}{L_F}\right)}{D_M \cosh\left(\frac{a}{L_F}\right)} \left(1 - \frac{\cosh\left(\frac{a}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+b}{L_M}\right)}\right) dx + \int_{a/2}^b \frac{S_0 L_M^2}{D_M} \left(1 - \frac{\cosh\left(\frac{x}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+b}{L_M}\right)}\right) dx \right]} \\ f_s &= \frac{\frac{a+2b}{a \cosh\left(\frac{a}{L_F}\right)} \left(1 - \frac{\cosh\left(\frac{a}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+b}{L_M}\right)}\right) \int_0^{a/2} \cosh\left(\frac{x}{L_F}\right) dx}{\left[\frac{1}{\cosh\left(\frac{a}{L_F}\right)} \left(1 - \frac{\cosh\left(\frac{a}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+b}{L_M}\right)}\right) \int_0^{a/2} \cosh\left(\frac{x}{L_F}\right) dx + \int_{a/2}^b dx - \int_{a/2}^b \frac{\cosh\left(\frac{x}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+b}{L_M}\right)} dx \right]} \\ f_s &= \frac{\frac{a+2b}{a \cosh\left(\frac{a}{L_F}\right)} \left(1 - \frac{\cosh\left(\frac{a}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+b}{L_M}\right)}\right) \left[L_F \sinh\left(\frac{x}{L_F}\right) \right]_0^{a/2}}{\left[\frac{1}{\cosh\left(\frac{a}{L_F}\right)} \left(1 - \frac{\cosh\left(\frac{a}{L_M}\right)}{\cosh\left(\frac{\frac{a}{2}+b}{L_M}\right)}\right) \left[L_F \sinh\left(\frac{x}{L_F}\right) \right]_0^{a/2} + [x]_{a/2}^b - \left[\frac{L_M}{\cosh\left(\frac{\frac{a}{2}+b}{L_M}\right)} \sinh\left(\frac{x}{L_M}\right) \right]_{a/2}^b \right]} \end{aligned}$$

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

Problem 8

Problem 9

Problem 10

Problem 11

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, $\mathbf{C}_\mathbf{A}$.

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

```
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 .

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

```
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 multiplied 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,

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

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

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

or

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

We can solve this now for lambda:

$$\begin{aligned} (3-\lambda)^2 - (-1)^2 &= 0 \\ 9 - 6\lambda + \lambda^2 - 1 &= 0 \\ 8 - 6\lambda + \lambda^2 &= 0 \end{aligned}$$

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} .

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

This gives the equations

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

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

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

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

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

Then, taking the square root of both sides,

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

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

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

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

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

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

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

$$a = 1$$

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

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

```
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])
```

```

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)

```

```

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.]]

```