NUEN 629
Numerical Methods in Reactor Analysis
Homework 4 & 5 & Project

Due on:
Thursday, November 19, 2015
&
Thursday, December 3, 2015

Dr. McClarren

Paul Mendoza

Contents

Homework 4 Problem Statement

Solve the following problem and submit a detailed report, including a justification of why a reader should believe your results and a description of your methods and iteration strategies.

1. (150 points + 50 points extra credit) In class we discussed the diamond-difference spatial discretization. Another discretization is the step discretization (this has several other names from other disciplines). It writes the discrete ordinates equations with isotropic scattering as, for $\mu_n > 0$ to

$$\mu_n \frac{\psi_{i,n} - \psi_{i-1,n}}{h_x} + \Sigma_t \psi_{i,n} = \frac{\Sigma_s}{2}\phi_i + \frac{Q}{2} \tag{1}$$

and for $\mu_n < 0$

$$\mu_n \frac{\psi_{i+1,n} - \psi_{i,n}}{h_x} + \Sigma_t \psi_{i,n} = \frac{\Sigma_s}{2}\phi_i + \frac{Q}{2} \tag{2}$$

The codes provided in class should be modified to implement this discretization.

(a) (50 Points) Your task (should you choose to accept it) is to solve a problem with uniform source of $Q = 0.01$, $\Sigma_t = \Sigma_s = 100$ for a slab in vacuum of width 10 using step and diamond difference discretizations. Use, 10, 50, and 100 zones ($h_x = 1, 0.02, 0.01$) and your expert choice of angular quadratures. Discuss your results and how the two methods compare at each number of zones.

(b) (10 points) Discuss why there is a different form of the discretization for the different signs of μ.

(c) (40 points) Plot the error after each iteration using a 0 initial guess for the step discretization with source iteration and GMRES.

(d) (50 points) Solve Reed's problem (see finite difference diffusion codes). Present convergence plots for the solution in space and angle to a "refined" solution in space and angle.

(e) (50 points extra credit) Solve a time dependant problem for a slab surrounded by vacuum with $\Sigma_t = \Sigma_s = 1$ and initial condition given by $\boldsymbol{\psi(0) = 1/h_x}$ (original problem statement said $\phi(0) = 1/h_x$ and I'm not sure how to solve that). Plot the solution at $t = 1$ s, using step and diamond difference. The particles have a speed of 1 cm/s. Which discretization is better with a small time step? What do you see with a small number of ordinates compared to a really large number (100s)?

Homework 4 Problem Background

Due to the complicated nature of this course, I provided this background for the lay person (me), so that they might have some grounding for the solution and hopefully believe the results. It should be noted that most of this background information is copied from various points in Dr. McClarren's notes, and is in no way original. Anything intelligent in the following is due to this fact and for any errors, I blame myself.

Beginning with the weighty neutron transport equation.

$$\left(\frac{1}{v}\frac{\delta}{\delta t} + \hat{\Omega}\cdot\nabla + \Sigma_t\right)\psi = \int_0^\infty dE' \int_{4\pi} d\hat{\Omega}' \ K(\hat{\Omega}'\cdot\hat{\Omega}, v'\to v)\Sigma_s\psi + \frac{1}{4\pi}\chi\int_0^\infty dE' \ \bar{\nu}\Sigma_f\phi + q$$

Where $K(\hat{\Omega}'\cdot\hat{\Omega}, v'\to v)$ represents the probability of scattering from one angle and energy to another given a scattering event occured and Σ_s is the macroscopic scattering cross section. The dependencies for the variables are shown below.

$$\Sigma_t(\vec{x}, v, t)$$
$$\psi(\vec{x}, \hat{\Omega}, v, t)$$
$$\Sigma_s(\vec{x}, v, t)$$
$$\chi(\vec{x}, v)$$
$$\Sigma_f(\vec{x}, v, t)$$
$$\phi(\vec{x}, v, t)$$
$$q(\vec{x}, \hat{\Omega}, v, t)$$

There are 7 free variables (three spatial $[\vec{x}]$, two angular $[\hat{\Omega}]$, one energy $[v]$ and one time $[t]$) in this equation. In the steady state $\left(\frac{\delta\psi}{\delta t} = 0, \text{ i.e. no time dependence}\right)$, non fissioning ($\Sigma_f = 0$) case the transport equation reduces to,

$$\left(\hat{\Omega}\cdot\nabla + \Sigma_t\right)\psi = \int_0^\infty dE' \int_{4\pi} d\hat{\Omega}' \ K(\hat{\Omega}'\cdot\hat{\Omega}, v'\to v)\Sigma_s\psi + q.$$

In order to reduce this to a single energy the following definitions are helpful (remembering all time dependence is gone).

$$\psi(\vec{x}, \hat{\Omega}) = \int_0^\infty dE \ \psi(\vec{x}, \hat{\Omega}, v(E))$$

$$\Sigma_t(\vec{x}) = \frac{\int_0^\infty dE \ \Sigma_t(\vec{x}, v(E))\psi(\vec{x}, \hat{\Omega}, v(E))}{\psi(\vec{x}, \hat{\Omega})}$$

$$K(\hat{\Omega}'\cdot\hat{\Omega}, v'\to v) = K(\hat{\Omega}'\cdot\hat{\Omega})K(v'\to v)$$

$$\Sigma_s(\vec{x}) = \frac{\int_0^\infty dE \int_0^\infty dE' \ \Sigma_s(\vec{x}, v(E))K(v'\to v)\psi(\vec{x}, \hat{\Omega}, v(E))}{\psi(\vec{x}, \hat{\Omega})}$$

$$q(\vec{x}, \hat{\Omega}) = \int_0^\infty dE \ q(\vec{x}, \hat{\Omega}, v(E))$$

Using these definitions, integrating the transport equation over all energy, and assuming cross sections and sources do not vary in space or angle, our transport equation reduces again to,

$$\left(\hat{\Omega}\cdot\nabla + \Sigma_t\right)\psi(\vec{x}, \hat{\Omega}) = \int_{4\pi} d\hat{\Omega}' \ K(\hat{\Omega}'\cdot\hat{\Omega})\Sigma_s\psi(\vec{x}, \hat{\Omega}') + q.$$

Where the double differential was assumed to be separable in angle and energy. The final simplification for our problem will be in space. If we assume that our geometry is infinite in y $\left(\frac{\delta}{\delta y} = 0\right)$ and x $\left(\frac{\delta}{\delta x} = 0\right)$. This also means that ψ depends only on z and mu, and if we recall that

$$\hat{\Omega} = (\sqrt{1-\mu^2}cos(\rho), \sqrt{1-\mu^2}sin(\rho), \mu),$$

and

$$\nabla = \left(\frac{\delta}{\delta x}, \frac{\delta}{\delta y}, \frac{\delta}{\delta x}\right)$$

also assuming that

$$K(\hat{\Omega}' \cdot \hat{\Omega}) = \frac{1}{4\pi} \text{ Isotropic Scattering}$$

then our transport equation, and the equation I think we are trying to solve for this homework is.

$$\left(\mu\frac{\delta}{\delta z} + \Sigma_t\right)\psi(z,\mu) = \Sigma_s\frac{2\pi}{4\pi}\int_{-1}^{1}d\mu' \ \psi(z,\mu') + q.$$

Checking units,

$$\left(\mu\frac{\delta}{\delta z} + \Sigma_t\right)\left[\frac{1}{cm}\right]\psi(z,\mu)\left[\frac{n\cdot cm}{str\cdot cm^3\cdot s}\right] = \Sigma_s\frac{1}{2}\left[\frac{1}{cm\cdot rad}\right]\int_{-1}^{1}d\mu' \ \psi(z,\mu')\left[\frac{n\cdot cm}{rad\cdot cm^3\cdot s}\right] + q\left[\frac{n}{str\cdot cm^3\cdot s}\right].$$

Σ_s was moved outside the integral because it has no angular dependence integration over the azimuthal angle occured because $\psi(z,\hat{\Omega})$ is assumed to be uniform and not depend on that angle.

Using Gauss-Legendre Quadrature for the integration term

$$\phi = \int_{-1}^{1}d\mu'\psi(z,\mu') = \sum_{i=1}^{n}w_i\psi(z,\mu_i')$$

where

$$w_i = \frac{2}{(1-\mu_i^2)[P_n'(\mu_i)]^2}$$

P_n' is the differential of the legendre polynomial n, and μ_i' are the roots of P_n. The weights of even n's of the legendre polynomials should sum to 2, the value of $\int_{-1}^{1}d\mu$, which they do.

Putting this all together with time dependence:

$$\left(\frac{1}{v}\frac{\delta}{\delta t} + \mu\frac{\delta}{\delta z} + \Sigma_t\right)\psi_n(z) = \Sigma_s\frac{1}{2}\sum_{n'=1}^{N}w_{n'}\psi_{n'}(z) + q$$

Where n and n' denote the direction being solved for and N is the total number of angles being solved for. Also unites of w are rad.

Diamond difference discretization

$$\frac{1}{v}\frac{\psi_{n,i}^{\ell+1,j+1} - \psi_{n,i}^{L,j}}{\Delta t} + \mu_n\frac{\psi_{n,i+1/2}^{\ell+1,j+1} - \psi_{n,i-1/2}^{\ell+1,j+1}}{hz} + \Sigma_t\psi_{n,i}^{\ell+1,j+1} = \Sigma_s\frac{1}{2}\sum_{n'=1}^{N}w_i\psi_{n',i}^{\ell,j+1} + q.$$

Where n is for angle, i is the midplane of a spacial discretization, ℓ is the iteration index for spacial convergence, j is for a time step and

$$\psi_{n,i}^{\ell+1,j+1} = \frac{1}{2}(\psi_{n,i+1/2}^{\ell+1,j+1} + \psi_{n,i-1/2}^{\ell+1,j+1})$$

Writing this in terms of a steady state

$$\mu_n \frac{\psi_{n,i+1/2}^{\ell+1,j+1} - \psi_{n,i-1/2}^{\ell+1,j+1}}{hz} + \Sigma_t^* \psi_{n,i}^{\ell+1,j+1} = \Sigma_s \frac{1}{2} \sum_{n'=1}^{N} w_i \psi_{n',i}^{\ell,j+1} + q^*.$$

where

$$\Sigma_t^* = \Sigma_t + \frac{1}{v\Delta t}$$

$$q^* = q + \frac{\psi_{n,i}^{L,j}}{v\Delta t}$$

The above equation has L for the iteration index to indicate that its value was iteratively determined in the previous time step.

Step discretization

Writing this in terms of a steady state for $\mu > 0$

$$\mu_n \frac{\psi_{n,i}^{\ell+1,j+1} - \psi_{n,i-1}^{\ell+1,j+1}}{hz} + \Sigma_t^* \psi_{n,i}^{\ell+1,j+1} = \Sigma_s \frac{1}{2} \sum_{n'=1}^{N} w_i \psi_{n',i}^{\ell,j+1} + q^*.$$

and for $\mu < 0$

$$\mu_n \frac{\psi_{n,i+1}^{\ell+1,j+1} - \psi_{n,i}^{\ell+1,j+1}}{hz} + \Sigma_t^* \psi_{n,i}^{\ell+1,j+1} = \Sigma_s \frac{1}{2} \sum_{n'=1}^{N} w_i \psi_{n',i}^{\ell,j+1} + q^*.$$

where

$$\Sigma_t^* = \Sigma_t + \frac{1}{v\Delta t}$$

$$q^* = q + \frac{\psi_{n,i}^{L,j}}{v\Delta t}$$

GMRES

The generalized minimium residual (GMRES) method is an iterative method for solving linear systems of equations. The method approximates the solution by the vector in a Krylov subspace with a minimum residual (see wikipedia or Dr. McClarren's notes, I'm not really sure how this method works, but python has a solver for it).

The system $A\vec{\phi} = b$ is solved with GMRES, where for our situation,

$$A = \left(I - \sum_{n'=1}^{N} L^{-1} \Sigma_s \frac{1}{2} \right)$$

where L^{-1} is a sweep solve for our system and acts as an operator (I think), and

$$b = \sum_{n'=1}^{N} L^{-1} q^*$$

Reeds Problem

Reeds problem is a similiar system as above, except the source and scattering and total cross sections are variable in z, and the width of z is 16.

Homework 4 Problem Solution

The code for this problem will be at the end of this section. The answers are below.

(a) (50 Points) Your task is to solve a problem with uniform source of $Q = 0.01$, $\Sigma_t = \Sigma_s = 100$ for a slab in vacuum of width 10 using step and diamond difference discretizations. Use, 10, 50, and 100 zones ($h_x = 1, 0.02, 0.01$) and your expert choice of angular quadratures. Discuss your results and how the two methods compare at each number of zones.

The angular quadrature used was the Gauss-Legendre Quadrature because of the integration range. Its form was shown in the background section. The plot below was produced with the GMRES method, but the source iteration scheme produced the same results.
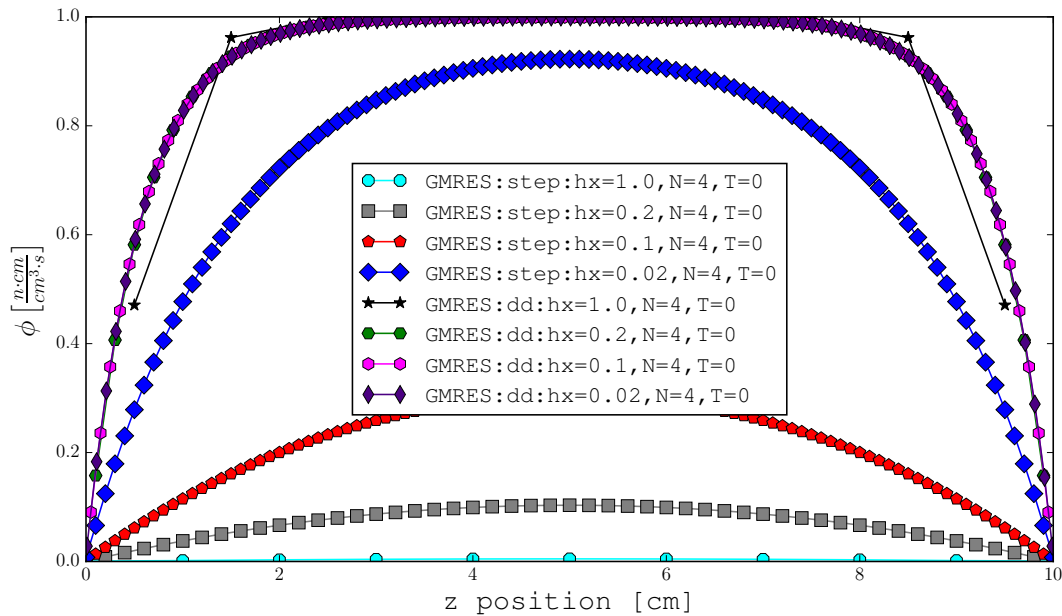
Both of the iterative solutions converged with max iterations of 100,000 and a slight modification on cross section ($\Sigma_t = \Sigma_t \cdot 1.0001$) to help the system converge. As the number of zones increased for the step solution, the flux magnitude kept increasing to match with the diamond difference and maintained a cosine(ish) shape. As the number of zones increased with the diamond difference, the shape started to converge towards the cosine, but maintained the proper magnitude.

Something else I would like to point out in the solution is that the step solution always had one more point plotted than the diamond difference. The reason for this is due to how each solution was solved. This is easier highlighted (for me) with an example, which is shown in the case where the number of zones is 10.

For the Diamond difference, the average locations (remember they were averaged), $\psi_{n,i}^{L,j+1}$, being solved for were,

$$z = [0.5, 1.5, 2.5, 3.5, 4.5, 5.5, 6.5, 7.5, 8.5, 9.5]$$

The points for $\psi_{n,i+1/2}^{L,j+1}$ and $\psi_{n,i-1/2}^{L,j+1}$ were at the points,

$$z = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$$

When sweeping to the right, $\psi_n(z = 0)$ was set to zero, because the incoming flux is zero, and all points were solved for up to where $z = 10$, and $\psi_{n,i}$ values were determined with averaging. This same thing occured when sweeping to the left (except here $\psi_n(z = 10)$ was set to zero). This would yield 10 values at the points $[0.5, 1.5, ..., 9.5]$.

For the step discretization scheme, the locations (non averaged), $\psi_{n,i}^{L,j+1}$, being solved for were,

$$x = \begin{cases} [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] & \mu > 0 \\ [0, 1, 2, 3, 4, 5, 6, 7, 8, 9], & \mu < 0 \end{cases}$$

When combining these two lists for ϕ, this was considered, and hence the step discretization scheme had one extra point (both lists have 10 points, but the location 10 is unqiue in the first list, and 0 in the second).

(b) (10 points) Discuss why there is a different form of the discretization for the different signs of μ.

The different forms are needed in the step discretization because in both the diamond and step approaches to the solution a value is needed from a previous zone. Our vacuum boundary condition states that the incoming neutrons are zero, which at the left side of the boundary, determines the angular flux moving to the right, and at the right side of the boundary, the angular flux moving to the left (these values are 0).

(c) (40 points) Plot the error after each iteration using a 0 initial guess for the step discretization with source iteration and GMRES.

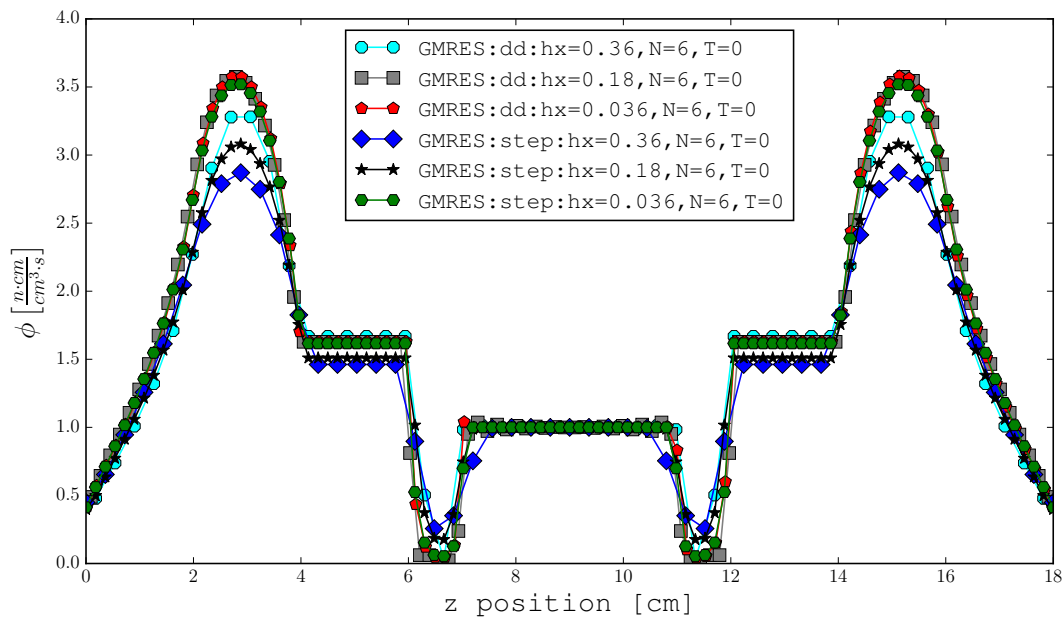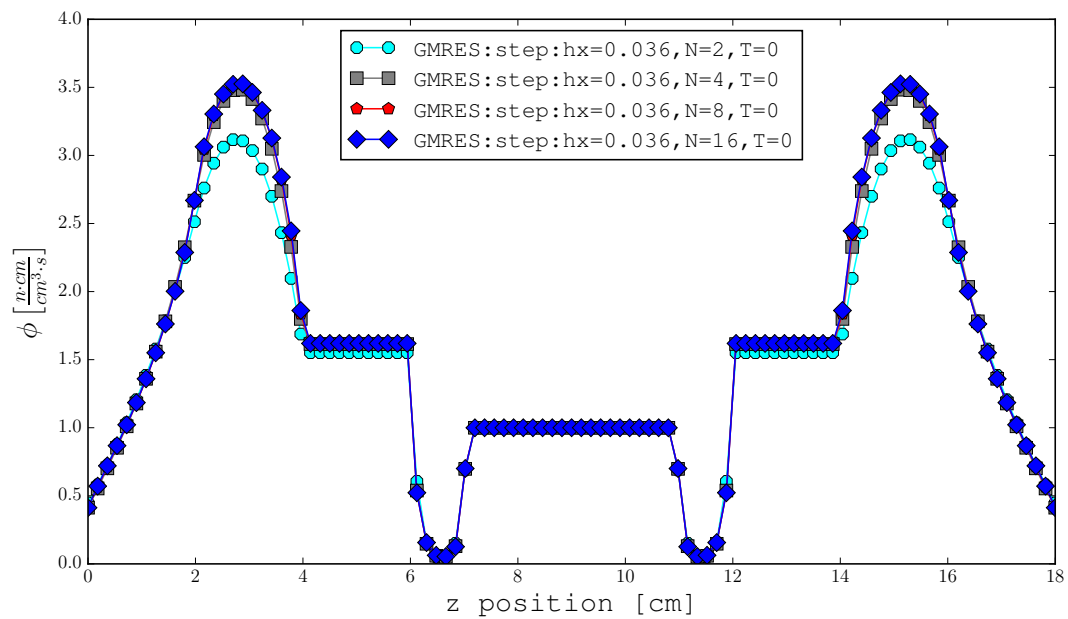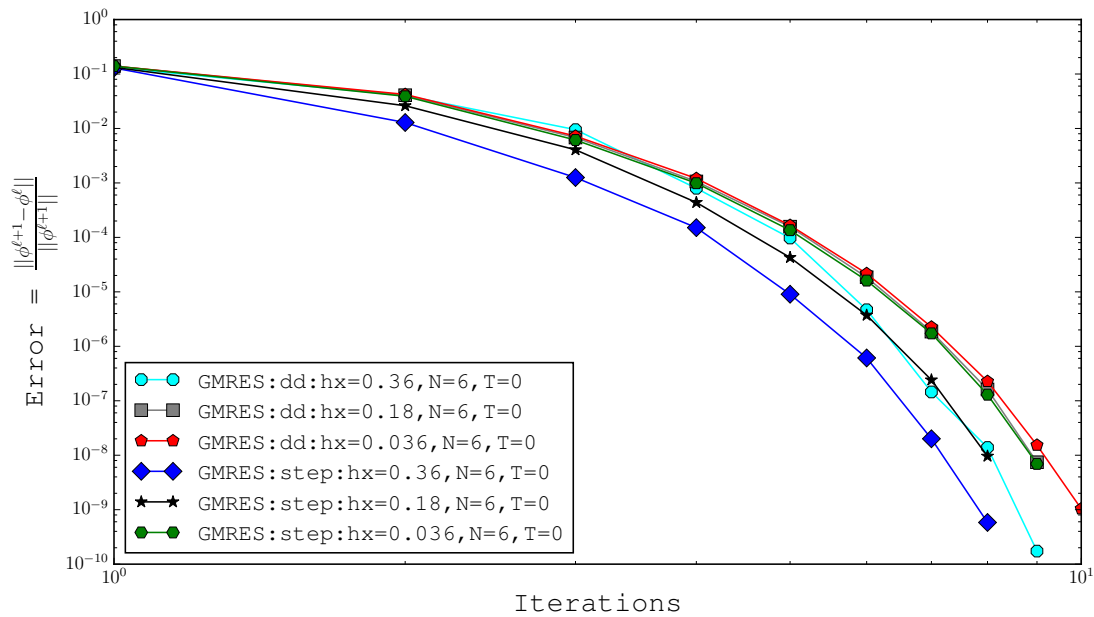**Error will be determined with the following:**

$$\text{Error} = \frac{||\phi^{\ell+1} - \phi^{\ell}||}{||\phi^{\ell+1}||}$$
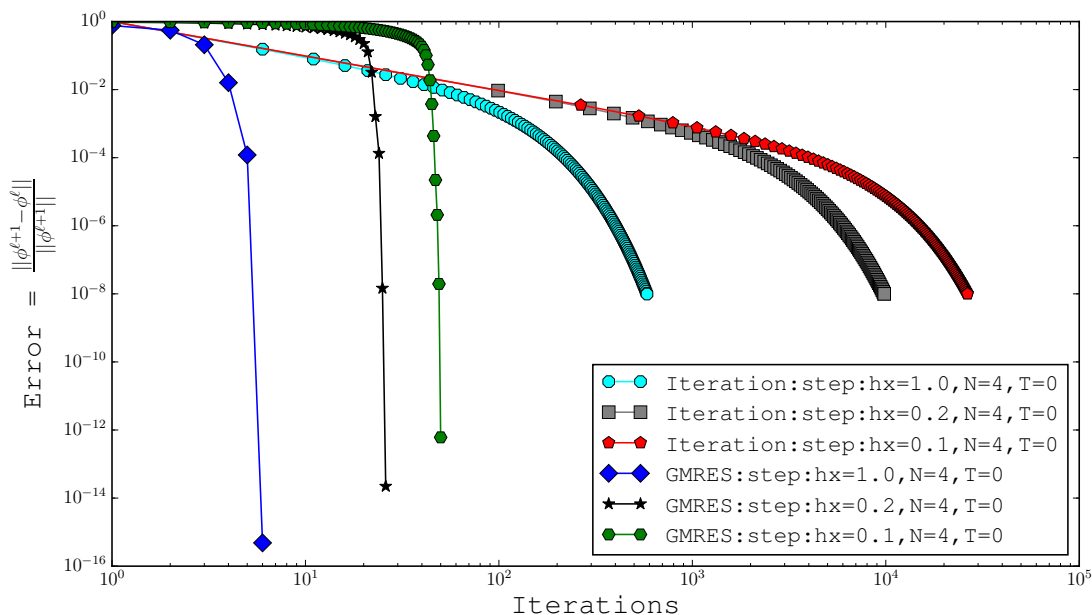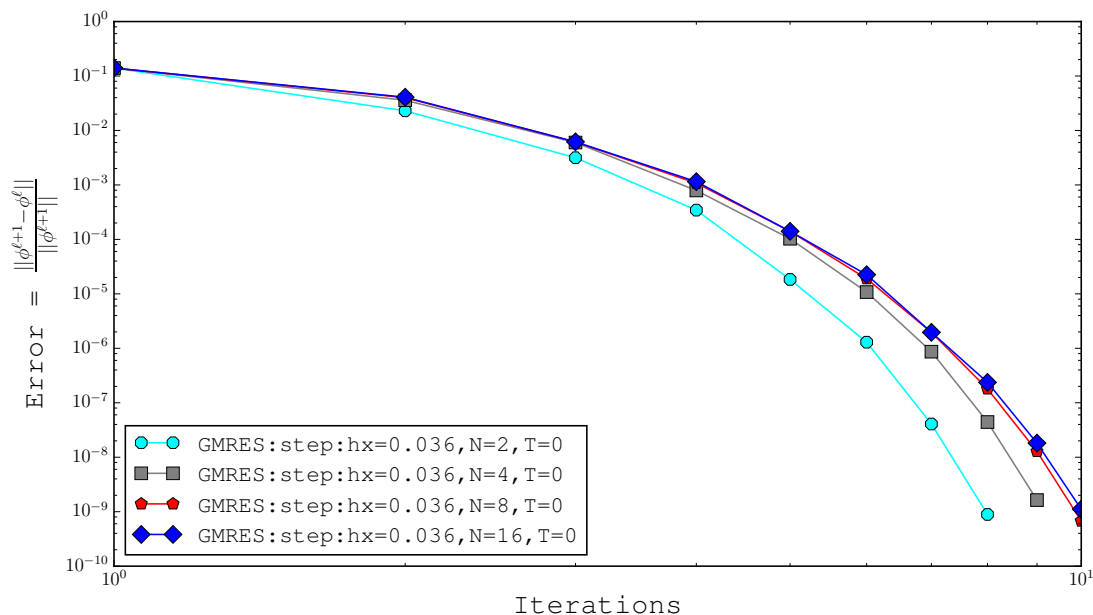
(d) (50 points) Solve Reed's problem (see finite difference diffusion codes). Present convergence plots for the solution in space and angle to a "refined" solution in space and angle.

Plots are below, reduced the number of points so that figures wouldn't take so long to load.

The solution converges with more spatial slices. Increasing the number of angular slices helps upto when N=4, but beyond that it doesn't do much.

(e) (50 points extra credit) Solve a time dependant problem for a slab surrounded by vacuum with $\Sigma_t = \Sigma_s = 1$ and initial condition given by $\boldsymbol{\psi(0) = 1/h_x}$ (original problem statement said $\phi(0) = 1/h_x$ and I'm not sure how to solve that). Plot the solution at $t = 1$ s, using step and diamond difference. The particles have a speed of 1 cm/s. Which discretization is better with a small time step? What do you see with a small number of ordinates compared to a really large number (100s)?

The Sweeps

Listing 1: **Diamond Difference Spatial Discretization**

```
###############################################################

def Sigma_tReed(r):
    value = 0 + ((1.0*(r>=14) + 1.0*(r<=4)) +
                 5.0 *((np.abs(r-11.5)<0.5) or (np.abs(r-6.5)<0.5)) +
                 50.0 * (np.abs(r-9)<=2) )
    return value;
def Sigma_aReed(r):
    value = 0 + (0.1*(r>=14) + 0.1*(r<=4) +
                 5.0 *((np.abs(r-11.5)<0.5) or (np.abs(r-6.5)<0.5)) +
                 50.0 * (np.abs(r-9)<=2) )
    return value;
def QReed(r):
    value = 0 + 1.0*((r<16) * (r>14))+ 1.0*((r>2) * (r<4)) + 50.0*(np.abs(r-9)<=2)
    return value;

def Timevector(T,dt):
    Time=[dt]
```

```
        while Time[-1]<T:
120         Time.append(Time[-1]+dt)
        return(Time)


    def diamond_sweep1D(I,hx,q,sigma_t,mu,boundary):
        """Compute a transport diamond difference sweep for a given
125     Inputs:
            I:              number of zones
            hx:             size of each zone
            q:              source array
            sigma_t:        array of total cross-sections
130         mu:             direction to sweep
```

Listing 2: **Step Discretization**

```
    Outputs:
        psi:              value of angular flux in each zone
        """
135     assert(np.abs(mu) > 1e-10)
        psi = np.zeros(I)
        ihx = 1./hx
        if (mu > 0):
            psi_left = boundary
140         for i in range(I):
                psi_right = (q[i] + (mu*ihx-0.5*sigma_t[i])*psi_left)\
                            /(0.5*sigma_t[i] + mu*ihx)
                psi[i] = 0.5*(psi_right + psi_left)
                psi_left = psi_right
145     else:
            psi_right = boundary
            for i in reversed(range(I)):
                psi_left = (q[i]+ (-mu*ihx-0.5*sigma_t[i])*psi_right)\
                            /(0.5*sigma_t[i] - mu*ihx)
150             psi[i] = 0.5*(psi_right + psi_left)
                psi_right = psi_left
        return psi


    def step_sweep1D(I,hx,q,sigma_t,mu,boundary):
155     """Compute a transport step sweep for a given
        Inputs:
            I:              number of zones
            hx:             size of each zone
            q:              source array
160         sigma_t:        array of total cross-sections
            mu:             direction to sweep
```

Main Program

Listing 3: Main Program

```
    #!/usr/bin/env python3


    ############################################################
```

```python
##################### Import packages #######################
#############################################################

import time
start_time = time.time()
import Functions as f


#############################################################
######################### Inputs ############################
#############################################################

# Constants
Q = 0.01
Sigma_t = 100;Sigma_s=100
# Add adsorption to help converge
if Sigma_t==Sigma_s:
    Sigma_t=Sigma_t*1.0001

# Geometry
L = 10.                        # Width of slab
slices=[10,50,100,500]         # Number of cuts in slab (looped)
N = 4                          # Number of angle slices
BCs = f.np.zeros(N)            # Zero incoming flux

#Time
T=0                            # total Time (A plot made at T)
dt=1                           # Time steps width
v=1                            # Velocity

MAXITS=100000                  # Max iterations for source iter
loud=False                     # Echo every Iteration?

#Method
Methods=['GMRES:step',         # 'Iteration' or 'GMRES'
         'GMRES:dd']        # Methods to solve with?
                               # 'step' or 'dd'

tol=1e-8

PlotError=False                # Do we plot the error?

NumOfPoints=100                # Max Number of points for plots


#############################################################
#################### Initialize Figures #####################
#############################################################

Check=0
fig=f.plt.figure(figsize=f.FigureSize)   # Plot all Methods
ax=fig.add_subplot(111)
if PlotError:
    erfig=f.plt.figure(figsize=f.FigureSize) # Err Plot
    erax=erfig.add_subplot(111)                 # at T=0
```

```
      ################################################################
60    #################### Calculations ############################
      ################################################################

      for Scheme in Methods:

65        Method=Scheme.split(':')[1]
          ########################################################
          #################### Set Up ############################
          ########################################################

70        for II in slices:
              if Method == 'step': #Step Dude needs one extra
                  I=II+1
              elif Method == 'dd':
                  I=II

75
              #Width, ang lists for materials
              hx = L/II
              q = f.np.ones(I)*Q
              Sig_t_discr = f.np.ones(I)*Sigma_t
80            Sig_s_discr = f.np.ones(I)*Sigma_s

              #Initialize psi (for time steps)
              if T==0:
                  psi=f.np.zeros((N,I))
85                Time=[0]
              else:
                  psi=f.np.ones((N,I))*(1/hx)
                  Time=f.Timevector(T,dt)

90            label_tmp=Scheme+":hx="+str(hx)+",N="+str(N)+",T="
              ########################################################
              #################### Determine phi ####################
              ########################################################

95            for t in Time: #Loop over time

                  label=label_tmp+str(t)

                  #Determine phi (new psi is determined for time steps)
100               x,phi,it,er,psi=f.solver(I,hx,q,Sig_t_discr,
                  Sig_s_discr,N,psi,v,dt,Time,BCs,Scheme,tol,MAXITS,loud)

                  ########################################################
                  ############### Plot Information ##################
```

Homework 5

$$\psi(\Omega, \bar{x}, t, E)$$

Project