PHY411 Report: Assignment 3

Name: Athira Sreejith
Roll no: MS18033
email: ms18033@iisermohali.ac.in
Date of submission: 25-10-2021

Contents

---

1 Introduction

The ionisation energy losses for 10 different particles through 10cm of copper is simulated and analysed. The minimum energy loss for muons and electrons are calculated and plotted as a function of atomic number and atomic mass.

2 Problem 1

The ionisation energy loss for 10 different types of particles (Electron, Muon, Tauon, Pion, Kaon, Proton, Neutron, Hydrogen, Deuterium, and Helium) incident on a 10 cm thick copper slab, is simulated. The incoming energies of the particles are drawn from a uniform distribution from 1 Mev to 10 GeV. Since the energy range is much below radiative loss regime, we can safely assume that the energy loss is only due to the ionization, and radiative effects are also neglected.

- There are 10,000 of each of these particles have passed through the detectors with uniformly distributed energies. i.e. 10 $\times$ 10,000 particles in total for above table.
- Use only Bethe-Block equation, don't use any aproximation.

2.1 Sub problem 1

2.1.1 Problem

Plot the energy spectrum of the particles.

2.1.2 Algorithm

1. Obtain the incoming energies of the incident particles by drawing 10,000 different random samples from a uniform distribution from 1 to 10,000.
2. Plot a histogram of the frequency distribution of the energies to obtain the energy spectrum.

2.1.3 Code

The code for this part of the sub problem has been attached with the next sub problem.

2.1.4 Results

Figure 1: Energy Spectrum of electron.png

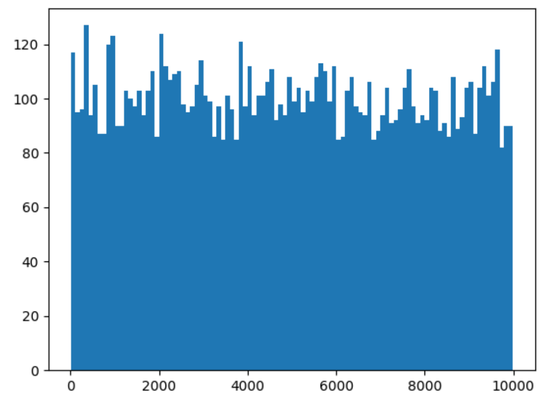


Figure 2: Energy Spectrum of muon.png

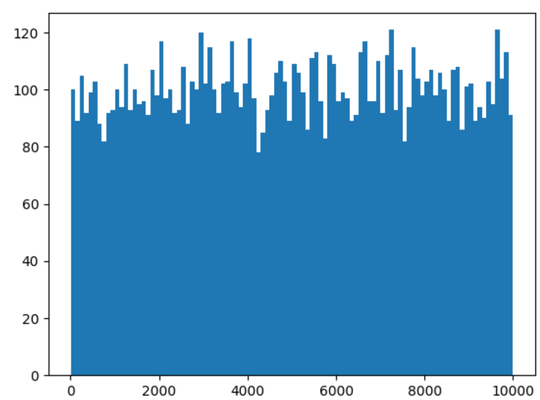


Figure 3: Energy Spectrum of tauon.png

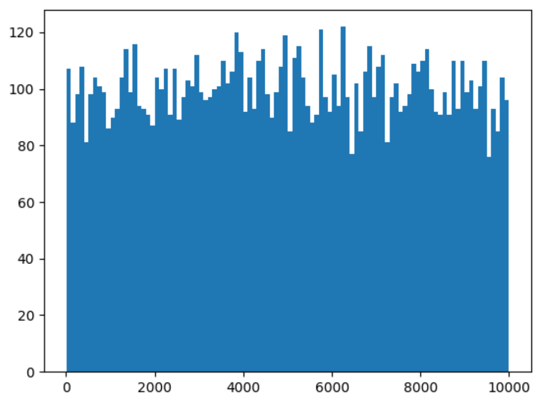


Figure 4: Energy Spectrum of pion.png

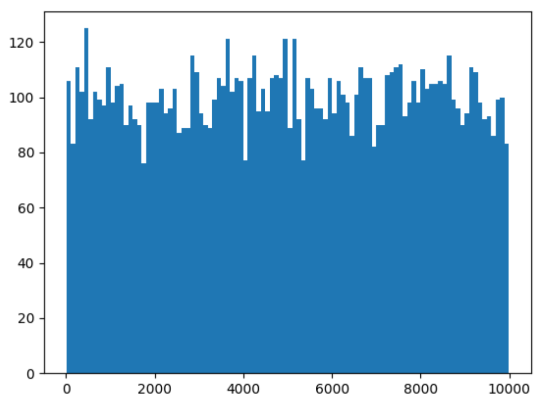


Figure 5: Energy Spectrum of kaon.png

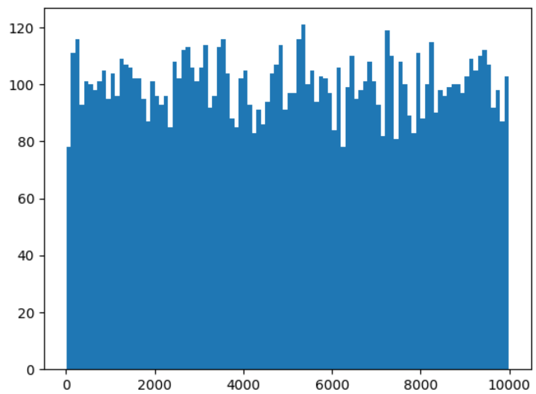


Figure 6: Energy Spectrum of proton.png

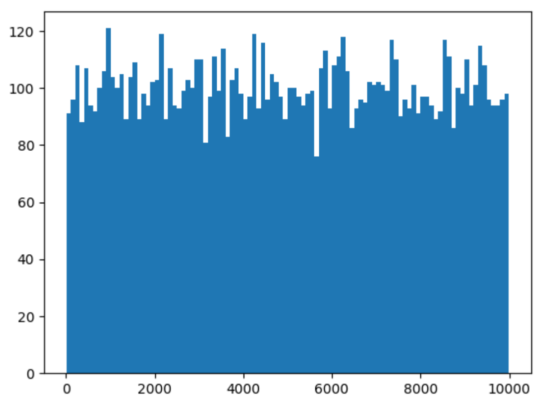


Figure 7: Energy Spectrum of neutron.png

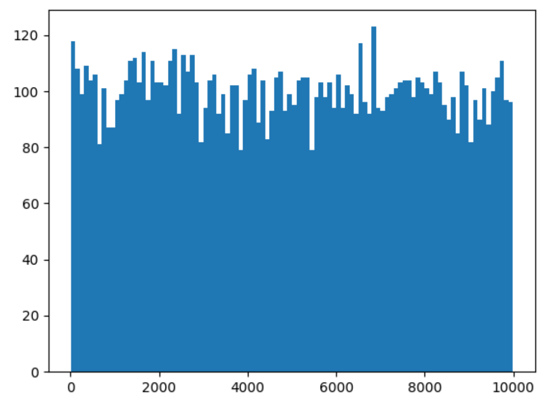


Figure 8: Energy Spectrum of hydrogen.png

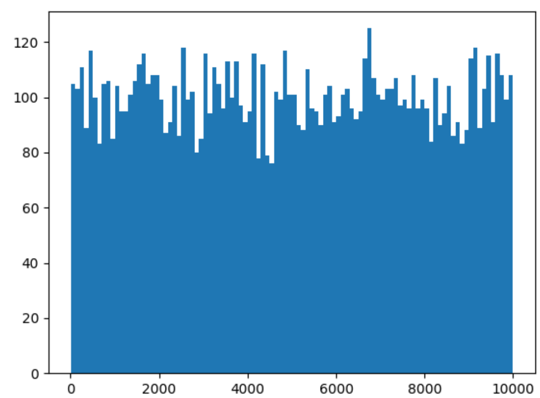


Figure 9: Energy Spectrum of deterium.png

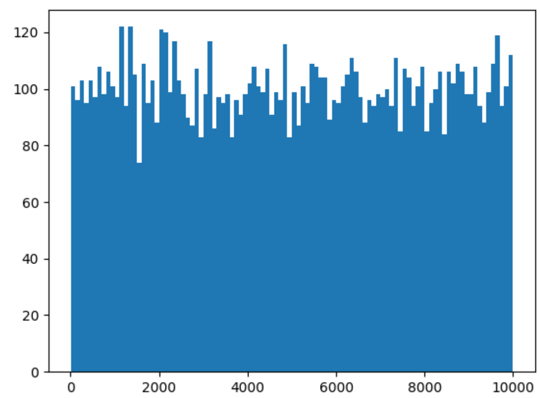
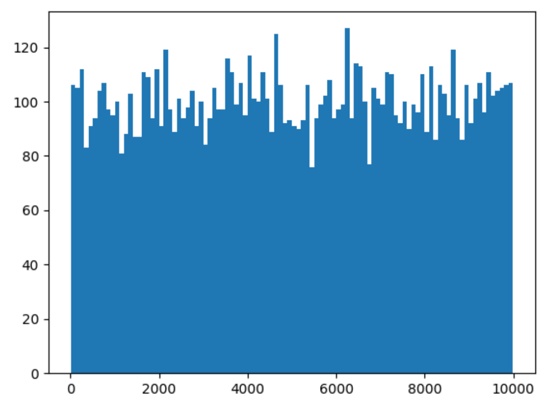


Figure 10: Energy Spectrum of Helium.png



2.1.5 Observations

The energy spectrum obtained for each particle forms a uniform distribution, as expected from the simulation.

2.2 Subproblem 2

2.2.1 Problem

Calculate energy loss for each particles, considering only ionisation energy loss.

- Plot momentum vs dE/dx plot for each particle
- Plot momentum vs dE/dx plot for all particles in single plot.

2.2.2 Algorithm

1. Define n = number of slices of the slab
2. Using the energies of incident particles, calculate the incident momentum of each particle using, $p = (1 + \frac{E}{m})^2 - 1)m$, where E is the kinetic energy of the incident particle obtained, and m is the mass energy of the particle.
3. Iterate through the list of energies of particles, and do the following for each type of particle.
   - Calculate the energy loss for each incident particle(that is, the energy loss for each incident energy of that particle)
   - Subtract the energy loss from the initial energies.
   - Repeat the above 2 steps for all the slices.
4. For each particle, calculate the net energy loss through the entire slab.
5. Plot momentum vs net energy loss of each type of particle.
6. Plot momentum vs net energy loss of all particles in a single plot.

2.2.3 Code

```
import numpy as np
import matplotlib.pyplot as pl

class Material:
    def __init__(self, I, A, B, C, a, m, X1, X0, aZ, aA, rho):

        self.I = I * 13.6056e-6 #parameter for ionization loss

        #parameters for density correction
        self.A, self.B, self.C = A, B, C
        self.a, self.m = a, m
        self.X0, self.X1 = X0, X1


        self.aZ, self.aA = aZ, aA #atomic number, atomic mass


        self.rho = rho #material density in gm/cc
        self.b = 4e-6 * rho #MeV/cm


def density_correction(beta, gamma, material):
    X = np.log10(beta*gamma)
    delta = 0
    if(X < material.X0):
        delta = 0
    elif(X < material.X1):
        delta = 4.6052*X + material.a*(material.X1 - X)**(material.m) + material.C
    elif(material.X1 < X):
        delta = 4.6052*X + material.C
    return delta


```

```
def ionisation_loss(E,mass, material):
    m_e = 0.511 #MeV/c^2
    m_u = 105.7 #MeV/c^2

    gamma = E/mass + 1

    beta = np.sqrt(1 - (1/gamma**2))

    K = 0.307075 #MeV g^-1 cm^2

    delta = density_correction(beta, gamma, material)
    Em = 2*m_e*(beta*gamma)**2/(1 + 2*gamma*m_e/m_u + (m_e/m_u)**2)
    coeff = K * material.aZ/material.aA * 1/(beta**2)
    dEdx = coeff*(0.5*np.log((2*m_e*((beta*gamma)**2)*Em)/(material.I**2)) + Em/(8*E) - beta**2 - 0.5*delta)

    return material.rho * dEdx


Copper = Material(27.7, 0.0701, 15.09, -4.74, -0.119, 3.38, 3, 0.20, 29, 63.5, 8.92)

np.random.seed(1)

particles = ['Electron', 'Muon', 'Tauon', 'Pion', 'Kaon', 'Proton', 'Neutron', 'Hydrogen', 'Deuterium', 'Helium']
mass = [0.511,105.658,1776.86,139.57,493.677,938.272,939.565,938.781,1875.6127,3727.379]

n_particles = 10000 #number of particles
E_max = 1
E_min = 10000

thickness = 100 #in mm

E = []
p_all = []



#Plotting the energy spectrum

for j in range(len(particles)):

    E_i = np.random.uniform(E_max, E_min, n_particles)

    p_i = [((1 + (E_i[k]/mass[j]))**2 - 1)*mass[j] for k in range(len(E_i))]
    E.append(E_i)
    p_all.append(p_i)
'''
    pl.hist(E_i,int(n_particles/100))
    pl.xlabel("Energy in Mev")
    pl.ylabel("Number of particles")
    pl.title("Energy spectrum of {}".format(j))
    pl.savefig("Energy_Spectrum_{}.png".format(j))
    pl.show()
'''


E_loss = []
E_net_loss_all = []


for j in range(len(particles)):
    E_loss_j = []


    for i in range(thickness):
        E_loss_i = []

```

```
        for k in range(n_particles):

            if E[j][k] != 0:
                E_loss_i.append(ionisation_loss(E[j][k],mass[j],Copper)) #Energy loss for all particles
            else:
                E_loss_i.append(0)

        E[j] -= E_loss_i[j]

        E_loss_j.append(E_loss_i)


    #Calculating net energy loss of each particle
    E_net_loss = []
    for each in range(len(E_loss_j[0])):
        net_loss = 0
        for ay in range(len(E_loss_j)):
            net_loss += E_loss_j[ay][each]
        E_net_loss.append(net_loss)


    E_net_loss_all.append(E_net_loss)
    E_loss.append(E_loss_j)

    #momentum vs dE/dx plot for each particle
    pl.plot(p_all[j],E_net_loss,".")
    pl.xlabel("Incident momentum")
    pl.ylabel("dE/dx")
    pl.title("{}".format(particles[j]))
    pl.savefig("plot2_{}.png".format(particles[j]))
    pl.show()
```

2.2.4 Results

Figure 11: Electron.png

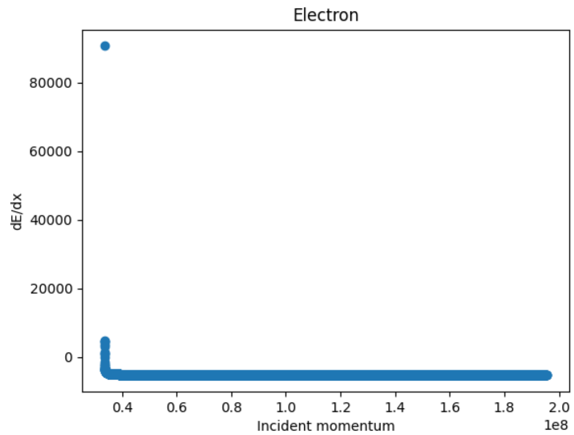


Figure 12: Muon.png

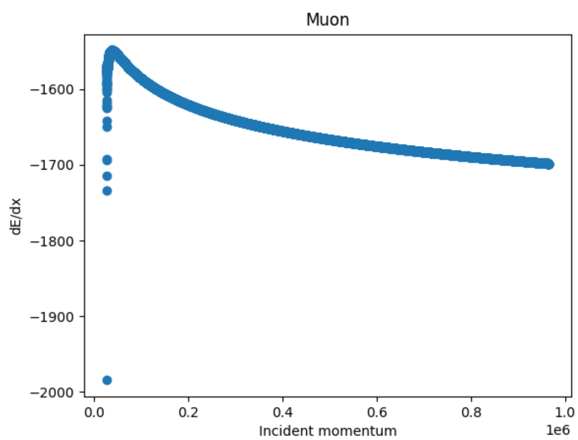


Figure 13: Tauon.png

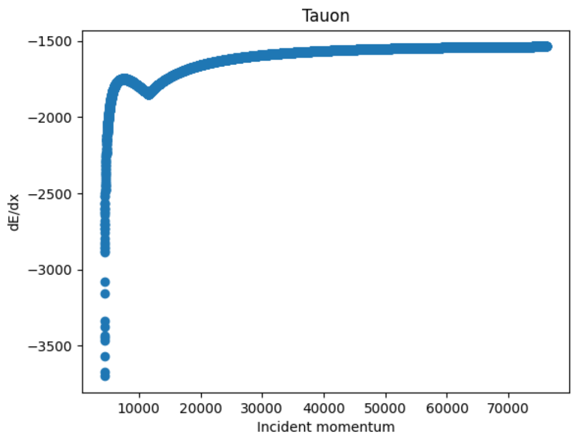


Figure 14: Neutron.png

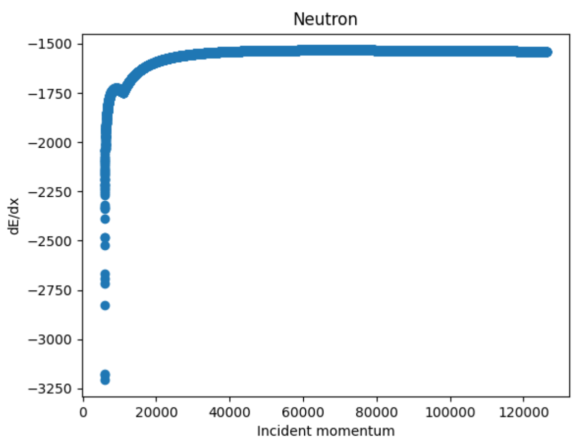


Figure 15: Energy Spectrum of hydrogen.png

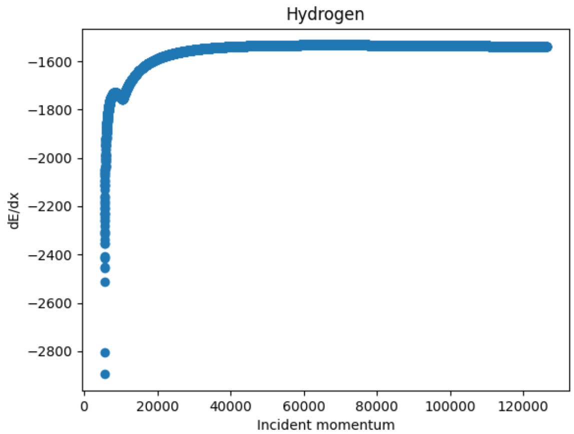


Figure 16: Dueterium.png

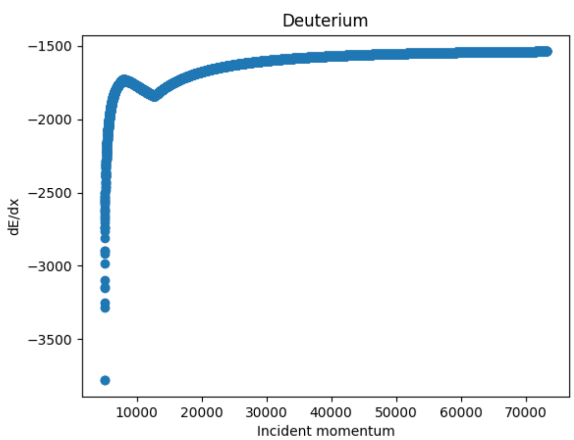
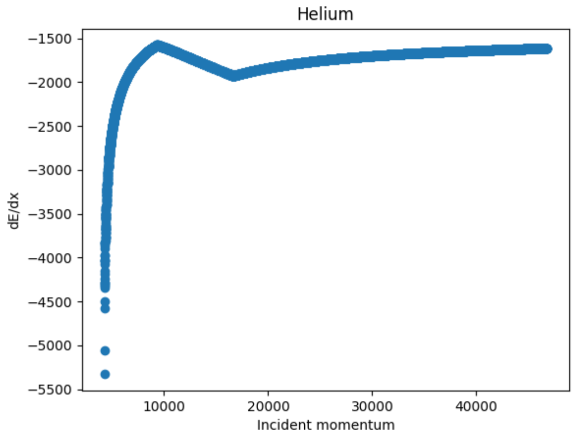


Figure 17: Helium.png



3 List of files submitted

1. PHY411-Report:Assignment 3.tex
2. PHY411-Report:Assignment 3.pdf
3. PHY411_3.py