

# Final Project

Due May 1, 11:59pm

150 points

CS 4499/5599

NE 4499/5599

Computational Engineering with C++

Dr. Leslie Kerby

Write a C++ program that randomly samples from the Watt Fission Spectrum.

## The Watt Distribution (Spectrum)

The distribution of the number of neutrons with energy in the fission spectrum is well represented by a mathematical function, the *Watt function*.

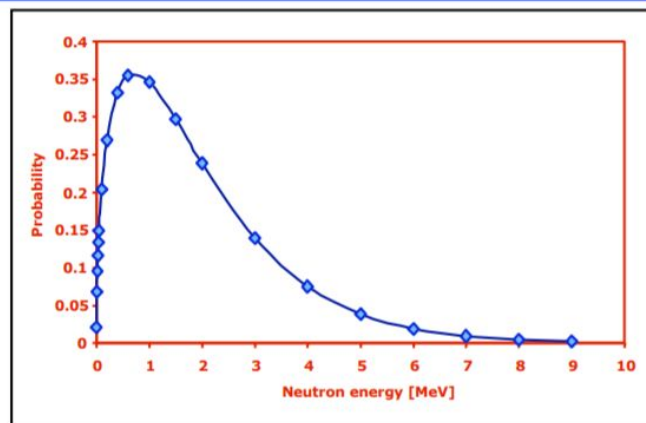
The probability of a neutron from fission having an energy between  $E$  and  $E+dE$  is the function  $P(E)dE$ .

$$P(E) = 0.4865 \sinh(\sqrt{2E}) e^{-E} \text{ MeV}^{-1}$$

This is an *empirical* formula – it is just a convenient way of expressing the experimental distribution.

This distribution is for neutrons from the fission of  $^{235}\text{U}$  with a slow neutron and is shown in the figure.

**Note:** It only changes slightly for other types of fission.



Say you have a Monte Carlo neutronics code. Part of this code will involve randomly sampling what energy a new neutron created from fission will have. This can be done using the Watt distribution. For this problem, we will assume the probability distribution of the energy is

$$P(E) = 0.4865 \sinh(\sqrt{2E}) e^{-E}$$

1. Setup.
  - a. First create a lambda function **fx** within main() which is passed **E** and returns the value of  $P(E)$  from the equation above.
  - b. Check that you have **fx** correct by integrating using **trapezoidal** from  $[0,8]$  and showing your result is 1.00 (accurate to about the hundredth place). Print the integral to the screen.
2. Find the **maximum** of **fx** ( $P(E)$  ).
  - a. You can do this either by finding the roots of the derivative of **fx** (ie using **finite\_difference\_derivative** and **bisect**) and evaluating **fx** at the root, or by using **minima**. Print the ( $E, P(E)$ ) to the screen.
3. Random point.
  - a. Randomly create a point in the range  $x=[0,8]$  and  $y=[0, \text{max from part 2}]$ . You will need to randomly select two values. Create this as a pair called **point**.
4. Use rejection method.
  - a. Check to see if the **point** selected in part 3 is under the  $P(E)$  curve. If it is, save the energy to a vector of doubles called **sample**. If it is not, reject the point and randomly create a new one and check again. Repeat until a valid point is found and the selected energy is stored.
5. Create a sample.
  - a. Randomly sample 100,000 energies from the Watt Fission Spectrum. They should be stored in **sample**. Print the size of **sample** to the screen.
6. Binning.
  - a. Create a vector of 100 energy bins, from 0 to 8 MeV, called **bins**. Find the number of energies in **sample** that are in each bin. Print to the screen the number of selections found in each of the 100 bins.
7. Histogram plot.
  - a. Make sure your random sampler works by plotting a histogram of **sample**, by plotting the value in each bin of **bins**. You can use Excel, Python, C++, etc. If not in your C++ code, then attach the other file with the image of the plot.

Attach all relevant code and screenshots.