EE 381
Project 4
Alexander Fielding

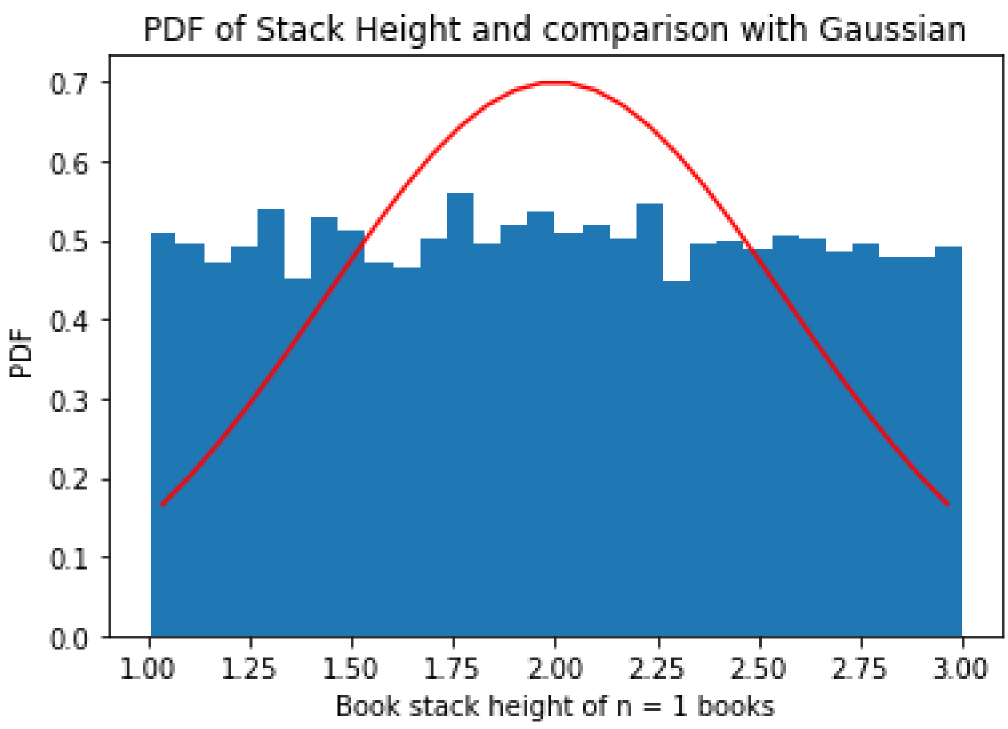
Problem 1
Introduction

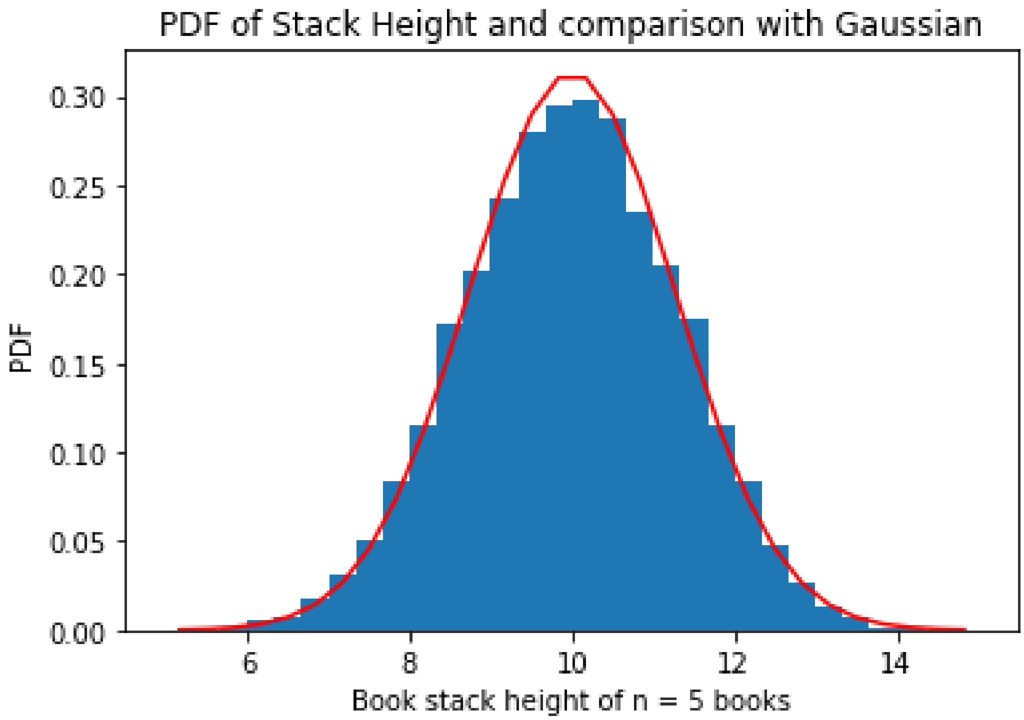
The purpose is to display the Central Limit Theorem which states that as sample sizes of independent random variables increase a normal distribution will occur. The simulation occurs 10,000 times and is compared with the Gaussian function.

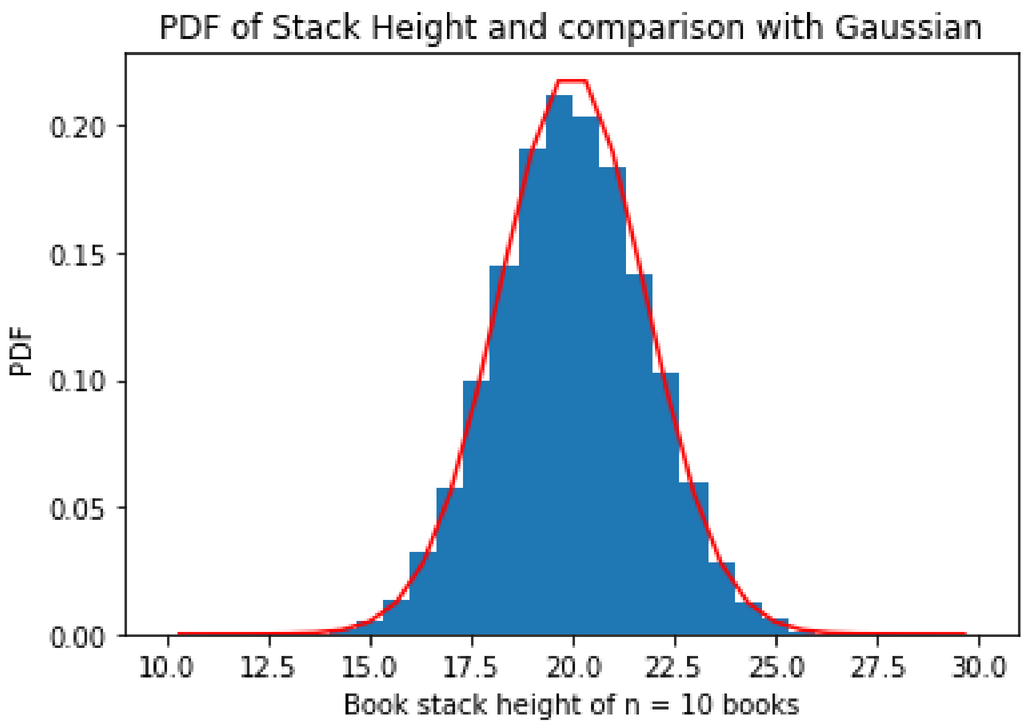
Methodology

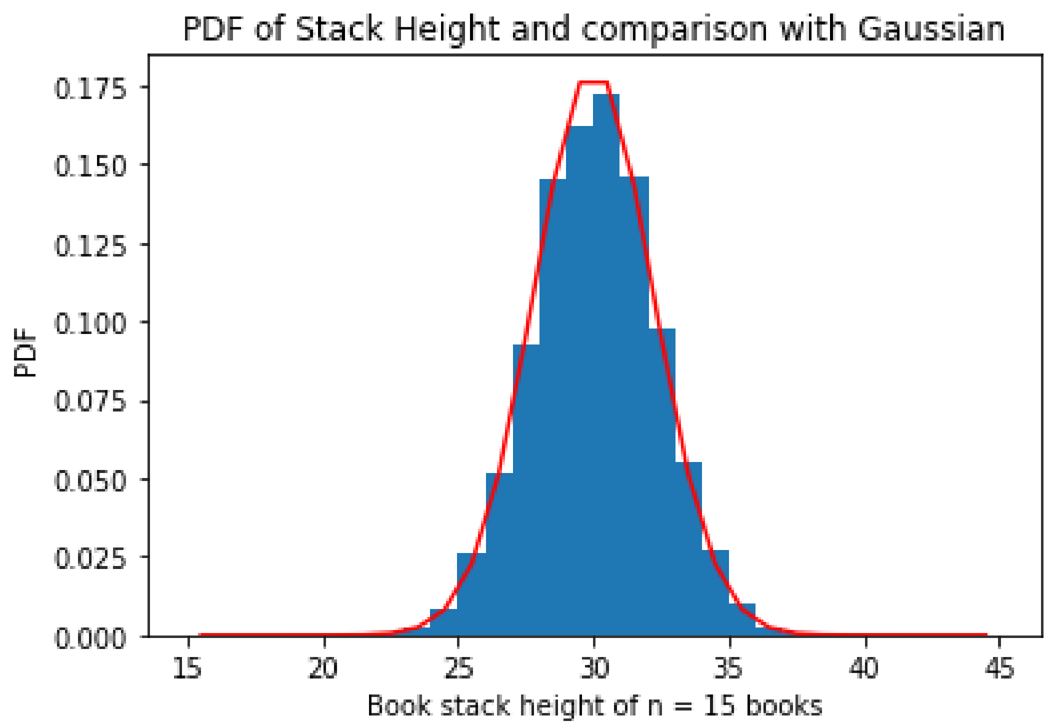
To accomplish this, we utilize the numpy and pyplot libraries to handle generating the random variables based on the range of book thicknesses, the mean of the book thicknesses and the standard deviation. 10,000 random variables are generated with uniform distribution and then the Sum of all are stored in a list. The list is then plotted alongside with the Gaussian function.

Results




PDF of Stack Height and comparison with Gaussian
PDF
0.00
0.05
0.10
0.15
0.20
0.25
0.30
6
8
10
12
14
Book stack height of n = 5 books


PDF of Stack Height and comparison with Gaussian
PDF
0.00
0.05
0.10
0.15
0.20
10.0
12.5
15.0
17.5
20.0
22.5
25.0
27.5
30.0
Book stack height of n = 10 books



Appendix

```
# EE 381 project 4
# Alexander Fielding
# problem 1

import pylab as plt
import numpy as numpy

# Globals
nbooks=1;
# a=min bookwidth ; b=max bookwidth
a=1; b=3;
meanthickness = 2
stnddeviation = 0.57
X = []      # stores 10,000 results from experiments
Result = []

def findSumofWidths(numbooks,alist):
   Sum = 0
   for i in alist:
      Sum += i
   return Sum
```

```
def gaussian(mu,sig,z):
    f=numpy.exp(-(z-  mu)**2/(2*sig**2))/(sig*numpy.sqrt(2*numpy.pi))
    return f



#Main
count = 0
while(count < 10001):
   X = numpy.random.uniform(1,3,nbooks)
   Result.append(findSumofWidths(nbooks,X))
   count = count + 1

nbins=30;
# Number of books ; Number of bins
edgecolor='w';
# Color separating bars in the bargraph
#
# Create bins and histogram
bins=[float(x) for x in plt.linspace(nbooks*a, nbooks*b,nbins+1)]
h1, bin_edges = plt.histogram(Result,bins,density=True)
   # Define points on the horizontal axis
be1=bin_edges[0:plt.size(bin_edges)-1]
be2=bin_edges[1:plt.size(bin_edges)]
b1=(be1+be2)/2
barwidth=b1[1]-b1[0] # Width of bars in the bargraph
plt.close('all')
# PLOT THE BAR GRAPH
fig1=plt.figure(1)
plt.bar(b1,h1, width=barwidth, edgecolor=edgecolor)
 #PLOT THE GAUSSIAN FUNCTION

f=gaussian(meanthickness*nbooks,stnddeviation*numpy.sqrt(nbooks),b1)
plt.plot(b1,f,'r')
plt.title('PDF of Stack Height and comparison with Gaussian')
plt.ylabel('PDF')
plt.xlabel('Book stack height of n = 1 books')
```

Problem 2

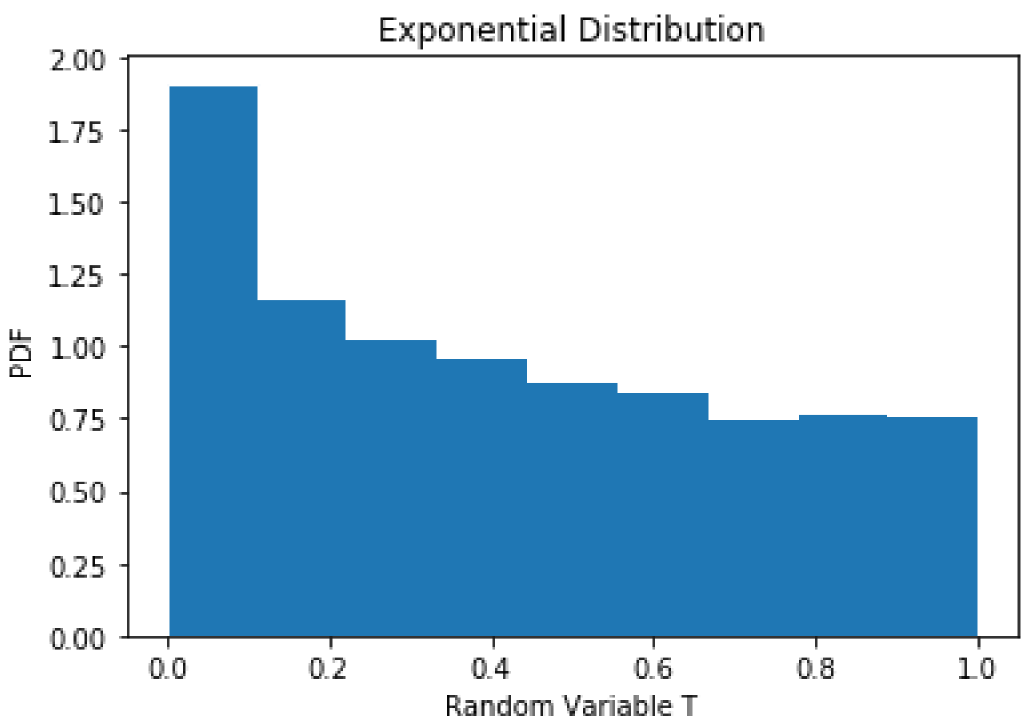
Introduction

The purpose of this experiment was to simulate the exponential distribution using the numpy library and the provided function.

Methodology

To implement this experiment the pyplot library and the numpty library are used to store the data into a list and then plot them will comparing them to the function itself.

Results



Appendix

```
# EE 381 project 4
# Alexander Fielding
# problem 2

import numpy as np
import pylab as plt

X = [] # Results
F = []
```

```
def expdist(t):
    if t.any() < 0:
        return 0
    else:
        result = 2 ** (-2 * t)
        return result


#Main
count = 0
t = np.random.exponential(1,10000)
f = np.random.exponential(1,10000)

while(count < 9999):
    count = count + 1
    X.append(expdist(t[count]))
    F.append(expdist(f[count]))


edgecolor='w';
bins=[float(x) for x in plt.linspace(0,1,10)]
h1, bin_edges = plt.histogram(X,bins,density=True)
    # Define points on the horizontal axis
be1=bin_edges[0:plt.size(bin_edges)-1]
be2=bin_edges[1:plt.size(bin_edges)]
b1=(be1+be2)/2
barwidth=b1[1]-b1[0] # Width of bars in the bargraph
plt.close('all')
# PLOT THE BAR GRAPH
fig1=plt.figure(1)
plt.bar(b1,h1, width=barwidth, edgecolor=edgecolor)
plt.title('Exponential Distribution')
plt.ylabel('PDF')
plt.xlabel('Random Variable T')
 #PLOT THE GAUSSIAN FUNCTION

plt.plot(b1,,'r')
```

Problem 3

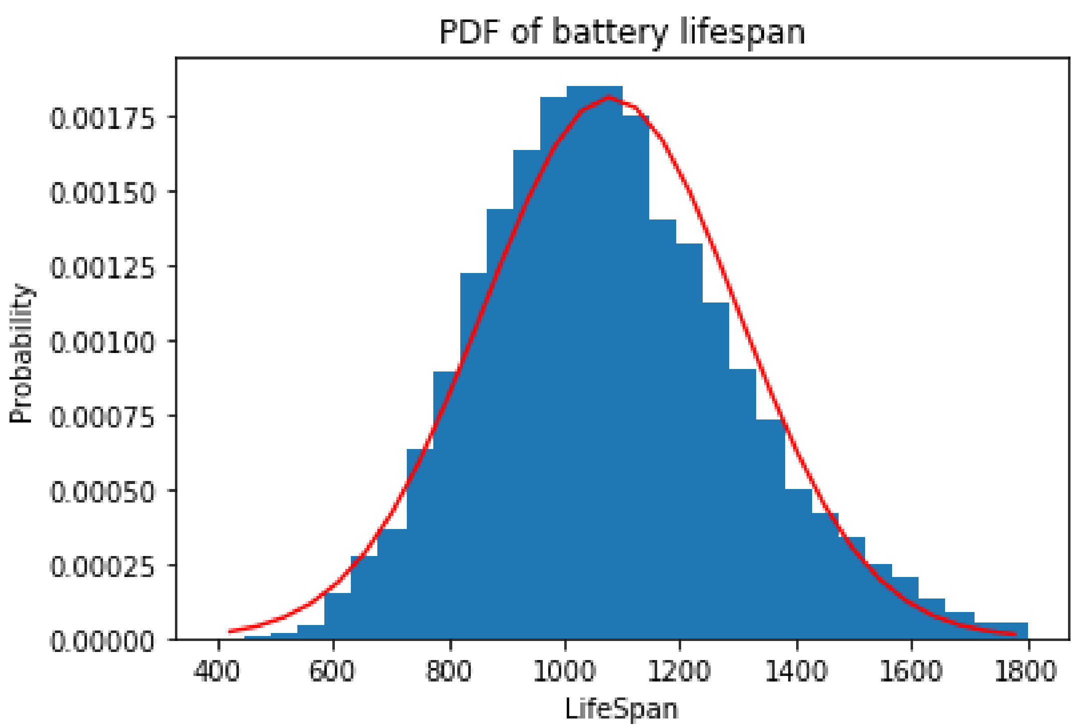
Introduction

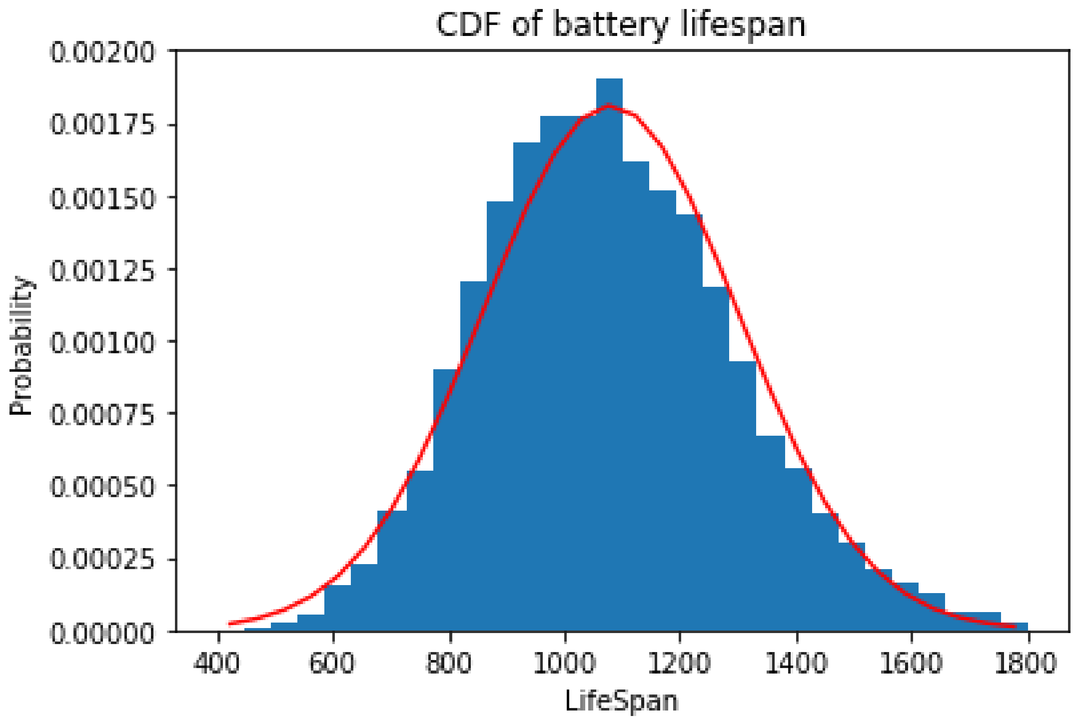
Real – World application of the central limit theorem using the same methodologies as in the previous problems. The goal is to visualize the lifespan of batteries using the normal probability distribution function. In addition, to use the same data and plot in a CDF plot to shot the probability that the battery life will be lower than 45 days.

Methodology

This simulation uses the numpy and pyplot libraries to handle the random variable distribution and the PDF/ CDF plotting. The normal distribution curve is obtained from the Gaussian function.

Results





Question	Answer
1. Probability that the carton will last longer than three year	0.00175
2. Probability that the carton will last between 2 to 2.5 years	0.00075

Appendix

```
# EE 381 project 4
# Alexander Fielding
# problem 3

import pylab as plt
import numpy as np

def gaussian(mu,sig,z):
    f=np.exp(-(z-  mu)**2/(2*sig**2))/(sig*np.sqrt(2*np.pi))
    return f

lifespan = 45
stnddeviation = 45 * np.sqrt(24)
mean = 24 * 45
```

```
count = 0
X = []
Result = []

while(count < 10000):
    count = count + 1
    carton = np.random.exponential(lifespan,24)
    C = 0 #Sum
    for i in carton:
        C += i
    X.append(C)
    Result.append(np.cumsum(C))


nbins=30;
# Number of books ; Number of bins
edgecolor='w';
# Color separating bars in the bargraph
#
# Create bins and histogram
bins=[float(x) for x in plt.linspace(400,1800,nbins+1)]
h1, bin_edges = plt.histogram(Result,bins,density=True)
    # Define points on the horizontal axis
be1=bin_edges[0:plt.size(bin_edges)-1]
be2=bin_edges[1:plt.size(bin_edges)]
b1=(be1+be2)/2
barwidth=b1[1]-b1[0] # Width of bars in the bargraph
plt.close('all')
# PLOT THE BAR GRAPH
fig1=plt.figure(1)
plt.bar(b1,h1, width=barwidth, edgecolor=edgecolor)
 #PLOT THE GAUSSIAN FUNCTION
f=gaussian(mean,stnddeviation,b1)

plt.plot(b1,f,'r')
plt.title('CDF of battery lifespan')
plt.ylabel('CDF')
plt.xlabel('Number of Batteries')
```