NE 334 Assignment 1
Due October 8, 2024

The purpose of this assignment is to gain experience with numerical simulations and data analysis. The required Python scripts are available on the UW Learn page, in the same content section this Assignment PDF file was found. You can run the calculations on your own computer. You will need to have Python installed in your computer. Upload a *single* PDF file to the learn dropbox for Assignment 1.

1. **Normal distribution**. The python script normal.py samples the normal distribution for a random variable *x*. Open the script in a text editor to modify calculation parameters. After running the script, a file named xnormal.dat is created and contains the sampled data. The input parameters are xmean, standard deviation, and sample size.

   (a) Modify the input parameters in normal.py and calculate the average value of *x*: h*x*i. This can be done by either modifying the python script to compute and print the average value, or by importing the xnormal.dat datafile using tools such as MS Excel. Increase the value of samplesize and recompute h*x*i. What happens to |h*x*i − xmean|^2 as you increase samplesize? Report all your input parameters. Be careful as not to increase samplesize too much as each real number stored in computer RAM occupies 8 bytes and each character saved in the x_normal.dat files requires 1 byte of disk storage.

```
#imput parameters

x_mean=1.

standard_deviation=2.

sample_size=10000
```

Code:

```
#Calculate the average value of x  x_average
= sum(x_norm)/len(x_norm) print("average
(x):",x_average)

#Calculate  average  -  x_mean  x_abs_squared  =
(np.abs(x_average  -  x_mean))**2  print  ("The
x_abs_squared is:", x_abs_squared)
```

The initial average (x): 0.996626902582589

I multiplied the sample size by 2 and got average (x): 0.998216591236791

The x_abs_squared is: 3.18054681669079e-06

The x_abs_squared value increases as the sample size increases but there are some descrepencies.

```
#imput parameters
x_mean=1.
```

```
standard_deviation=2.
sample_size=30000
```

- Multiplied sample size by 3 to get
- average (x): 0.9971580742408167
- The x_abs_squared is: 8.076542020709545e-06

```
#imput parameters
x_mean=1.
standard_deviation=2.
sample_size=40000
```

- Multiplied sample size by 4 to get
- average (x): 0.9890085723215594
- The x_abs_squared is: 0.00012081148241039088

(b) Choose input parameters and compute the variance of x, σ_x^2, based on the sampled data. This can be done by modifying the python script or by analyzing the data in a program such as MS Excel. Also calculate the standard deviation and compare your result to the input parameter standard deviation. Report all your input parameters.

```
#imput parameters
x_mean=5.
standard_deviation=7.
sample_size=5000
```

```
#Calculate variance and standard deviation
x_variance        =        np.var(x_norm)
x_std_deviation = np.std(x_norm)
 print("variance (x):", x_variance) print("Standard
Deviation (x):", x_std_deviation) print("Initial
Standard Deviation:", standard_deviation)
```

average (x): 5.051575757283553

The x_abs_squared is: 0.0026600587393719602 variance

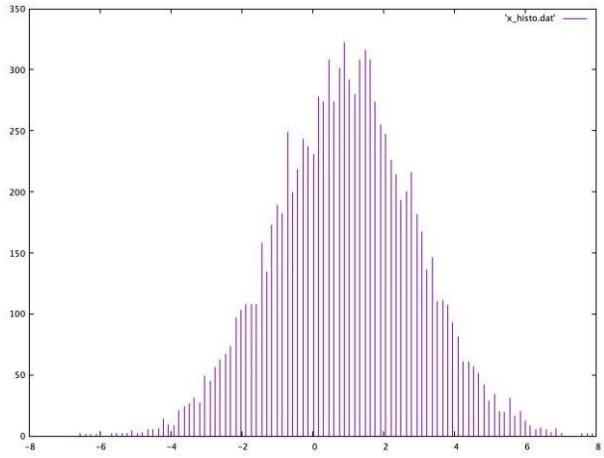
(x): 49.54937138883668

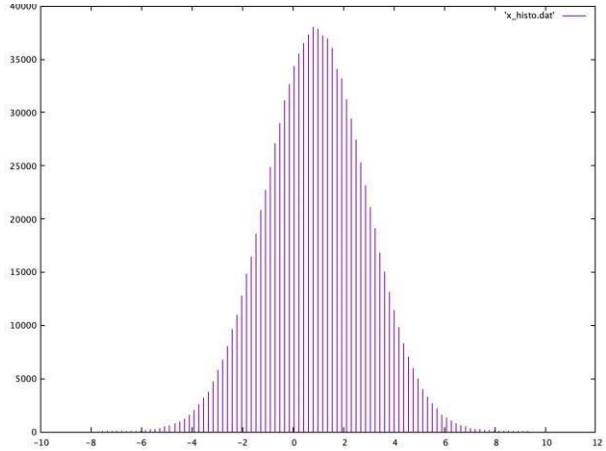
Standard Deviation (x): 7.039131437104771

Initial Standard Deviation: 7.0

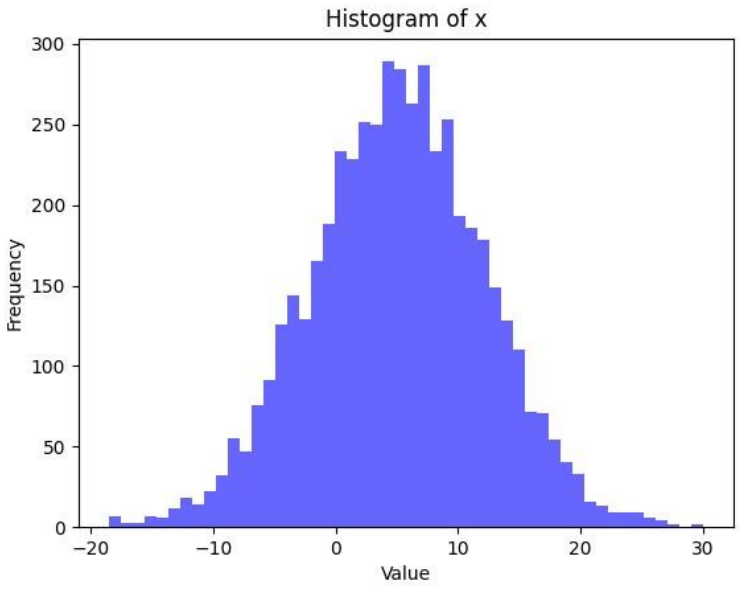
The standard deviation from input parameters is very close to the calculated standard deviation in proximity.

(c) The file xhisto.dat contains the histogram of *x* sampled from the normal distribution. Plot the histogram of *x* for different values of samplesize and your choice of input parameters. The graphs should look like this:





The script parameters nbins can be adjusted to generate as smoother graph. When sample size is smaller, nbins should also be smaller. Report all your input parameters.



2. **Uniform distribution**. The python script uniform.py samples a uniform distribution of random numbers in the range $0 \le x \le 1$. A file named xuniform.dat is created and contains the sample data.

   (a) What do you expect average of x to be?

   It should be approximately 0.5

   (b) Use a text editor to modify the input parameters in uniform.py and calculate h*x*i. How does h*x*i change when sample size increases?

   The initial average of the sample size given is: 0.49653098972791715 I

   multiplied sample size by 2 and the average of x is: 0.4964968972954596

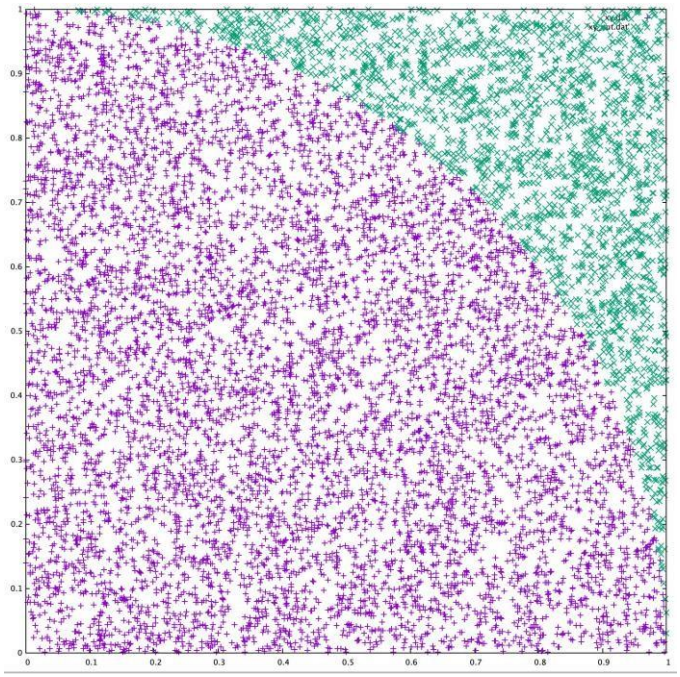
   Average did not change with change in sample size.

   (c) Modify the script to generate random numbers between −1 and 3. Include your script with your submission.

```python
import numpy as np
sample_size=10000*2
x_uniform=np.random.uniform(-1,3, sample_size)
f=open('x_uniform.dat','w')
for x in x_uniform:
    print(x)
    f.write(str(x)+'\n')
f.close()

#Calculate the average value of x
average = np.mean(x_uniform)
print("The average is:", average)
```

3. **Can you calculate** π ? The python script pi.py samples uniform distributions for two sets of random numbers, $0 \le x \le 1$ and $0 \le y \le 1$. The total number of samples is samplesize. The code also reports the number of data points, countcircle, for which $x^2 + y^2 \le 1$ and those data points are stored in xy.dat. Below is a graph of the sample points in the (x,y) plane. The purple points are those for which $x^2 + y^2 \le 1$ holds true. The points in green are those for which the $x^2 + y^2 \le 1$ condition is NOT met. Those points are stored in the file xyout.dat.



(a) Devise an approach to estimate the value of π based on the data generated by running pi.py. Describe your procedure.

```
        """
        To estimate pi, count the number of points inside the unit circle,
        represented by count_circle.
        Divide this count by the toal sample size and then multiply by 4 because
        the ratio of the area of a unit circle
        to the area of the square that bounds the circle is pi/4. Solving for pi =
4*(count_circle/ sample_size)
       """

        estimated_pi = 4*(count_circle/sample_size)
        print("Estimated value of pi:", estimated_pi)
```

Estimated value of pi: 3.156

(b) Report your estimate of π for a series of samplesize values.

At sample size 20000 ❾ Estimated value of pi: 3.154

At sample size 40000❾ Estimated value of pi: 3.153

At sample size 60000 ❾ Estimated value of pi: 3.1386666666666665

At sample size 80000❾ Estimated value of pi: 3.14845

At sample size 90000❾ Estimated value of pi: 3.14

(c) Estimate the standard error of your computed π value.

Hint: for N independent samples, the standard error of the mean of property A, hAi, is defined as $\sigma_A/\sqrt{N}$.

Standard Error of pi: 0.004078057258058057

NOTE: You can use matplotlib to make your plots if you wish. This requires modifying the python scripts.