STOCHASTIC PROCESSES

HW01

Hanie Hatami(99100614)
Stochastic processes course

Contents

1 Abstract

In this exercise, we want to generate datasets with the number of data 10^2, 10^3, 10^4, 10^5 and 10^6 with normal distribution (mean 0 and variance 1) and uniform (0 to 1). Then we want to draw their histogram with a special interval according to the number of data and then determine the shape of the probability distribution function. After that, we specify the error for each interval and draw the error bar of each interval in the graph.

2 Results

First, we shall simulate the normal distribution.

2.1 Normal distribution

2.1.1 Normal distribution n=10^2:

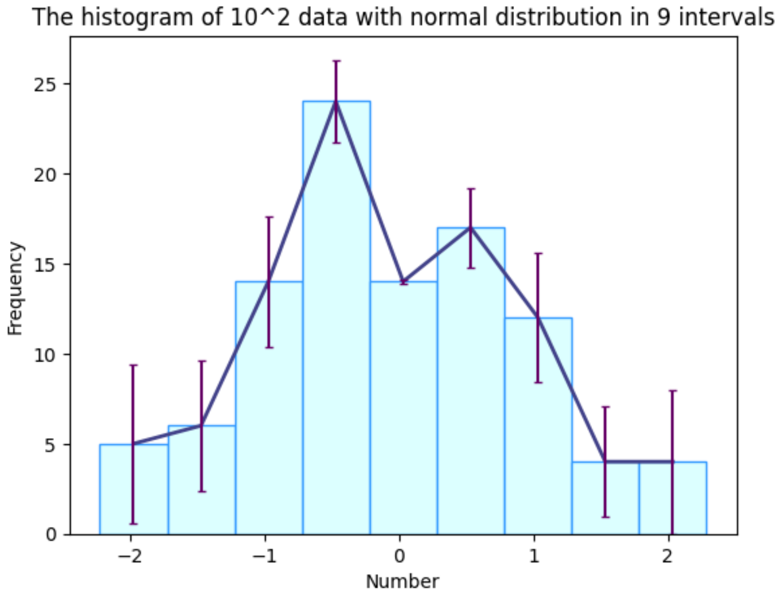

Figure 1: The histogram of 10^2 data with normal distribution in 9 intervals

2.1.2 Normal distribution n=10^3:

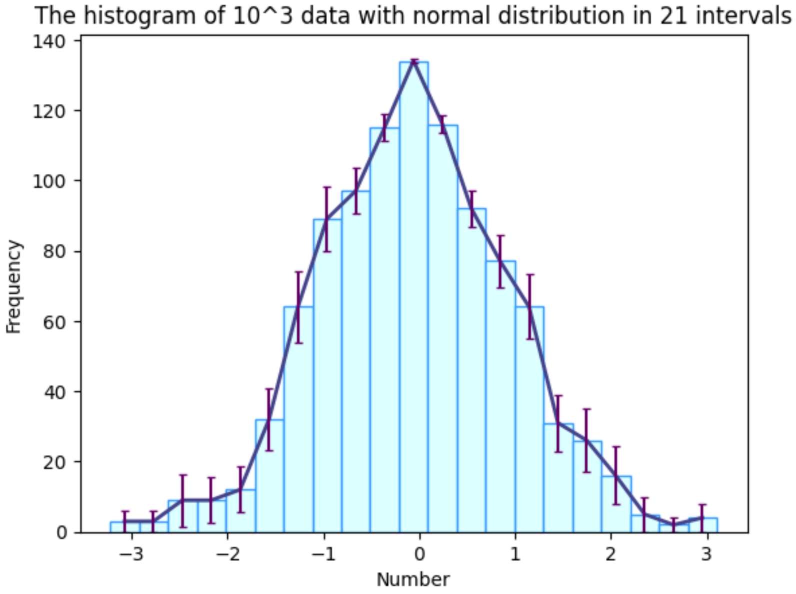

Figure 2: The histogram of 10^3 data with normal distribution in 21 intervals

2.1.3 Normal distribution n=10^4:

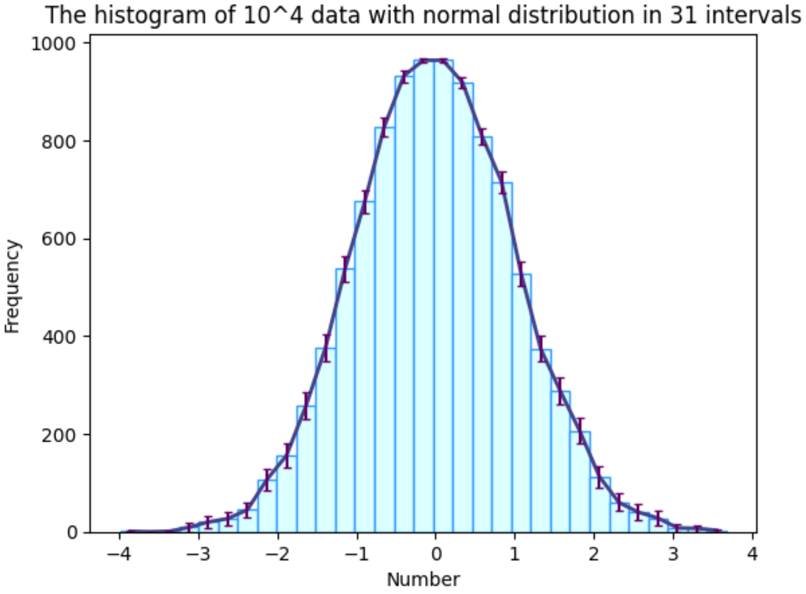

Figure 3: The histogram of 10^4 data with normal distribution in 31 intervals

2.1.4 Normal distribution n=10^5:

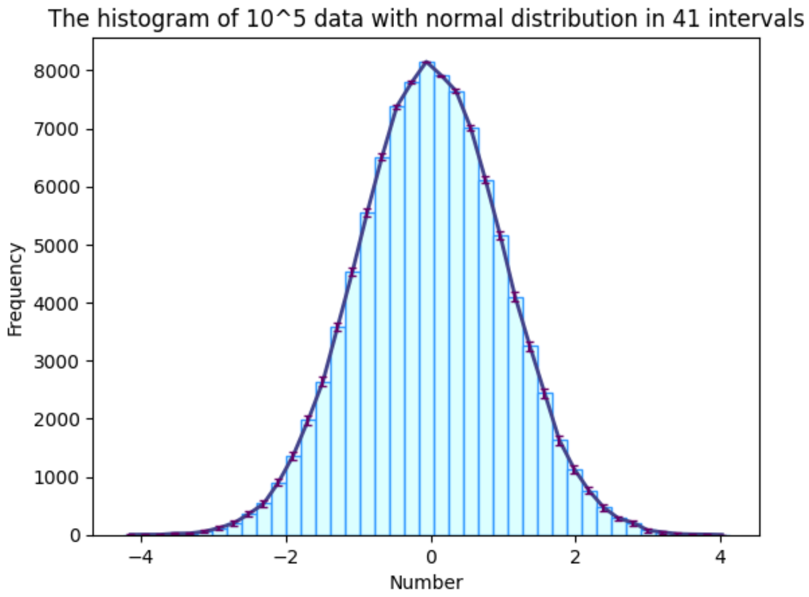

Figure 4: The histogram of 10^5 data with normal distribution in 41 intervals

2.1.5 Normal distribution n=10^6:

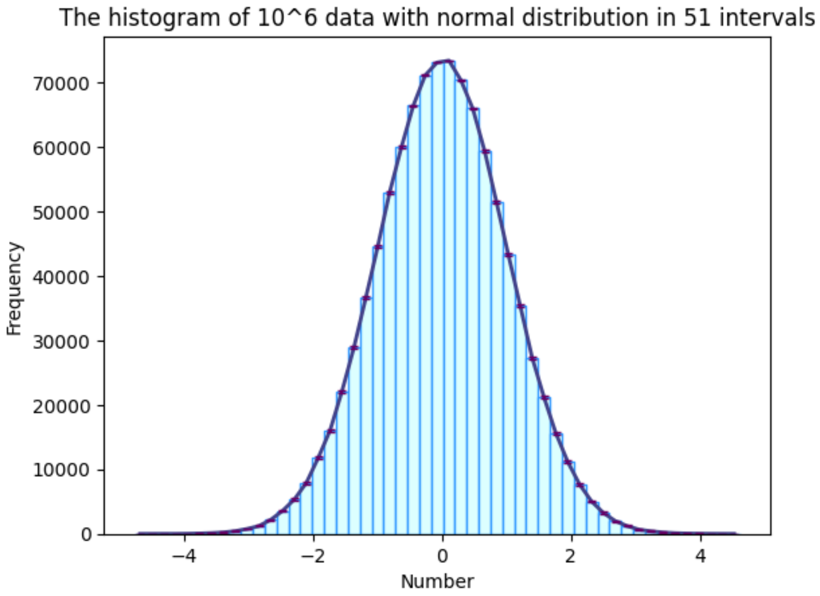

Figure 5: The histogram of 10^6 data with normal distribution in 51 intervals

Now, It's time for the uniform distribution.

2.2 Uniform distribution

2.2.1 Uniform distribution n=10^2:

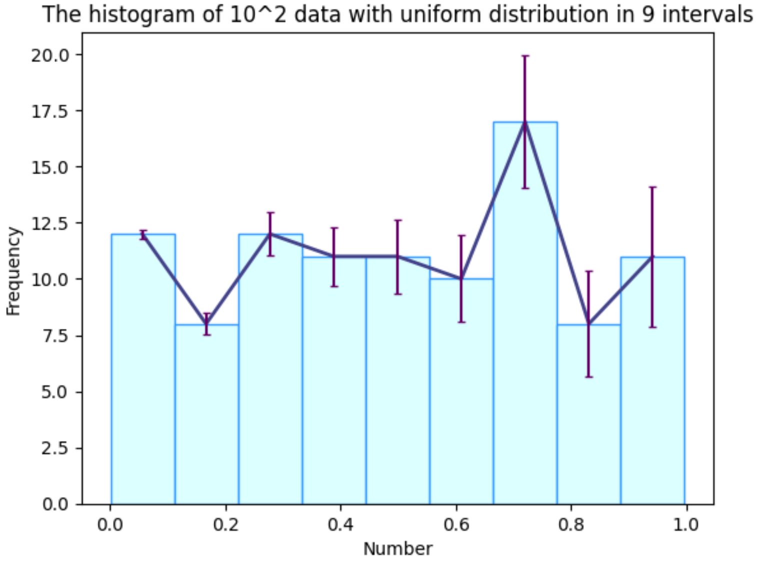

Figure 6: The histogram of 10^2 data with uniform distribution in 9 intervals

2.2.2 Uniform distribution n=10^3:

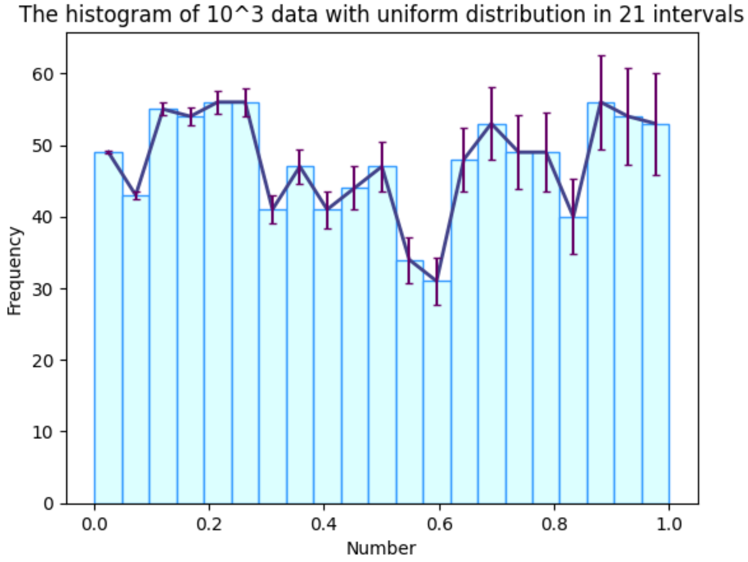

Figure 7: The histogram of 10^3 data with unifrom distribution in 21 intervals

2.2.3 Uniform distribution n=10^4:

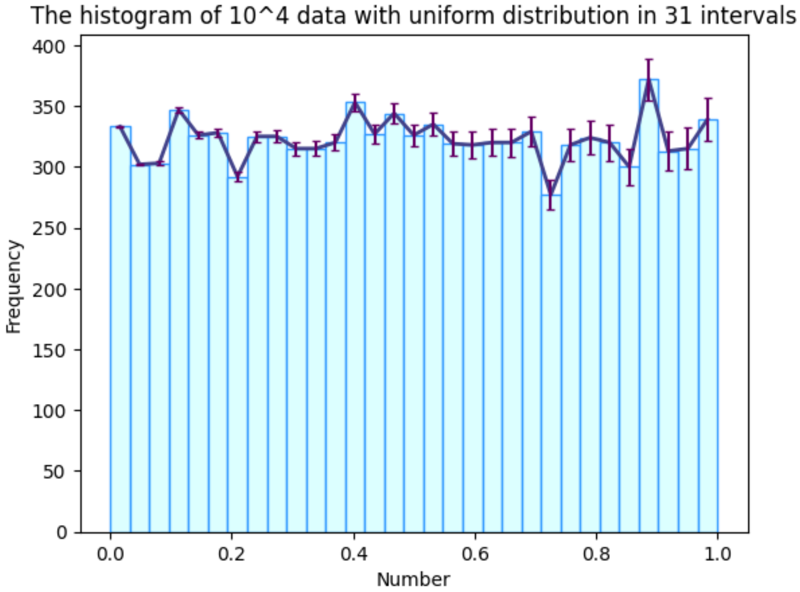

Figure 8: The histogram of 10^4 data with unifrom distribution in 31 intervals

2.2.4 Uniform distribution n=10^5:

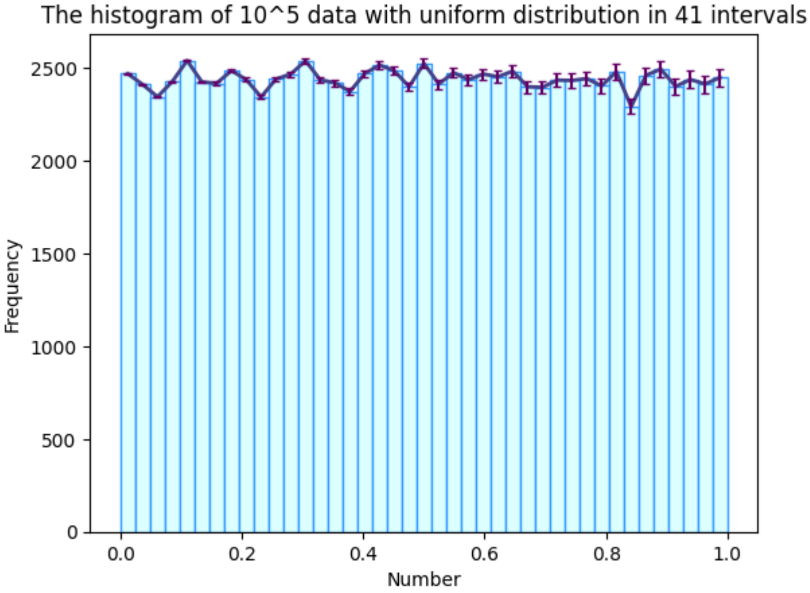

Figure 9: The histogram of 10^5 data with unifrom distribution in 41 intervals

2.2.5 Uniform distribution n=10^6:

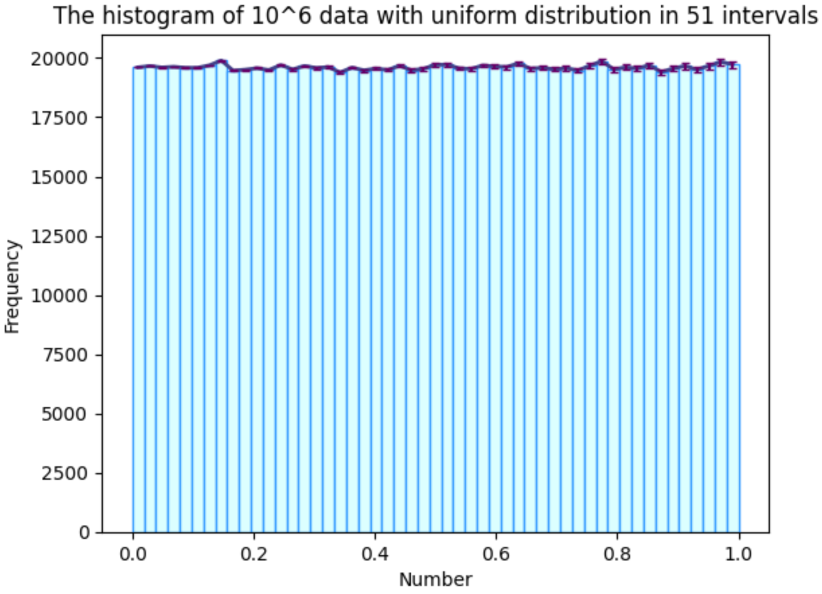

Figure 10: The histogram of 10^6 data with unifrom distribution in 51 intervals

3 Conclusion

As expected, as the number of data increases, the shape of probability distribution becomes closer to reality and the error bars become smaller. Also, the error often increases from the data center to the corners.
All these items were part of our expectations from the outputs.

4 Code

```python
# Import needed libraries
import numpy as np
import matplotlib.pyplot as plt

# Generate n random numbers with normal distribution(for normal distribution run this block.)
n = int(input('Enter numbers of random data you want to gerenate: '))
random_number = np.random.normal(0, 1, 1000000)
data = np.random.choice(random_number, n , replace = False)
# number of bins we want
bins = int(input('Enter the number of bins you want:'))

# Generate n random numbers with normal distribution(for uniform distribution run this block.)
n = int(input('Enter numbers of random data you want to gerenate: '))
data = np.random.uniform(0, 1, n)
# number of bins we want
bins = int(input('Enter the number of bins you want:'))

# Calculate the bin centers and the histogram values and the bin wodth
hist_values, bin_edges = np.histogram(data, bins=bins)
bin_centers = (bin_edges[:-1] + bin_edges[1:]) / 2

# Calculate the error for each interval
def Error_for_each_interval(data, bins, bin_edges):
  errors = []
  for i in range(bins):
    numinbins = []
    for j in data:
      if(j >= bin_edges[i]  and j <= bin_edges[i+1]):
        numinbins.append(j)
        binmid = (bin_edges[i] + bin_edges[i+1]) /2
    if len(numinbins) != 0:
      mu = binmid * len(numinbins) / (n)
      sigma2 = ((binmid**2)*len(numinbins))- (n * (mu**2)) / (n-1)
      vari = (sigma2)**(1/2)
      filter_arr = []
      for element in data:
        if element >=bin_edges[i]  and element <= bin_edges[i+1]:
          filter_arr.append(element)
      # Remove Probability < 0
      if (len(filter_arr) - vari) <= 0:
          vari = len(filter_arr)
      errors.append(vari)
    if len(numinbins) == 0:
      errors.append(0.0)
  errors = np.array(errors)
  return errors

# Create the histogram
def plot(x,bins,bin_centers, hist_values,yerrors):

  plt.hist(x, bins=bins , color=’#DCFFFF’ , edgecolor=’#3399FF’)

  # Plot the distribution function
  plt.plot(bin_centers, hist_values, ’#43438A’, linewidth=2)

  # Plot the error bars
  plt.errorbar(bin_centers, hist_values, yerr=yerrors, fmt=’none’, color=’#660066’, capsize=2)

  # Set the labels and title
  plt.xlabel(’Number’)
  plt.ylabel(’Frequency’)
  plt.title(’The histogram of 10^6 data with uniform distribution in 51 intervals’)

  # Display the plot
  plt.show()

plot(data, bins,bin_centers,hist_values , Error_for_each_interval(data, bins, bin_edges))
```