Project 2

Qidi Han

February 4, 2020

Question 1

Simulate sampling uniformly (10000) on the interval [-3,2]

Question 1(a)

Generate a histogram of the outcomes.

Theory

In this question, I simulate sampling uniformly (10000) on interval[-3, 2]. And I plot the histogram

Algorithm 1: Code for generating three different histogram.

Input: Number$_s$*ampling*n=10000,a =-3, b =2
Output: Three different histogram

```
/* import library                                    */
import numpy as np
import matplotlib.pyplot as plt
import statistics
/* for loop                                          */
for i from 0 to 10000 do
    num=(b-a)*np.random.rand() + a
    rand-arr1.append(num)
/* code for histogram                                */
N, bins, patches = plt.hist(rand-arr1)
plt.ylabel('Sample count')
plt.xlabel('sample value')
plt.title('Histogram of
plt.xlim(-3.5,2.5 )

```

```
-0.4790893317337787
2.058792562549812
```

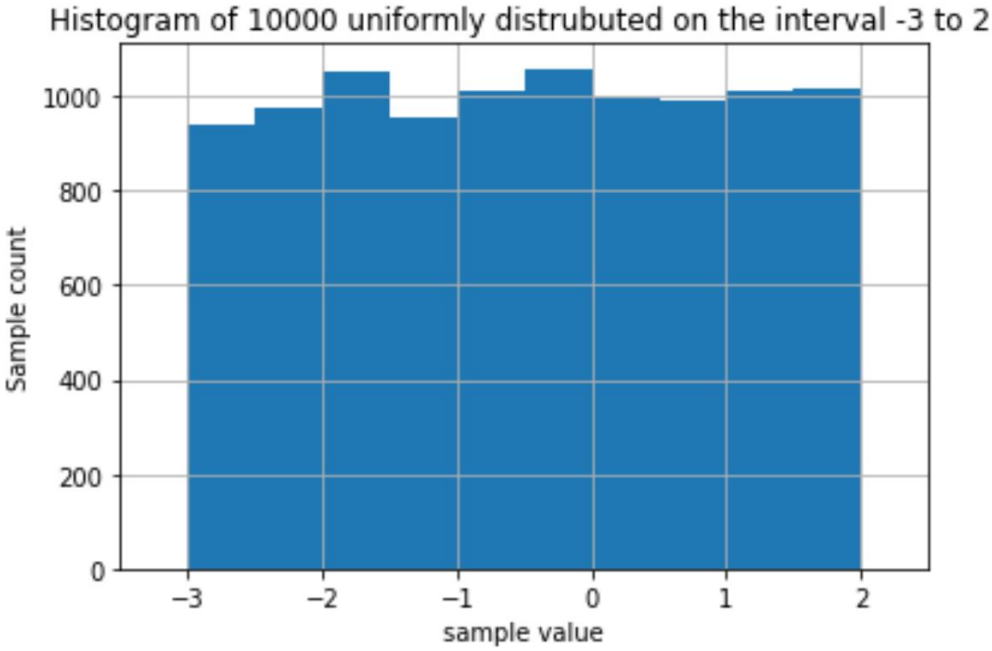


Figure 1: Histogram of 10000 outcomes.

In fig 1 histogram of 10000 outcomes

Explanation

In this question, I simulate the sampling uniformly on interval[-3, 2]. In Figure 1, some of the number are below 1000 and some of the number are above 1000, but still seems uniform

Question 1(b)

Compute the sample mean and sample variance for your samples. How do these values compare to the theoretical values? If you repeat the experiment will you compute a different sample mean or sample variance?

Theory

In this question, I use following code to compute the sample mean and sample variance of my 10000 samples, and use the formula below to calculate the theoretical values. I will repeat

this experiment with different size.

Algorithm 2: sample mean and sample variance

```
   /* Sample mean and sample variance                                   */
1  statistics.mean(rand_arr1)
2  statistics.variance(rand_arr1)
   /* Formula for theoretical mean and theoretical variance             */
3
```

$$mean = \frac{a+b}{2} \tag{1}$$

$$variance = \frac{(a+b)^2}{12} \tag{2}$$

```
4
```

Explanation

In this question, The sample mean is -0.479089, the sample variance is 2.05879. Use the formula 1 and formula 2, I got the theoretical mean is -0.5, and the theoretical variance is 2.0833. Comparing with our sample mean and sample variance, there are slight different. From the graph, we also can see the result of this difference, as I mentioned before, some of the number below 1000 and some of the number above 1000.When I repeat the experiment with different sample size, I will got different sample mean and sample variance, but larger sample size, give me a approached sample mean and sample variance with our theoretical number.

Question 1(c)

Compute the bootstrap confidence interval (what width?) for the sample mean and sample standard deviation.

Theory

In this question, I will compute the bootstrap confidence interval for the sample mean and sample standard deviation, the following formula is for calculate the sample mean and variance. The following code is using for plotting the confidence interval and compute 2.5 and 97.5 percent of mean and std. I will repeat the experiment with 1000 times, therefore, there are 1000 different means and variance. I will use the code from part a for this part. Using resample() function to re-sample the means and variance.

Algorithm 3: Bootstrap confidence interval for the sample mean and sample standard deviation

```
/* Formula for mean and variance */
```

$$Xn = \frac{1}{n} * (\sum_{k=1}^{n} Xk) \tag{3}$$

$$S^2 = \frac{1}{n-1} * (\sum_{k=1}^{n} (Xk - Xn)^2) \tag{4}$$

```
/* Plot the re-sample mean and re-sample variance */
```

$\mathrm{X} = \mathrm{resample}(\mathrm{arr}_mean, n_samples = len(arr_mean), replace = True)$

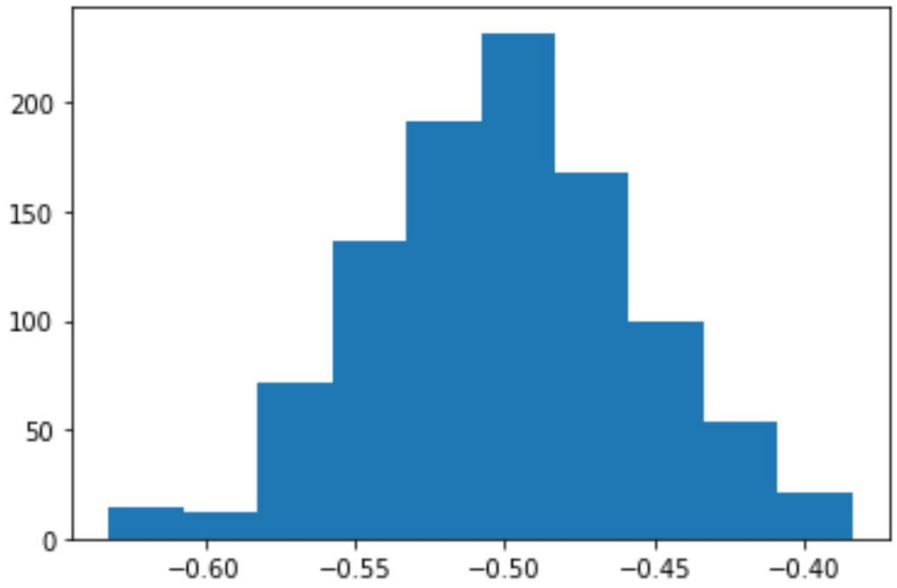
$\mathrm{X2} = \mathrm{resample}(\mathrm{arr}_std, n_samples = len(arr_std), replace = True)$

plt.hist(X)

```
/* Code for calculate the percentiles */
```

$\mathrm{per}_std1 = np.percentile(X2, (100 - a)/2, interpolation =' nearest')$

$\mathrm{per}_std2 = np.percentile(X2, (100 + a)/2, interpolation =' nearest')$




```
The  2.5 % percentile of mean is:  -0.5849706518950519
The  97.5 % percentile of mean is:  -0.4175611827857542
```

Figure 2: Confidence interval for sample mean

```
The  2.5 % percentile of std is:  1.4020919677173678
The  97.5 % percentile of std is:  1.4819601782565495
```

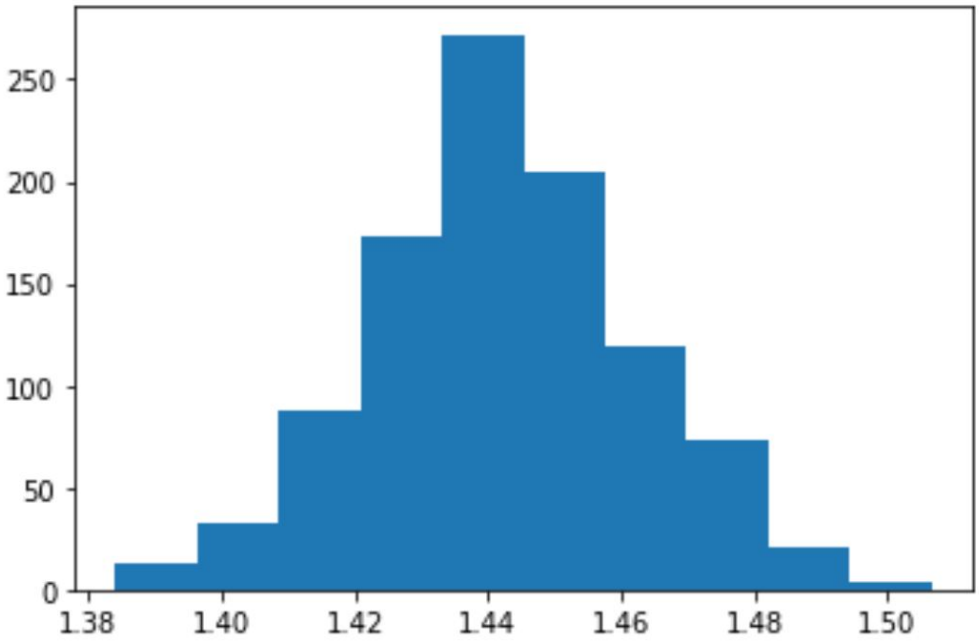


Figure 3: Confidence interval for Std

Explanation

From our experiment, Using the formula 3 and 4, and programming the code for getting the answer 2.5 percentile of mean is -0.5497, the 97.5 percentile of mean is -0.41756, the 2.5 percentile of std is 1.40209, the 97.5 percentile of std is 1.48196. This is a reasonable number for the experiment.

Question 2

Produce a sequence X by drawing samples from a standard uniform random variable

Question 2(a)

Compute Cov[Xk,Xk+1].Are Xk and Xk+1 uncorrelated? What can you conclude about the independence of Xk and Xk+1.

Theory

In this question, I will compute the covariance for our sample data. First I generate 1000 random number as your Xk data, then I use np.roll() function to shift the data to the left with one position and fill the last position with a 0. Finally I use np.cov() function to compute the covariance.The out put for the covariance are shows in below

Algorithm 4: Code for compute Covariance for Xk and Xk+1

Input: Number$_s amplingn$=10000
Output: Covariance data

```
/* Generate 1000 random number                                    */
number_value = 1000; arr = [];
for i from 0 to 1000 do
    arr.append(random.random())
/* Compute the Covariance, and shift to left with one position    */
k=np.roll(arr,-1);
k[number_value − 1] = 0;
/* Print the output                                               */
print(ans)
print(ans[0][1])
/* Order of output                                                */
( Cov(A,A)  Cov(A,B) )
( Cov(B,A)  Cov(B,B) )
```

```
[[ 0.08018419 -0.0027201 ]
 [-0.0027201   0.08036043]]
-0.0027200971292824965
```

Figure 4: Covariance output

Explanation

From the line 8 in our algorithm, the number of the covariance that we want is (1,2). Covariance of Xk and Xk+1 is -0.0006073530463881886. The number is very close to 0. Therefore, Xk and Xk+1 uncorreltated, but we cannot conclude about the independence of Xk and Xk+1. @e only can conclude xk and xk+1 is uncorrelated when they are independent, but

we cannot make a conclusion with opposite way

Question 2(b)

Compute a new sequence Y[k] where: Y[k]=X[k]-2X[k-1]+0.5X[k-2]-X[k-3], Assume X[k]=0 for k<=0. Compute Cov[Xk,Yk]. Are Xk and Yk correlated?

Theory

This question has similar process with last question, but for this question, we are not going to shift the position, we are going to make a new function Y[k]. Before to make this equation, we are extend the X[k] to three different X1[k],X2[k],X3[k], and make the first few element to be 0. The following code will use to explain the idea of extension. Finally, I will discuss the uncorrelated relation in the explanation part

Algorithm 5: Code for compute Covariance for Xk and Xk+1

Input: Number$_s amplingn$=10000
Output: Covariance data

```
/* Generate 1000 random number                                    */
number_value = 1000; arr = [];
for i from 0 to 1000 do
    arr.append(random.random())
/* Copy the number from X and make few number to be 0 at front    */
X_1 = copy.copy(X); X_2 = copy.copy(X); X_3 = copy.copy(X);
X_1[0] = 0;
X_2[0] = 0; X_2[1] = 0;
X_3[0] = 0; X_3[1] = 0; X_3[2] = 0;
/* Build the function Y[k]                                        */
y=[]
for i from 0 to 1000 do
    sum = X[j]-2*X_1[j] + 0.5 * X_2[j] - X_3[j]
    y.append(sum)
/* Order of output                                                */
( Cov(A,A)  Cov(A,B) )
( Cov(B,A)  Cov(B,B) )
```

```
[[ 0.08366754 -0.12557006]
 [-0.12557006  0.18946306]]
-0.12557005952759026
```

Figure 5: Covariance output

Explanation

The reason of building those three X1, X2, X3 array, it's because X[k]=0 for k<=0, I want to avoid the boundary error for our question, the answer of our covariance is around -0.12557005952759026, it's not perfectly close to 0, therefore, Xk and Yk is correlated.

Question 3

Let M = 10. Simulate (uniform) sampling with replacement from the outcomes 0, 1, 2, 3, . . . , M-1.

Question 3(a)

Generate a histogram of the outcomes.

Theory

This is similar with question 1, but in this question, we will generate 10000 times from number from 0 to 9. I will generate a histogram of the outcomes.

Algorithm 6: Code for compute Covariance for Xk and Xk+1

```
/* Histogram of the outcomes                                    */
M=10; number_count = 100; nums = np.random.randint(M, size = number_count)
bins=np.arange(M+1)-0.5
N, bins, patches = plt.hist(nums, bins, alpha=0.75)

```

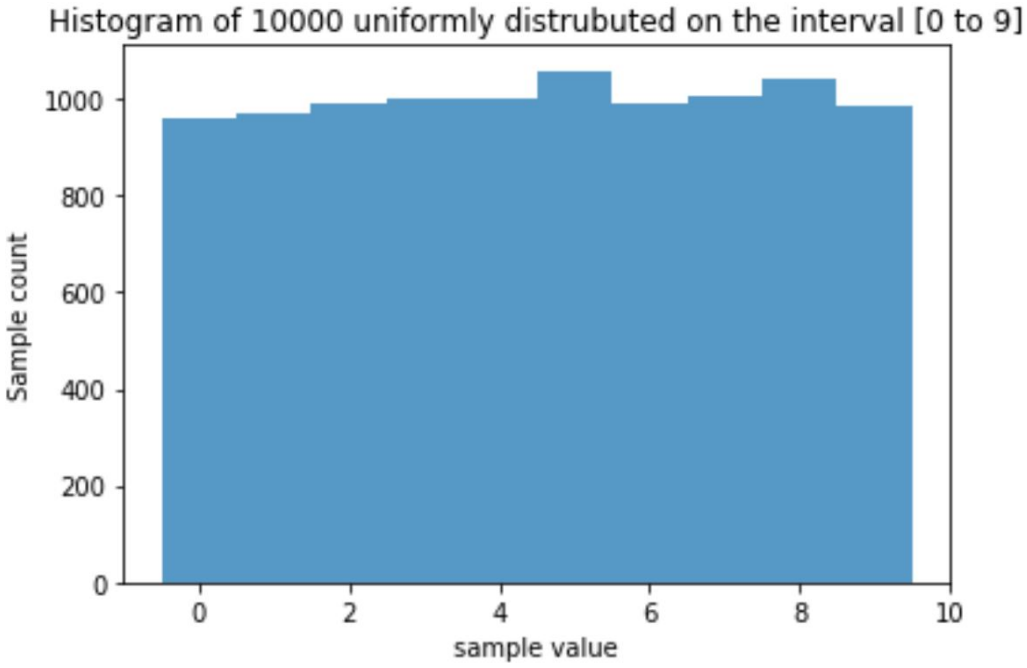


Figure 6: Histogram of the outcomes

Explanation

This graph is similar with our Figure 1, but the range is changed for our question. The range for our question is 0 to 9. As the number of sampling increasing, the graph will more likely to be uniform.

Question 3(b)

Perform a statistical goodness-of-fit test to conclude at the 95 percent confidence level if your data fits samples from a discrete uniform distribution 0, 1, 2, . . . , 9.

Theory

In this question, we are going to make a statistical goodness of fit test to conclude at the 95 percent confidence level if the data samples from 0 to 9. Below is the code for this question, and I will discuss the conclusion in the explanation part. First, I will use the formula to compute chisquare. The formula is below. Then I will use chi2.ppf() to determine our theoretical result. Finally, I will compare those two answers to determine whether the line is good or bad.

```
   /* Formula for compute chi-square test statistic                    */
1
```

$$X^2 = \sum \frac{(observed - expected)^2}{expected} \tag{5}$$

```
   /* chi-square test statistic                                        */
2  for i from 0 to 10 do
3      di=x[j]-x_theo[j]
4      di2=di*di
5      out=di2/x_theo[j]
6      sum1 = sum1+out
   /* Compute theoretical result                                       */
7  chi2.ppf(0.95,len(x)-1)
```

```
[987, 1006, 977, 1047, 1037, 999, 1005, 968, 980, 994]
[1000. 1000. 1000. 1000. 1000. 1000. 1000. 1000. 1000. 1000.]
5.798000000000001
Chi square test statistic is:  5.798000000000001
Chi square test p-value is:  0.75995272785559
The 95 percentile of chi square dist with 9 dof, is: 16.918977604620448
good fit
```

Figure 7: Goodness of fit test output

Explanation

The result is our data is good fit at the 95 percent confidence level. I tried many test result, sometime, the data will give us a bad fit at the 95 percent confidence level, When the data is not that uniform, the data will show us the bad fit.

Question 3(c)

Repeat (b) to see if your data (the same data from b) instead fit an alternate uniform distribution 1, 2, 3, . . . , 10.

Theory

This is the same step with the last question. But we are going to make a statistical goodness of fit test to conclude at the 95 percent confidence level if the data samples from 1 to 10. I

will use equation 5 and same code to do the program in this question and I will discuss the conclusion in the explanation part. First, I will use the formula to compute chisquare. The formula is below. Then I will use chi2.ppf() to determine our theoretical result. Finally, I will compare those two answers to determine whether the line is good or bad.

```
[0, 17, 4, 10, 13, 9, 7, 10, 9, 13]
[10. 10. 10. 10. 10. 10. 10. 10. 10. 10.]
21.4
Chi square test statistic is:  21.4
Chi square test p-value is:  0.010988016145239983
The 95 percentile of chi square dist with 9 dof, is: 16.918977604620448
bad fit
```

Figure 8: Goodness of fit test output

Explanation

The result is our data is bad fit at the 95 percent confidence level. Therefore, the data is not good sample to determine the result. I tried many examples, sometimes, the sample data will give me a good fit for this equation. I think the goodness of fit test is depend on the sample data of questions.