Project 2

Wenjun Li

February 5, 2020

Question 1

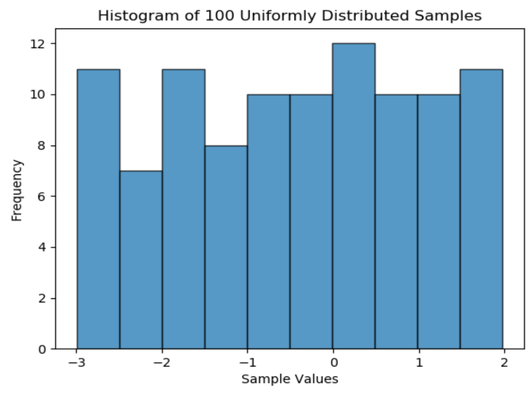
Simulate sampling uniformly (how many?) on the interval [-3, 2].

a. Generate a histogram of the outcomes.

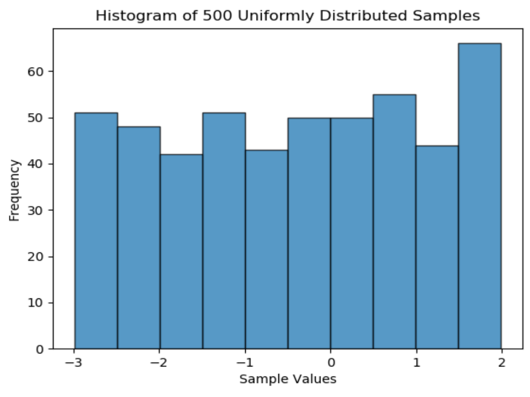
b. Compute the sample mean and sample variance for your samples. How do these values compare to the theoretical values? If you repeat the experiment will you compute a different sample mean or sample variance?

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

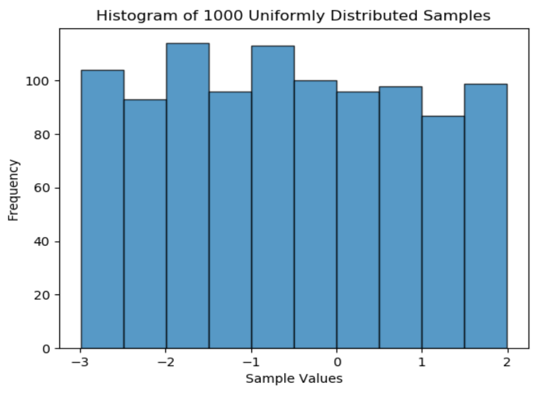
Part (a) - Results and Analysis



(a)



(b)



(c)

Figure 1: Histograms of (a)100, (b)500 and (c)1000 Uniformly Distributed Samples.

In this simulation, I randomly sample 100, 500, and 1000 uniformly distributed data points on the interval [-3, 2] and plot the corresponding histograms in Figure 1. As the histograms suggest, the frequency distribution is closer to uniform when the number of sampled data points increases.

Part (b) - Results and Analysis

Here, I use the results from 1000 uniformly distributed data points. The sample mean is -0.4725 and the sample variance is 2.1624. In this simulation, the theoretical mean and theoretical variance would be as follows:

$$TheoreticalMean = (b + a)/2 = -0.5 \tag{1}$$

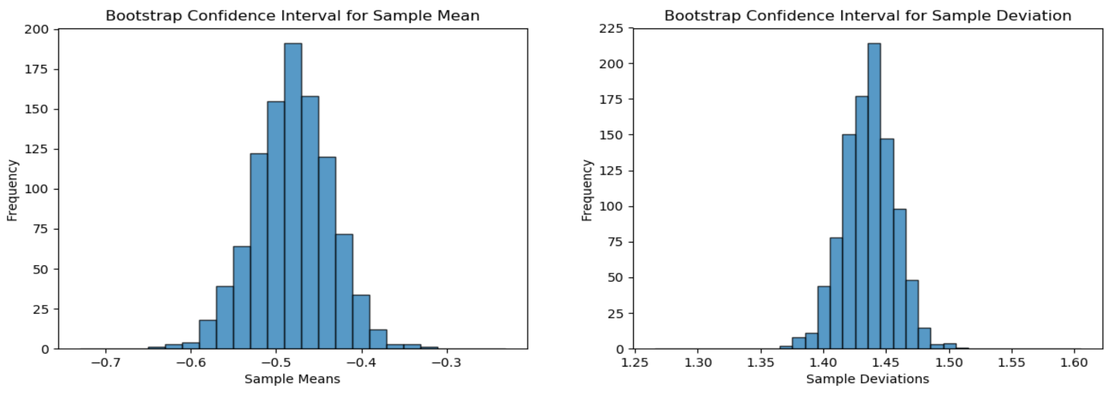
$$TheoreticalVariance = (b - a)^2/12 = 2.0833 \tag{2}$$

When repeating the experiment, the sample mean and sample variance will be slightly different, but the difference is quite small and acceptable. When I use 100 samples to repeat the experiment, the difference between sample mean and sample variance is larger, which again shows that the 1000-sample experiment is more stable and closer to uniform distribution.

Part (c) - Results and Analysis

To compute the Bootstrap confidence interval, we need to decide the number of resampled sequences. Here, I set N = 1000, the same number as the sequence length. The confidence interval width used is 95%, which is the standard width. The 2.5% percentile mean is: -0.5705, the 97.5% percentile mean is: -0.3954; the 2.5% percentile std is: 1.3969, and the 97.5% percentile std is: 1.4741.

The plots of sample mean and sample variance are shown in Figure 2.



(a) (b)

Figure 2: Histograms of Bootstrap CI for (a)Sample Mean and (b)Sample Deviation.

Question 2

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

a. Compute $Cov[X_k, X_{k+1}]$. Are X_k and X_{k+1} uncorrelated? What can you conclude about the independence of X_k and X_{k+1}?

b. Compute a new sequence Y where: $Y[k] = X[k] - 2 \cdot X[k-1] + 0.5 \cdot X[k-2] - X[k-3]$. Assume $X[k] = 0$ for $k \leq 0$. Compute $Cov[X_k, Y_k]$. Are X_k and Y_k uncorrelated?

Part (a) - Results and Analysis

The covariance measures the linear relationship between a pair of variables or sequences, and the correleation coefficient (denoted as ρ) is the measuring metric. A low covariance implies low correlation while a large covariance implies high correlation. The covariance and correlation equations are shown below:

$$Covariance : Cov(X,Y) = E[(X - E[X])(Y - E[Y])] \tag{3}$$

$$Correlation : \rho_{X,Y} = \frac{Cov(X,Y)}{\sigma_X \sigma_Y} \tag{4}$$

Note that, if the covariance is zero, then correlation is zero, and this can tell us X and Y are uncorrelated (*i.e.* linearly uncorrelated). But, the ρ cannot tell us about the independence (specifically, non-linear dependence) between X and Y, because there might be non-linear dependence between them, which cannot be implied from correlation.

In this simulation, I set the sequence length as 100. X_k is a sequence of 100 uniformly distributed random variables, and X_{k+1} is sequence that shifted by one unit from X_k. The simulation outcomes are as follows:

$$Cov[X_k, X_{k+1}] = \text{-0.0008}$$
$$\rho_{X,X_{k+1}} = \text{-0.0105}$$

We can conclude that X_k and X_{k+1} are uncorrelated and linerarly independent because their covariance and correlation coefficient are very close to zero. This is expected since X_k and X_{k+1} are generated independently.

Part (b) - Results and Analysis

In part (b), I first generate X_k and then shift X_k by one, two and three units to obtain X_{k-1}, X_{k-2} and X_{k-3}, repectively. For $k \leq 0$, the sequences are padded with zeros. The simulation outcomes are as below:

$$Cov[X_k, Y_k] = 0.0889$$
$$\rho_{X,X_{k+1}} = 0.4514$$

From the above results, we can conclude that X_k and Y_k are correlated because the values are not zero. To analyze this, first we should see that Y_k and X_k are dependent and they should produce a correlation coefficient of 1. Second, X_{k-1}, X_{k-2} and X_{k-3} will produce some corruption to the coeeficient, because they have no dependence with X_k, and thus no dependence with Y_k. But still, the result implies a positive dependence between X_k and y_k, which is expected.

Question 3

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

a. Generate a histogram of the outcomes.

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

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

Part (a) - Results and Analysis

The result is shown in Figure 3. I choose N = 10000 samples and the histogram is very close to uniform distribution.

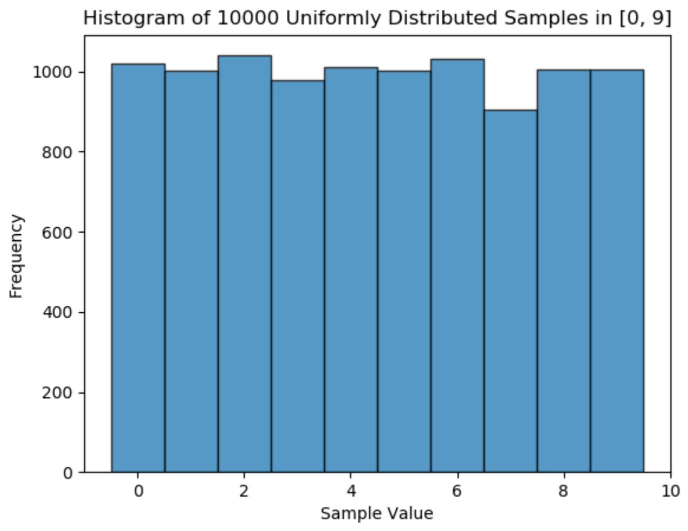


Figure 3: Histogram of 10000 Uniformly Distributed Samples in [0, 9].

Part (b) - Results and Analysis

To computer χ^2 goodness of fit, we need to count the number of samples in each histogram bin and compute the expected count for each bin. In this experiment, the expected count is 1000 (*i.e.* N/M = 10000/10 = 10) and the degree of freedom (dof) is (M-1) = 9. Theoretically, the 95 percentile of chi square distribution with 9 dof is 16.929. The experiment results are as follows:

chi square test statistic = 12.668
chi square test p-vale = 0.178

From the results, we conclude that our outcome is a good fit because it is less than the expected value (16.929).

Part (c) - Results and Analysis

In part (c), we change the uniform distribution to 1, 2, ..., 10 and repeat the experiment with the same data from part (a). Thus, we will have zero observed sample in the bin of `10` and zero expected sample in the bin of `0`. The results are as below:

chi square test statistic = 1005.136
chi square test p-vale = 1.346e-210

From the results, we conclude that our outcome cannot fit the new data because the chi square test statistic is much larger than the expected value.

Code Input

Listing 1: Code Input for Question 1

```
# Question 1
import numpy as np
import matplotlib.pyplot as plt


# part (a)
N = 1000     # switch N among: 100, 500, 1000
samples = np.random.uniform(−3, 2, N)
counts, bins, ignored = plt.hist(samples, edgecolor='black', alpha=0.75)

plt.figure(1)
plt.title(Histogram of  Uniformly Distributed Samples.format(N))
plt.xlabel(Sample Values)
plt.ylabel(Frequency)
plt.show()


# part (b)
sample_mean = np.mean(samples)
sample_variance = np.var(samples)

print(sample mean =  and sample variance = .format(sample_mean, sample_variance))


# part (c)
from sklearn.utils import resample


N_resample = 1000       # number of resample
std = []
mean = []

# re−sample, compute corresponding mean & std, and store them
for i in range(N_resample):
    re_samples = resample(samples, n_samples=len(samples), replace=True)

    mean.append(np.mean(re_samples))
    std.append(np.std(re_samples))

# sort mean and std
mean = np.sort(mean)
```

```
std = np.sort(std)


# compute 95% confidence interval
a = 95
per_1_mean = np.percentile(mean, (100-a)/2, interpolation='nearest')
per_2_mean = np.percentile(mean, (100+a)/2, interpolation='nearest')
print('The ', str((100-a)/2), '% percentile mean is: ', str(per_1_mean))
print('The ', str((100+a)/2), '% percentile mean is: ', str(per_2_mean))

per_1_std = np.percentile(std, (100-a)/2, interpolation='nearest')
per_2_std = np.percentile(std, (100+a)/2, interpolation='nearest')
print('The ', str((100-a)/2), '% percentile std is: ', str(per_1_std))
print('The ', str((100+a)/2), '% percentile std is: ', str(per_2_std))


# plot
plt.figure(2)
bins = np.arange(min(mean) - 0.1, max(mean)+0.1, 0.02)
plt.hist(mean, bins, edgecolor='black', alpha=0.75)
plt.title(Bootstrap Confidence Interval for Sample Mean)
plt.ylabel(Frequency)
plt.xlabel(Sample Means)
plt.show()

plt.figure(3)
bins = np.arange(min(std) - 0.1, max(std)+0.1, 0.01)
plt.hist(std, bins, edgecolor='black', alpha=0.75)
plt.title(Bootstrap Confidence Interval for Sample Deviation)
plt.ylabel(Frequency)
plt.xlabel(Sample Deviations)
plt.show()
```

Console Output:

```
sample mean = -0.4809508864628063 and sample variance = 2.0667579463453083
The  2.5 % percentile mean is:  -0.5705873122643091
The  97.5 % percentile mean is:  -0.3954027267514725
The  2.5 % percentile std is:  1.3969358452225769
The  97.5 % percentile std is:  1.4741610748960001
```

Listing 2: Code Input for Question 2

```
import numpy as np

```

```

# part (a)
N = 100
X = np.random.uniform(0, 1, N+1)        # generate n uniformly distributed samples

X_k = X[:N]                             # samples of: 0 ~ N
X_k_plus_1 = X[1:]                      # samples of: 1 ~ N+1

cov = np.sum((X_k - np.mean(X_k)) * (X_k_plus_1 - np.mean(X_k_plus_1))) / N
corr = cov / (np.std(X_k) * np.std(X_k_plus_1))
print('the covariance between Xk and Xk+1 is: {}'.format(cov))
print('the correlation between Xk and Xk+1 is: {}'.format(corr))


# part (b)
X_k_minus_1 = X[:N-1]
X_k_minus_2 = X[:N-2]
X_k_minus_3 = X[:N-3]

X_k_minus_1 = np.insert(X_k_minus_1, 0, [0])        # padding 0
X_k_minus_2 = np.insert(X_k_minus_2, 0, [0, 0])
X_k_minus_3 = np.insert(X_k_minus_3, 0, [0, 0, 0])

Y = X_k - 2*X_k_minus_1 + 0.5*X_k_minus_2 - X_k_minus_3

cov_XY = np.sum((X_k-np.mean(X_k)) * (Y-np.mean(Y))) / N
corr_XY = cov_XY / (np.std(X_k) * np.std(Y))
print('the covariance between XK and Yk is: {}'.format(cov_XY))
print('the correlation between Xk and Yk is: {}'.format(corr_XY))
```

Console Output:

```
the covariance between Xk and Xk+1 is: -0.000799562551825313
the correlation between Xk and Xk+1 is: -0.010538704878738196
the covariance between XK and Yk is: 0.08892730612477137
the correlation between Xk and Yk is: 0.4514282553175321
```

Listing 3: Code Input for Question 3

```
import numpy as np
import matplotlib.pyplot as plt


# part (a)
M = 10
```

```
N = 10000        # number of samples
sample = np.random.randint(0, M, N)


# plot
plt.figure(1)
bins = np.arange(0, M+1, 1) - 0.5
plt.hist(sample, bins, edgecolor=black, alpha=0.75)
plt.title(Histogram of 10000 Uniformly Distributed Samples in [0, 9])
plt.xlabel(Sample Value)
plt.ylabel(Frequency)
plt.show()


# part (b)
from scipy.stats import chisquare
from scipy.stats import chi2


dof = M - 1     # degree of freedom
x = np.array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0])     # store original data
x_theo = np.array([1000, 1000, 1000, 1000, 1000, 1000, 1000, 1000, 1000, 1000])
        # expected = N / M = 1000

for i in range(M):
    x[i] = np.count_nonzero(sample == i)

print(x)
print(x_theo)

chi_sq_res = chisquare(x, x_theo)
print('chi square test statistic is {}'.format(chi_sq_res.statistic))
print('chi square test p-vale is: {}'.format(chi_sq_res.pvalue))
print('The 95 percentile of chi square dist with 9 dof, is: {}'.format(chi2.ppf
    (0.95, dof)))


# part (c)
sample_new = np.random.randint(1, M+1, N)       # generate new samples
x_new = np.array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0])

for i in range(M):
    x_new[i] = np.count_nonzero(sample_new == i)

```

```python
print(x_new)
print(x_theo)

chi_sq_res_new = chisquare(x_new, x_theo)
print('chi square test statistic is {}'.format(chi_sq_res_new.statistic))
print('chi square test p-vale is: {}'.format(chi_sq_res_new.pvalue))
print('The 95 percentile of chi square dist with 9 dof, is: {}'.format(chi2.ppf
    (0.95, dof)))
```

Console Output:

```
[1020 1002 1040  979 1011 1001 1032  905 1006 1004]
[1000 1000 1000 1000 1000 1000 1000 1000 1000 1000]
chi square test statistic is 12.668
chi square test p-vale is: 0.17821324369628383
The 95 percentile of chi square dist with 9 dof, is: 16.918977604620448
[   0  989  999 1007 1023  961  977 1036  999  967]
[1000 1000 1000 1000 1000 1000 1000 1000 1000 1000]
chi square test statistic is 1005.136
chi square test p-vale is: 1.3460385843473524e-210
The 95 percentile of chi square dist with 9 dof, is: 16.918977604620448
```