Kevin Long

Professor Chassiakos

EE 381

22 April 2019

Project 5: Confidence Intervals

1) Effect of Sample Size on Confidence Intervals

a) Introduction

Assume that you are measuring a statistic in a large population of size *N*. The statistic has mean μand standard deviation σ; drawing a sample of size *n* from the population, produces a distribution for the sample mean (X*)* with:

$$E[\bar{X}] = \mu_{\bar{X}} = \mu \quad \text{and} \quad E[(\bar{X}-\mu_{\bar{X}})^2] = \sigma_{\bar{X}}^2 = \frac{\sigma^2}{n} \Rightarrow \sigma_{\bar{X}} = \frac{\sigma}{\sqrt{n}}$$

In this project we will explore the relation of *X* to the population mean μ.

• As a first example consider a barrel of a million ball bearings (i.e. population size *N* = 1,000,000) where someone has actually weighed all one million of them and found the exact mean to be μ= 100 grams and the exact standard deviation to be σ = 12 grams. This is obviously an unrealistic assumption, but assume for the time being that these parameters have been measured exactly.

• Now pick a sample of size *n* (for example *n* = 5) of bearings from the barrel, weigh them and find the mean of the sample

$$\bar{X}_5 = \frac{X_1 + X_2 + X_3 + X_4 + X_5}{5}.$$

• Next take a larger sample (for example *n* = 10) and find the new mean

$$\bar{X}_{10} = \frac{X_1 + X_2 + \cdots + X_{10}}{10}.$$

• Continue this process for larger and larger *n*, until *n* = 100.

• Next for each value of *n*, calculate the standard deviation of the sample from

$$\sigma_{\bar{X}_n} = \frac{\sigma}{\sqrt{n}}$$ and plot:

(i) The values of $u \pm 1.96\frac{\sigma}{\sqrt{n}}$ as a function of *n*

(ii) The values of $\mu \pm 2.58\frac{\sigma}{\sqrt{n}}$ as a function of *n*

b) Methodology

The libraries numpy, matplotlib.pyplot, math, and random are imported to help with calculations. There are five lists that are created: “mean,” “top95,” “bottom95,” “top99,” and “bottom99;” all initialized with *n* “none” values. For each iteration through the lists, the formulas mentioned in the introduction are applied and values are calculated and placed in their respective lists. Then, for each value in the respective lists, are then plotted as confidence intervals.

c) **Result(s) and Conclusion(s)**

Figure 1. Sample means & 95% confidence intervals

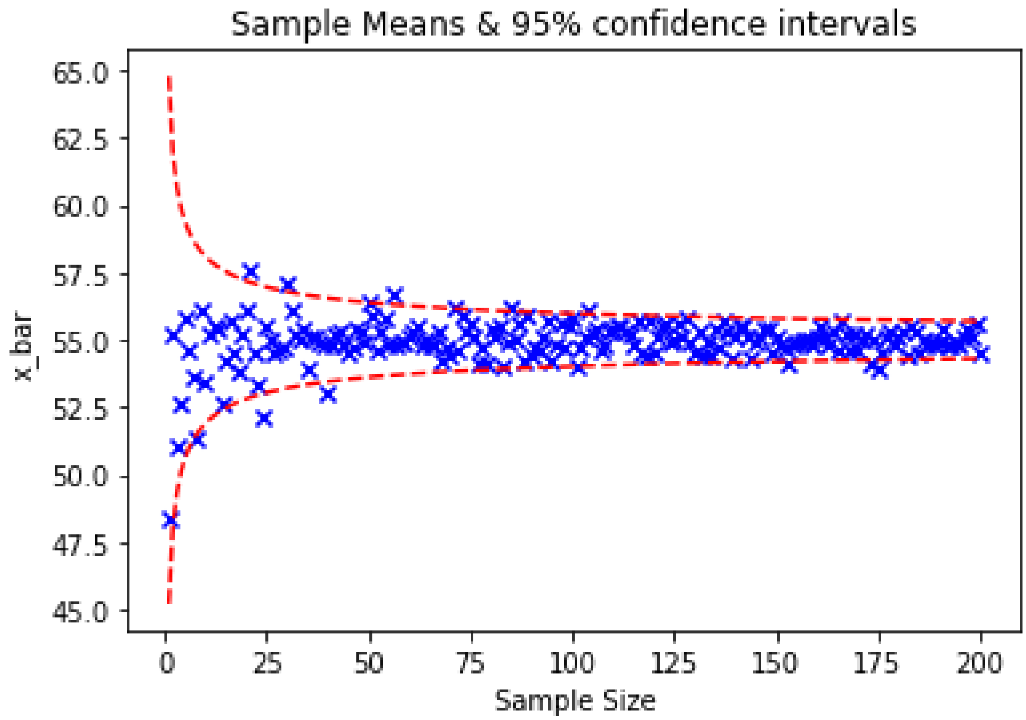
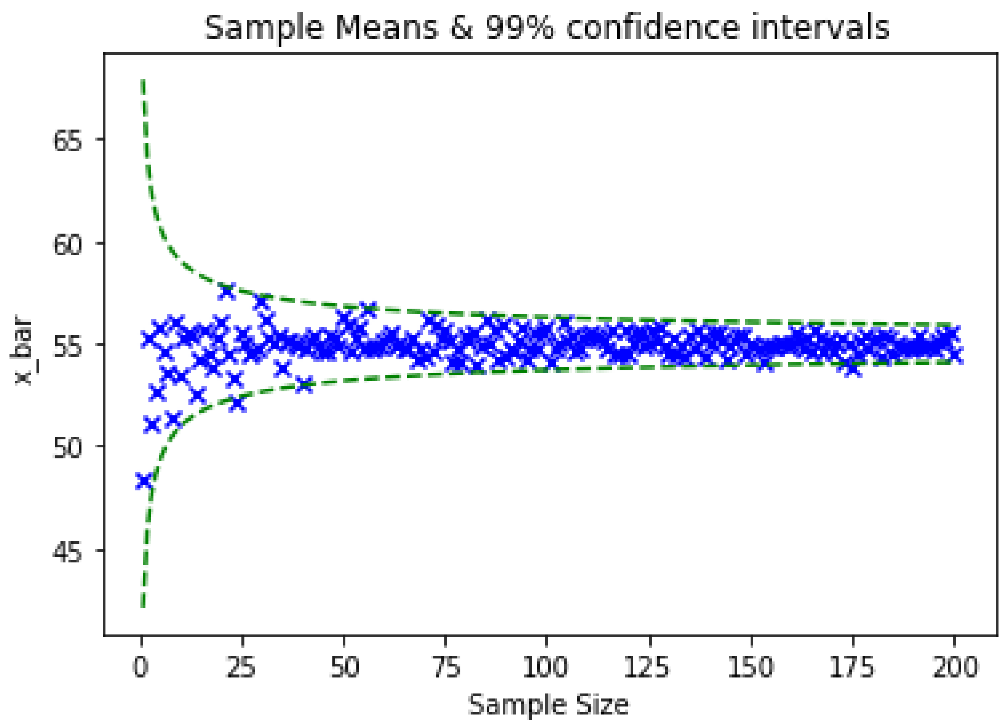


Figure 2. Sample Means & 99% confidence intervals



d) **Source Code**

```
import numpy as np
import matplotlib.pyplot as plt
import math as m
import random as r

def problem1():
    N = 1500000
    mu = 55
    sig = 5
    n = 200
    B = np.random.normal(mu, sig, N)

    mean = [None] * n
    top95 = [None] * n
    bottom95 = [None] * n
    top99 = [None] * n
    bottom99 = [None] * n

    for i in range (0, n):
        count = i + 1
        x = B[r.sample(range(N), count)]
        mean[i] = np.sum(x)/count
        std = sig/m.sqrt(count)
        top95[i] = mu + 1.96 * std
        bottom95[i] = mu - 1.96 * std
        top99[i] = mu + 2.58 * std
        bottom99[i] = mu - 2.58 * std

    listo = [x for x in range(1, count + 1)]

    fig1 = plt.figure(1)
    plt.scatter(listo, mean, c = 'Blue', marker = 'x')
    plt.plot(listo, top95, 'r--')
    plt.plot(listo, bottom95, 'r--')
    plt.title('Sample Means & 95% confidence intervals')
    plt.xlabel('Sample Size')
    plt.ylabel('x_bar')

    fig2 = plt.figure(2)
    plt.scatter(listo, mean, c = 'Blue', marker = 'x')
    plt.plot(listo, top99, 'g--')
    plt.plot(listo, bottom99, 'g--')
    plt.title('Sample Means & 99% confidence intervals')
    plt.xlabel('Sample Size')
    plt.ylabel('x_bar')
```

```
#Part 2
trials = 10000
problem2(B, N, mu, trials, 2.78, 4.6, 5)
problem2(B, N, mu, trials, 2.02, 2.7, 40)
problem2(B, N, mu, trials, 1.98, 2.62, 120)
```

2) Using the Sample Mean to Estimate the Population Mean

a) Introduction

In reference to the previous section, it is obviously unrealistic to think that anyone actually measured the exact mean and standard deviation of all one million ball bearings. More realistically, you would not have any idea what the mean or standard deviation was, and you would need to weigh random samples of different sizes (for example n = 5, 35, or 100 bearings) and then draw reasonable conclusions about the weight distribution of all one million bearings. To simulate this problem, generate a barrel of a million ball bearings with weights normally distributed, with a mean μ and a standard deviation σ. As an example, take a sample of n bearings from the population of N = 1,000,000. Then calculate the mean of the sample:

$$\bar{X} = \frac{X_1 + X_2 + \cdots + X_n}{n}$$

The standard deviation of the sample mean can be calculated by:

$$\sigma_n = \frac{\hat{S}}{\sqrt{n}} \text{ , where } \hat{S} = \left\{ \frac{(X_1 - \bar{X})^2 + (X_2 - \bar{X})^2 + \cdots + (X_n - \bar{X})^2}{n-1} \right\}^{1/2}$$

The question is: *Can the value of X, which is calculated for a sample of size n, be used to estimate the mean* μ*of the population of* N = 1,000,000 *bearings?* The answer is given in terms of confidence intervals, typically the 95% and 99% confidence interval.

b) Methodology

This portion of the project relies on the variables “B,” “N,” and “mu” defined in problem1(). A “trials” variable is initialized with 10000 to simulate experiments. Respective variables are then passed as parameters into problem2() to be calculated. The variables “successZ95,” “successZ99,” “successT95,” and “successT99” are all initialized with 0. For each trial, the formulas mentioned in the introduction are applied. After calculations have been ran, they are printed out in string format.

c) Result(s) and Conclusion(s)

Table 1. Success rate (percentage) for different sample sizes

Sample size (n)	95% Confidence (Using Normal distribution)	99% Confidence (Using Normal distribution)	95% Confidence (Using Student’s *t* distribution)	99% Confidence (Using Student’s *t* distribution)
5	0.8783	0.9405	0.9516	0.9897
40	0.9417	0.9857	0.9481	0.9896
120	0.9449	0.9881	0.9473	0.9892

d) **Source Code**

```
def problem2(b, N, mu, trials, t95, t99, size):
    successZ95 = 0
    successZ99 = 0
    successT95 = 0
    successT99 = 0
    sample = size
    for z in range (0, trials):
        y = b[r.sample(range(N), sample)]
        yMean = np.sum(y)/sample
        total = 0
        for a in range(0, len(y)):
            total = total + (y[a] - yMean) ** 2
        yS = total/(sample - 1)
        yS = m.sqrt(yS)
        yStd = yS/m.sqrt(sample)

        yTop95 = yMean + 1.96 * yStd
        yBottom95 = yMean - 1.96 * yStd
        yTop99 = yMean + 2.58 * yStd
        yBottom99 = yMean - 2.58 * yStd

        tTop95 = yMean + t95 * (yStd)
        tBottom95 = yMean - t95 * (yStd)
        tTop99 = yMean + t99 * (yStd)
        tBottom99 = yMean - t99 * (yStd)

        if yBottom95 <= mu and yTop95 >= mu:
            successZ95 += 1
        if yBottom99 <= mu and yTop99 >= mu:
            successZ99 += 1
        if tBottom95 <= mu and tTop95 >= mu:
            successT95 += 1
        if tBottom99 <= mu and tTop99 >= mu:
            successT99 += 1

    print('Success Rate using normal, sample = %d,' % sample, '95% confidence 
interval')
    print(successZ95/trials)
    print('Success Rate using normal, sample = %d,' % sample, '99% confidence 
interval')
    print(successZ99/trials)
    print('Success Rate using student t, sample = %d,' % sample, '95% 
confidence interval')
```

```
        print(successT95/trials)
        print('Success Rate using student t, sample = %d,' % sample, '99% confidence interval')
        print(successT99/trials)
print('')
```