Vinh Vu

Dr. Chassiakos

EE 381

April 22nd, 2019

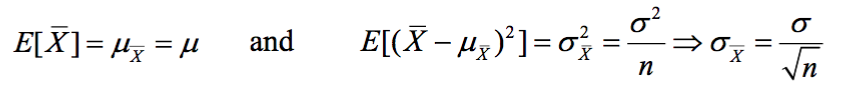
Project 5: Confidence Interval

**Problem 1: Effect of sample size on confidence intervals**

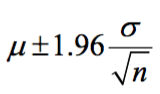
**Introduction:** The problem demonstrates the effect of sample size on confidence intervals. We slowly increase the sample size, and graph the sample size with its corresponding sample mean. Then we graph the upper and lower bounds of the standard deviation as it corresponds to the samples. The parameters are provided as follows:

* Total number of bearings: N
* Population mean: μ (grams)
* Population standard deviation: σ (grams)
* Sample sizes: n = 1, 2, …, 200

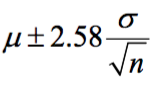
Drawing a sample size of n from the population, produces a distribution for the sample mean with:



For 95% confidence interval, the values of standard deviations are calculated as follows:

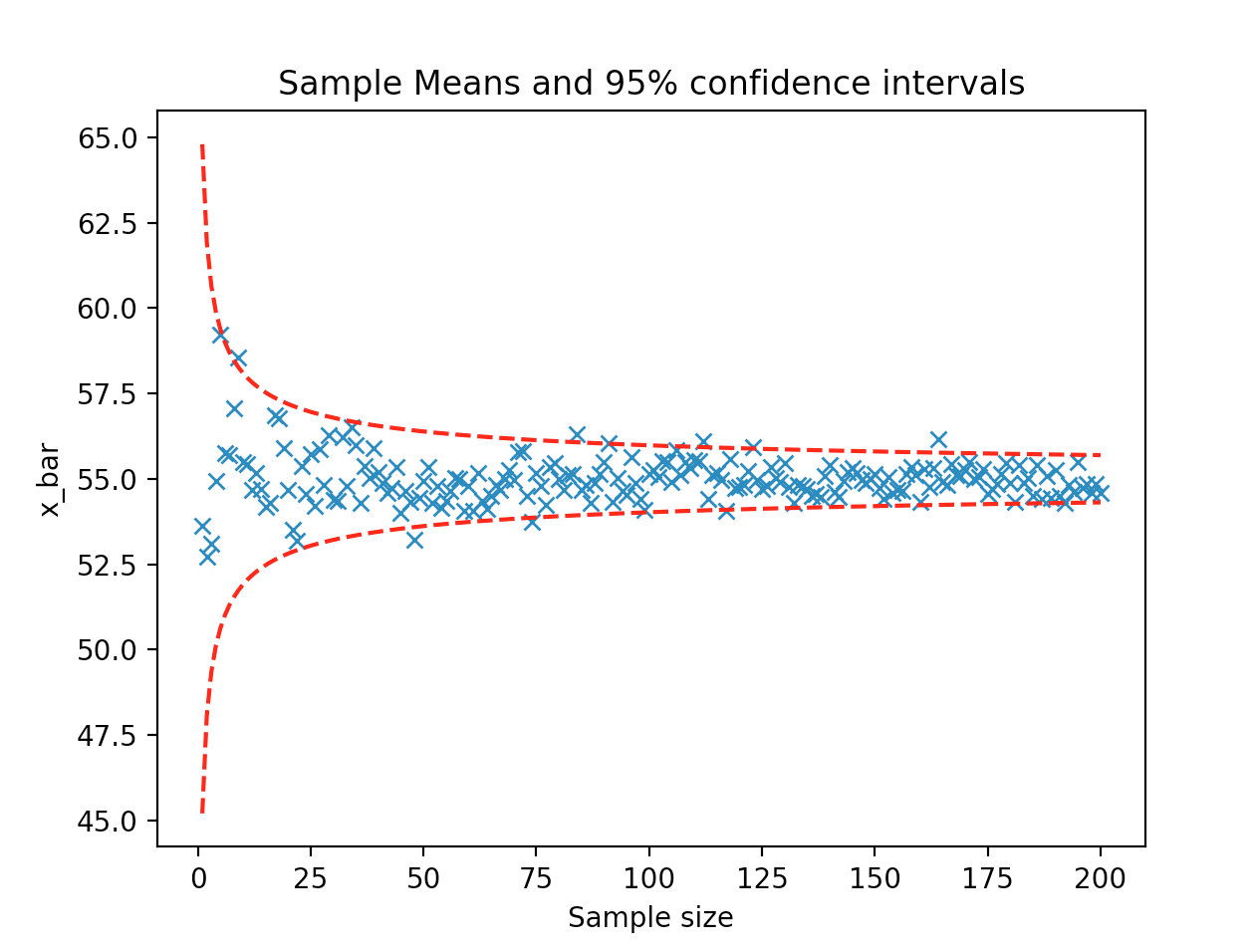


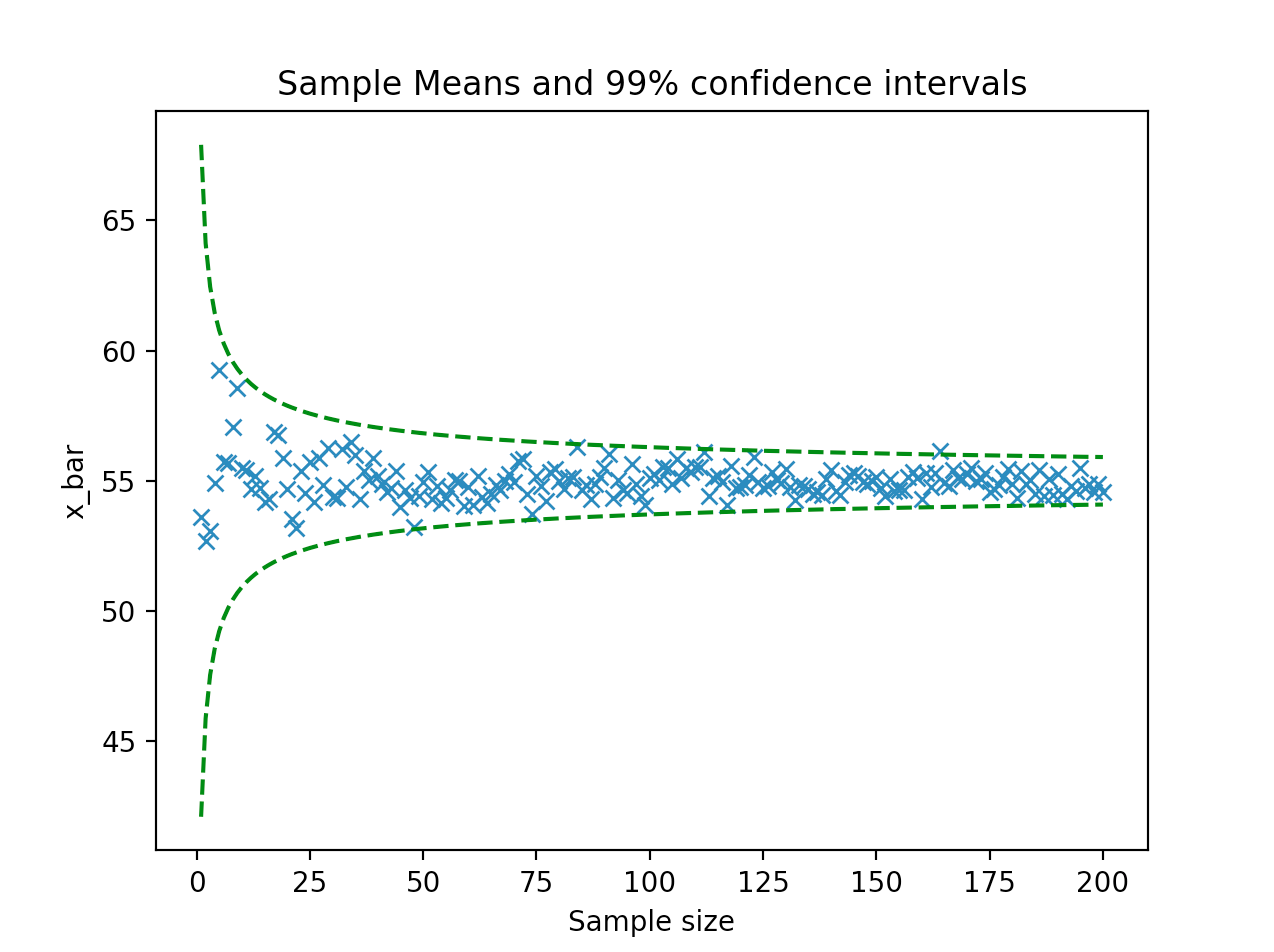
For 99% confidence interval, the values of standard deviations are calculated as follows:



**Methodology**: In order to create the samples of incrementing size, we implement a for loop for the sizes ranges from 1 to 200. For each iteration of the for loop, the sample size is randomly selected from the population of 1,500,00 created by the function generatePopulation(). Sample mean and standard deviation are calculated based on the sample size for that iteration. Positive and negative intervals for the standard deviations for 95% and 99% are saved into four lists of values. The four lists are used to generate two separate graphs: 95% confidence interval and 99% confidence interval.

**Results and Conclusion:**

****

****

**Appendix:**

import numpy as np

import random as rand

import matplotlib

import matplotlib.pyplot as plt

import math

def generatePopulation():

N = 1500000

mean = 55

sigma = 5

randomVariable =list(np.random.randn(N)\*sigma + mean)

return randomVariable

def effectSampleSize():

population = generatePopulation()

mean = 55

sigma = 5

sampleMean = []

sampleSigmaPos95 = []

sampleSigmaNeg95 = []

sampleSigmaPos99 = []

sampleSigmaNeg99 = []

n = list(range(1,201))

for size in n:

sample = list(rand.sample(population,size))

sMean = sum(sample)/size

sampleMean.append(sMean)

sSigma = sigma/math.sqrt(size)

sampleSigmaPos95.append(mean+(1.96\*sSigma))

sampleSigmaNeg95.append(mean-(1.96\*sSigma))

sampleSigmaPos99.append(mean+(2.58\*sSigma))

sampleSigmaNeg99.append(mean-(2.58\*sSigma))

plt.plot(n,sampleMean,'x')

plt.plot(n,sampleSigmaPos95,'r--')

plt.plot(n,sampleSigmaNeg95,'r--')

#plt.axhline(y=75)

plt.title("Sample Means and 95% confidence intervals")

plt.ylabel("x\_bar")

plt.xlabel("Sample size")

plt.show()

plt.plot(n,sampleMean,'x')

plt.plot(n,sampleSigmaPos99,'g--')

plt.plot(n,sampleSigmaNeg99,'g--')

#plt.axhline(y=75)

plt.title("Sample Means and 99% confidence intervals")

plt.ylabel("x\_bar")

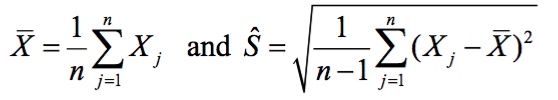
plt.xlabel("Sample size")

plt.show()

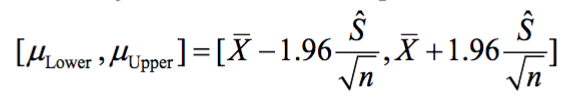
effectSampleSize()

**Problem 2:**

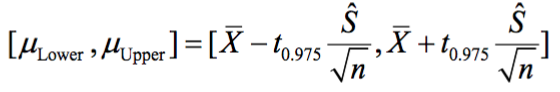
**Introduction:** In this problem, we are analyzing the accuracy of utilizing the normal distribution tables versus utilizing the student’s distribution tables for samples of different sizes. For smaller sizes, student’s normal distribution table values are more accurate, and the consistency is expected to continue for larger sampling sizes. Calculating the sample mean and sample standard deviation requires:



There are two confidence intervals to be generated: 95% and 99%. Creating the 95% confidence interval with standard deviation of:



The appropriate distribution for small samples (n 30) is the t-distribution. Creating the 95% confidence interval using t-distribution with v = n – 1 = 4



**Methodology:** There are three different sample sizes to generate the confidence intervals: size of 5, 40, and 120. For each of the sample size, we generate the sample standard deviation and the 95% and 99% confidence intervals. Each sample is considered a success when the mean of the entire population is contained in the confidence interval. For each size, we take a sample of 10,000. In addition, we generate the confidence intervals with both the normal distribution and the student’s t distribution. We then calculate the total number of successes and record the percentile on a table.

**Results and Conclusion:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **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** | 0.8763 | 0.9336 | 0.9473 | 0.9889 |
| **40** | 0.9443 | 0.9865 | 0.9513 | 0.9907 |
| **120** | 0.9439 | 0.9864 | 0.9466 | 0.9880 |

**Appendix:**

import numpy as np

import random as rand

import matplotlib

import matplotlib.pyplot as plt

import math

def generatePopulation():

N = 1500000

mean = 55

sigma = 5

randomVariable =list(np.random.randn(N)\*sigma + mean)

return randomVariable

def estimatingPopulationMean(sampleSize):

M = 10000

mean = 55

sigma = 5

success95normal = 0

success99normal = 0

success95t = 0

success99t = 0

population = generatePopulation()

count = 0

v = sampleSize - 1

while count < M:

sample = list(rand.sample(population,sampleSize))

sampleMean = sum(sample) / sampleSize

sHatCalc = [(x - sampleMean)\*\*2 for x in sample]

sHat = math.sqrt(sum(sHatCalc)/(sampleSize-1))

normal95StandardError = 1.96\*(sHat / math.sqrt(sampleSize))

normal95Interval = [sampleMean - normal95StandardError, sampleMean + normal95StandardError]

if mean >= normal95Interval[0] and mean <= normal95Interval[1]:

success95normal += 1

normal99StandardError = 2.576 \* (sHat / math.sqrt(sampleSize))

normal99Interval = [sampleMean - normal99StandardError, sampleMean + normal99StandardError]

if mean >= normal99Interval[0] and mean <= normal99Interval[1]:

success99normal += 1

if v == 4:

t95 = 2.78

t99 = 4.6

if v == 39:

t95 = 2.02

t99 = 2.7

if v == 119:

t95 = 1.98

t99 = 2.62

t95StandardError = t95 \* (sHat / math.sqrt(sampleSize))

t95Interval = [sampleMean -t95StandardError, sampleMean + t95StandardError]

if mean >= t95Interval[0] and mean <= t95Interval[1]:

success95t += 1

t99StandardError = t99\*(sHat / math.sqrt(sampleSize))

t99Interval = [sampleMean-t99StandardError, sampleMean + t99StandardError]

if mean >= t99Interval[0] and mean <= t99Interval[1]:

success99t += 1

count += 1

percentSuccess95Normal = success95normal / M

percentSuccess99Normal = success99normal / M

percentSuccess95t = success95t / M

percentSuccess99t = success99t / M

print("Percent success of 95% confidence interval normal distribution: ", percentSuccess95Normal)

print("Percent success of 99% confidence interval normal distribution: ", percentSuccess99Normal)

print("Percent success of 95% confidence interval t-distribution: ", percentSuccess95t)

print("Percent success of 99% confidence interval t-distribution: ", percentSuccess99t)

estimatingPopulationMean(120)