Project 5 - Confidence Intervals

California State University Long Beach

EE 381

Probability and Statistics

Christopher Masferrer

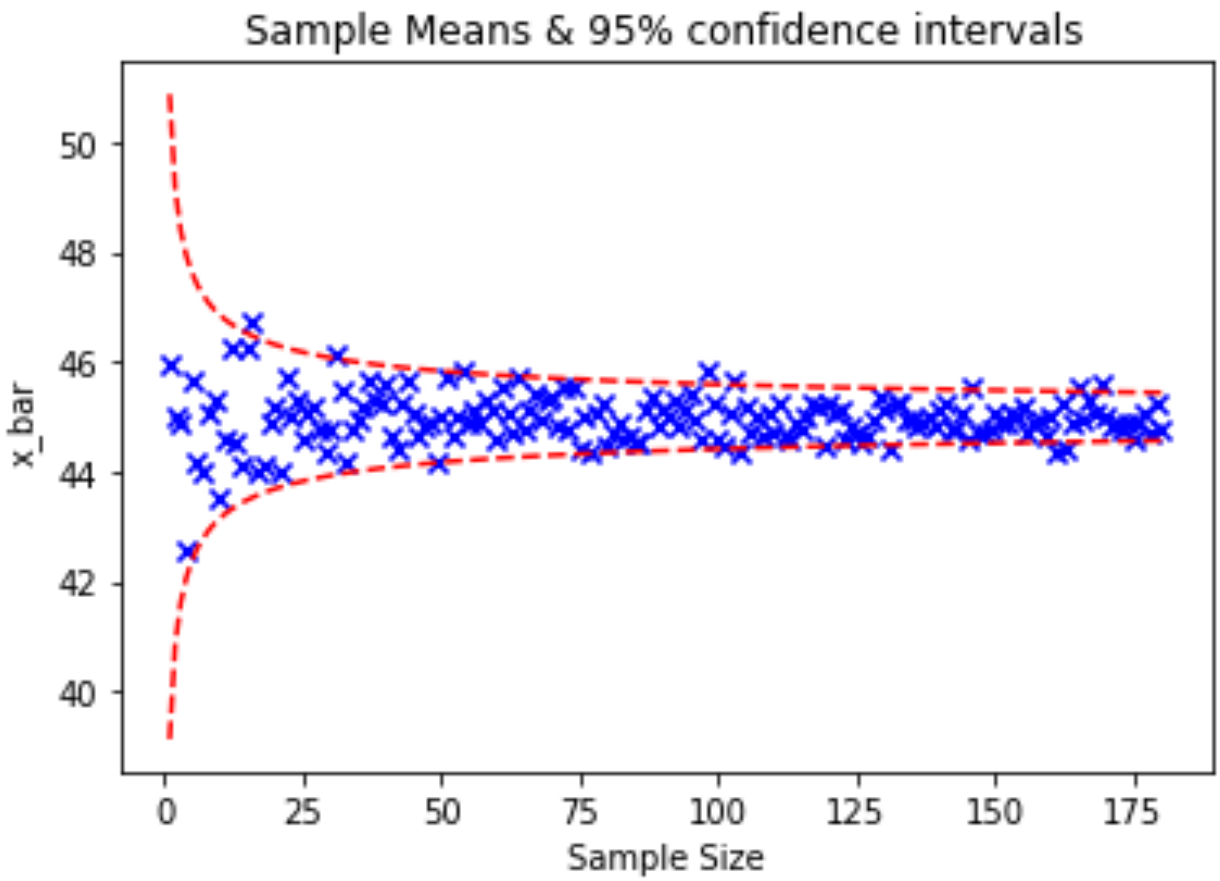
T/TH 5:30 - 6:20

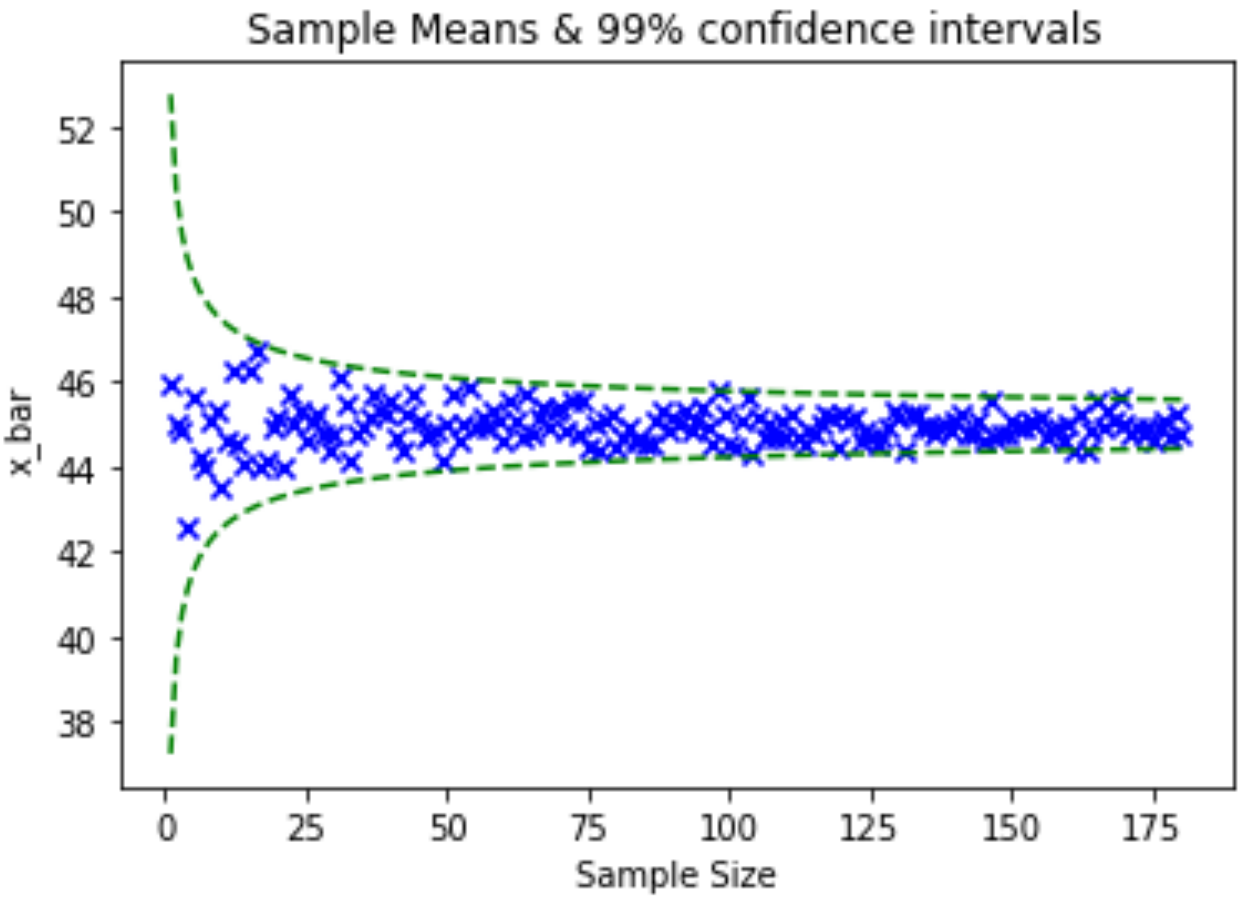
Experiment 1: Effect of sample size on confidence interval

Intro: For this experiment I will be creating two graphs. The first is a sample mean with a 95% confidence interval and the other is 99%. I will be using the provided values to generate the result.

Methodology: I created a function that generates both charts. Calculations are performed first. The top 95 and 99 percent are calculated as well as the bottom 95 and 99 percent. The charts were then generated and displayed.

Results:




Sample Means & 99% confidence intervals
x_bar
52
50
48
46
44
42
40
38
0
25
50
75
100
125
150
175
Sample Size

Code:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Thu Nov  7 13:18:04 2019

@author: christophermasferrer
"""

#Christopher Masferrer
#EE 381
#Lab 5
import numpy as np
import matplotlib.pyplot as plt
import random as r
import math as m

N = 1200000
mu = 45
sig = 3
B = np.random.normal(mu,sig,N)

def sSize():
    n = 180
    mean = [None] * n
    top95 = [None] * n
    bottom95 = [None] * n
    top99 = [None] * n
    bottom99 = [None] * n
    for i in range (0,n):
        counter = i+1
        x = B[r.sample(range(N), counter)]
        mean[i] = np.sum(x)/counter
        std = sig/m.sqrt(counter)
        top95[i] = mu + 1.96*(std)
        bottom95[i] = mu - 1.96*(std)
        top99[i] = mu + 2.58*(std)
        bottom99[i] = mu - 2.58*(std)

    coll = [x for x in range(1, counter+1)]

    plt.close('all')

    fig1 = plt.figure(1)
    plt.scatter(coll, mean, c = 'Blue', marker = 'x')
    plt.plot(coll, top95, 'r--')
```

```
    plt.plot(coll, bottom95, 'r--')
    plt.title('Sample Means & 95% confidence intervals')
    plt.xlabel('Sample Size')
    plt.ylabel('x_bar')

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

sSize()
```

Experiment 2: Using the sample mean to estimate the population mean

Intro: For this experiment I will be calculating the population mean using the sample mean that is generated from the previous experiment. The experiment will be repeated 10,000 times and the results of 95 and 99 percent confidence will be recorded for the sample sizes of 5, 40, and 120 and placed in a chart.

Methodology: The code from the previous part has been used to calculate the sample mean. The information is then sent to a new function called sMean which calculates the population mean for sizes 5, 40, 120.

Results:

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	0.8792	0.9427	0.9527	0.9899
40	0.9453	0.9858	0.9514	0.9894
120	0.9476	0.99	0.9504	0.9912

Code:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Sun Nov 10 08:21:27 2019

@author: christophermasferrer
"""

#Christopher Masferrer
#EE 381
#Lab 5
import numpy as np
import matplotlib.pyplot as plt
import random as r
import math as m

N = 1200000
mu = 45
sig = 3
B = np.random.normal(mu,sig,N)

def sSize():
    n = 180
    mean = [None] * n
    top95 = [None] * n
    bottom95 = [None] * n
    top99 = [None] * n
    bottom99 = [None] * n
    for i in range (0,n):
        counter = i+1
        x = B[r.sample(range(N), counter)]
        mean[i] = np.sum(x)/counter
        std = sig/m.sqrt(counter)
        top95[i] = mu + 1.96*(std)
        bottom95[i] = mu - 1.96*(std)
        top99[i] = mu + 2.58*(std)
        bottom99[i] = mu - 2.58*(std)

    coll = [x for x in range(1, counter+1)]

    plt.close('all')

    fig1 = plt.figure(1)
    plt.scatter(coll, mean, c = 'Blue', marker = 'x')
```

```
    plt.plot(coll, top95, 'r--')
    plt.plot(coll, bottom95, 'r--')
    plt.title('Sample Means & 95% confidence intervals')
    plt.xlabel('Sample Size')
    plt.ylabel('x_bar')

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

    sMean(2.78, 4.6, 5)
    sMean(2.02, 2.7, 40)
    sMean(1.98, 2.62, 120)

def sMean(t95, t99, size):
    trials = 10000
    successZ95 = 0
    successZ99 = 0
    successT95 = 0
    successT99 = 0
    sample = size
    for i in range (0, trials):
        y = B[r.sample(range(N), sample)]
        yMean = np.sum(y)/sample
        total = 0
        for n in range(0, len(y)):
            total = total + (y[n]-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('')

sSize()
```