SOC-GA 2332 Intro to Stats Problem Set 1 Solution

Di Zhou

2/27/2021

Contents

Load packages

```
knitr::opts_chunk$set(echo = TRUE)

library(tidyverse)
library(kableExtra)
library(gridExtra)
```

Import data

```
gapminder <- read.csv("gapminder.csv")
parent_inc <- read.csv("parent_inc.csv")
```

1. Functions

Recall the formulas for population mean:

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{1}$$

and variance:

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N} (x_i - \mu)^2 \tag{2}$$

where N is the population size.

1.1 Write a function in R that calculates the **population mean** according to Equation 1 without using any R functions that directly calculate the mean. For example, you cannot use `mean()` from base R, or `summarise(., mean = mean())` from `tidyverse`.

```
pop_mean <- function(v){    #define function name

  mean = sum(v)/length(v)   #sum of values in the vector divided by the number of values

  return(mean)              #return mean value

}
```

1.2 Write a function in R that calculates the **population variance** according to Equation 2 without using any R functions that directly calculate the variance. For example, you cannot use `var()` from base R, or `summarise(., var = var())` from `tidyverse`.

```
pop_var <- function(v){                          #defind function name

  var = sum((v - pop_mean(v))^2)/length(v) #sum of (each value - mean)^2, divided by N

  return(var)                                    #return result

}
```

1.3 Import `gapminder.csv` to your R environment.
- Apply the two functions you just created to the `lifeExp` variable in `gapminder`.
- Use R functions that directly calculate mean and variance to the same `lifeExp` variable vector.
- Report your results of the above two steps either in text or in a table. The results for the mean should be equal, but the results for variance should be different. Find out and explain why the results in variance differ.

```
# R function
mean(gapminder$lifeExp)
```

```
## [1] 59.47444
```

```
var(gapminder$lifeExp)
```

```
## [1] 166.8517
```

```
# Hand-code function
pop_mean(gapminder$lifeExp)
```

```
## [1] 59.47444
```

```
pop_var(gapminder$lifeExp)
```

```
## [1] 166.7537
```

The result for `pop_var()` differs from `var()` because in `pop_var()` we used **N, the population size instead of n-1 (sample size -1) as the denominator**.
In practice, as long as we are analyzing samples, we use the variance and standard deviation formula that has n-1 as the denominator.

2. Data Transformation Using `tidyverse`

- **Method 1**: Use `pivot_longer()` to bring out `father_name` and `mother_name`, then use `ifelse()` to assign value to a new variable called `income` that equals to the value in `fincome` when detect "father" in `type`, otherwise equals to the value in `mincome`. (We haven't covered `ifelse()` yet, but you can read its documentation in R and run the examples in the documentation to learn how to use it.)

```
# Method 1:
parent_inc %>%
  pivot_longer(c(father_name, mother_name),
               names_to = "type",
               values_to = "name") %>%
  mutate(income = ifelse(type == "father_name", fincome, mincome),
         type = str_remove(type, "_name")) %>%
  select(famid, type, name, income) %>%
  kbl(booktabs = T) %>%
  kable_styling(position = "center")
```

famid	type	name	income
1	father	Arthur	42000
1	mother	Jess	45000
2	father	Harry	35000
2	mother	Pam	24000
3	father	Matt	78000
3	mother	Mary	55000

- **Method 2**: Separate original df to `mother_df` and `father_df`, transform each df and then combine them.

```
# Method 2:

# Select mother related variables
mother_df <- parent_inc %>%
  select(famid, mother_name, mincome) %>%
  pivot_longer(c(mother_name),
               names_to = "type",
               values_to = "name") %>%
  mutate(type = str_remove(type, "_name")) %>%
  rename(income = mincome)

# Display
mother_df %>%
  kbl(booktabs = T) %>%
  kable_styling(position = "center")
```

famid	income	type	name
1	45000	mother	Jess
2	24000	mother	Pam
3	55000	mother	Mary

```
# Select father related variables
father_df <- parent_inc %>%
  select(famid, father_name, fincome) %>%
  pivot_longer(c(father_name),
               names_to = "type",
               values_to = "name") %>%
  mutate(type = str_remove(type, "_name")) %>%
  rename(income = fincome)

# Display
father_df %>%
  kbl(booktabs = T) %>%
  kable_styling(position = "center")
```

famid	income	type	name
1	42000	father	Arthur
2	35000	father	Harry
3	78000	father	Matt

```r
# Before combine, make sure all the variable names are the same in the two df
rbind(father_df, mother_df) %>%          # Use rbind to combine rows
  arrange(famid) %>%                     # Rank by family ID
  select(famid, type, name, income) %>%  # Reorder columns using select
  kbl(booktabs = T) %>%
  kable_styling(position = "center")
```

famid	type	name	income
1	father	Arthur	42000
1	mother	Jess	45000
2	father	Harry	35000
2	mother	Pam	24000
3	father	Matt	78000
3	mother	Mary	55000

3. Population, sample, and sampling distribution

3.1 Create a population data frame that has one variable called "value", whose value follows a normal distribution with population mean $\mu = 5$ and population variance $\sigma^2 = 1$ with 100,000 observations.

```r
# Set seed for replication
set.seed(1001)
# generate population df
pop <- rnorm(100000, mean = 5, sd = 1) %>% as_tibble()

# Mean & sd of population
mean(pop$value)
```

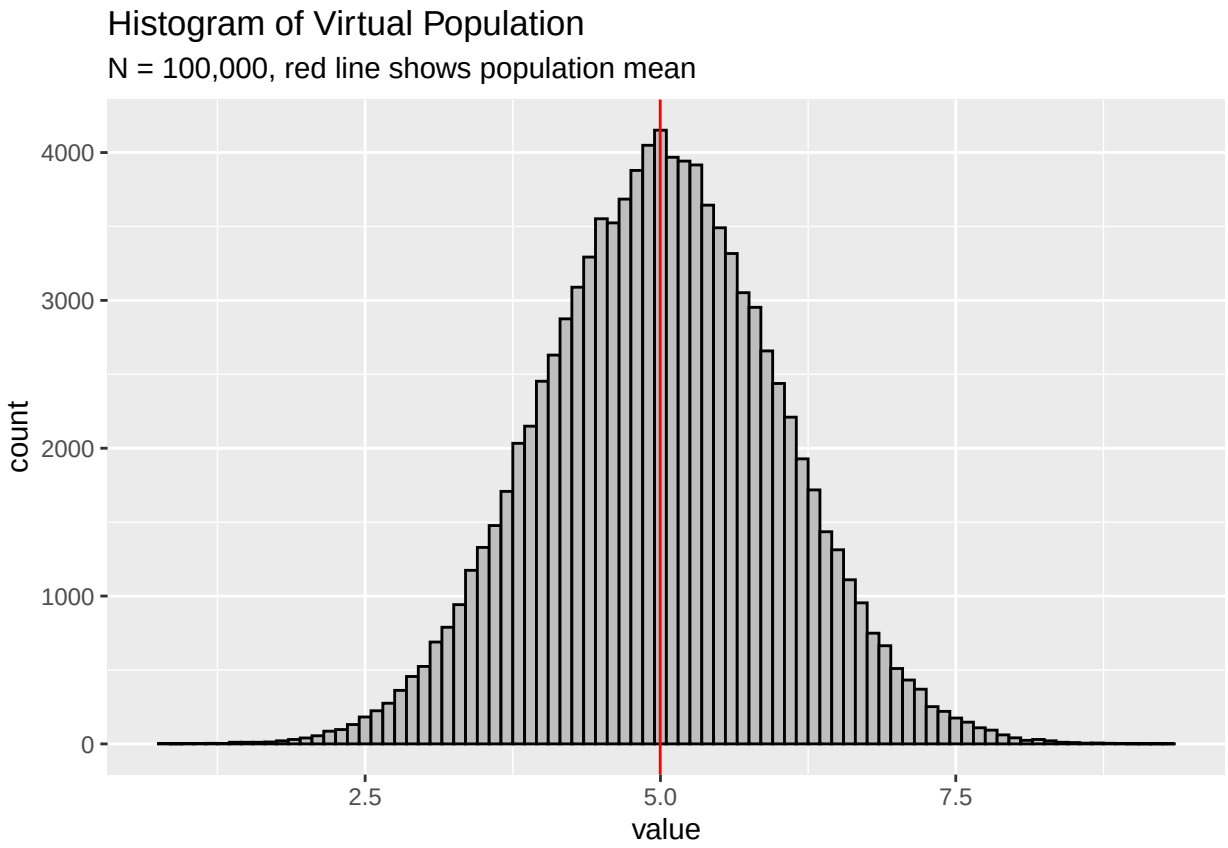
```
## [1] 4.998119
```

```r
pop_var(pop$value) %>% sqrt()
```

```
## [1] 1.000214
```

3.2 Create a histogram of the population with appropriate title and labels. Add a vertical line at the population mean.

```r
# Plot histogram
pop %>%
  ggplot() +
  geom_histogram(aes(value),
                 binwidth = 0.1,
                 fill = "grey",
                 color = "black") +
  geom_vline(xintercept = mean(pop$value), color = "red") +
  labs(title = "Histogram of Virtual Population",
       subtitle = "N = 100,000, red line shows population mean")
```



3.3 Draw a random sample from the population, with sample size n = 50.

```
# Set seed for replication
set.seed(1001)
# Draw a sample n=50 from pop, without replacement
sample_n50 = pop %>% sample_n(50, replace = F)

# Sample mean and sd
mean(sample_n50$value)
```

```
## [1] 4.999257
```

```
sd(sample_n50$value)
```

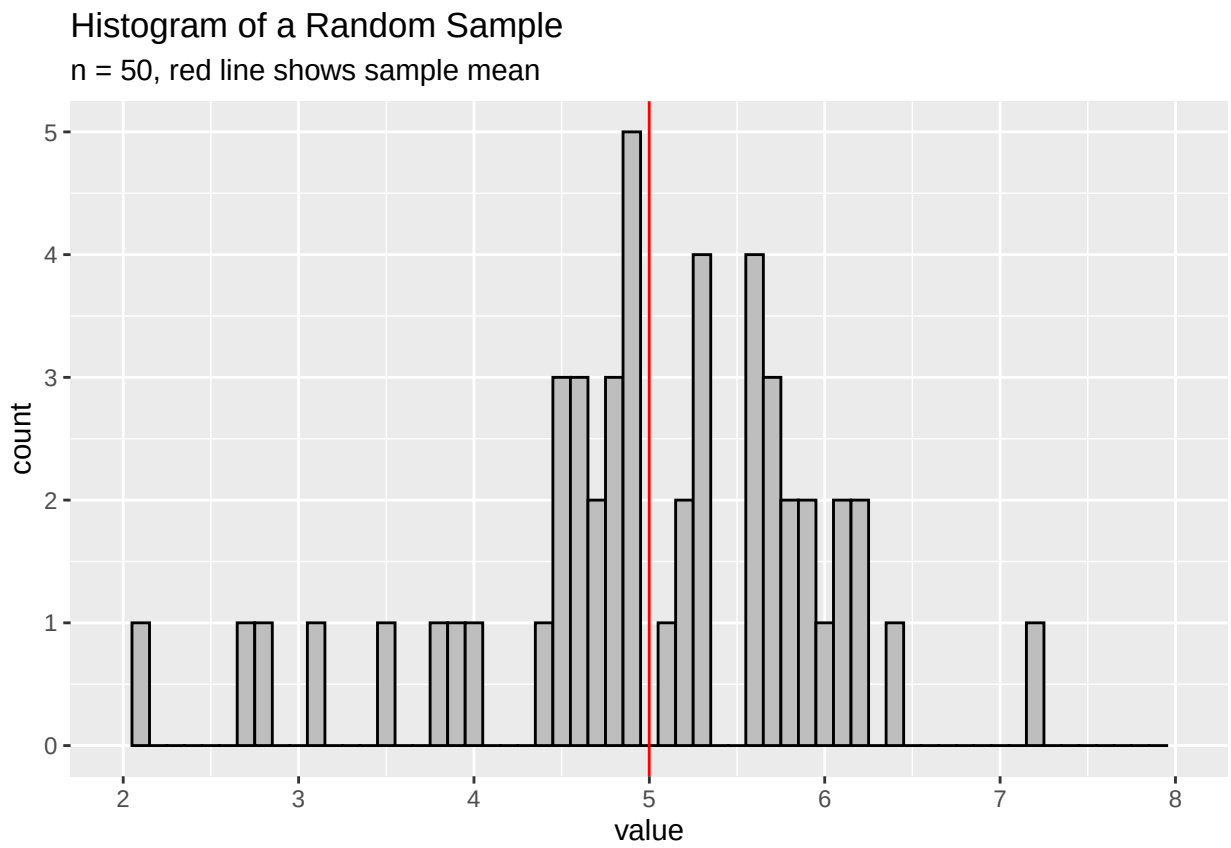
```
## [1] 1.002066
```

3.4 Plot a histogram of the sample with appropriate title and labels. Add a vertical line at your point estimate of the population mean. How does this histogram compare to the one you created in question 2?

```
hist_sample_n50 <- sample_n50 %>%
  ggplot() +
  geom_histogram(aes(value),
                 binwidth = 0.1,
                 fill = "grey",
                 color = "black") +
  geom_vline(xintercept = mean(sample_n50$value), color = "red") +
  labs(title = "Histogram of a Random Sample ",
       subtitle = "n = 50, red line shows sample mean") +
  scale_x_continuous(breaks = seq(2, 8, 1), limits = c(2, 8))    # set x axis tick breaks and limit
```

```
hist_sample_n50
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```



While the population distribution has a normal, bell shape, this random sample does not reflect the shape of the population. This is because we have a relatively small sample size (n = 50) in comparison to the population (N = 100,000). The randomness in sampling leads to this sample distribution.

3.5 Based on your sample, report your point estimate of the population mean $\hat{\mu}$, the standard error of this estimate, and its 95% confidence interval. Show the formulas you used for calculating these statistics.

```
# sample mean
mean(sample_n50$value)
```

```
## [1] 4.999257
```

```
# standard error
se_sample_n50 <- sd(sample_n50$value)/sqrt(50)

# 95% CI using the critical value of Z at 0.025 (z = 1.96)
ci_upper_sample_n50 <- mean(sample_n50$value) + 1.96*se_sample_n50
ci_lower_sample_n50 <- mean(sample_n50$value) - 1.96*se_sample_n50

# 95% CI using the critical value of t at 0.025, df = 49
# find the critical value of t:
t <- qt(p = 0.025, df = 50 - 1) %>% abs() # abs() is for absolute value
# calculate CI
ci_upper_sample_n50_t <- mean(sample_n50$value) + t*se_sample_n50
```

```
ci_lower_sample_n50_t <- mean(sample_n50$value) - t*se_sample_n50
```

- Based on my random sample (n = 50), the point estimate of the population mean $\hat{\mu} = \overline{y} = 4.9992566$
- Based on this random sample, the standard error $se = \frac{s}{\sqrt{n}} = 0.1417136$
- The 95% confidence interval, using the formula $\hat{\mu} \pm 1.96se$, is (4.721498, 5.2770153)
- The 95% confidence interval, using the formula $\hat{\mu} \pm t_{0.025,df=49}se$, where t = 2.0095752 is (4.7144725, 5.2840408)
- The confidence interval estimated using t score is larger than using the z score, because our sample size is relatively small, thus uncertainty is larger than when sample size is large. But in practice, since our sample is usually larger than 100, we use the critical z value 1.96 to calculate confidence interval of the sample mean.

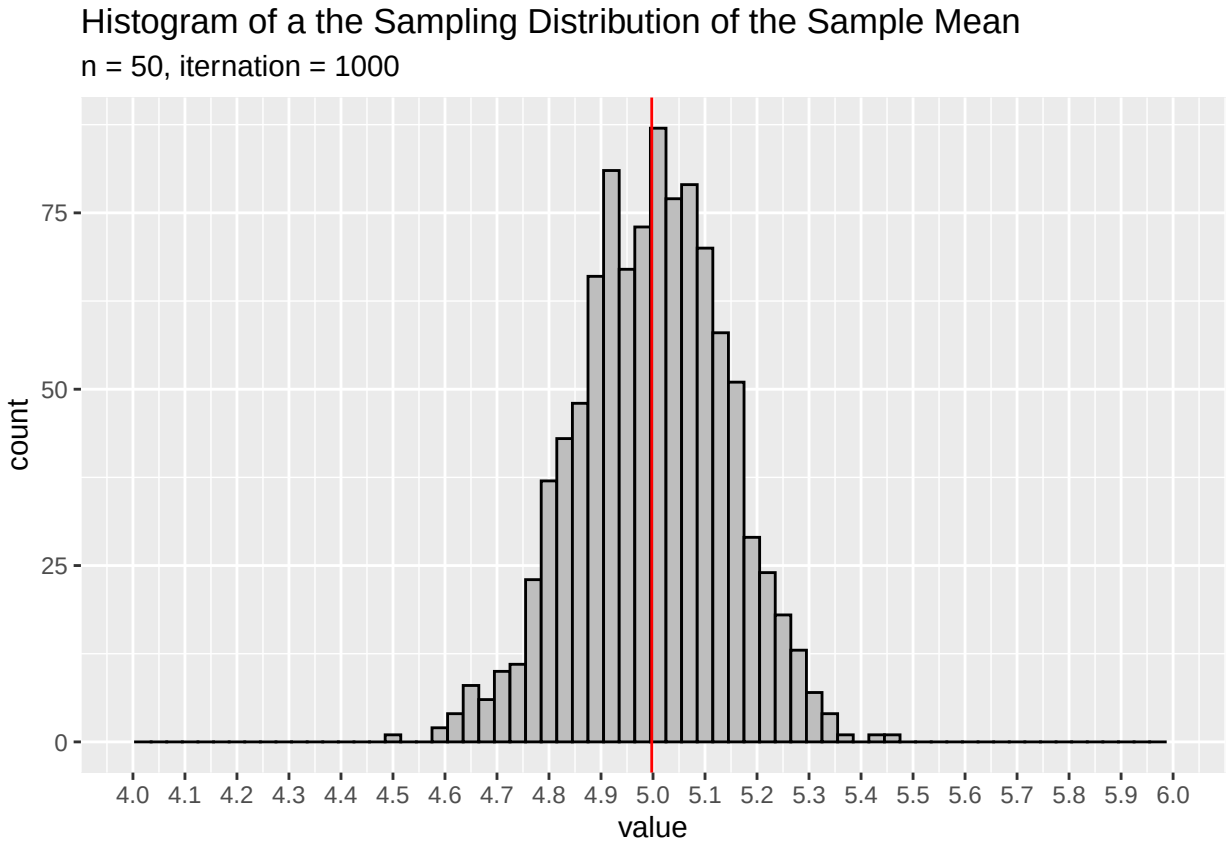
3.6 Simulate the sampling distribution of the sample mean (n = 50) using 1,000 draws. That is, repeat the action you took for question 3 for 1,000 times and save the mean you get for each repetition to a data object.

```
# Repeat sampling for 1000 times, sample without replacement
mean_vector_n50 <- vector(mode = "numeric", length = 1000)

# Set seed for replication
set.seed(1001)
# Save the mean value for each random sampling iteration
for (i in 1:1000){

  # sample 50 from pop, calculate mean
  sample <- pop %>% sample_n(50, replace = F)
  sample_mean = mean(sample$value)

  mean_vector_n50[i] <- sample_mean
}

# Convert mean vector to df
mean_df_n50 <- mean_vector_n50 %>% as_tibble()
```

3.7 Create a histogram of the sampling distribution of the sample mean you simulated in question 6 with appropriate title and labels. Add a vertical line at your point estimate of the population mean.

```
hist_sampling_n50 <- mean_df_n50 %>%
  ggplot() +
  geom_histogram(aes(value),
                 binwidth = 0.03,
                 fill = "grey",
                 color = "black") +
  geom_vline(xintercept = mean(mean_df_n50$value), color = "red") +
  labs(title = "Histogram of a the Sampling Distribution of the Sample Mean ",
       subtitle = "n = 50, iternation = 1000") +
  scale_x_continuous(breaks = seq(4, 6, 0.1), limits = c(4, 6))    # set x axis breaks and limit

hist_sampling_n50
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```



3.8 Using the sampling distribution you obtained in question 6, report your point estimate of the population mean $\hat{\mu}$, the standard error of this estimate, and the 95% confidence interval of this estimate. Show the definitions or formulas you used for calculating these statistics.

```
# mean of the sampling distribution is the mu hat
mean(mean_df_n50$value)
```

```
## [1] 4.997582
```

```
# se of mean is the sd of the sampling distribution of the mean
sd(mean_df_n50$value)
```

```
## [1] 0.1422061
```

```
# 95% CI using z value
ci_upper_sampling_n50 <- mean(mean_df_n50$value) + 1.96*sd(mean_df_n50$value)
ci_lower_sampling_n50 <- mean(mean_df_n50$value) - 1.96*sd(mean_df_n50$value)

# 95% CI using t value (same as the t calculated in Q3.5)
ci_upper_sampling_n50_t <- mean(mean_df_n50$value) + t*sd(mean_df_n50$value)
ci_lower_sampling_n50_t <- mean(mean_df_n50$value) - t*sd(mean_df_n50$value)
```

- Based on the sampling distribution of the sample mean (n = 50, iteration = 1000), the point estimate of the population mean $\hat{\mu} = \overline{y} = 4.9975824$
- Based on the sampling distribution of the sample mean (n = 50, iteration = 1000), the standard error of the sample mean is the standard deviation of the sampling distribution of the sample mean, 0.1422061
- The 95% confidence interval, using the formula $\hat{\mu} \pm 1.96se$, is (4.7188584, 5.2763063)

- The 95% confidence interval, using the formula $\hat{\mu} \pm t_{0.025,df=49} se$, where t is (4.7118085, 5.2833562)

3.9 Repeat questions 3 to 8 increasing the size of your sample to n = 1,000. Plot and report your results. Then, using the concepts that we learned in class, summarize the differences with respect to what you obtained with a sample of 50.

3.9.1 Draw a random sample n = 1,000

```
# Set seed for replication
set.seed(1001)
# Draw a sample n=50 from pop, without replacement
sample_n1000 = pop %>% sample_n(1000, replace = F)

# Sample mean and sd
mean(sample_n1000$value)
```

```
## [1] 5.020228
```

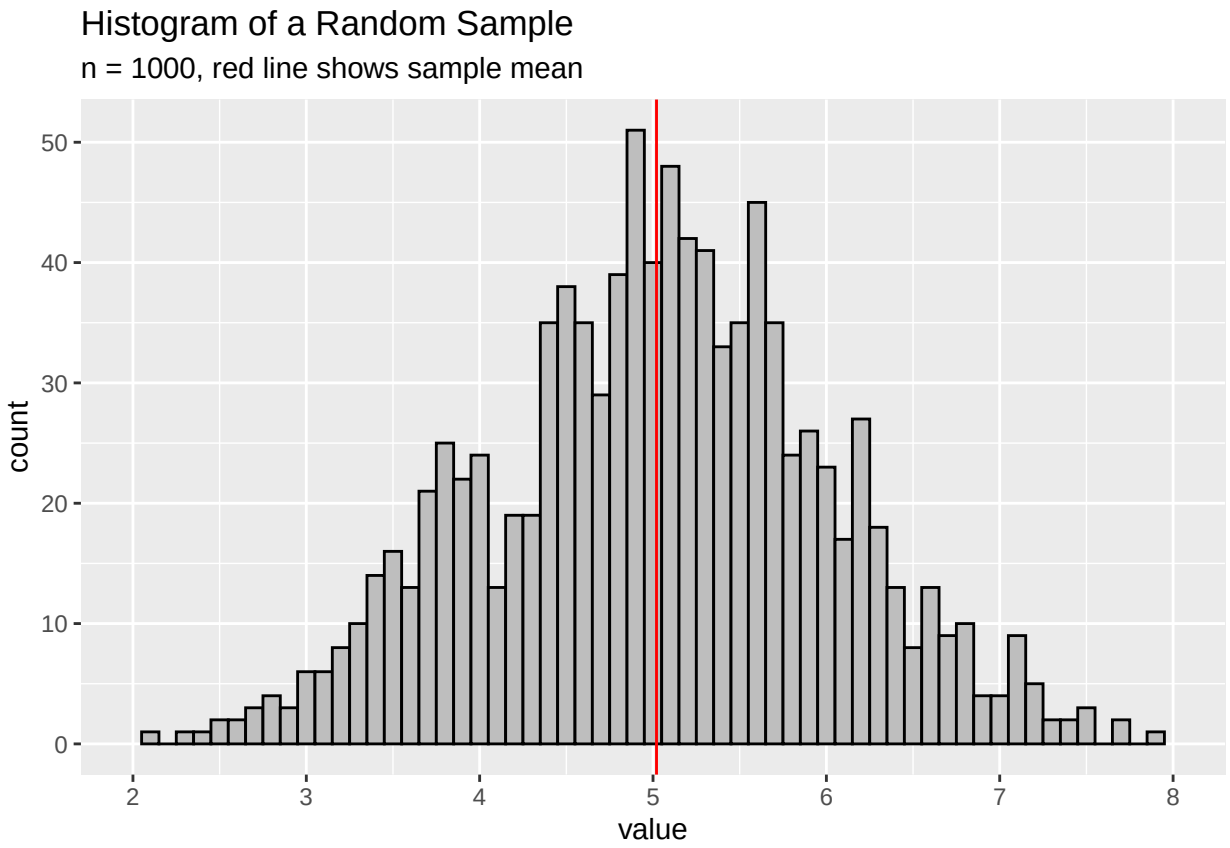
```
sd(sample_n1000$value)
```

```
## [1] 0.9919429
```

3.9.2 Histogram of the random sample n = 1,000

```
hist_sample_n1000 <- sample_n1000 %>%
  ggplot() +
  geom_histogram(aes(value),
                 binwidth = 0.1,
                 fill = "grey",
                 color = "black") +
  geom_vline(xintercept = mean(sample_n1000$value), color = "red") +
  labs(title = "Histogram of a Random Sample ",
       subtitle = "n = 1000, red line shows sample mean") +
  scale_x_continuous(breaks = seq(2, 8, 1), limits = c(2, 8))  # set x axis breaks and limit

hist_sample_n1000
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```



3.9.3 Based on your sample, report your point estimate of the population mean $\hat{\mu}$, the standard error of this estimate, and its 95% confidence interval. Show the formulas you used for calculating these statistics.

```
# mu hat is the sample mean
mean(sample_n1000$value)
```

```
## [1] 5.020228
```

```
# standard error
se_sample_n1000 <- sd(sample_n1000$value)/sqrt(1000)

# 95% CI using the critical value of Z at 0.025 (z = 1.96)
ci_upper_sample_n1000 <- mean(sample_n1000$value) + 1.96*se_sample_n1000
ci_lower_sample_n1000 <- mean(sample_n1000$value) - 1.96*se_sample_n1000
```

- Based on my random sample (n = 1000), the point estimate of the population mean $\hat{\mu} = \overline{y} = 5.0202277$
- Based on this random sample, the standard error $se = \frac{s}{\sqrt{n}} = 0.031368$
- The 95% confidence interval, using the formula $\hat{\mu} \pm 1.96se$, is (4.9587464, 5.0817089)

3.9.4 Simulate the sampling distribution of the sample mean (n = 1000) using 1,000 draws

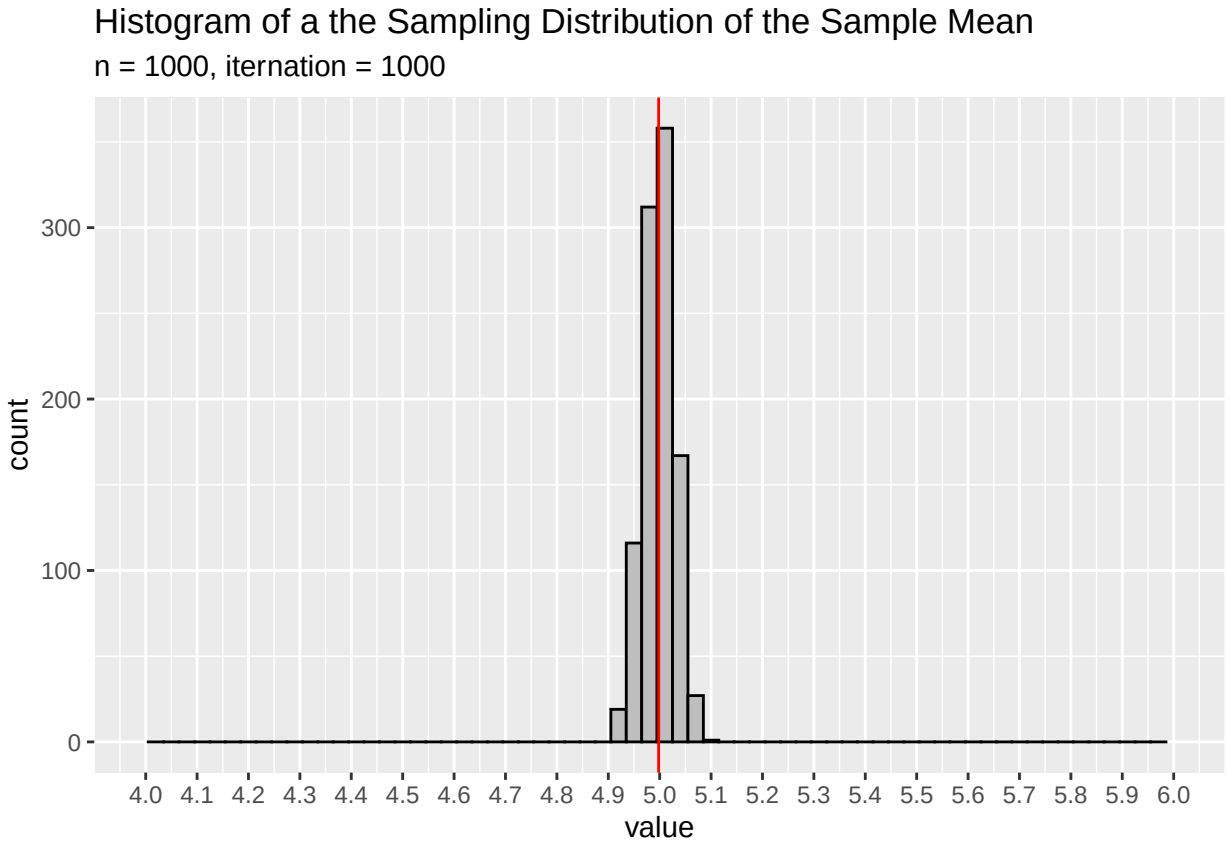
```
# Repeat sampling for 1000 times, sample without replacement
mean_vector_n1000 <- vector(mode = "numeric", length = 1000)

# Set seed for replication
set.seed(1001)
# Save the mean value for each random sampling iteration
```

```r
for (i in 1:1000){

  # sample 1000 rows from pop, calculate mean
  sample <- pop %>% sample_n(1000, replace = F)
  sample_mean = mean(sample$value)

  mean_vector_n1000[i] <- sample_mean
}

# Convert mean vector to df
mean_df_n1000 <- mean_vector_n1000 %>% as_tibble()
```

3.9.5 Create a histogram of the sampling distribution of the sample mean you simulated in question 6 with appropriate title and labels. Add a vertical line at your point estimate of the population mean.

```r
hist_sampling_n1000 <- mean_df_n1000 %>%
  ggplot() +
  geom_histogram(aes(value),
                 binwidth = 0.03,
                 fill = "grey",
                 color = "black") +
  geom_vline(xintercept = mean(mean_df_n1000$value), color = "red") +
  labs(title = "Histogram of a the Sampling Distribution of the Sample Mean ",
       subtitle = "n = 1000, iternation = 1000") +
  scale_x_continuous(breaks = seq(4, 6, 0.1), limits = c(4, 6))  # set x axis tick breaks and limit

hist_sampling_n1000
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```



3.9.6 Using the sampling distribution, report your point estimate of the population mean $\hat{\mu}$, the standard error of this estimate, and the 95% confidence interval of this estimate. Show the definitions or formulas you used for calculating these statistics.

```
# mean of the sampling distribution is the mu hat
mean(mean_df_n1000$value)
```

```
## [1] 4.998081
```

```
# se of mean is the sd of the sampling distribution of the mean
sd(mean_df_n1000$value)
```

```
## [1] 0.02974794
```

```
# 95% CI using z value
ci_upper_sampling_n1000 <- mean(mean_df_n1000$value) + 1.96*sd(mean_df_n1000$value)
ci_lower_sampling_n1000 <- mean(mean_df_n1000$value) - 1.96*sd(mean_df_n1000$value)
```

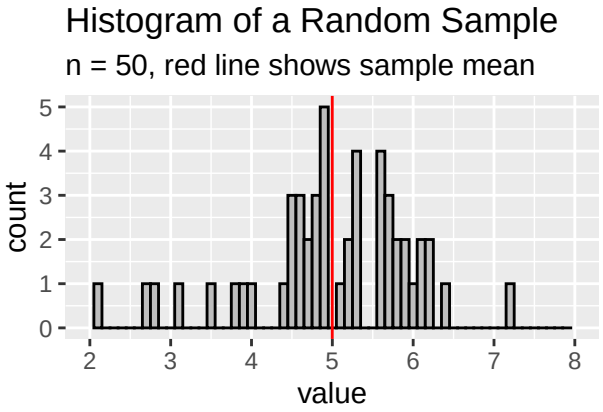
- Based on the sampling distribution of the sample mean (n = 1000, iteration = 1000), the standard error of the sample mean is the standard deviation of the sampling distribution of the sample mean, 0.0297479

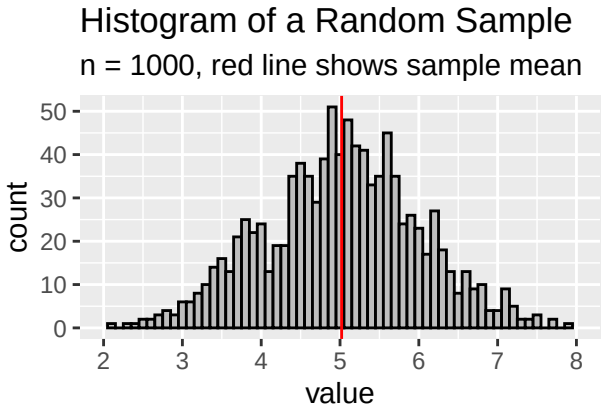
- The 95% confidence interval, using the formula $\hat{\mu} \pm 1.96se$, is (4.9397746, 5.0563865)

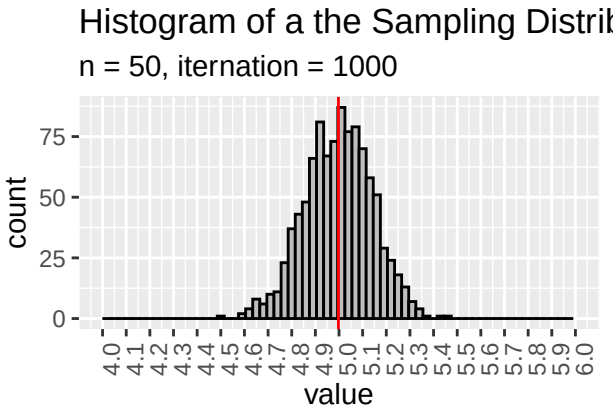
Then, using the concepts that we learned in class, summarize the differences with respect to what you obtained with a sample of 50. Hint: Which law or theorem that we learned in class is being demonstrated here?

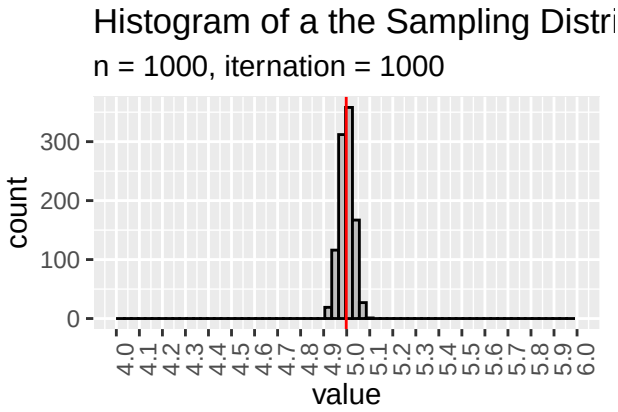
The results of above exercise is summarized in this table and graph:

Type	Mean	Standard Deviation	SE of Mean	CI of Mean
Population (N = 100,000)	4.998	1.000	NA	NA
Sample (n = 50)	4.999	1.002	0.142	(4.714, 5.284)
Sampling Dist. (n = 50, iter = 1000)	4.998	0.142	NA	(4.712, 5.283)
Sample (n = 1000)	5.020	0.992	0.031	(4.959, 5.082)
Sampling Dist.(n = 1000, iter = 1000)	4.998	0.030	NA	(4.940, 5.056)









By the **(Weak) Law of Large Numbers**, we are more likely to get a sample whose mean converges to the population mean as sample size gets larger. (But in my case, the sample mean for n = 50 happens to be a good approximation of the population mean, even better than the n = 1000 sample.)

This exercise also demonstrates the relationship between sample size and the confidence interval of the mean. As shown in the two histograms of the sampling distribution of the mean, as our sample size gets larger, our uncertainty about the point estimate of the mean gets smaller, which means our CI shrinks as sample size n gets larger.