Day 13 - the Central Limit Theorem

Learning outcomes and terms to know

In this section, we introduce two fundamental concepts in statistics: sampling distributions and the **Central Limit Theorem**. By the end of this activity, you should understand:

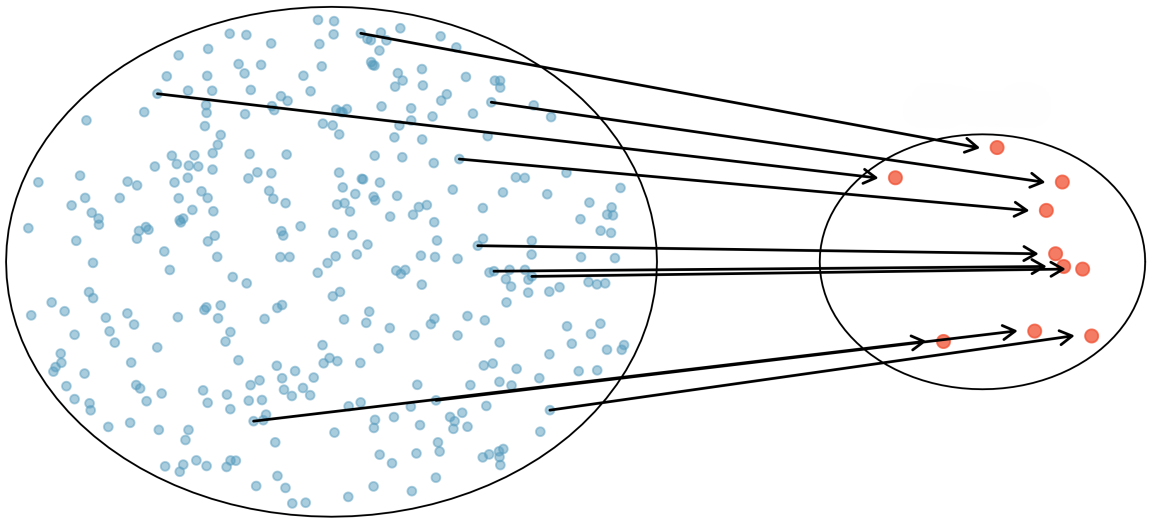
- sample statistics and population parameters
- sampling variability, sampling distributions, and standard errors
- the Central Limit Theorem

```
library(tidyverse)
```

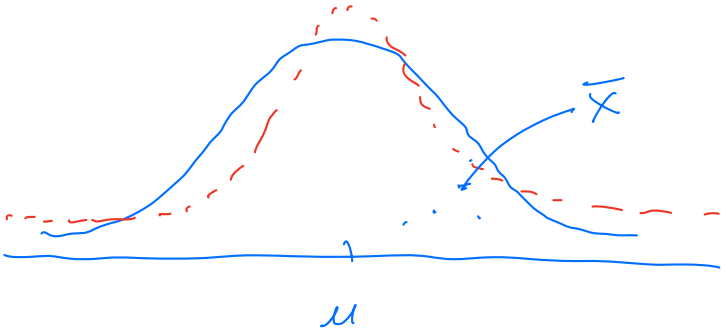


Source: r/mathmemes

Review: population parameters and sample statistics



Do we expect the sample mean to exactly equal the population mean every time we take a sample from the population?



Note

The difference between the observed sample statistic and the population parameter is generally attributed to one of two sources:

- Sampling Variability describes how much a statistic will tend to vary from one sample to the next. Much of statistics is focused on understanding and quantifying this value. **This is the reason we need/care about statistics!**
- Bias describes systematic tendencies to over- or under-estimate the true population value. We try to avoid bias by obtaining samples that are representative of the population.

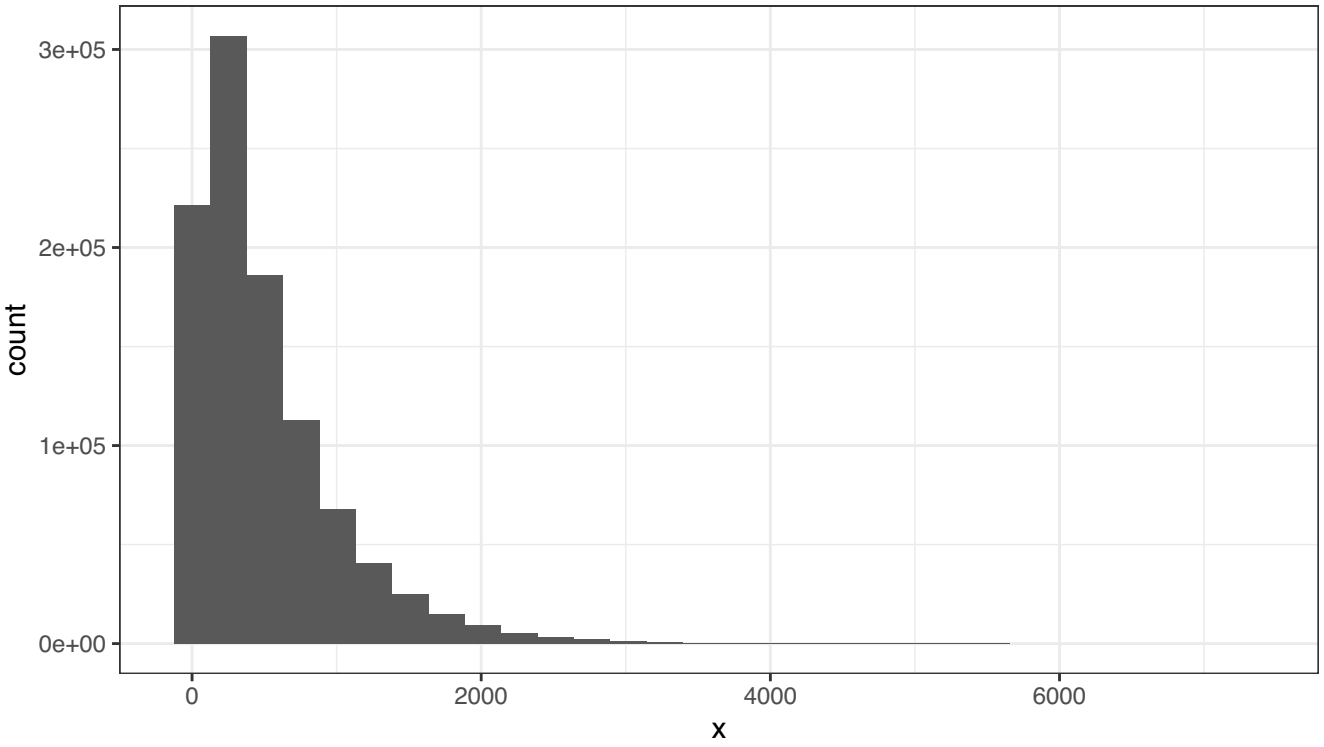
Sampling distributions

Note

The sampling distribution of a statistic is the probability distribution of the statistic. The sampling distributions describes the central tendency and variability of statistics resulting from infinitely many samples of size n from the population.

The code below simulates 10,000 samples of size 25 from a hypothetical population and calculates the sample mean and sample variance for each sample.

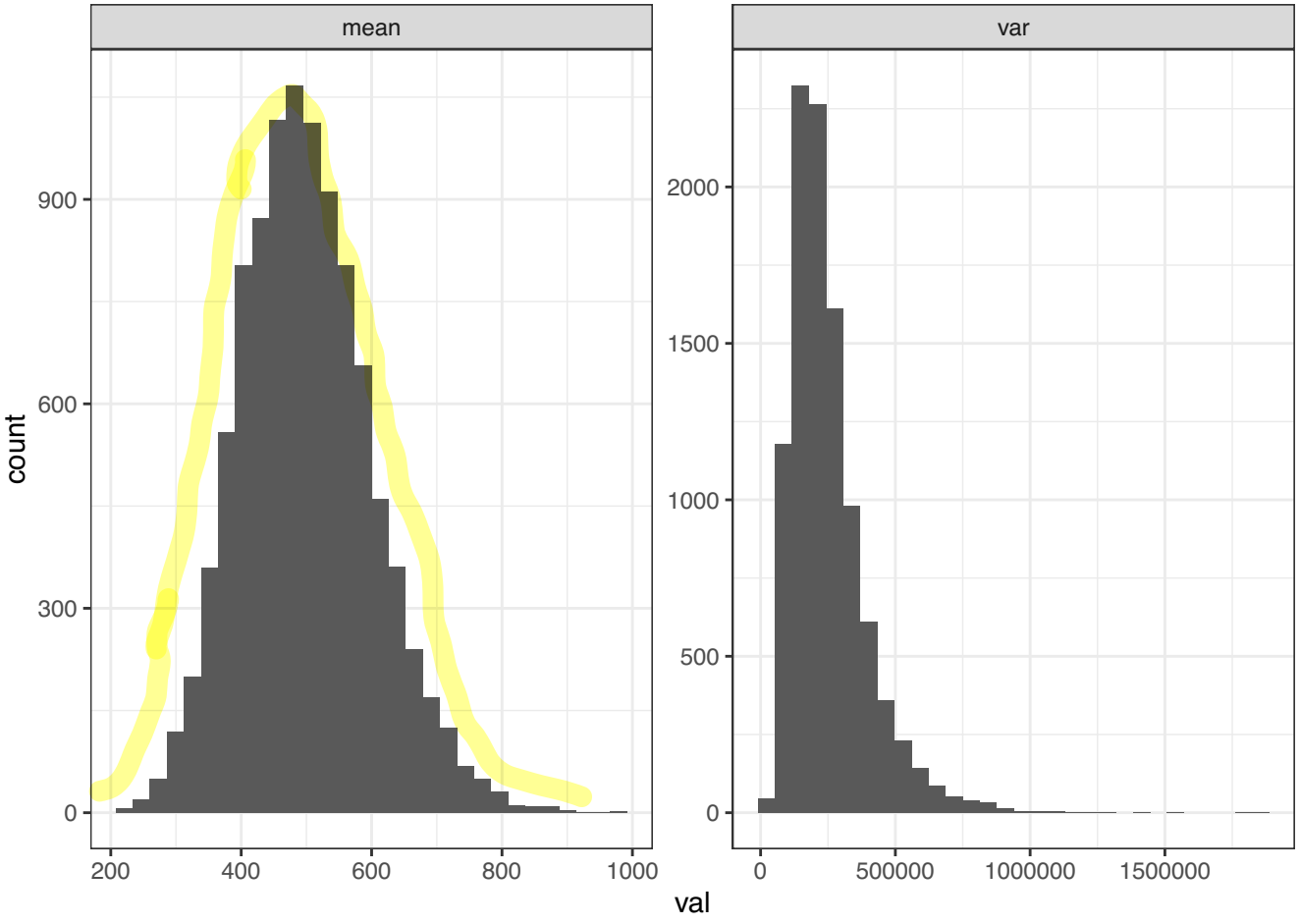
```
# generate 1,000,000 gamma observations to represent the population
population <- rgamma(1e6, 1, 1/500)
tibble(x = population) %>% ggplot() + geom_histogram(aes(x = x)) + theme_bw()
```




```
# example sampling distribution of the means
means <- replicate(
  1e4, expr = {
    sample(population, size = 25, replace = T) %>% mean
  }
)
```

```r
# example sampling distribution of the variance
vars <- replicate(
  1e4, expr = {
    sample(population, size = 25, replace = T) %>% var
  }
)

# plot
bind_rows(
  tibble(val = means) %>% mutate(stat = "mean"),
  tibble(val = vars) %>% mutate(stat = "var")
) %>%
  ggplot() + geom_histogram(aes(x = val)) + theme_bw() +
  facet_wrap(~ stat, nrow = 1, scales = "free")
```



Note

The Central Limit Theorem states that the sampling distribution of the **sample mean** is approximately normally distributed with mean μ and standard deviation $\frac{\sigma}{\sqrt{n}}$.

$$W(X) = \bar{X} \sim N\left(\mu, \frac{\sigma}{\sqrt{n}}\right)$$

Typically, the population standard deviation, σ, is unknown, which means we cannot calculate the standard deviation of the sample mean. In that case, we may estimate the standard deviation of the sample mean by substituting the sample standard deviation in for the population standard deviation, $\frac{s}{\sqrt{n}}$. This quantity is called the standard error of the sample mean,

$$SE(\bar{X}) = \frac{s}{\sqrt{n}}$$

A few notes on the Central Limit Theorem:

- The CLT applies to samples from *any* distribution, even if the population distribution is heavily skewed, multimodal, or some other strangeness!
- For the version of the CLT we consider in this class, the observational units in the sample must be independent of one another.
- Note that by definition, the distribution of the sample mean *approaches* normality as the sample size increases. In practice, the exact sample size required to ensure that the sample mean is approximately normal depends upon the shape of the population distribution.
- The CLT applies to proportions as well!! Why is this?

$$\hat{p} = \frac{\text{ppl w green eyes}}{\text{all people}} = \frac{\sum x_i}{n}, \quad x_i = \{0, 1\}$$

Looking ahead

So why do we care about the Central Limit Theorem? In statistics, we are often interested in asking questions of a population, and using a sample to answer those questions. The Central Limit Theorem provides us with a way to understand the distribution of a sample mean with only a **single** sample. We can then use our knowledge of the distribution of the sample mean to answer the questions asked of the population.

$$s^2 = \frac{1}{n-1}\sum (x_i - \bar{x})^2$$

A note on standard deviations

We have now learned about about **four** different quantities that measure spread and start with the word "standard." All four are defined below - try to remember the difference between them!

Note

- σ**: the population standard deviation (population-level, unknown)**

  This quantity can be thought of as the average deviation from the **population mean** for each **member of the population**.

$$\sigma = \sqrt{\frac{1}{N}\sum(y_i - \mu)^2}$$

- s**: the sample standard deviation (sample-level, known)**

  This quantity can be thought of as the average deviation from the **sample mean** for each **observational unit in our sample.**

$$s = \sqrt{\frac{1}{n-1}\sum(y_i - \bar{x})^2}$$

- $SD(\bar{X})$**: the standard deviation of the sample mean (population-level, unknown)**

  This quantity can be thought of as the average deviation from the **population mean** for **sample means of size** n **from this population**.

$$SD(\bar{X}) = \frac{\sigma}{\sqrt{n}}$$

- $SE(\bar{X})$**: the standard error of the sample mean (sample-level, known)**

  This quantity can be thought of as the **estimated standard deviation of the sample mean**.

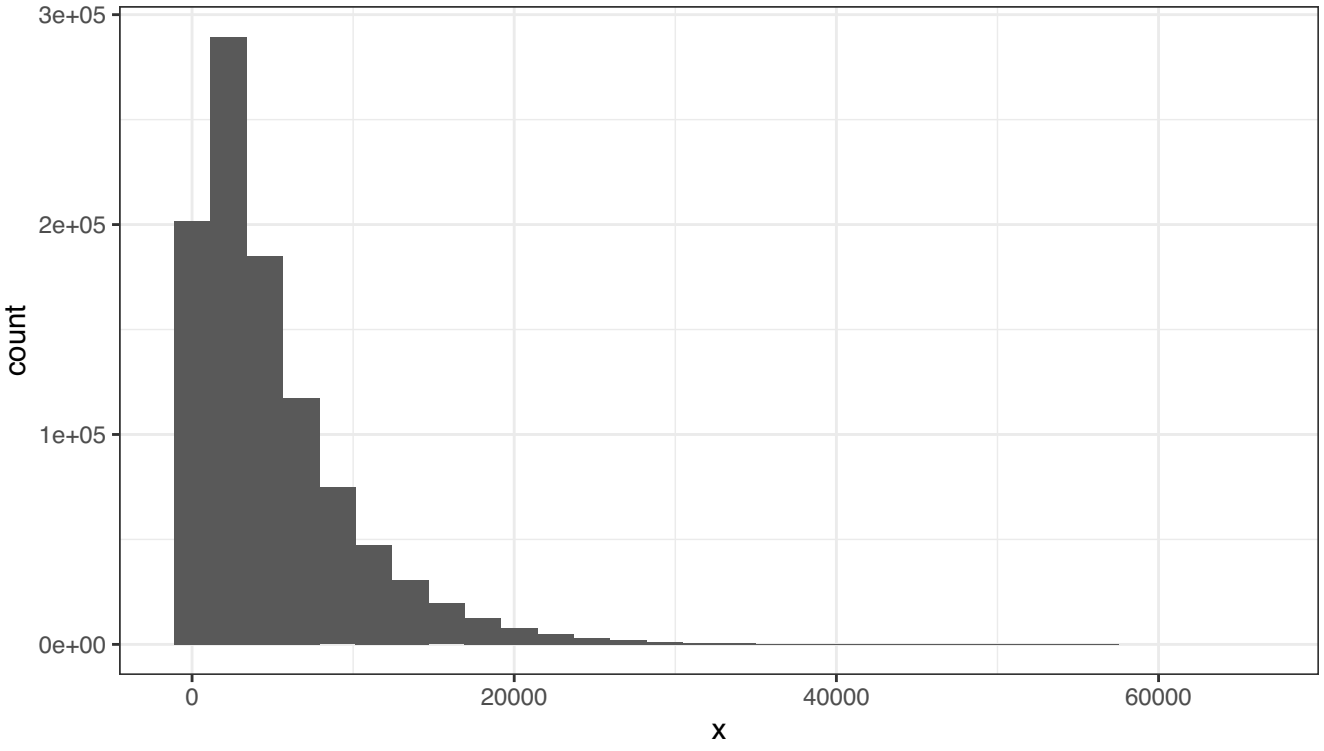
$$SE(\bar{X}) = \frac{s}{\sqrt{n}}$$

Examples

For each of the following population distributions, use computer simulation to visualize the sampling distribution of the sample mean for sample sizes of 5, 25, 100, and 1000. The code printed above may be helpful.

Case 1 - Heavily skewed population

```
# generate a skewed population
population <- rgamma(1e6, 1, 1/5000)
tibble(x = population) %>% ggplot() + geom_histogram(aes(x = x)) + theme_bw()
```



Case 2 - Bimodal

```
# generate a bimodal pop
population <- c(rgamma(1e6, 1, 1/500), rgamma(1e6, 100, 1/20))
tibble(x = population) %>% ggplot() + geom_histogram(aes(x = x)) + theme_bw()
```

