PS01 - Statistics Review

Zachary Tipton

2026-01-21

```r
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.1     v stringr   1.5.2
v ggplot2   4.0.0     v tibble    3.3.0
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.1.0
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

Simulation and Sampling Distributions

In this problem set, we'll explore the central limit theorem and sampling distributions through simulation.

We'll create a general simulation function that can work with different sampling distributions and statistics.

Creating a Simulation Function

Let's write a function that simulates drawing samples and calculating statistics.

The function will take three main parameters: - `N`: sample size - `draw_sample`: a function that draws a sample of size N - `calculate_statistic`: a function that calculates a statistic from the sample

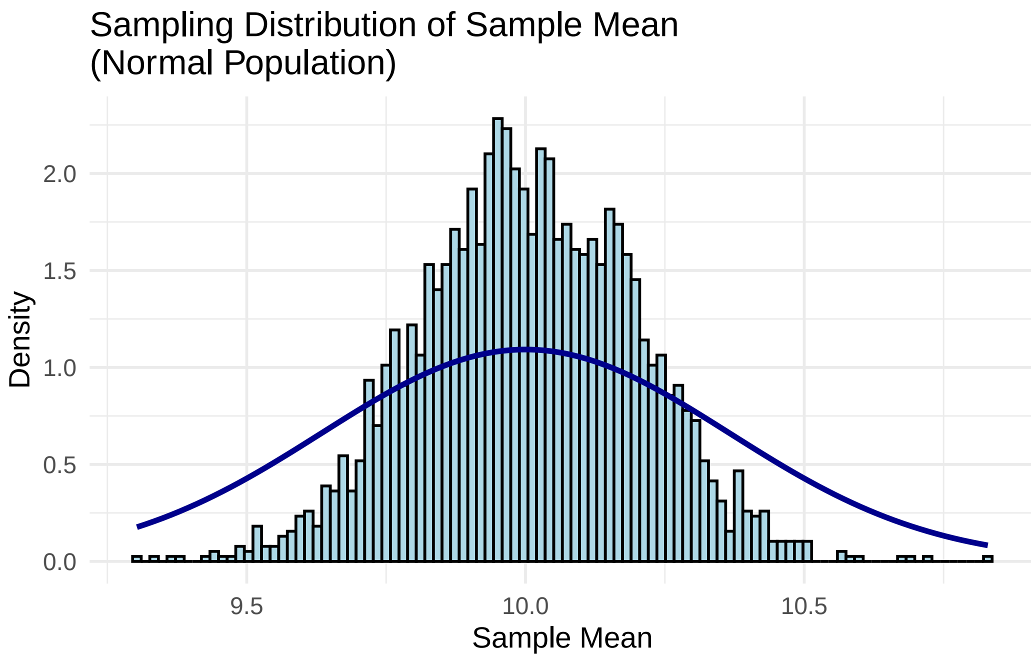
```r
## Helper function that will draw B samples using `draw_sample` and calcuating the statistic
## Returns the set of statistics generated
simulate_sampling_distribution <- function(
  N,
  draw_sample,
  calculate_statistic,
  B = 2500
) {
  # Create a vector to store the statistics
  statistics <- numeric(B)

  # Run B simulations
  for (i in 1:B) {
    # Draw a sample of size N
    x <- draw_sample(N)

    # Calculate the statistic and store it
    statistics[i] <- calculate_statistic(x)
  }

  return(statistics)
}
```

Example 1: Normal Distribution

Let's test our function with samples drawn from a normal distribution. We'll calculate the sample mean.

```r
## Sample from normal function
draw_normal_sample <- function(N) {
  rnorm(N, mean = 10, sd = 2)
}

## Calculate the sample mean
calculate_sample_mean <- function(x) {
  mean(x)
}


## Run simulation and analyze results
set.seed(123)
```

```r
normal_means <- simulate_sampling_distribution(
  N = 100,
  draw_sample = draw_normal_sample,
  calculate_statistic = calculate_sample_mean,
  B = 2500
)

## Plot the sampling distribution
ggplot() +
  ## empirical sampling distribution
  geom_histogram(
    aes(x = normal_means, y = after_stat(density)),
    fill = "lightblue",
    color = "black",
    bins = 100
  ) +
  ## theoretical sampling distribution
  stat_function(
    fun = function(x) dnorm(x, mean = 10, sd = 2 / sqrt(30)),
    color = "darkblue",
    linewidth = 1
  ) +
  labs(
    title = "Sampling Distribution of Sample Mean\n(Normal Population)",
    x = "Sample Mean",
    y = "Density"
  ) +
  theme_minimal()
```

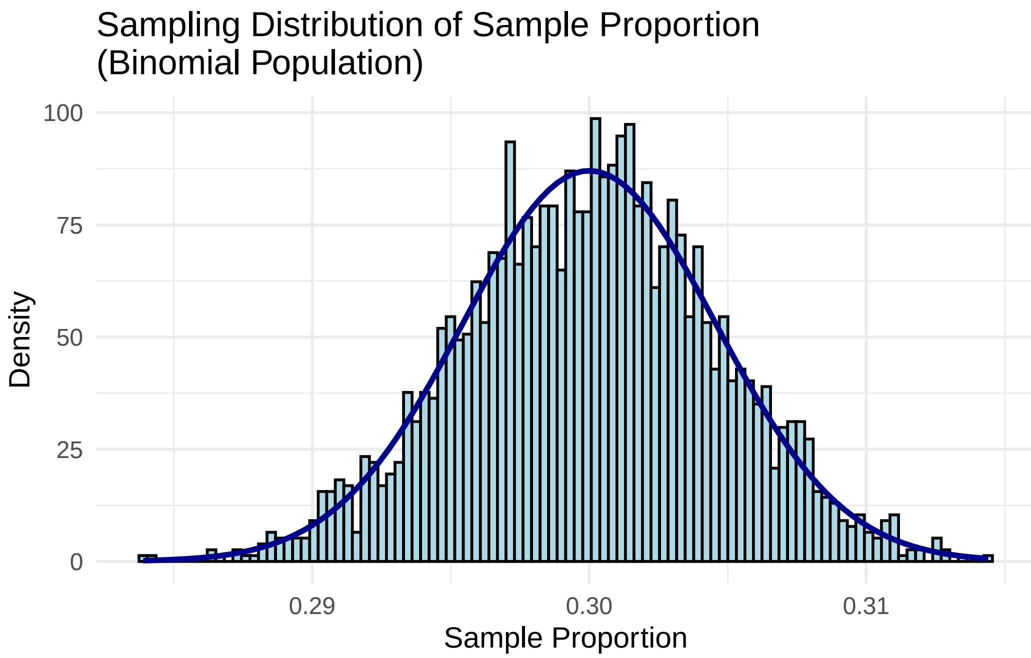


Example 2: Binomial Distribution

Now let's try with a binomial distribution. We'll calculate the sample proportion of successes.

```
N <- 10000
p <- 0.3
draw_binomial_sample <- function(N) {
  rbinom(N, size = 1, prob = p) # Bernoulli trials with probability of success = 0.3
}

# Notes ####
if (FALSE) {
  "Why would a distribution not be normal? small sample size.
  N moves the distribution from binomial to normal n > 30 typically
  B does not change the distribuion, it just draws more or less from the same distribution"}

calculate_sample_proportion <- function(sample_data) {
  mean(sample_data) # For 0/1 data, mean = proportion
}

## Run simulation and analyze results
set.seed(456)
```

```r
binomial_props <- simulate_sampling_distribution(
  N = N,
  draw_sample = draw_binomial_sample,
  calculate_statistic = calculate_sample_proportion,
  B = 2500
)

ggplot() +
  ## empirical sampling distribution
  geom_histogram(
    aes(x = binomial_props, y = after_stat(density)),
    fill = "lightblue",
    color = "black",
    bins = 100
  ) +
  ## theoretical sampling distribution
  stat_function(
    fun = function(x) dnorm(x, mean = p, sd = sqrt(p * (1 - p) / N)),
    color = "darkblue",
    linewidth = 1
  ) +
  labs(
    title = "Sampling Distribution of Sample Proportion\n(Binomial Population)",
    x = "Sample Proportion",
    y = "Density"
  ) +
  theme_minimal()
```



Exercise

1. What happens when `N` is small? How does the normal approximation of the sampling distribution perform?

When `N` is small, the distribution looks more different from a normal approximation than when `N` increases. `N` needs to get fairly high (~10000) for the binomial distribution to more approximate the normal distribution.

2. Does the number of simulations (B) change the sample distribution?

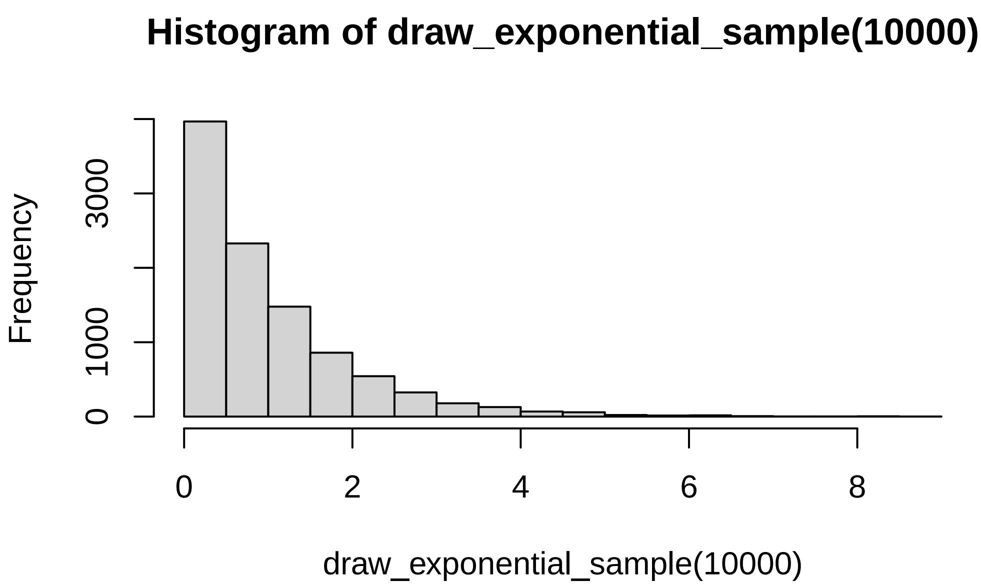
No - changing `B` decreases/increases how many draws are taken out of the distribution but does not change the structure of the distribution.

Example 3: Exponential Distribution

Let's try with an exponential distribution, which is skewed. We'll see how the sample mean behaves.

```
## Define function for exponential distribution
draw_exponential_sample <- function(N) {
  rexp(N, rate = 1) # Mean = 1, variance = 1
}
```

```r
## Example to see the distribution
hist(draw_exponential_sample(10000))
```



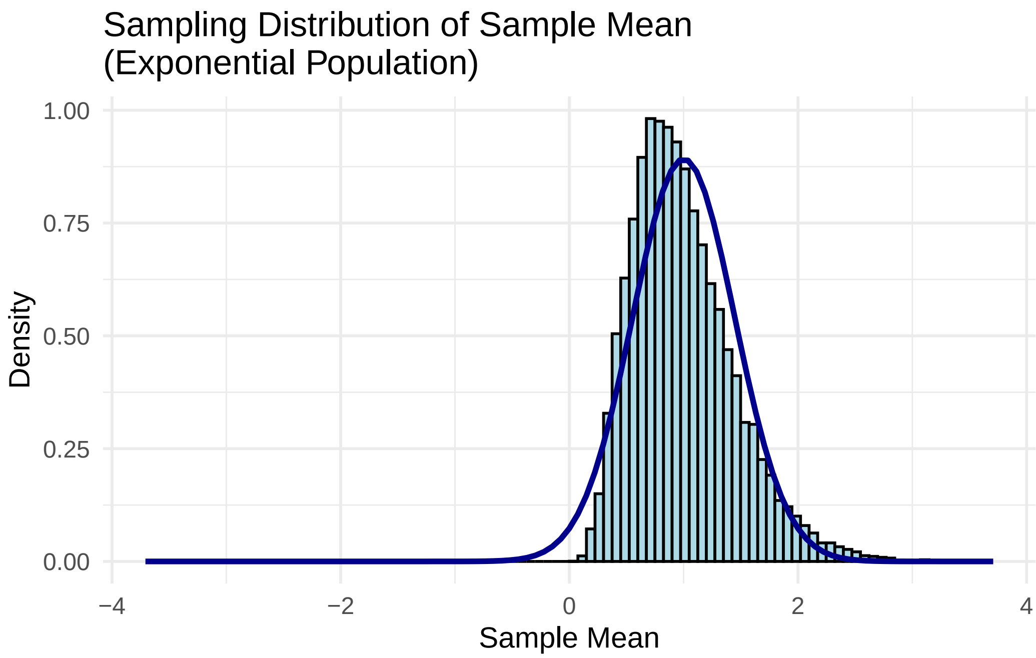

```r
N = 5

## Run simulation and analyze results
set.seed(789)
exp_means <- simulate_sampling_distribution(
  N = N,
  draw_sample = draw_exponential_sample,
  calculate_statistic = calculate_sample_mean,
  B = 25000
)

# Summary of the sampling distribution
summary(exp_means)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.02766 0.67320 0.93207 0.99736 1.25485 3.70770
```

```r
## Plot the sampling distribution
ggplot() +
  ## empirical sampling distribution
  geom_histogram(
```

```r
    aes(x = exp_means, y = after_stat(density)),
    fill = "lightblue",
    color = "black",
    bins = 100
  ) +
  ## theoretical sampling distribution
  stat_function(
    fun = dnorm,
    args = list(mean = 1, sd = 1 / sqrt(N)),
    color = "darkblue",
    linewidth = 1
  ) +
  scale_x_continuous(
    limits = c(-max(abs(exp_means)), max(abs(exp_means)))
  ) +
  labs(
    title = "Sampling Distribution of Sample Mean\n(Exponential Population)",
    x = "Sample Mean",
    y = "Density"
  ) +
  theme_minimal()
```

```
Warning: Removed 2 rows containing missing values or values outside the scale range
(`geom_bar()`).
```



Exercise

1. Try a few different values of `N`, e.g. try 5, 10, 20, 40, and 100. How "quickly" does the normal approximation start to work well?

Around 500 our exponential distribution started to look more approximately normal.

2. When `N` is small, the normal approximation is a poor approximation of the true sample distribution of the statistic. Why is it particularly problematic when conducting inference using the normal approximation (1.96 * standard error)? (Hint: consider the tails / t-distribution)

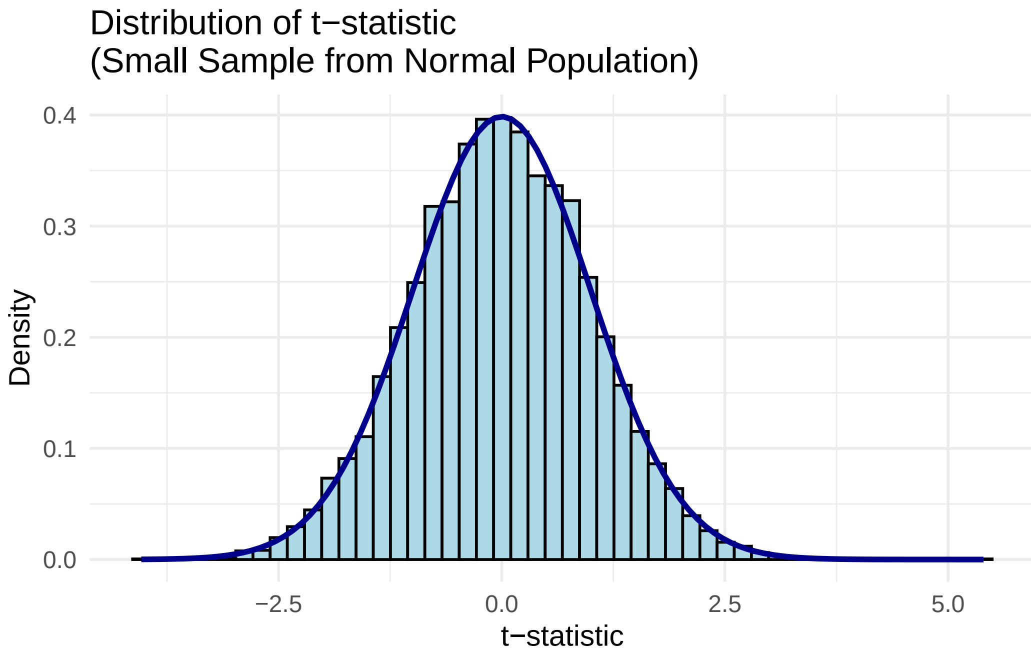
It's fine if he data distribution is skewed, but we need our sample distribution to approximate normal - extreme values more frequently $>$ 3 sd. Normal approximation is saying that extreme values are not happening that often - but when our tails are thicker - then we're getting more extreme values than we would otherwise expect

t-statistic

One of the most common statistic we will calculate is the t test-statistic. This is because we will often want to test whether a sample mean is statistically significantly different from a hypothesized value. Let's look at the sample distribution of our test statistic.

```r
N = 300

## Define t-statistic function
calculate_t_statistic <- function(sample_data) {
  n <- length(sample_data)
  sample_mean <- mean(sample_data)
  sample_sd <- sd(sample_data)
# two estimates

  # t-statistic: (x̄ -  ) / (s/√n)
  # Here we test against true mean = 10
  (sample_mean - 10) / (sample_sd / sqrt(n))
}
# estimated standard deviation of the sample distribution
# (sample_sd / sqrt(n)) == standard error
# dividing a normal distribution by another normal distribution

## Run t-distribution simulation
set.seed(1111) # random number generators work better with large values - doesn't matter but
t_stats <- simulate_sampling_distribution(
  N = N, # Small sample size
  draw_sample = draw_normal_sample,
  calculate_statistic = calculate_t_statistic,
  B = 10000
)

## Plot t-statistics
ggplot() +
  ## empirical sampling distribution
  geom_histogram(
    aes(x = t_stats, y = after_stat(density)),
    fill = "lightblue",
    color = "black",
    bins = 50
  ) +
  ## theoretical sampling distribution
  stat_function(
    fun = function(x) dt(x, df = N - 1),
    color = "darkblue",
    linewidth = 1
  ) +
  labs(
```

```
    title = "Distribution of t-statistic\n(Small Sample from Normal Population)",
    x = "t-statistic",
    y = "Density",
  ) +
  theme_minimal()
```



Perhaps unsurprisingly, our statistic follows the `t` distribution with $N-1$ degrees of freedom.

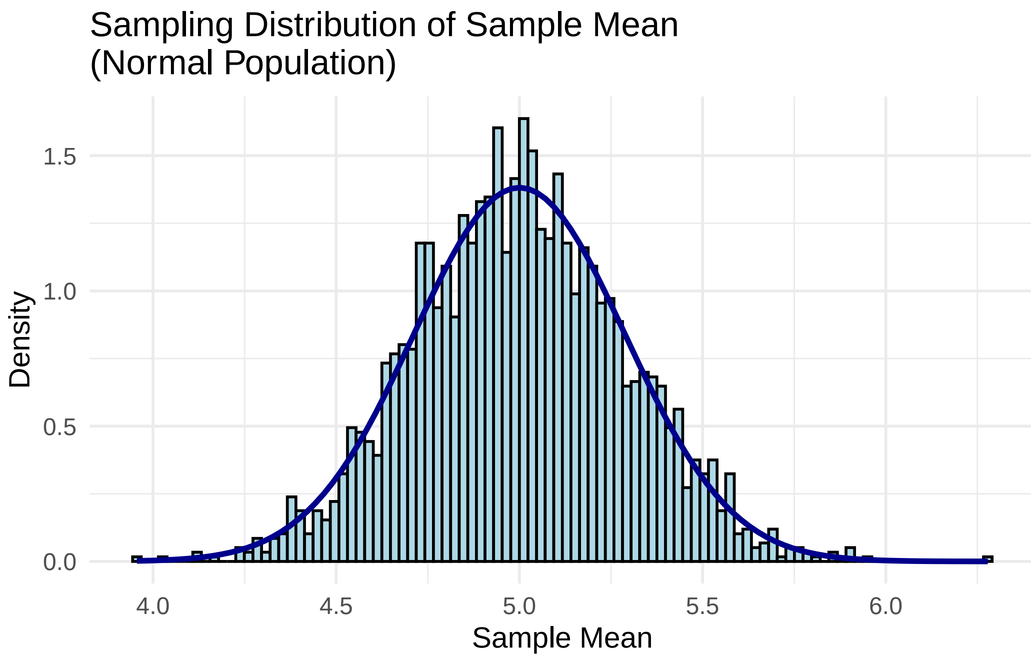
What we learnt There is a distribution we are drawing from Draw observations from the distribution - sampling Estimate some parameter of the data the value you would observe if you could infinitely draw samples pop - mean, sd, quartiles, different estimators pick an estimator to estimate the population parameter - routine you promise you will do when you collect a sample (calculate mean - simple; can be complicated) repeat - repeated sampling - if you got everyone you would have the population values sampling distribution - distribution of the statistic under repeated sampling CLT - if sample size is large, sample distribution approximates population distribution steps in your estimator - mess up the normal distribution standard error * 1.96 = confidence intervals

Exercises

Exercise 1

Modify the simulation to use a uniform distribution U(0, 10). Draw $N = 100$ Calculate the sample mean.

```r
N = 100

## Sample from uniform function
draw_uniform_sample <- function(N) {
  runif(N, min = 0, max = 10)
}

## Calculate the sample mean
calculate_sample_mean <- function(x) {
  mean(x)
}

## Run simulation and analyze results
set.seed(20260120)
uniform_means <- simulate_sampling_distribution(
  N = N,
  draw_sample = draw_uniform_sample,
  calculate_statistic = calculate_sample_mean,
  B = 2500
)

## Plot the sampling distribution
ggplot() +
  ## empirical sampling distribution
  geom_histogram(
    aes(x = uniform_means, y = after_stat(density)),
    fill = "lightblue",
    color = "black",
    bins = 100
  ) +
  ## theoretical sampling distribution
  stat_function(
    fun = function(x) dnorm(x, mean = 5, sd = 2.886751 / sqrt(N)),
    color = "darkblue",
    linewidth = 1
  ) +
  labs(
    title = "Sampling Distribution of Sample Mean\n(Normal Population)",
    x = "Sample Mean",
    y = "Density"
  ) +
  theme_minimal()
```



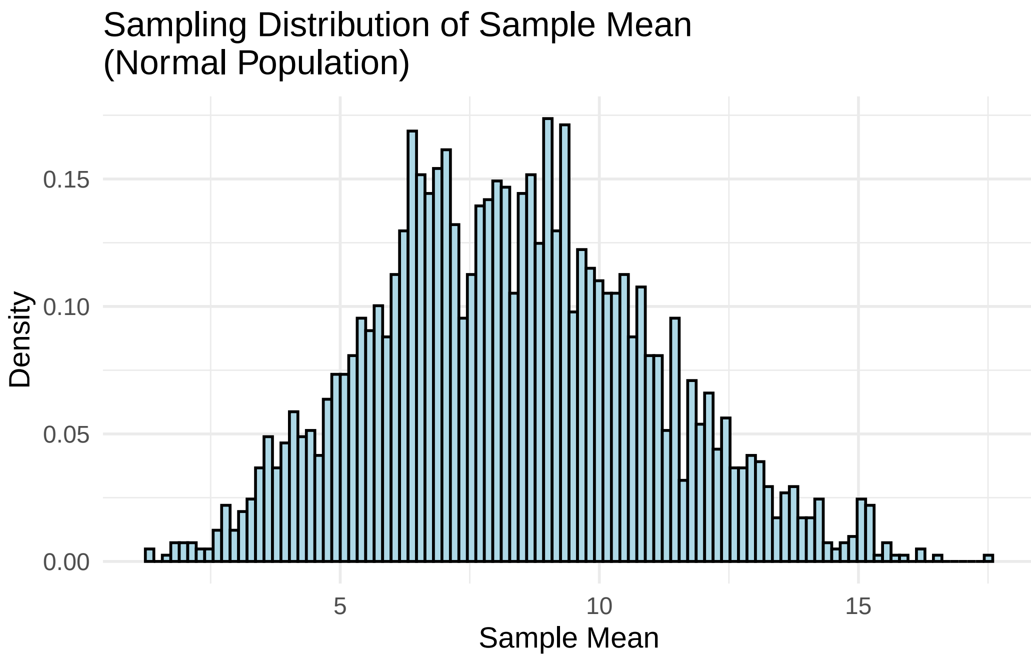

```
# Compare the sampling distribution to the theoretical normal distribution.
```

Exercise 2

Create a function to calculate the sample variance (use the `var` function). Run simulations with different sample sizes ($N = 10$, $N = 30$, $N = 100$).

```
N = 10

## Sample from normal function
draw_uniform_sample <- function(N) {
  runif(N, min = 0, max = 10)
}

## Calculate the sample mean
calculate_sample_mean <- function(x) {
  mean(x)
}

## Calculate variance
calculate_sample_variance <- function(x) {
    var(x)
}
```

```r
## Run simulation and analyze results
set.seed(20260120)
uniform_means <- simulate_sampling_distribution(
  N = N,
  draw_sample = draw_uniform_sample,
  calculate_statistic = calculate_sample_variance,
  B = 2500
)

## Plot the sampling distribution
ggplot() +
  ## empirical sampling distribution
  geom_histogram(
    aes(x = uniform_means, y = after_stat(density)),
    fill = "lightblue",
    color = "black",
    bins = 100
  ) +
  labs(
    title = "Sampling Distribution of Sample Mean\n(Normal Population)",
    x = "Sample Mean",
    y = "Density"
  ) +
  theme_minimal()
```




```
# Compare the sampling distribution to the theoretical normal distribution.
```

How does the sampling distribution of the variance change with sample size?

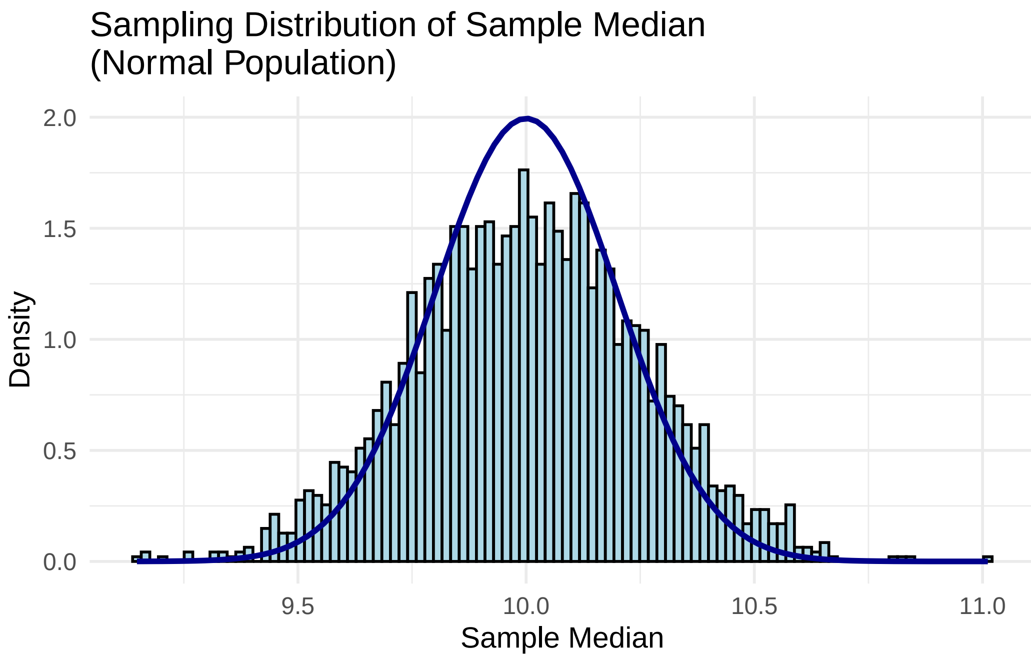
Smooths out to be more normally distributed looking as N increases

Exercise 3

Simulate the sampling distribution of the median for samples from a normal distribution. Compare it to the sampling distribution of the mean.

```
# Exercise 3

N = 100

## Sample from normal function
draw_normal_sample <- function(N) {
  rnorm(N, mean = 10, sd = 2)
}

## Calculate the sample mean
calculate_sample_median <- function(x) {
  median(x)
```

```r
}


## Run simulation and analyze results
set.seed(123)
normal_median <- simulate_sampling_distribution(
  N = N,
  draw_sample = draw_normal_sample,
  calculate_statistic = calculate_sample_median,
  B = 2500
)

## Plot the sampling distribution
ggplot() +
  ## empirical sampling distribution
  geom_histogram(
    aes(x = normal_median, y = after_stat(density)),
    fill = "lightblue",
    color = "black",
    bins = 100
  ) +
  ## theoretical sampling distribution
  stat_function(
    fun = function(x) dnorm(x, mean = 10, sd = 2 / sqrt(N)),
    color = "darkblue",
    linewidth = 1
  ) +
  labs(
    title = "Sampling Distribution of Sample Median\n(Normal Population)",
    x = "Sample Median",
    y = "Density"
  ) +
  theme_minimal()
```



Which sample distribution has the larger variance (use the `var` function)?

```
# Calculate the variance for both distributions
var_median <- var(normal_median)
var_mean   <- var(normal_means)

# Print the results
cat("Variance of the Sample Median:", var_median, "\n")
```

```
Variance of the Sample Median: 0.06188399
```

```
cat("Variance of the Sample Mean:  ", var_mean, "\n")
```

```
Variance of the Sample Mean:   0.03781717
```

```
# Calculate the Relative Efficiency
# (If > 1, the mean is more efficient)
efficiency <- var_median / var_mean
print(paste("The median is", round(efficiency, 3), "times more variable than the mean."))
```

```
[1] "The median is 1.636 times more variable than the mean."
```

Key Takeaways

For many statistics, the central limit theorem (CLT) will ensure that the sampling distribution is *approximately* normally distributed *in large samples.* However, this is not guaranteed and caution should be had. For most distributions, sample sizes of 30 or more are sufficient for the CLT to provide a good approximation, though this varies with the degree of skewness in the population.