IS5 in R: Confidence Intervals for Means (Chapter 14)

Nicholas Horton (nhorton@amherst.edu)

2025-01-15

Table of contents

Introduction and background

This document is intended to help describe how to undertake analyses introduced as examples in the Fifth Edition of *Intro Stats* (2018) by De Veaux, Velleman, and Bock. This file as well as the associated Quarto reproducible analysis source file used to create it can be found at http://nhorton.people.amherst.edu/is5.

This work leverages initiatives undertaken by Project MOSAIC (http://www.mosaic-web.org), an NSF-funded effort to improve the teaching of statistics, calculus, science and computing in the undergraduate curriculum. In particular, we utilize the `mosaic` package, which was written to simplify the use of R for introductory statistics courses. A short summary of the R needed to teach introductory statistics can be found in the mosaic package vignettes (https://cran.r-project.org/web/packages/mosaic). A paper describing the mosaic approach was published in the *R Journal*: https://journal.r-project.org/archive/2017/RJ-2017-024.

We begin by loading packages that will be required for our analyses.

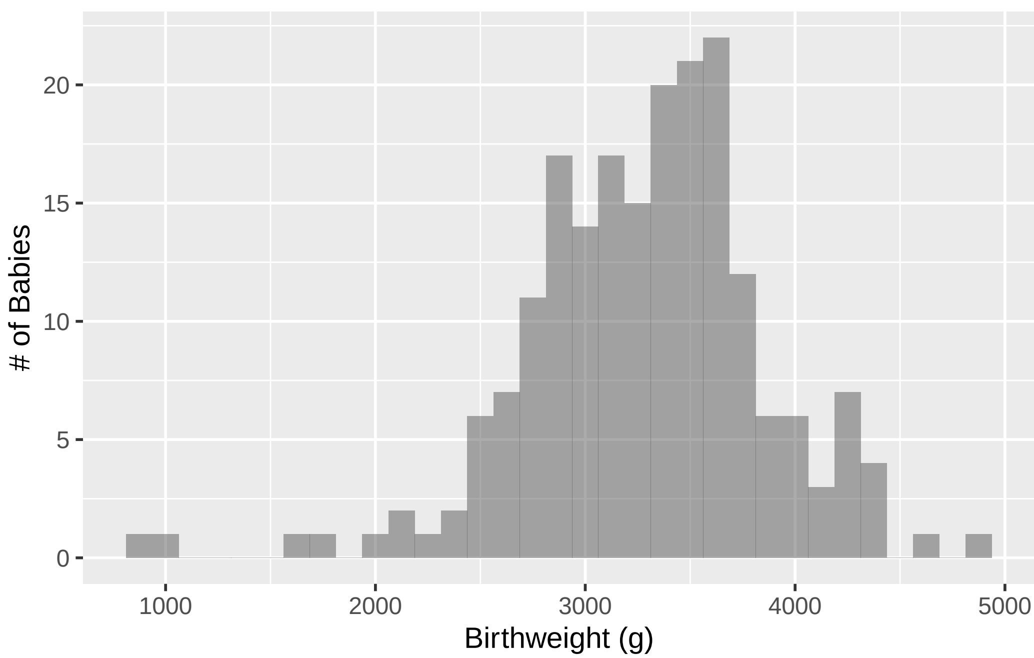
```r
library(mosaic)
library(tidyverse)
```

Chapter 14: Confidence Intervals for Means

```r
Babies <- read_csv("http://nhorton.people.amherst.edu/is5/data/Babysamp_98.csv") |>
  janitor::clean_names()
```

By default, `read_csv()` prints the variable names. These messages have been suppressed using the `message: false` code chunk option to save space and improve readability. Here we use the `clean_names()` function from the `janitor` package to sanitize the names of the columns (which would otherwise contain special characters or whitespace).

```r
# Figure 14.1, page 441
gf_histogram(~ weight, data = Babies, binwidth = 125) |>
  gf_labs(x = "Birthweight (g)", y = "# of Babies")
```




```r
set.seed(12346) # To ensure we get the same values when we run it multiple times
num_sim <- 10000 # Number of simulations
```

```r
# What does do() do?
mean(~ weight, data = sample(Babies, size = 100)) # Mean of a random sample of 100
```

```
[1] 3290.72
```

```r
mean(~ weight, data = sample(Babies, size = 100)) # Mean of another random sample
```

```
[1] 3247.51
```

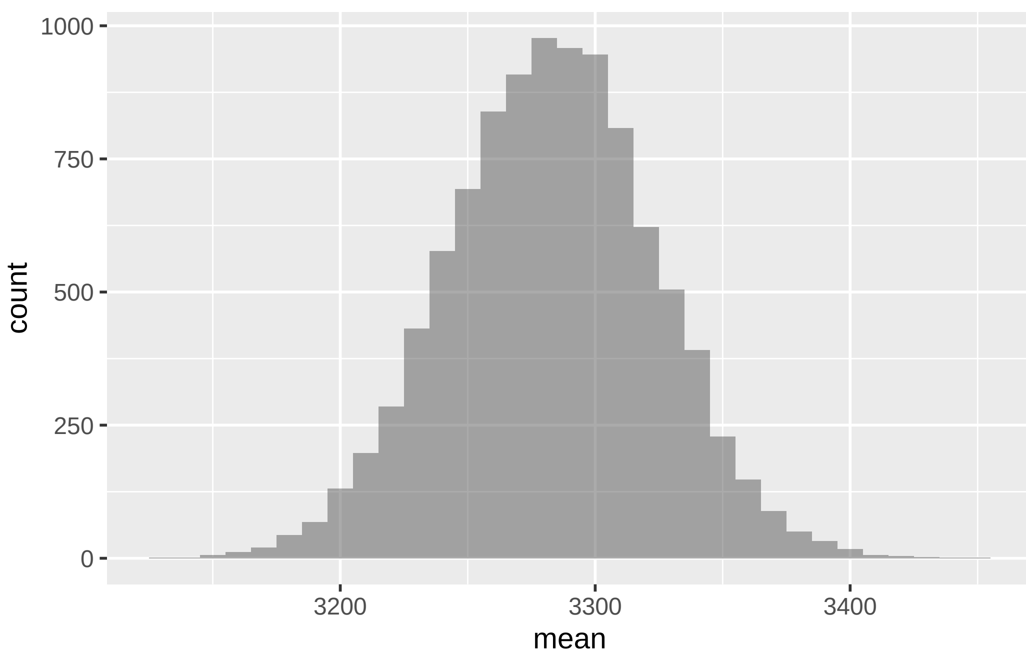
```r
do(2) * mean(~ weight, data = sample(Babies, size = 100)) # Calculates the mean twice
```

```
     mean
1 3296.01
2 3251.20
```

```r
# For the visualization, we need 10,000 means
WeightMeans <- do(num_sim) * mean(~ weight, data = sample(Babies, size = 100))
```

The `do()` function repeatedly calculates the mean of a random sample of 100 baby weights.

```r
# Figure 14.2
gf_histogram(~ mean, data = WeightMeans, binwidth = 10)
```



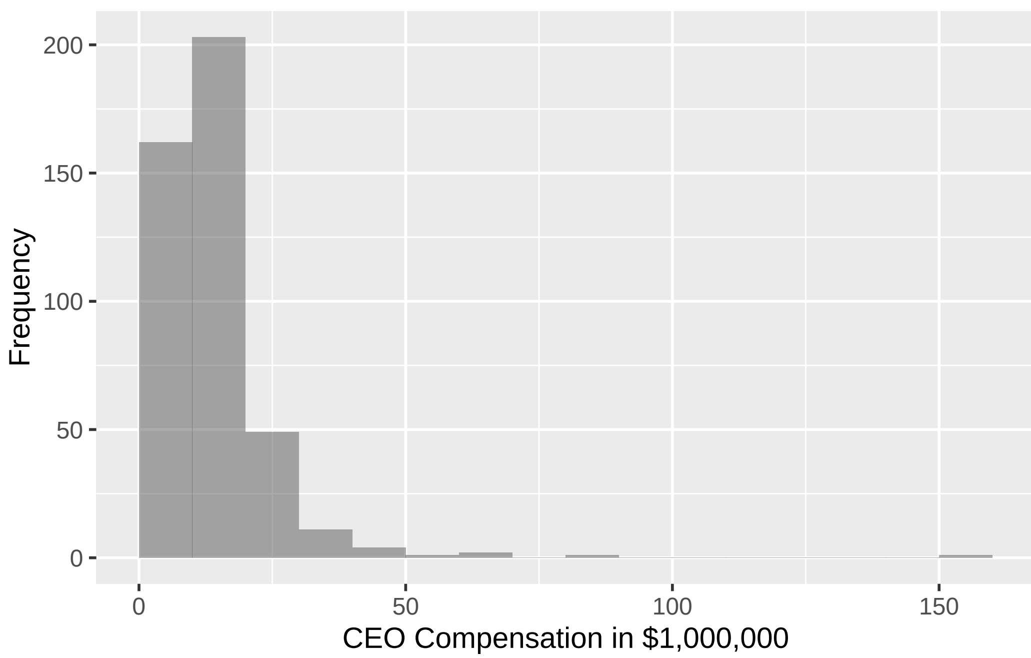
```
df_stats(~ mean, data = WeightMeans)
```

```
  response    min       Q1   median       Q3     max     mean       sd     n
1     mean 3133.8 3255.355 3283.065 3309.812 3450.96 3282.656 40.47944 10000
  missing
1       0
```

Section 14.1: The Central Limit Theorem

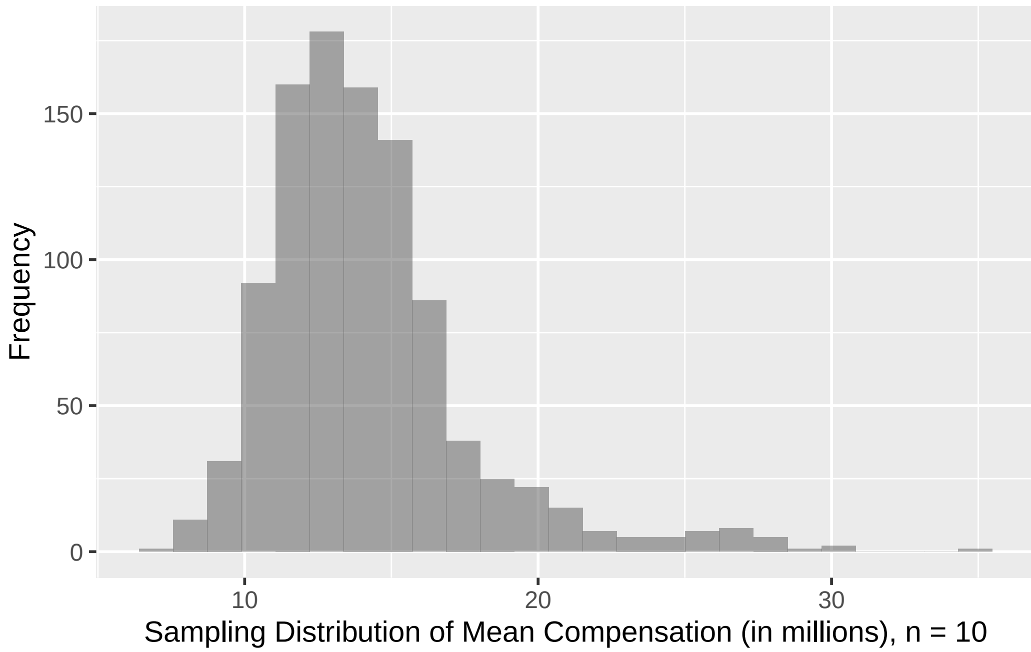
```
set.seed(1821)
CEOComp <-
  read_csv("http://nhorton.people.amherst.edu/is5/data/CEO_Compensation_2014.csv") |>
  janitor::clean_names()

# Figure 14.3, page 443
gf_histogram(~ ceo_compensation_m, data = CEOComp, binwidth = 10, center = 5) |>
  gf_labs(x = "CEO Compensation in $1,000,000", y = "Frequency")
```

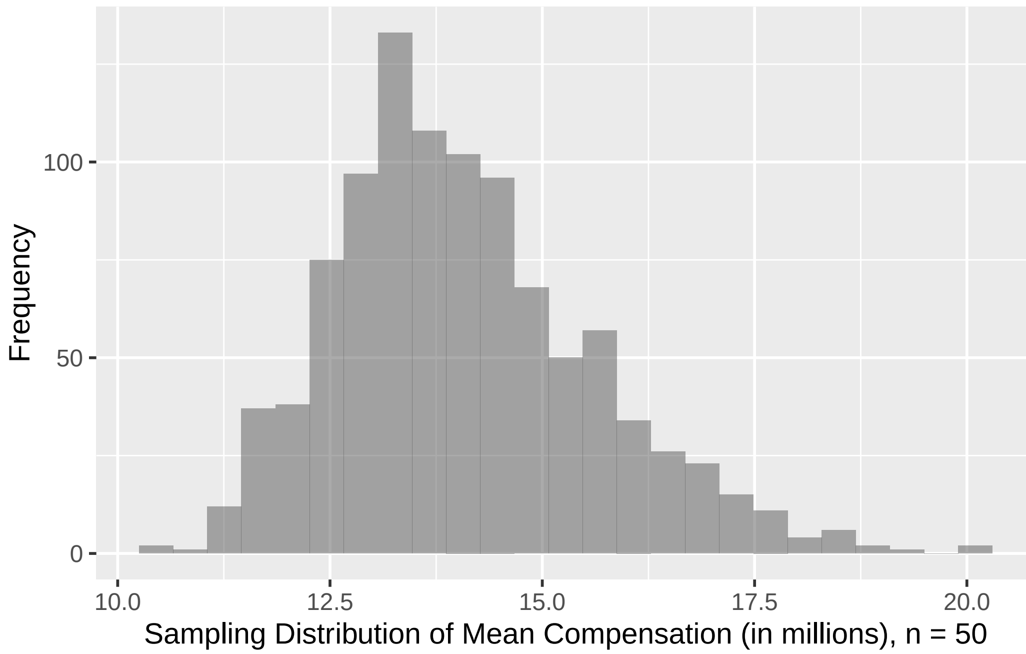



```
# Figure 14.4
num_sim <- 1000 # Here the number of simulations is 1,000
```

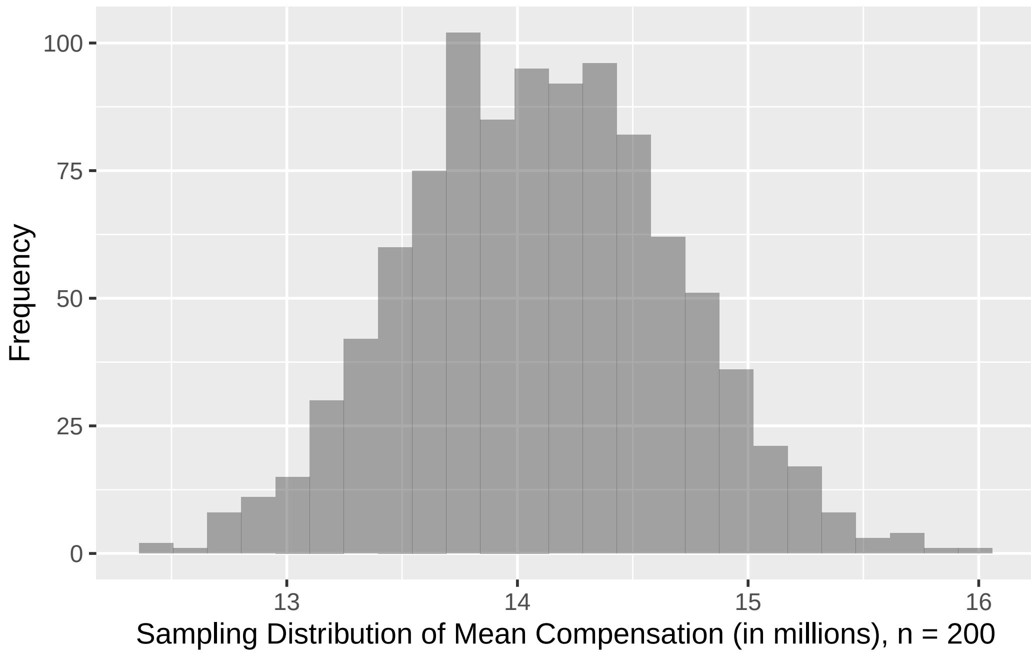
```r
CEOMeansn10 <- do(num_sim) * mean(~ ceo_compensation_m, data = sample(CEOComp, size = 10))
gf_histogram(~ mean, data = CEOMeansn10) |>
  gf_labs(
    x = "Sampling Distribution of Mean Compensation (in millions), n = 10",
    y = "Frequency"
  )
```




```r
# Figure 14.5
CEOMeansn50 <- do(num_sim) * mean(~ ceo_compensation_m, data = sample(CEOComp, size = 50))
gf_histogram(~ mean, data = CEOMeansn50) |>
  gf_labs(
    x = "Sampling Distribution of Mean Compensation (in millions), n = 50",
    y = "Frequency"
  )
```

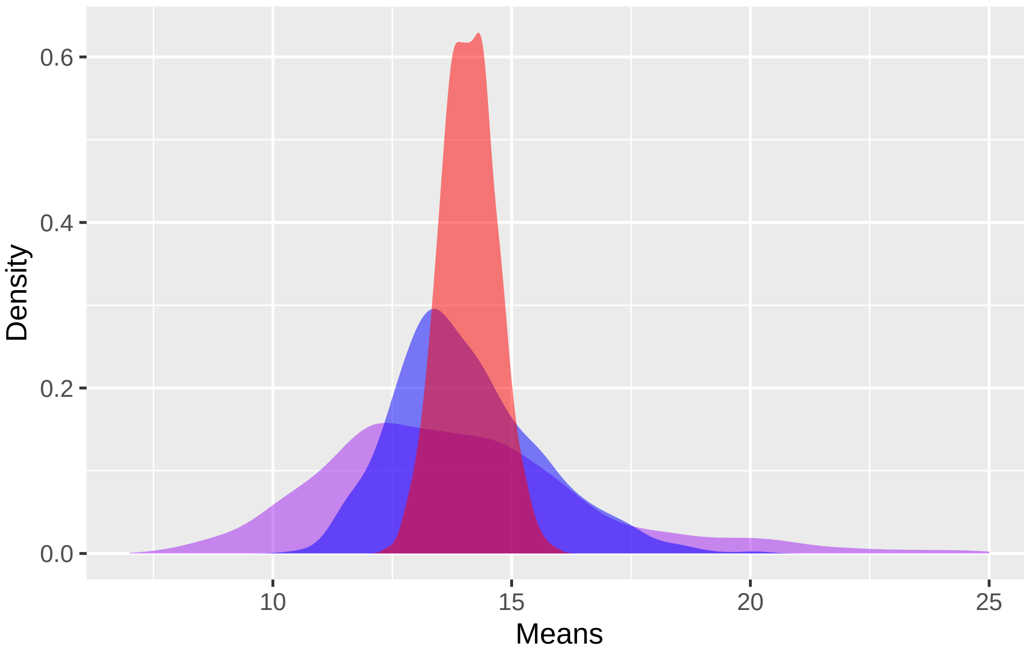



```
# Figure 14.7 (skipped 14.6 because it's similar)
CEOMeansn200 <- do(num_sim) * mean(~ ceo_compensation_m, data = sample(CEOComp, size = 200))
gf_histogram(~ mean, data = CEOMeansn200) |>
  gf_labs(
    x = "Sampling Distribution of Mean Compensation (in millions), n = 200",
    y = "Frequency"
  )
```



For each example sample size, the `do()` function calculates the mean of random samples of that specified size.

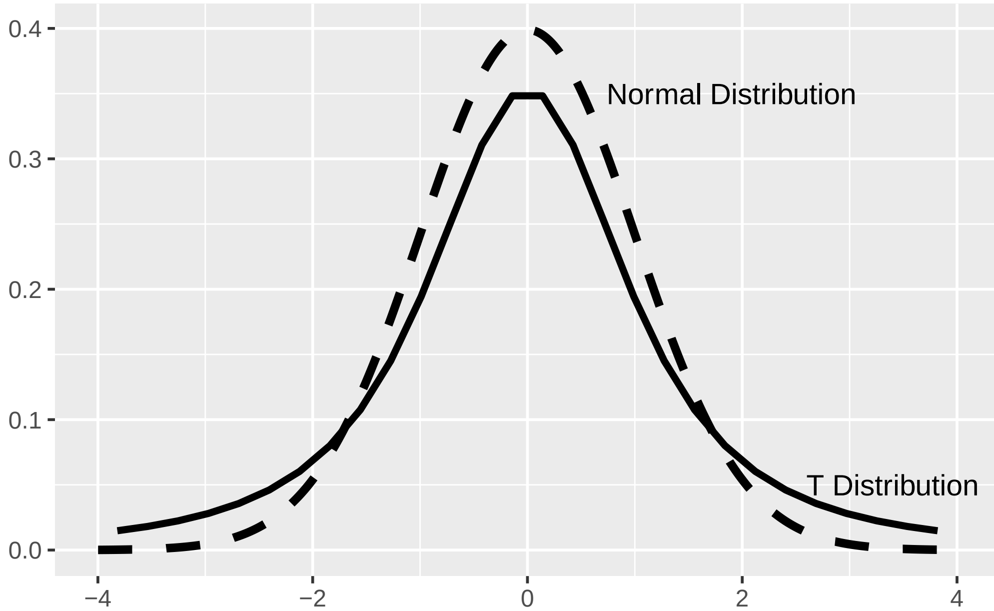
```
# Samples as overlaid density plots
gf_density(~ mean, data = CEOMeansn10, fill = "purple") |>
  gf_density(~ mean, data = CEOMeansn50, fill = "blue") |>
  gf_density(~ mean, data = CEOMeansn200, fill = "red") |>
  gf_labs(y = "Density", x = "Means") +
  xlim(7, 25)
```



The purple density reflects the distribution of the means from random samples of size 10. The blue density reflects the distribution of the means from random samples of size 50. The red density reflects the distribution of the means from random samples of size 200.

Section 14.2: A Confidence Interval for the Mean

```
# Figure 14.9, page 446
gf_dist(dist = "norm", linetype = 2, lwd = 1.5) |>
  gf_refine(annotate(geom = "text", x = 1.9, y = .35, label = "Normal Distribution")) |>
  gf_labs(x = "", y = "") |>
  gf_dist(dist = "t", df = 2, lwd = 1.25) |>
  gf_refine(annotate(geom = "text", x = 3.4, y = .05, label = "T Distribution")) +
  xlim(-4, 4)
```



Example 14.1: A One-Sample *t*-Interval for the Mean

```
# page 448
Salmon <- read_csv("http://nhorton.people.amherst.edu/is5/data/Farmed_salmon.csv") |>
  janitor::clean_names()
Salmon <- Salmon |>
  filter(mirex != "NA")
df_stats(~ mirex, data = Salmon)
```

```
  response min    Q1 median      Q3   max    mean         sd   n missing
1    mirex   0 0.056  0.079 0.13475 0.194 0.09134 0.04952388 150       0
```

```
t.test(~ mirex, data = Salmon)
```

```

    One Sample t-test

data:  mirex
t = 22.589, df = 149, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 0.08334978 0.09933022
sample estimates:
```

```
mean of x
  0.09134
```

```
salmonlm <- lm(mirex ~ 1, data = Salmon) # equiv to t.test(~ mirex)
# replication of interval from page 448
confint(salmonlm, data = Salmon)
```

```
                 2.5 %     97.5 %
(Intercept) 0.08334978 0.09933022
```

```
tstats <- xqt(df = 149, p = c(.025, .975), plot = FALSE)
# finds the t statistics when given quantiles and df (default will plot a graph)
sey <- sd(~ mirex, data = Salmon) / (150^(1 / 2)) # standard error
mean(~ mirex, data = Salmon) + sey * tstats # calculations match those from confint
```

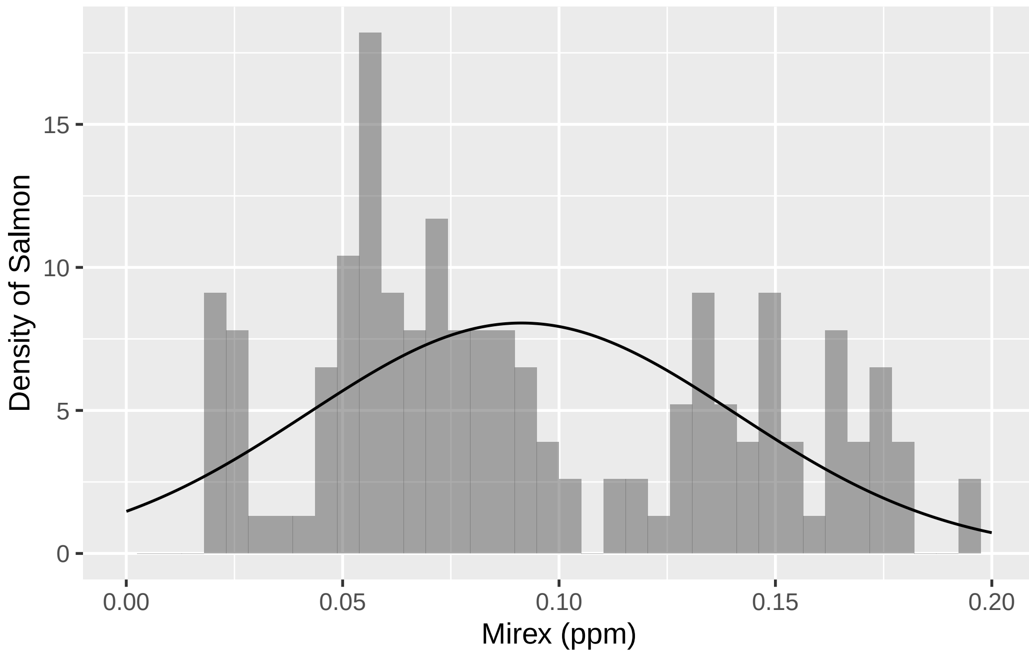
```
[1] 0.08334978 0.09933022
```

The `confint()` function takes an object, in this case a linear regression model, as an argument.

Example 14.2: Checking Assumptions and Conditions for Student's *t*

We can generate a histogram to check the assumptions of the model.

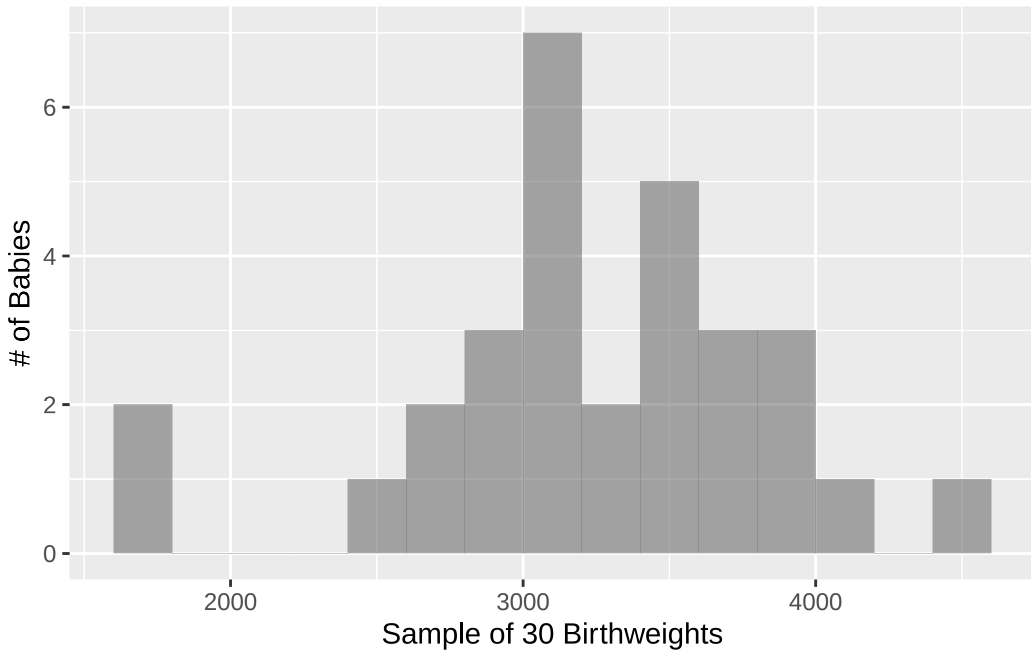
```
# With a normal distribution (page 450)
gf_dhistogram(~ mirex, data = Salmon, bins = 40) |>
  gf_dist(
    dist = "norm",
    mean = mean(~ mirex, data = Salmon),
    sd = sd(~ mirex, data = Salmon)
  ) |>
  gf_labs(x = "Mirex (ppm)", y = "Density of Salmon") +
  xlim(0, .2)
```



Step-By-Step Example: A One-Sample *t*-Interval for the Mean

We can do the same for a sample of size 30 from the `Babies` dataset.

```
BabiesSample <- sample(Babies, size = 30)
gf_histogram(~ weight, data = BabiesSample, binwidth = 200, center = 100) |>
  gf_labs(x = "Sample of 30 Birthweights", y = "# of Babies")
```




```
df_stats(~ weight, data = BabiesSample)
```

```
  response  min   Q1 median      Q3  max   mean       sd  n missing
1   weight 1671 2984   3230 3607.75 4593 3239.5 617.6798 30       0
```

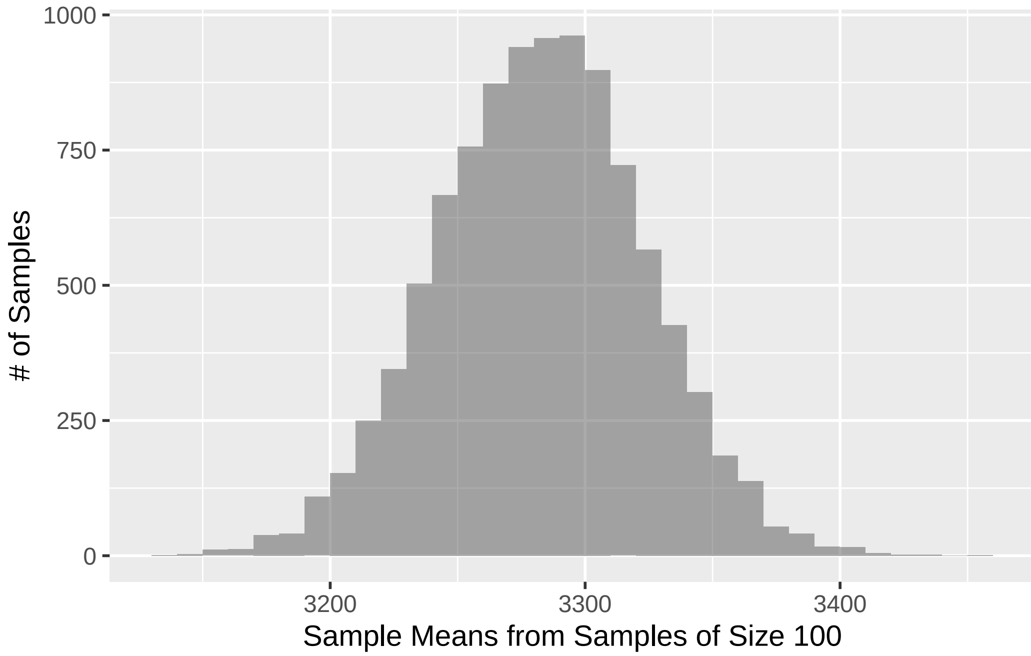
```
babieslm <- lm(weight ~ 1, data = BabiesSample)
confint(babieslm, level = 0.90)
```

```
                 5 %     95 %
(Intercept) 3047.885 3431.115
```

Section 14.3: Interpreting Confidence Intervals

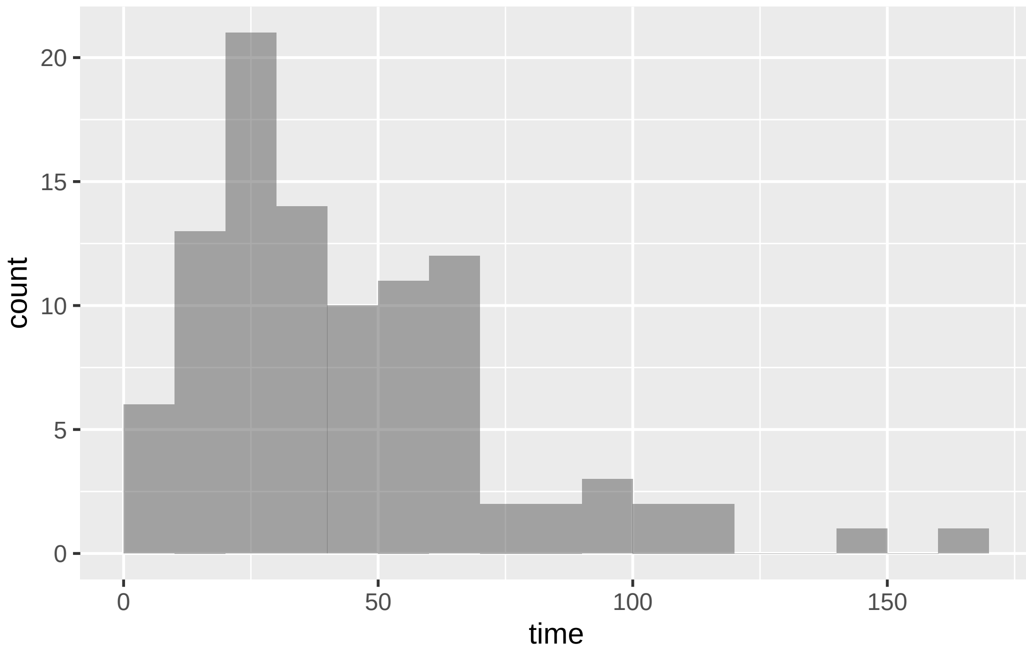
Section 14.4: Picking Our Interval up by Our Bootstraps

```
# page 453
gf_histogram(~ mean, data = WeightMeans, binwidth = 10, center = 5) |>
  gf_labs(x = "Sample Means from Samples of Size 100", y = "# of Samples")
```



Step-By-Step Example: A Bootstrap Confidence Interval for the Mean

```
# page 455
CommuteSample <-
  read_csv("http://nhorton.people.amherst.edu/is5/data/Commuter_sample.csv")
gf_histogram(~ time, data = CommuteSample, binwidth = 10, center = 5)
```




```
# Bootstrap
num_sim <- 10000
commutebootstrap <- do(num_sim) * mean(~ time, data = resample(CommuteSample))
```

The `resample()` function samples of that data set size with replacement. For more information about `resample()`, refer to the resampling vignette: https://cran.r-project.org/web/packages/mosaic/vignettes/

```
qdata(~ mean, p = c(.025, .975), data = commutebootstrap) # grab the percentiles with qdata
```

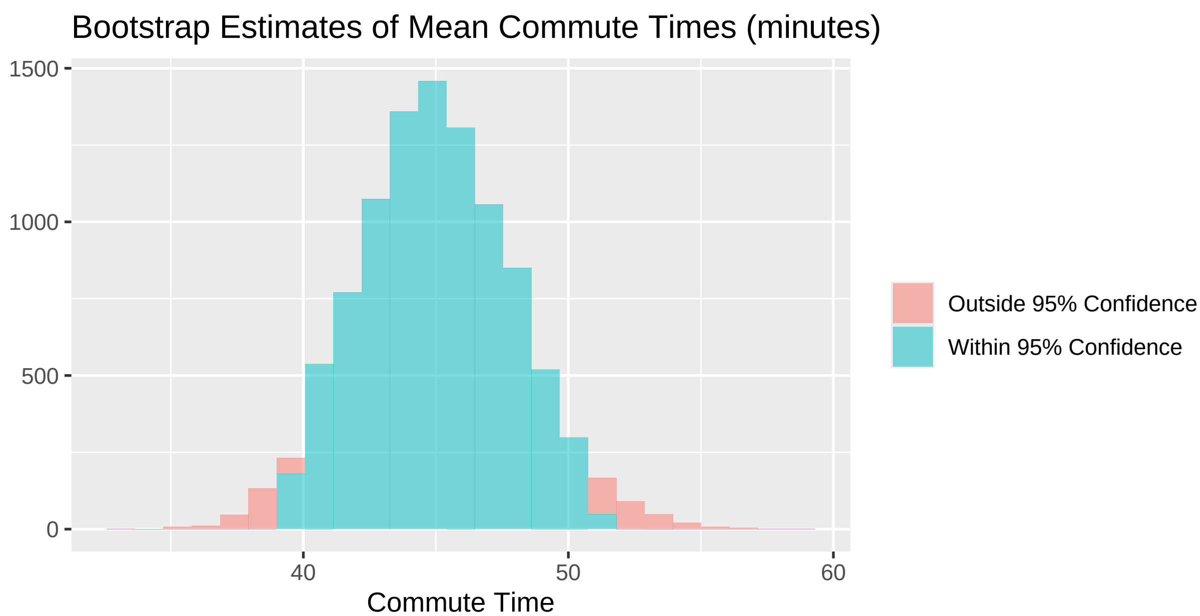
```
    2.5%    97.5%
39.27975 51.20025
```

```
confint(commutebootstrap, method = "quantile") # an equivalent quantile approach
```

```
  name    lower    upper level     method estimate
1 mean 39.27975 51.20025  0.95 percentile    44.98
```

```
commutebootstrap <- commutebootstrap |>
  mutate(interval = ifelse(mean > 39.28 & mean < 50.98, "Within 95% Confidence",
    "Outside 95% Confidence"
  )) # for fill
```

```
gf_histogram(
  ~ mean,
  fill = ~ interval, data = commutebootstrap,
  title = "Bootstrap Estimates of Mean Commute Times (minutes)"
) |>
  gf_labs(x = "Commute Time", y = "", fill = "")
```



Section 14.5: Thoughts About Confidence Intervals