STU22005 Laboratory Session 3

Professor Caroline Brophy

Week beginning 22nd February 2021

Instructions

- In this lab sheet, there will be a series of notes for you to work through. For each section, you should run the R code yourself on your own computer and verify the output provided. Ask your lab demonstrator any questions you have as you work through the material.
- In the last section of the sheet, there are several questions where you will write your own code. Your lab demonstrator is there to help if you get stuck in this section.
- Create a script for all the code that you write in this lab session. Save the script into the folder you have for the module. Save your code regularly throughout the class. (Do not write your code into the "Console" window as this will not save in an R script.)

Introduction

In Section 4.2 and 4.3 of the lecture notes, we looked at how samples are constructed for population means.

Specifically, we saw that when $Y_1, Y_2, ..., Y_n \sim N(\mu, \sigma^2)$ and IID and where σ^2 is unknown, then

$$P(\bar{Y} - t_{\nu,\frac{\alpha}{2}}\frac{S}{\sqrt{n}} < \mu < \bar{Y} + t_{\nu,\frac{\alpha}{2}}\frac{S}{\sqrt{n}}) = 1 - \alpha$$

Today, we are going to examine this theory using a simulation study. First we will look at how to use R to generate a confidence interval for a set of data.

Constructing confidence intervals

Examine the data

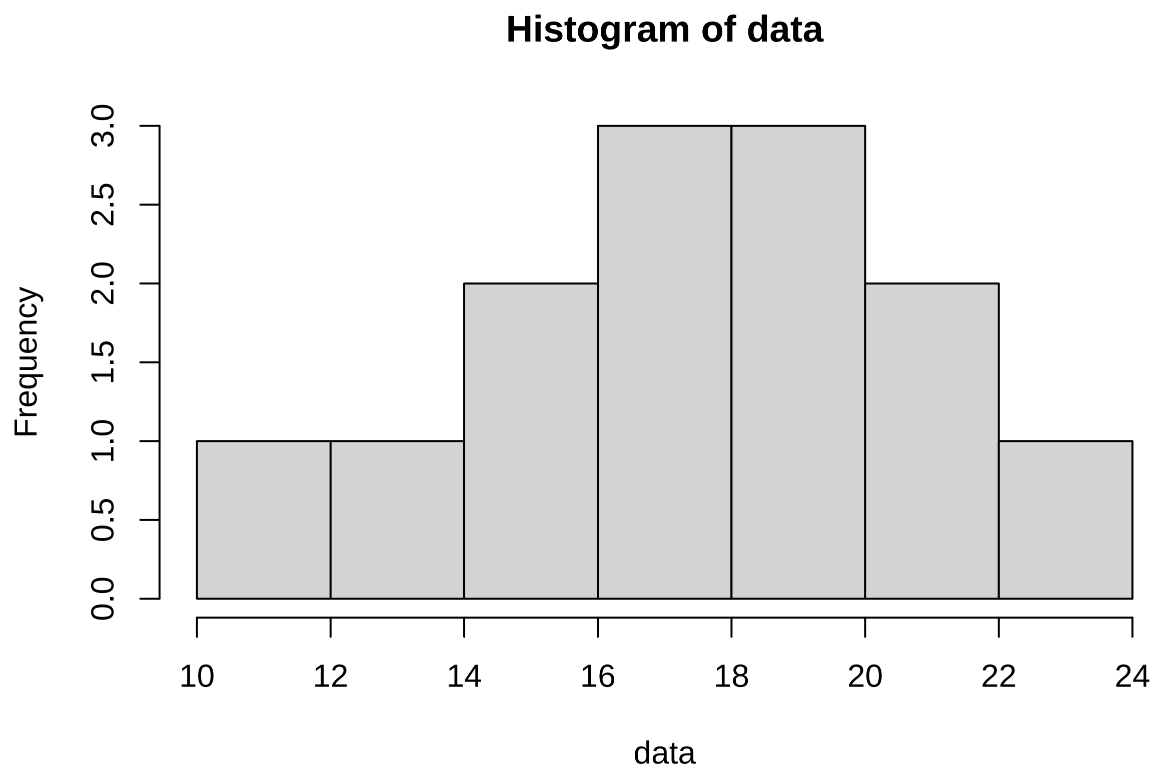
A random sample of data was collected from a population and recorded as:

```r
# Create a vector containing the data
data <- c(13.1, 19.6, 21.3, 11.6, 15.4, 23.7, 18.6, 16.1, 19.3, 17.4, 21.5, 16.8, 14.9)
# Print out the data
data
```

```
##  [1] 13.1 19.6 21.3 11.6 15.4 23.7 18.6 16.1 19.3 17.4 21.5 16.8 14.9
```

It was assumed that this data comes from a normally distributed population. To see if this is reasonable, we will construct a histogram for the data.

```r
# Generate a histogram
hist(data)
```



Does it seem reasonable to assume a normal population?

Make the CI using `t.test`.

We can generate CIs using the `t.test()` function in R. The default is 95% confidence.

```r
# Find the confidence interval using t.test (default is 95% confidence interval)
t.test(data)
```

```
##
##  One Sample t-test
##
## data:  data
## t = 18.277, df = 12, p-value = 3.982e-10
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  15.53576 19.74116
## sample estimates:
## mean of x
##  17.63846
```

We see that the 95% confidence interval is: (15.54, 19.74).

- What is the interpretation of this confidence interval?

We are 95% confident that the true population mean μ lies between 15.54 and 19.74.

- What is the hypothesis test that is also being tested in the above output?

$\mathrm{H}_0 : \mu = 0$, versus $\mathrm{H}_0 : \mu \neq 0$

- Is this a sensible hypothesis here?

We have no idea! No context was given, and hypotheses should be specified based on prior belief about the mean of the population. By default, R will test against 0, but we can change the default. Check out all the options by running `?t.test`

By default, t.test returns a 95% confidence interval. We can also find a 90% confidence interval.

- Before looking at the output below, would you expect to see a narrower or wider interval than the 95% interval above?

```r
# Find the 90% confidence interval using t.test
t.test(data, conf.level = 0.9)
```

```
##
##  One Sample t-test
##
## data:  data
## t = 18.277, df = 12, p-value = 3.982e-10
## alternative hypothesis: true mean is not equal to 0
## 90 percent confidence interval:
##  15.91844 19.35849
## sample estimates:
## mean of x
##  17.63846
```

We see that the 90% confidence interval is: (15.92, 19.36).

Make the CI 'manually'

Functions in R are a useful way to put together lines of code for use over and over.

For example, the following function called `CI` will produce a confidence interval for data `x`, a specified alpha level and presented rounded to a certain number of decimal places. Go through each line of code in the function and search the R help files to make sure you know what each line is doing.

```r
# Create a function where x is the data and alpha to determine the level
# of confidence (100*(1-alpha)%)
CI <- function(x, alpha, round) {
  n <- length(x)
  m <- mean(x)
  s <- sd(x)
  se <- s/sqrt(n)
  tval <- qt(0.5*alpha, df = n-1, lower.tail=FALSE )
  round(c( m - tval * se, m + tval * se ), round)
}
```

We can use this function now to construct confidence intervals for the data above.

- A 95% confidence interval:

```r
# Create a function where x is the data and alpha to determine the level
# of confidence (100*(1-alpha)%)
CI(x = data, alpha = 0.05, round = 2)
```

```
## [1] 15.54 19.74
```

- A 90% confidence interval:

```r
# Create a function where x is the data and alpha to determine the level
# of confidence (100*(1-alpha)%)
CI(x = data, alpha = 0.1, round = 2)
```

```
## [1] 15.92 19.36
```

Try some other confidence levels and different number of decimal places yourself.

Simulation study

Simulate 1000 datasets each of size 15 from N(35, 9).

```r
# Simulate 1000 datasets of size 15 from N(35, 9) and store in a matrix
set.seed(2646537)
mean_val = 35
sd_val = 3
samples = 1000
size = 15
x_sim <- matrix(rnorm(samples*size, mean = mean_val, sd = sd_val), nrow=samples,
                ncol=size, byrow=TRUE)
head(x_sim)
```

```
##          [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]     [,8]
## [1,] 34.83475 34.62040 37.36633 37.82874 35.32423 31.55374 33.88428 37.57255
## [2,] 39.53413 32.33123 31.67806 37.52939 28.51531 38.17218 32.47591 34.81324
## [3,] 36.02773 37.47317 37.68937 37.10155 31.18523 32.64913 30.91685 33.28539
## [4,] 34.93994 38.70333 35.47057 40.84601 35.98842 35.91212 34.38294 36.27744
## [5,] 37.28475 33.17796 39.62568 33.43506 30.44311 32.41080 30.28986 37.21542
## [6,] 42.51327 36.41066 32.23130 38.07373 37.03873 35.56509 34.02546 32.30357
##          [,9]    [,10]    [,11]    [,12]    [,13]    [,14]    [,15]
## [1,] 29.60297 35.36716 39.94571 34.37661 33.50760 30.00947 32.36061
## [2,] 36.77427 35.26131 39.19979 38.02758 30.66132 36.25693 33.88787
## [3,] 34.09606 37.08055 41.52477 38.81536 39.81109 36.34767 32.68370
## [4,] 33.63959 37.65694 36.20192 37.76382 34.19711 32.18568 32.12954
## [5,] 39.62225 41.47868 36.20215 31.82571 39.86455 40.24877 36.11197
## [6,] 34.76550 42.96254 35.48996 36.69347 31.47773 35.53573 36.61273
```

Construct 95% confidence intervals for each dataset and find the proportion of confidence intervals that contain the true μ value.

To do this, we will write a new function called `coverage`.

```r
# Create the function coverage
coverage <- function(X, alpha, mu, round) {
  CIs <- matrix(nrow=nrow(X), ncol=2)
  for(k in 1:nrow(X)) CIs[k, ] <- CI(X[k,], alpha, round)
  z <- (CIs[,1] < mu) * (CIs[ , 2 ] > mu)
  sum(z)/nrow(X)
}
```

To use the function, we specify the input values as follows. It will return the proportion of intervals from our simulated dataset that contained the true μ value (here $\mu = 35$ as specified above).

```r
# Call the function coverage:
coverage(X = x_sim, alpha = 0.05, mu = mean_val, round = 2)
```

```
## [1] 0.947
```

Putting your code into practice

Save the code you have written so far in this lab session. Generally, it is a good idea to save your code every five minutes!

For the remainder of this laboratory session, please try not to use copy and paste, but rather Write out your own code. Try to remember the code yourself, but you can look back at the earlier code whenever you need to. Typing out your own code gives you a better chance of understanding what each part of the code does, and will help your R programming skills to develop.

It is very good practice to include good comments for each part of your R code. Please get into the habit of this every time you write R code.

Questions

1. A sample of data was collected from a normally distributed population. The values are: 69, 74, 79, 81, 85, 86, 89, 90, 94, 97, 100, 105. Generate a 99% confidence interval for the data using `t.test`.
2. Generate the same 99% CI using the manual function created in the earlier section.
3. Repeat the previous two questions for a 90% confidence interval. Before you write the code, do you expect to get a wider or narrower interval here?
4. Using the code and functions above, do a simulation study to find the coverage of 95% confidence intervals constructed from samples of size 20 drawn from a N(50, 16) population. Use a seed value of 9498. Use 1000 simulated samples. Repeat for 10,000, 100,000 and 1,000,000 (but keep the same seed). What do you find as the number of simulated samples increases?
5. If you have time at the end, rerun your simulation study under different conditions, for example, you could change the mean of the distribution or the sample size.