Assignment 2

Due Sunday September 27 at 11:59pm

Instructions (the same as for Assignment 1): Make an R Notebook and in it answer the two questions below (one Notebook for both questions). When you are done, hand in on Quercus the *output* from Previewing (or Knitting) your Notebook. Do *not* hand in the Notebook itself. You want to show that you can (i) write code that will answer the questions, (ii) run that code and get some sensible output, (iii) write some words that show you know what is going on and that reflect your conclusions about the data. Your goal is to convince the grader that you *understand* what you are doing: not only doing the right thing, but making it clear that you know *why* it's the right thing.

Do *not* expect to get help on this assignment. The purpose of the assignments is for you to see how much *you* have understood. You will find that you also learn something from grappling with the assignments. The time to get help is after you watch the lectures and work through the problems from PASIAS, via tutorial and the discussion board. The only reason to contact the instructor while working on the assignments is to report something missing like a data file that cannot be read.

You have 3 hours to complete this assignment after you start it.

My solutions to this assignment, with extra discussion, will be available after everyone has handed in their assignment.

1. The data set at http://ritsokiguess.site/STAC32/cholest.csv contains cholesterol measurements for heart attack patients (at several different times) as well as for a group of control patients. We will focus on the control patients in this question.

   (a) Read in and display (some of) the data.

Solution: This is (as you might guess) a `.csv`, so:

```
my_url <- "http://ritsokiguess.site/STAC32/cholest.csv"
cholest <- read_csv(my_url)
```

```
## Parsed with column specification:
## cols(
##   `2-Day` = col_double(),
##   `4-Day` = col_double(),
##   `14-Day` = col_double(),
##   control = col_double()
## )
```

```
cholest
```

```
## # A tibble: 30 x 4
##    `2-Day` `4-Day` `14-Day` control
##      <dbl>   <dbl>    <dbl>   <dbl>
##  1     270     218      156     196
##  2     236     234       NA     232
##  3     210     214      242     200
```

```
##  4     142     116      NA     242
##  5     280     200      NA     206
##  6     272     276     256     178
##  7     160     146     142     184
##  8     220     182     216     198
##  9     226     238     248     160
## 10     242     288      NA     182
## # ... with 20 more rows
```

Note for yourself that there are 30 observations (and some missing ones), and a column called `control` that is the one we'll be working with.

Extra: the 2-day, 4-day and 14-day columns have the funny "backticks" around their names, because a column name cannot contain a `-` or start with a number. This is not a problem here, since we won't be using those columns, but if we wanted to, this would not work:

```
cholest %>% summarize(xbar = mean(2-Day))
```

```
## Error: Problem with `summarise()` input `xbar`.
## x object 'Day' not found
## i Input `xbar` is `mean(2 - Day)`.
```

because it is looking for a column called `Day`, which doesn't exist. The meaning of `2-Day` is "take the column called `Day` and subtract it from 2". To make this work, we have to supply the backticks ourselves:

```
cholest %>% summarize(xbar = mean(`2-Day`, na.rm = TRUE))
```

```
## # A tibble: 1 x 1
##    xbar
##   <dbl>
## 1  254.
```

This column also has missing values (at the bottom), so here I've asked to remove the missing values before working out the mean. Otherwise the mean is, unhelpfully, missing as well.

You might imagine that dealing with column names like this would get annoying. There is a package called `janitor` that has a function called `clean_names` to save you the trouble. Install it first, then load it:

```
library(janitor)
```

and then pipe your dataframe into `clean_names` and see what happens:

```
cholest %>% clean_names() -> cholest1
cholest1
```

```
## # A tibble: 30 x 4
##    x2_day x4_day x14_day control
##     <dbl>  <dbl>   <dbl>   <dbl>
##  1    270    218     156     196
##  2    236    234      NA     232
##  3    210    214     242     200
##  4    142    116      NA     242
##  5    280    200      NA     206
##  6    272    276     256     178
##  7    160    146     142     184
```

```
##  8    220    182     216     198
##  9    226    238     248     160
## 10    242    288      NA     182
## # ... with 20 more rows
```

These are all legit column names; the `-` has been replaced by an underscore, and each of the first three column names has gained an `x` on the front so that it no longer starts with a number. This then works:

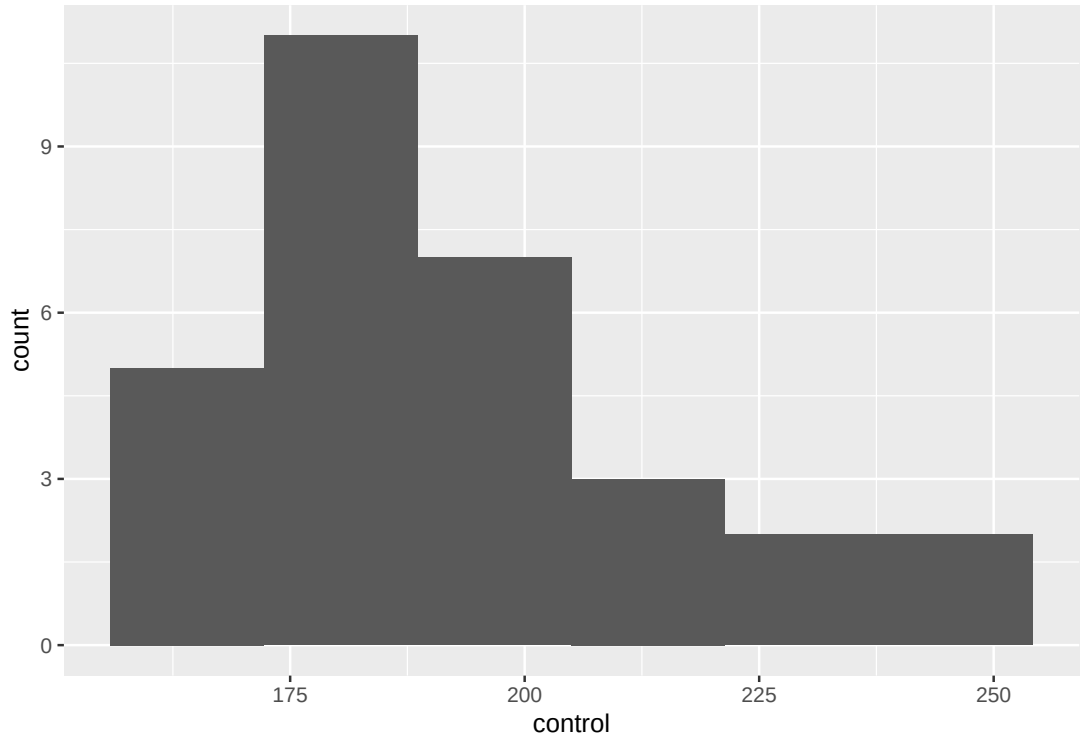
```
cholest1 %>% summarize(xbar = mean(x2_day, na.rm = TRUE))
```

```
## # A tibble: 1 x 1
##    xbar
##   <dbl>
## 1  254.
```

(b) Make a suitable plot of the cholesterol levels of the control patients, and comment briefly on the shape of the distribution.

Solution: There is one quantitative variable, so a histogram, as ever:

```
ggplot(cholest, aes(x=control)) + geom_histogram(bins=6)
```



Pick a number of bins that shows the shape reasonably well. Too many or too few won't. (Sturges' rule says 6, since there are 30 observations and $2^5 = 32$.) Seven bins also works, but by the time you get to 8 bins or more, you are starting to lose a clear picture of the shape. Four bins is, likewise, about as low as you can go before getting too crude a picture.

Choosing one of these numbers of bins will make it clear that the distribution is somewhat

skewed to the right.

(c) It is recommended that people in good health, such as the Control patients here, keep their cholesterol level below 200. Is there evidence that the mean cholesterol level of the population of people of which the Control patients are a sample is less than 200? State your conclusion in the context of the data.

Solution: The word "evidence" means to do a hypothesis test and get a P-value. Choose an α first, such as 0.05.

Testing a mean implies a one-sample t-test. We are trying to prove that the mean is less than 200, so that's our alternative: $H_a : \mu < 200$, and therefore the null is that the mean is equal to 200: $H_0 : \mu = 200$. (You might think it makes more logical sense to have $H_0 : \mu \geq 200$, which is also fine. As long as the null hypothesis has an equals in it in a logical place, you are good.)

```
with(cholest, t.test(control, mu=200, alternative = "less"))
```

```
##
##  One Sample t-test
##
## data:  control
## t = -1.6866, df = 29, p-value = 0.05121
## alternative hypothesis: true mean is less than 200
## 95 percent confidence interval:
##      -Inf 200.0512
## sample estimates:
## mean of x
##  193.1333
```

This is also good:

```
t.test(cholest$control, mu=200, alternative = "less")
```

```
##
##  One Sample t-test
##
## data:  cholest$control
## t = -1.6866, df = 29, p-value = 0.05121
## alternative hypothesis: true mean is less than 200
## 95 percent confidence interval:
##      -Inf 200.0512
## sample estimates:
## mean of x
##  193.1333
```

I like the first version better because a lot of what we do later involves giving a data frame, and then working with things in that data frame. This is more like that.

This test is *one*-sided because we are looking for evidence of *less*; if the mean is actually *more* than 200, we don't care about that. For a one-sided test, R requires you to say which side you are testing.

The P-value is not (quite) less than 0.05, so we cannot quite reject the null. Therefore, there is no evidence that the mean cholesterol level (of the people of which the control group are a

sample) is less than 200. Or, this mean is not significantly less than 200. Or, we conclude that this mean is equal to 200. Or, we conclude that this mean could be 200. Any of those.

If you chose a different α, draw the right conclusion for the α you chose. For example, with $\alpha = 0.10$, we *do* have evidence that the mean is less than 200. Being consistent is more important than getting the same answer as me.

(d) What could the population mean cholesterol level be? You might need to get some more output to determine this.

Solution:

This is *not* quoting the sample mean, giving that as your answer, and then stopping. The sample mean should, we hope, be somewhere the population mean, but it is almost certainly not the same as the population mean, because there is variability due to random sampling. (This is perhaps the most important thing in all of Statistics: recognizing that variability exists and dealing with it.)

With that in mind, the question means to get a range of values that the population mean could be: that is to say, a confidence interval. The one that came out of the previous output is one-sided, to go with the one-sided test, but confidence intervals for us are two-sided, so we have to run the test again, but two-sided, to get it. To do that, take out the "alternative", thus (you can also take out the null mean, since a confidence interval has no null hypothesis):

```
with(cholest, t.test(control))
```

```
##
##  One Sample t-test
##
## data:  control
## t = 47.436, df = 29, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  184.8064 201.4603
## sample estimates:
## mean of x
##  193.1333
```

With 95% confidence, the population mean cholesterol level is between 184.8 and 201.5.

You need to state the interval, and you also need to round off the decimal places to something sensible. This is because in your statistical life, you are providing results to someone else *in a manner that they can read and understand.* They do not have time to go searching in some output, or to fish through some excessive number of decimal places. If that's what you give them, they will ask you to rewrite your report, wasting everybody's time when you could have done it right the first time.

How many decimal places is a good number? Look back at your data. In this case, the cholesterol values are whole numbers (zero decimal places). A confidence interval is talking about a mean. In this case, we have a sample size of 30, which is between 10 and 100, so we can justify one extra decimal place beyond the data, here one decimal altogether, or two *at the absolute outside*. (Two is more justifiable if the sample size is bigger than 100.) See, for example, this, in particular the piece at the bottom.

(e) (2 points) Explain briefly why you would be reasonably happy to trust the t procedures in this question. (There are two points you need to make.)

Solution: The first thing is to look back at the graph you made earlier. This was skewed to the right ("moderately" or "somewhat" or however you described it). This would seem to say that the t procedures were not very trustworthy, since the population distribution doesn't look very normal in shape.

However, the second thing is to look at the sample size. We have the central limit theorem, which says (for us) that the larger the sample is, the less the normality matters, when it comes to estimating the mean. Here, the sample size is 30, which, for the central limit theorem, is large enough to overcome moderate non-normality in the data.

My take, which I was trying to guide you towards, is that our non-normality was not too bad, and so our sample size is large enough to trust the t procedures we used.

Extras:

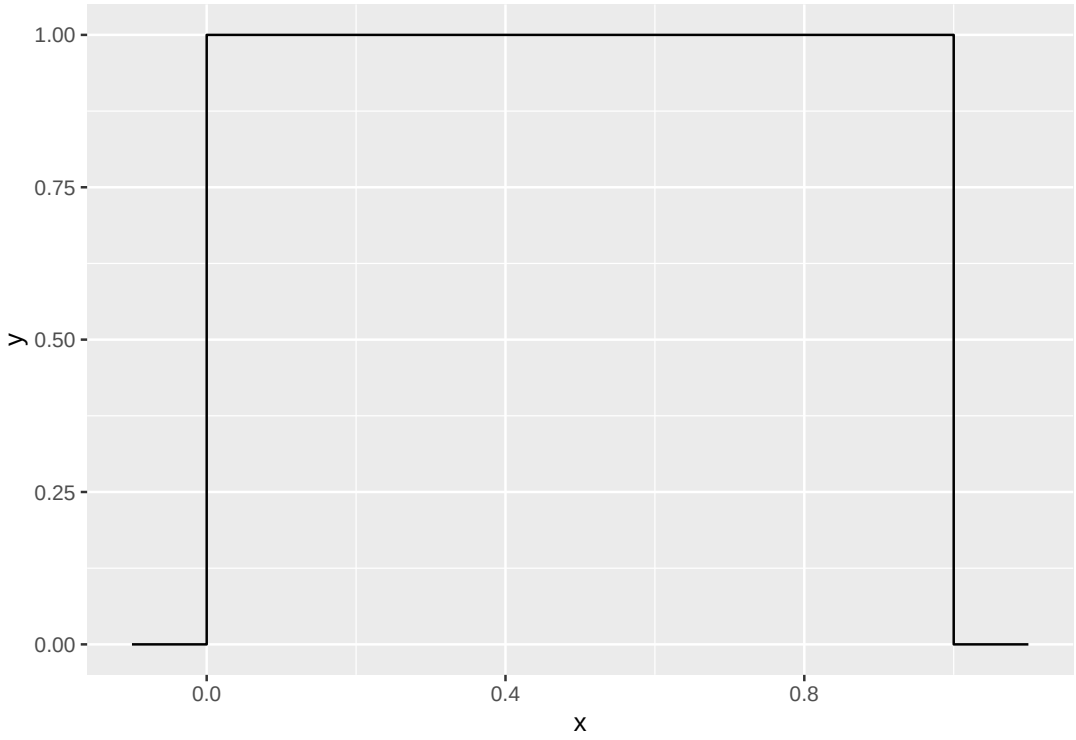
(i) **There is nothing magical about a sample size of 30.** What matters is the tradeoff between sample size and the extent of the non-normality. If your data is less normal, you need a larger sample size to overcome it. Even a sample size of 500 might not be enough if your distribution is very skewed, or if you have extreme outliers.

The place $n = 30$ comes from is back from the days when we only ever used printed tables. In most textbooks, if you printed the t-table on one page in a decent-sized font, you'd get to about 29 df before running out of space. Then they would say "∞ df" and put the normal-distribution z numbers in. If the df you needed was bigger than what you had in the table, you used this last line: that is, you called the sample "large". Try it in your stats textbooks: I bet the df go up to 30, then you get a few more, then the z numbers.

(ii) By now you are probably thinking that this is very subjective, and so it is. What actually matters is the shape of the thing called the *sampling distribution of the sample mean*. That is to say, what kind of sample means you might get in repeated samples from your population. The problem is that you don't know what the population looks like.[1] But we can fake it up, in a couple of ways: (a) we can play what-if and pretend we know what the population looks like (to get some understanding for "populations like that"), or (b) we can use a technique called the "bootstrap" that will tell us what kind of sample means we might get from the population that *our* sample came from (this seems like magic and, indeed, is).

The moral of the story is that the central limit theorem is more powerful than you think.

To illustrate my (a), let's pretend the population looks like this, with a flat top:



Only values between 0 and 1 are possible, and each of those is equally likely. Not very normal in shape. So let's take some random samples of size *three*, not in any sense a large sample, from this population, and see what kind of sample means we get. This technique is called *simulation*: rather than working out the answer by math, we're letting the computer approximate the answer for us. Here's one simulated sample:

```
u <- runif(3)
u
```

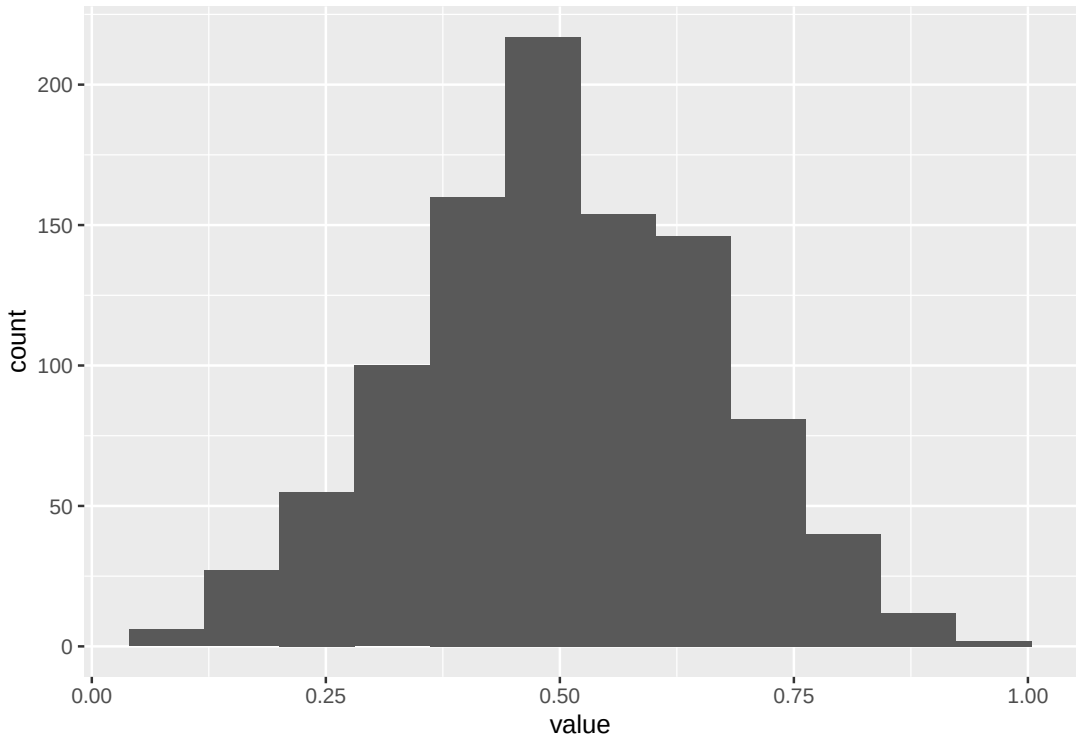
```
## [1] 0.9475841 0.1245953 0.2277288
```

```
mean(u)
```

```
## [1] 0.4333027
```

and here's the same thing 1000 times, including a histogram of the sample means:

```
rerun(1000, runif(3)) %>%
    map_dbl(~mean(.)) %>%
    enframe() %>%
    ggplot(aes(x=value)) + geom_histogram(bins=12)
```



This is our computer-generated assessment of what the sampling distribution of the sample mean looks like. Isn't this looking like a normal distribution?

Let's take a moment to realize what this is saying. If the population looks like the flat-topped uniform distribution, the central limit theorem kicks in for a sample of size *three*, and thus if your population looks like this, t procedures will be perfectly good for $n = 3$ or bigger, *even though the population isn't normal*.

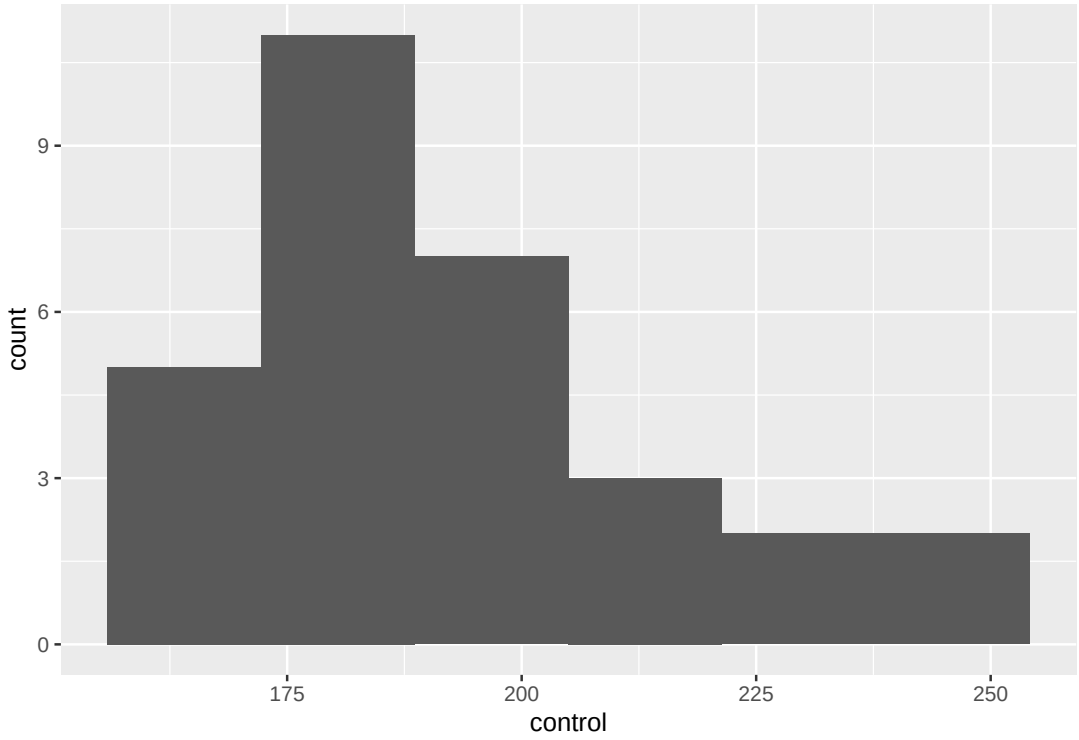
Thus, when you're thinking about whether to use a t-test or something else (that we'll learn about later), the distribution shape matters, *but so does the sample size*.

I should say a little about my code. I'm not expecting you to figure out details now (we see the ideas properly in simulating power of tests), but in words, one line at a time:

- generate 1000 ("many") samples each of 3 observations from a uniform distribution
- for each sample, work out the mean of it
- turn those sample means into a data frame with a column called `value`
- make a histogram of those.

Now, the central limit theorem doesn't always work as nicely as this, but maybe a sample size of 30 is large enough to overcome the skewness that we had:

```
ggplot(cholest, aes(x=control)) + geom_histogram(bins=6)
```



That brings us to my (b) above.

The sample that we had is in some sense an "estimate of the population". To think about the sampling distribution of the sample mean, we need more estimates of the population. How might we get those? The curious answer is to *sample from the sample*. This is the idea behind the *bootstrap*. (There will be, by the time you read this, an extra-insight lecture on the bootstrap.) The name comes from the expression "pulling yourself up by your own bootstraps", meaning "to begin an enterprise or recover from a setback without any outside help" (from here), something that should be difficult or impossible. How is it possible to understand a sampling distribution with only one sample?

We have to be a bit careful. Taking a sample from the sample would give us the original sample back. So, instead, we sample *with replacement*, so that each bootstrap sample is different:

```
sort(cholest$control)
```

```
##  [1] 160 162 164 166 170 176 178 178 182 182 182 182 182 184 186 188 196 198
## [19] 198 198 200 200 204 206 212 218 230 232 238 242
```

```
sort(sample(cholest$control, replace=TRUE))
```

```
##  [1] 164 166 166 166 166 176 178 178 182 182 182 182 182 188 198 198 198 200
## [19] 200 200 200 204 206 206 218 218 230 232 232 242
```

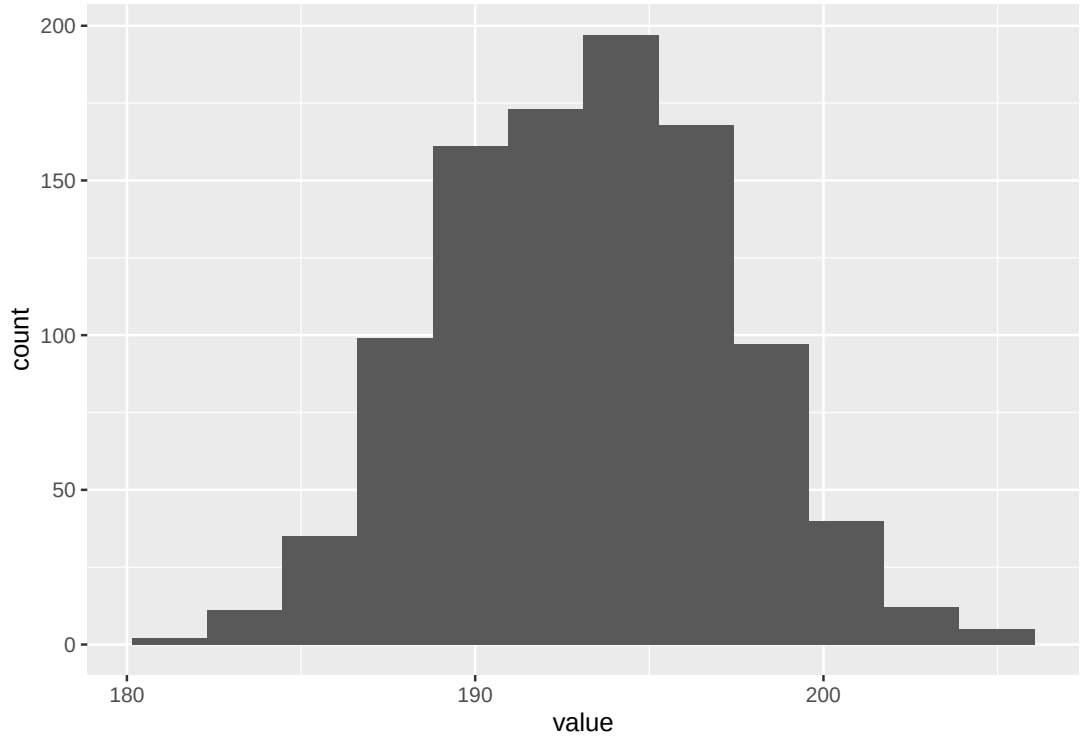
A bootstrap sample contains repeats of the original data values, and misses some of the others. Here, the original data had values 160 and 162 that are missing in the bootstrap sample; the original data had one value 166, but the bootstrap sample has *four*! I sorted the data and the bootstrap sample to make this clearer; you will not need to sort. This is a perfectly good bootstrap sample:

```
sample(cholest$control, replace = TRUE)
```

```
##  [1] 242 232 198 160 242 182 182 182 198 162 212 198 242 204 242 242 170 198
## [19] 182 206 232 170 218 188 166 178 164 160 218 196
```

So now we know what to do: take lots of bootstrap samples, work out the mean of each, plot the means, and see how normal it looks. The only new idea here is the sampling with replacement:

```
rerun(1000, sample(cholest$control, 30, replace = T)) %>%
  map_dbl(~mean(.)) %>%
  enframe() %>%
  ggplot(aes(x=value)) + geom_histogram(bins=12)
```



That looks pretty normal, not obviously skewed, and so the t procedures we used will be reliable enough.

2. Take your right hand, and stretch the fingers out as far as you can. The distance between the tip of your thumb and the tip of your little (pinky) finger is your handspan. The students in a Statistics class at Penn State measured their handspans and also whether they identified as male or female. The data are at http://ritsokiguess.site/STAC32/handspan.txt, with handspans measured in inches. Thinking of these as a random sample of all possible students, is it true that males have a larger mean handspan than females? This is what we will explore.

   (a) Read in and display (some of) the data.

Solution: This is a delimited (by spaces) file, so:

```
my_url <- "http://ritsokiguess.site/STAC32/handspan.txt"
span <- read_delim(my_url, " ")
```

```
## Parsed with column specification:
## cols(
##   sex = col_character(),
##   handspan = col_double()
## )
```

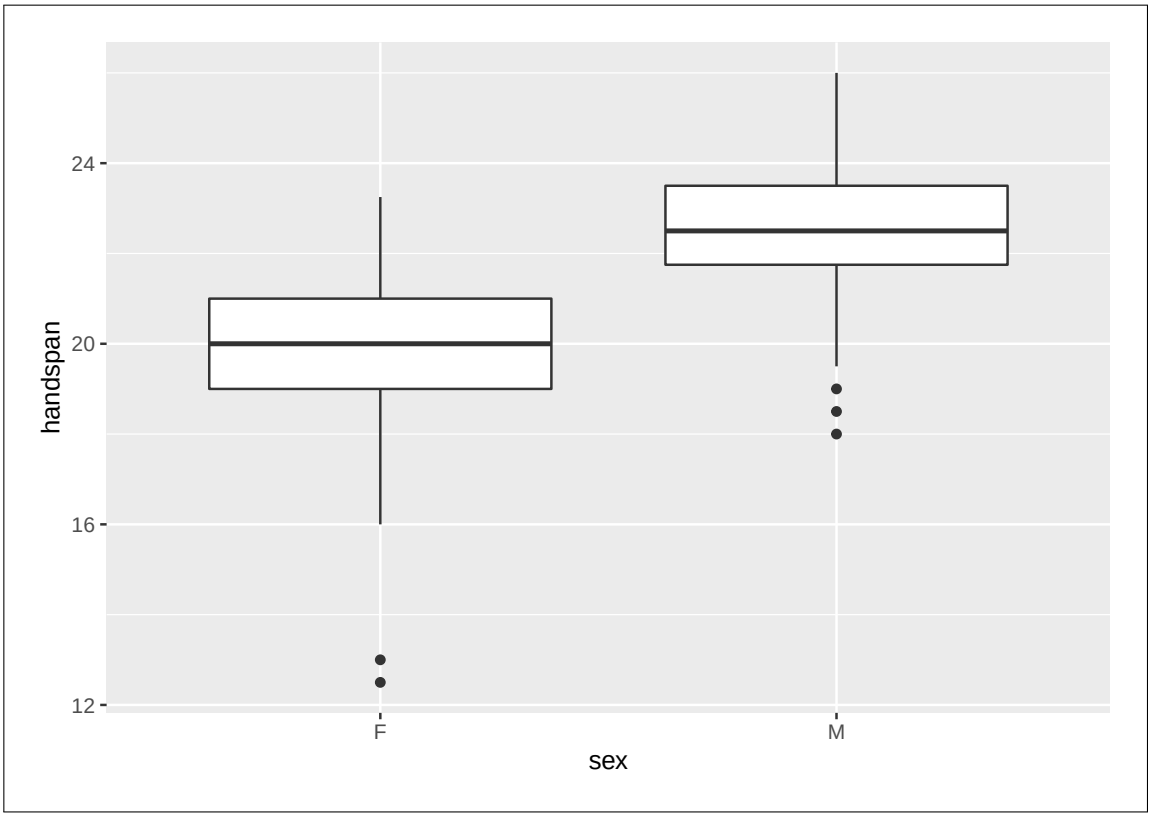
```
span
```

```
## # A tibble: 190 x 2
##    sex   handspan
##    <chr>    <dbl>
##  1 M         21.5
##  2 M         22.5
##  3 M         23.5
##  4 F         20
##  5 F         19
##  6 F         20.5
##  7 F         20.5
##  8 F         20.2
##  9 M         23
## 10 M         24.5
## # ... with 180 more rows
```

(b) Make a suitable graph of the two columns.

Solution: One quantitative variable and one categorical one, so a boxplot:

```
ggplot(span, aes(x=sex, y=handspan)) + geom_boxplot()
```



(c) Run a suitable two-sample t-test to address the question of interest. What do you conclude, in the context of the data?

Solution: We are trying to show that males have a *larger* mean handspan, so we need an `alternative`. To see which: there are two sexes, F and M in that order, and we are trying to show that F is less than M:

```
t.test(handspan~sex, data=span, alternative="less")
```

```
##
##  Welch Two Sample t-test
##
## data:  handspan by sex
## t = -10.871, df = 187.92, p-value < 2.2e-16
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##       -Inf -2.154173
## sample estimates:
## mean in group F mean in group M
##        20.01699        22.55747
```

The P-value is very small, so there is no doubt that males have larger average handspans than females.

(d) Obtain a 90% confidence interval for the difference in mean handspan between males and females. Do you need to run any more code? Explain briefly.

Solution: A confidence interval is two-sided, so we have to re-run the test without the `alternative` to make it two-sided. Note also that we need a 90% interval, which is different from the default 95%, so we have to ask for that too:

```
t.test(handspan~sex, data=span, conf.level=0.90)
```

```
##
##  Welch Two Sample t-test
##
## data:  handspan by sex
## t = -10.871, df = 187.92, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 90 percent confidence interval:
##  -2.926789 -2.154173
## sample estimates:
## mean in group F mean in group M
##        20.01699        22.55747
```

The interval is -2.93 to -2.15, *which you should say.* It would be even better to say that males have a mean handspan between 2.15 and 2.93 inches larger than that of females. You also need to round off your answer: the data are given to 0 or 1 decimals, so your interval should be given to 1 or 2 decimals (since the confidence interval is for a mean).

On a question like this, the grader is looking for three things:

- getting the output
- saying what the interval is
- rounding it to a suitable number of decimals.

Thus, getting the output alone is only one out of three things.

(e) Explain briefly why you might have some concerns about the validity of the t-tests you ran in this question. Or, if you don't have any concerns, explain briefly why that is.

Solution: The major assumption here is that the male and female handspans have (approximate) normal distributions. The boxplots we drew earlier both had low-end outliers, so the normality is questionable.

Also, say something about the sample sizes and whether or not you think they are large enough to be helpful.

How big are our sample sizes?

```
span %>% count(sex)
```

```
## # A tibble: 2 x 2
##   sex       n
##   <chr> <int>
## 1 F       103
## 2 M        87
```

My suspicion is that we are saved by two things: the sample sizes are large enough for the central limit theorem to help us, and in any case, the conclusion is so clear that the assumptions can afford to be off by a bit.

Extra: one way to think about whether we should be concerned about the lack of normality is

to use the *bootstrap* to see what the sampling distribution of the sample mean might look like for males and for females. (This is the stuff in Lecture 5a.) The way this works is to sample from each distribution *with replacement*, work out the mean of each sample, then repeat many times, once for the females and once for the males.

To start with the females, the first thing to do is to grab *only* the rows containing the females. This, using an idea from Lecture 5a that we see again properly later, is `filter`:

```
span %>% filter(sex=="F") -> females
females
```

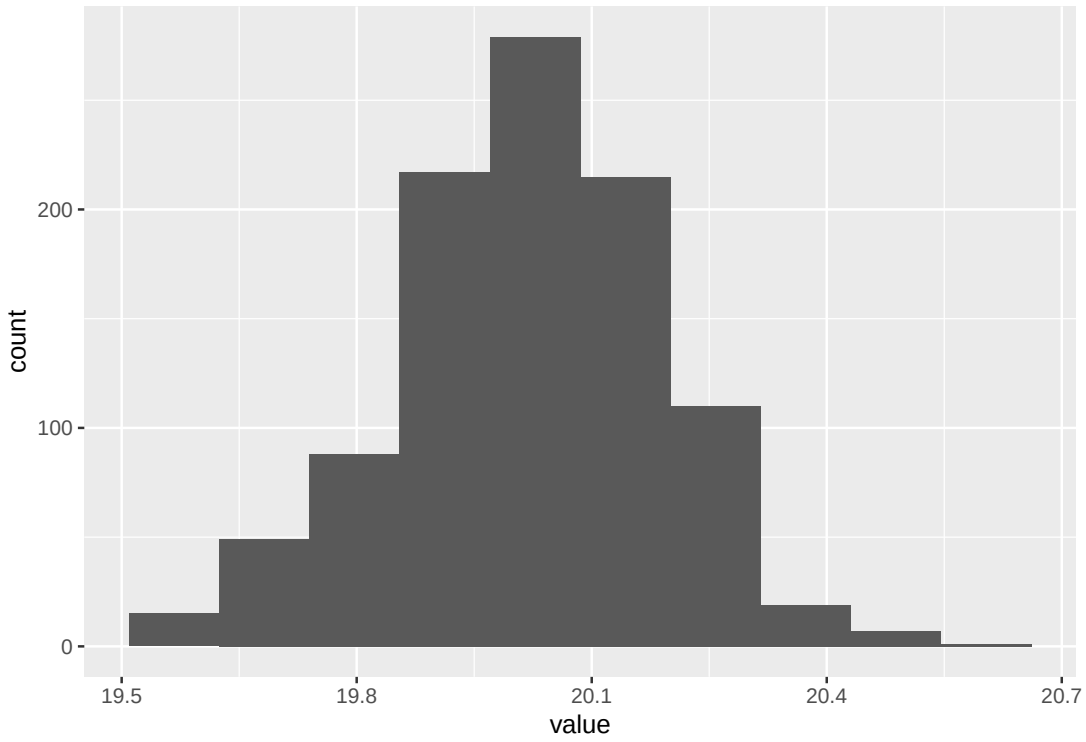
```
## # A tibble: 103 x 2
##    sex   handspan
##    <chr>    <dbl>
##  1 F         20
##  2 F         19
##  3 F         20.5
##  4 F         20.5
##  5 F         20.2
##  6 F         20
##  7 F         18
##  8 F         20.5
##  9 F         22
## 10 F         20
## # ... with 93 more rows
```

There are 103 females. From these we need to take a "large" number of bootstrap samples to get a sense of how the mean handspan of the females varies:

```
set.seed(457299)
rerun(1000, sample(females$handspan, replace = TRUE)) %>%
  map_dbl(~mean(.)) %>%
  enframe() -> d
```

Then we make a histogram of the bootstrap sampling distribution of the sample mean for the females:

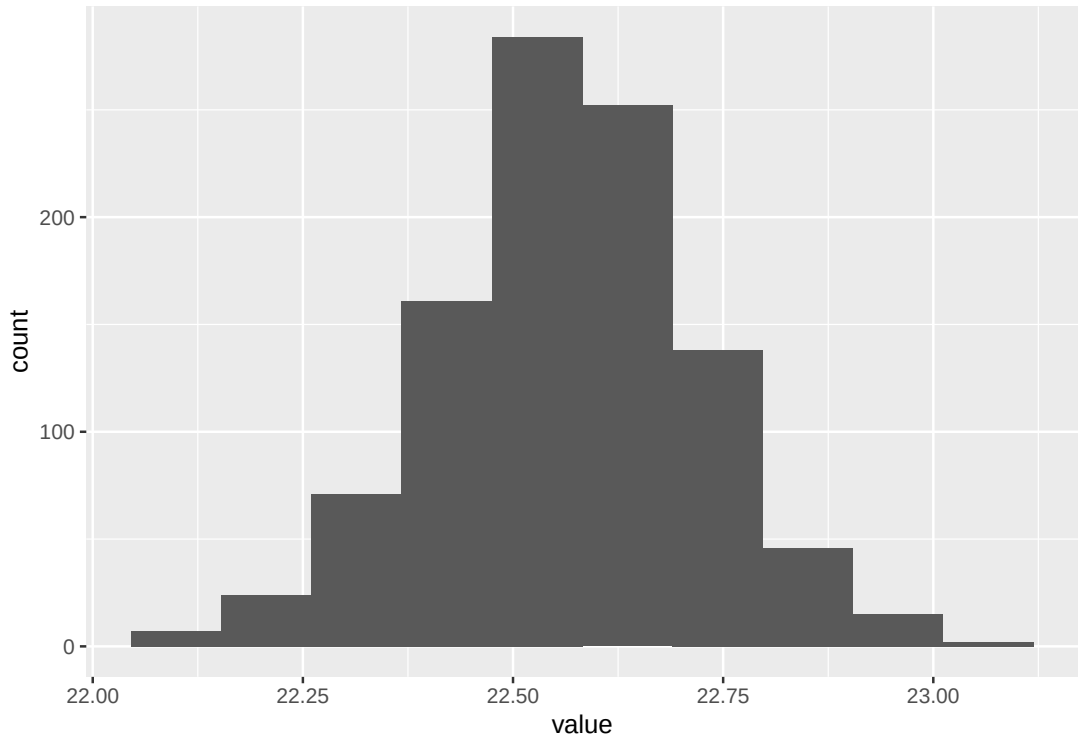
```
ggplot(d, aes(x = value)) + geom_histogram(bins = 10)
```



I don't know what you think of this. There are a few more extreme values than I would like, and it looks otherwise a bit left-skewed to me. But maybe I am worrying too much.

The males one works exactly the same way:

```
span %>% filter(sex=="M") -> males
rerun(1000, sample(males$handspan, replace = TRUE)) %>%
  map_dbl(~mean(.)) %>%
  enframe() -> d
ggplot(d, aes(x = value)) + geom_histogram(bins = 10)
```



There is a similar story here. I think these are good enough overall, and so I am happy with the two-sample t-test, but it is not as clear-cut as I was expecting.

Notes

1. If you did, all your problems would be over.