Assignment 8

Due Thursday November 15 at 11:59pm on Blackboard

As before, the questions without solutions are an assignment: you need to do these questions yourself and hand them in.

See `https://www.utsc.utoronto.ca/~butler/c32/quercus1.nb.html` for instructions on handing in assignments in Quercus. For SAS assignments, follow these steps:

- Follow the previous instructions to enable RTF output on SAS.
- When you have the output you want, click the button above the output with the W on it. This will download a file (in RTF format) that you can open.
- Copy and paste output from the RTF into a new Word document (which will contain what you hand in). Copy and paste your code from the Code tab, and write your answers/explanations.
- Finally, save your Word document *as PDF*. (If you see that you need to make changes, make them *in the Word document* and then save it again as PDF.) Hand in your PDF file.

1. Recall that we previously investigated the North Carolina births in SAS. Here we revisit that data set, which was at `http://www.utsc.utoronto.ca/~butler/c32/ncbirths.csv`.

   (a) Read the data set into SAS again.

   Solution: Since you've done it before, you can do it again. If you're on SAS Studio online, you'll need the first line (or you'll have to grapple with variable names containing spaces again):

```
options validvarname=v7;

filename myurl url "http://www.utsc.utoronto.ca/~butler/c32/ncbirths.csv";

proc import
  datafile=myurl
  dbms=csv
  out=ncbirths
  replace;
  getnames=yes;
```

   (b) Taking all the babies together, obtain a 95% confidence interval for the mean birth weight of all babies. Compare SAS's answer with R's from when you did this before. You will need to be careful to get the right name for this variable.

   Solution: In this variable, both the brackets and the space have been replaced by underscores, so there are *two* underscores between the words `Weight` and `pounds`, and one after:

proc ttest:

```
proc ttest;
  var Weight__pounds_;
```

N	Mean	Std Dev	Std Err	Minimum	Maximum
500	7.0688	1.5062	0.0674	1.1875	11.6250

Mean	95% CL Mean		Std Dev	95% CL Std Dev	
7.0688	6.9364	7.2011	1.5062	1.4183	1.6058

DF	t Value	Pr > |t|
499	104.94	<.0001

6.94 to 7.20 pounds. (Don't take the CI for the standard deviation by mistake!) This is the same as R's.

(c) Likewise taking all the babies together, test the null hypothesis that the mean birthweight is 7.3 pounds against the alternative that it is less. Obtain a P-value, and see whether it is the same as R's from before. (You did the actual test before, so I don't need the conclusion in context this time.)

Solution: Specifying a null hypothesis mean and a directional alternative in SAS, L for "less":

```
proc ttest h0=7.3 sides=L;
  var Weight__pounds_;
```

N	Mean	Std Dev	Std Err	Minimum	Maximum
500	7.0688	1.5062	0.0674	1.1875	11.6250

Mean	95% CL Mean		Std Dev	95% CL Std Dev	
7.0688	-Infty	7.1798	1.5062	1.4183	1.6058

DF	t Value	Pr < t
499	-3.43	0.0003

The alternative is that the mean is *lower* than 7.3. The P-value is 0.0003, which is the same (to the accuracy shown) as R's.

If you couldn't remember the `sides` thing, you can do three steps: get a two-sided test (the P-value comes out as 0.0006), check to see that you are on the correct side of the null (the sample mean is 7.07, which *is* less than 7.3), then halve the two-sided P-value (getting 0.0003). But you need all of this. If you have it, you're good.

2. Nenana, Alaska, is about 50 miles west of Fairbanks. Every spring, there is a contest in Nenana, called the Ice Classic. A wooden tripod is placed on the frozen river, and people try to guess the exact minute when the ice melts enough for the tripod to fall through the ice. The contest started in 1917 as an amusement for railway workers, and has taken place every year since. Now, hundreds of thousands of people enter their guesses on the Internet and the prize for the winner can be as much as $300,000. The contest has a website: `http://www.nenanaakiceclassic.com/`. On the website is a webcam that shows

the latest picture of the river, so that during the contest you can see how close you are to winning (or losing).

Because so much money is at stake, and because the exact same tripod is placed at the exact same spot on the ice every year, the data are consistent and accurate. The data are in `http://www.utsc.utoronto.ca/~butler/c32/nenana.txt`, separated by tabs (since the dates have spaces in them).

Yes, we saw these data before.

(a) Read in the data, and find the means for all the variables. You probably did this before; if you did, you can re-use your code.

Solution: This is the same as last time on this data set. Use the same code as you did before, once you got it working.

```
filename myurl url "http://www.utsc.utoronto.ca/~butler/c32/nenana.txt";

proc import
  datafile=myurl
  dbms=dlm
  out=mydata
  replace;
  delimiter='09'x;
  getnames=yes;

proc means;
```

```
                          The MEANS Procedure

Variable       N            Mean         Std Dev         Minimum         Maximum
--------------------------------------------------------------------------------
Year          87         1960.00      25.2586619         1917.00         2003.00
JulianDate    87     125.5443126       5.9317755     110.7045000     141.4872000
--------------------------------------------------------------------------------
```

(b) Obtain a 90% confidence interval for the mean Julian date.

Solution: The Julian dates are numbers, so it makes perfect sense to calculate the mean (and to find a confidence interval for the population mean that it is an estimate for).

A confidence interval at a non-standard confidence level requires you to note that SAS uses `alpha` as would be appropriate for a test, so a 90% CI needs $\alpha = 0.10$:

```
proc ttest alpha=0.10;
  var JulianDate;
```

```
           N        Mean     Std Dev     Std Err     Minimum     Maximum

          87       125.5      5.9318      0.6360       110.7       141.5
              Mean        90% CL Mean        Std Dev      90% CL Std Dev

             125.5       124.5    126.6      5.9318      5.2774   6.7906
                              DF    t Value    Pr > |t|

                              86     197.41      <.0001
```

124.5 to 126.6, same as R before. Ignore all the rest of the output.

(c) Test whether the mean Julian date is 130 or less than 130. (Remember the bit about the old-timer and his grey beard from before?)

Solution: Test that the mean Julian date is 130, against the alternative that it is less. This needs to be actually done, since a confidence interval is two-sided and this is one-sided (and therefore you can't just use the result of the previous part, at least not without thinking).

```
proc ttest h0=130 sides=L;
  var JulianDate;
```

N	Mean	Std Dev	Std Err	Minimum	Maximum
87	125.5	5.9318	0.6360	110.7	141.5

Mean	95% CL Mean		Std Dev	95% CL Std Dev	
125.5	-Infty	126.6	5.9318	5.1624	6.9727

DF	t Value	Pr < t
86	-7.01	<.0001

Either `side` or `sides` will work. Use whichever you like.

Consistent P-value with R.

You cannot do this just from the confidence interval output from the previous question, because we never specified a null mean there, and so in fact the P-value comes from a test of the population mean being *zero* (the default). Precisely, all the confidence interval tells us about the P-value for a test of $\mu = 130$ vs. $\mu < 130$ is that it is less than half of 0.10.

Not also that 130 is outside the confidence interval, so we'd expect a very small P-value (even though the test is one-sided, so strictly we'd expect a *two-sided* test to have a very small P-value, and therefore a one-sided test to have an *even smaller* P-value, given that the data is on the "right side" of 130).

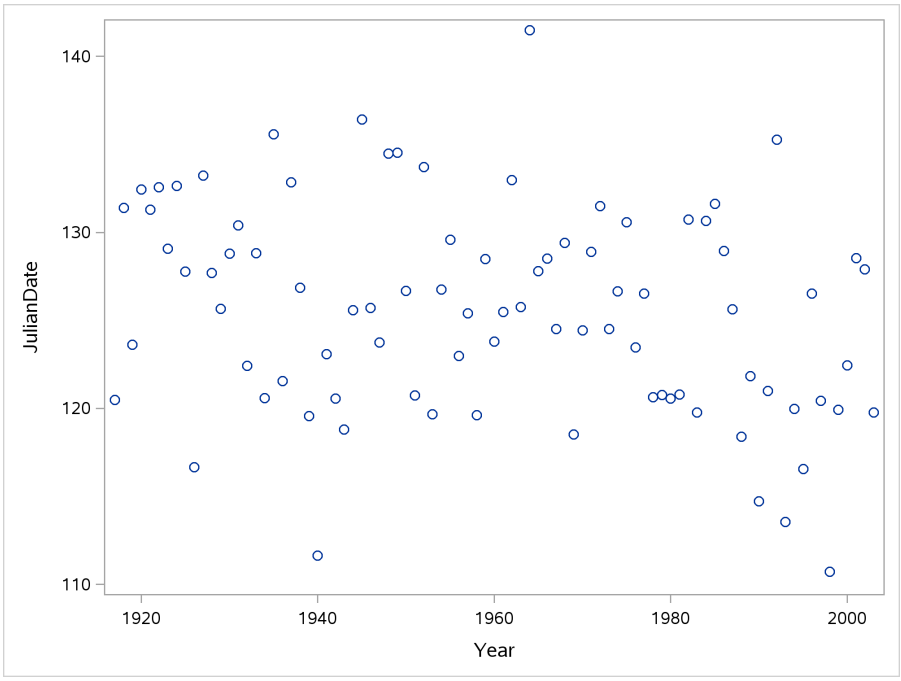
We have the same issues as before about the mean Julian date not being constant, so it doesn't make all that much sense to do inference for it. But that's by the way. Our aim here was to figure out how to do a test and confidence interval.

(d) If you feel up for some exploration, follow through this part, about making a plot of Julian date against year.

Solution: I want to see whether the typical date on which the tripod falls through the ice is changing over time. The obvious thing with two quantitative variables is a scatterplot:

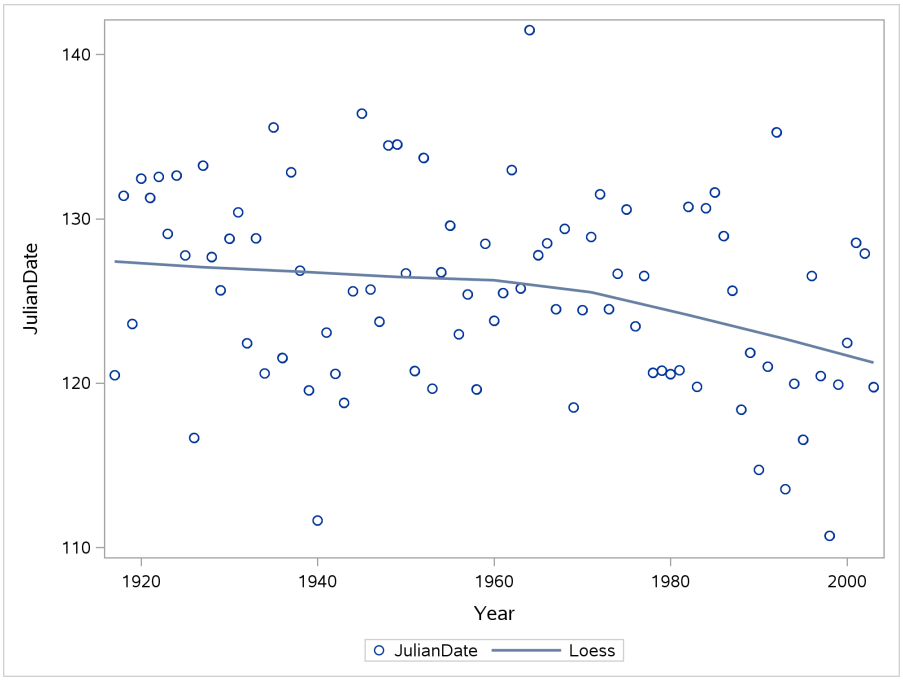
```
proc sgplot;
  scatter x=year y=juliandate;
```

SAS graphs (for me) seem to insist on a large amount of empty space around them:


140
130
120
110
JulianDate
1920
1940
1960
1980
2000
Year

That looks very random, but it is natural data with (as usual for natural data) a lot of variability. A smooth trend would be nice. Down near the end of the course I talk about `loess`, which is the SAS version of `geom_smooth`, so we'll borrow that to use now:

```
proc sgplot;
  scatter x=year y=juliandate;
  loess x=year y=juliandate;
```


140
130
120
110
JulianDate
1920
1940
1960
1980
2000
Year
JulianDate
Loess

There is a small but noticeable downward trend, more pronounced since about 1970. Environmental science people see a lot of graphs with this kind of shape. What this one means is that the ice is melting *earlier* on average every year, and that the trend has been faster since about 1970.

A way to test whether this is real or just chance is to fit a regression line, test the slope for significance and look at its size. We haven't done this in SAS yet, but to give you a preview:

```
proc reg;
  model juliandate=year;
```

```
                            The REG Procedure
                              Model: MODEL1
                        Dependent Variable: JulianDate

                    Number of Observations Read          87
                    Number of Observations Used          87
                              Analysis of Variance

                                       Sum of           Mean
Source                     DF         Squares         Square    F Value    Pr > F

Model                       1      238.06059      238.06059       7.26    0.0085
Error                      85     2787.93204       32.79920
Corrected Total            86     3025.99262
              Root MSE              5.72706    R-Square     0.0787
              Dependent Mean      125.54431    Adj R-Sq     0.0678
              Coeff Var             4.56178
                              Parameter Estimates

                           Parameter       Standard
    Variable      DF        Estimate          Error    t Value    Pr > |t|

    Intercept      1      254.64848       47.92518       5.31      <.0001
    Year           1       -0.06587        0.02445      -2.69      0.0085
```

The slope is significantly nonzero (P-value 0.0085) and negative, so that downward trend is real. As for its size, it's about 0.065 days per year, which doesn't seem like much, but if we scale that up to the 80 or so years the Ice Classic has been running:

```
-0.06587*80
```

```
## [1] -5.2696
```

Five and a bit days out of the hundred-and-something that the Julian dates typically are.

3. A professor collected some information about a random sample of 22 of his students. Specifically, the information collected was:
   - height (in inches)
   - weight (in pounds)
   - birthday (the number of the month it's in)

- the result of a coin toss by that student (1 is heads, 0 is tails)
- the gender of the student. These are recorded as 0 and 1, but unfortunately we don't know which one is male and which one is female.
- Two measurements of the student's pulse rate, taken a few minutes apart.

The data are in `http://www.utsc.utoronto.ca/~butler/c32/students.txt`. Use SAS for this question.

(a) Read in the data from the file. (The data values are all numbers.) Display the 22 rows of data.

Solution:

```
filename myurl url "http://www.utsc.utoronto.ca/~butler/c32/students.txt";
proc import
  datafile=myurl
  dbms=dlm
  out=students
  replace;
  delimiter=' ';
  getnames=yes;


proc print;
```

Obs	HEIGHT	WEIGHT	BIRTHDAY	COIN
1	65	135	4	1
2	63	119	9	1
3	72	175	11	0
4	60	106	9	1
5	65	135	8	0
6	72	170	10	1
7	64	180	8	1
8	71	205	10	1
9	75	195	6	0
10	71	185	8	1
11	71	182	6	0
12	65	108	8	0
13	73	150	4	1
14	67	128	6	0
15	74	175	6	1
16	66	160	9	1
17	65	143	9	0
18	72	190	11	0
19	64	180	2	1
20	61	195	5	0
21	72	220	7	1
22	69	235	7	1

Obs	SEX	PULSE1	PULSE2
1	0	73	73
2	0	62	70
3	1	72	73
4	0	73	73
5	0	75	74
6	1	69	69
7	0	68	73
8	1	72	73
9	1	68	67
10	1	68	73
11	1	70	74
12	0	73	75
13	1	70	70
14	0	74	80
15	1	68	69
16	0	74	77
17	0	70	72
18	1	68	69
19	1	68	68
20	1	90	95
21	1	75	78
22	1	74	72

22 rows of sensible-looking numbers. I think I am good, even though they are running off the bottom of the page.

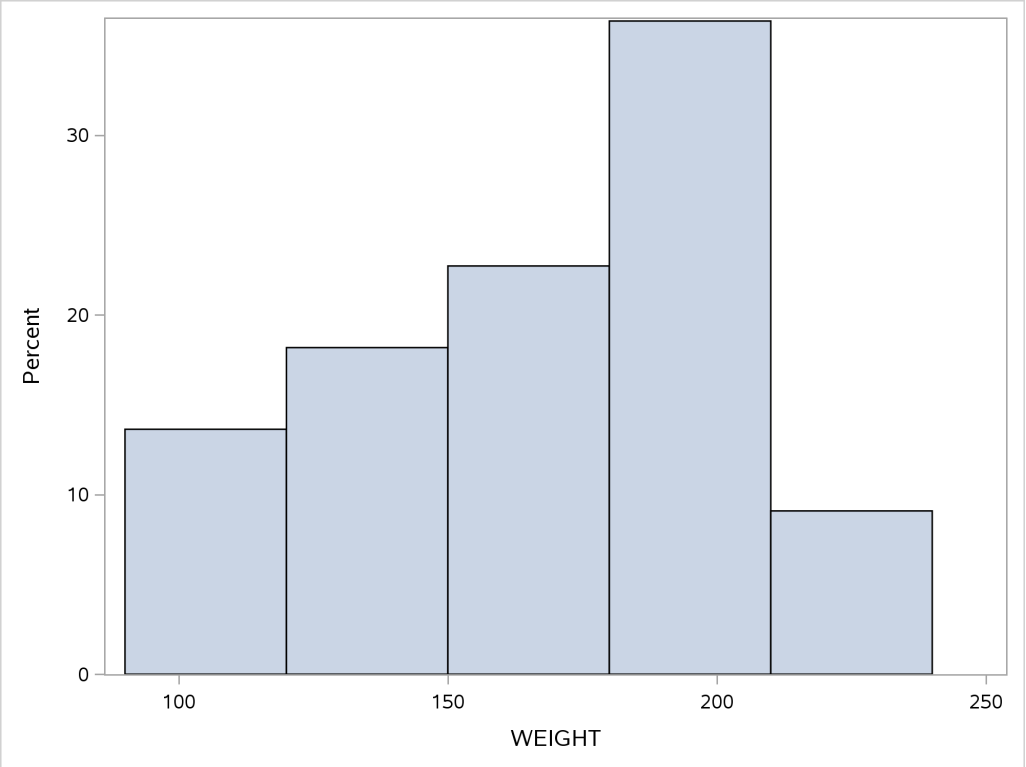
With only 22 rows, it's OK to display them all.

The variable names are in ALL CAPITALS, but SAS is not case-sensitive, so you can use things like `weight` the rest of the way. I'm going to do that, because I'm lazy.

(b) Obtain a histogram of the weights. How would you describe its shape? Comment briefly.

Solution: `proc sgplot`:

```
proc sgplot;
  histogram weight;
```



This is a little bit skewed to the left.

Sturges' rule would say six bins rather then five ($n = 22$ and $2^5 = 32$). SAS doesn't say what its default number of bins is.[1]

(c) Obtain a 99% confidence interval for the mean weight (of all students of whom these are a sample).

Solution: This requires a bit of thinking to get the confidence level right. In SAS, you specify `alpha` (that you would use for a test), which is one minus the confidence level, so:

```
proc ttest alpha=0.01;
```

```
var weight;
```

N	Mean	Std Dev	Std Err	Minimum	Maximum
22	166.9	35.1775	7.4999	106.0	235.0

Mean	99% CL Mean		Std Dev	99% CL Std Dev	
166.9	145.6	188.1	35.1775	25.0535	56.8746

DF	t Value	Pr > |t|
21	22.25	<.0001

The output confirms that we have indeed got a 99% interval for the mean, which goes from 145.6 to 188.1 pounds.

(d) Is it reasonable to believe that these students came from a population with mean 140 pounds, or is the mean bigger than that? Carry out a suitable test, and explain briefly what you conclude.

Solution: It's tempting to look at the CI and say "140 is not in there, so the mean is bigger than 140". That isn't quite right, though, because a confidence interval is a two-sided thing, and this test is one-sided (the alternative is "bigger than 140"). So we had better do the test.

The major concern is how to specify the null and alternative hypotheses. There are some choices about how to specify the null value 140: `mu0` or `location` or `h0`. I think they all work, though the documentation says the last one. The alternative is expressed by saying `side=` along with "2" (two-sided, the default), 'L" for lower or "U" for "upper". It is my habit to use uppercase for these, so that "lower" doesn't get confused with "1", whatever that means.[2] Here, the alternative is "bigger", so we want "U". `sides` instead of `side` also works:

```
proc ttest h0=140 sides=U;
  var weight;
```

N	Mean	Std Dev	Std Err	Minimum	Maximum
22	166.9	35.1775	7.4999	106.0	235.0

Mean	95% CL Mean		Std Dev	95% CL Std Dev	
166.9	154.0	Infty	35.1775	27.0639	50.2709

DF	t Value	Pr > t
21	3.58	0.0009

The P-value is 0.0009, a lot smaller than 0.05, so we reject the null mean of 140 and conclude that the mean weight of students in the population from which these students were drawn is indeed greater than 140 pounds.

(e) Explain briefly why this t-test should be reasonably trustworthy.

Solution: The (relevant) assumption behind the t-test is that the data come from a normal distribution. Our histogram suggested that the weight values are somewhat skewed (not badly

skewed, though). This is OK for two reasons (if you get one of them, that's OK): (i) the t-tests are robust to non-normality, so that even if the data are somewhat non-normal, as here, the P-value is still reasonable, or (ii) the sample is on the small side (22), so that the data can *look* a bit non-normal even if they were actually drawn from a normal distribution.

You might have noticed that these students are males and females mixed together, and so you would expect to see two different distributions of weights mixed together. This might be the case in fact, but our sample is too small to be sure about that.

(f) Can you determine from the data whether `SEX` 0 is males and 1 females, or whether it's the other way around? Obtain some numbers or graphs to help you decide, and explain briefly what you conclude.

Solution: The key here is to find something in the data set that is associated with gender (based on what you know or can guess). I don't mind so much what, as long as it is reasonable to believe that it would be associated with gender.

My best guess is that males tend to be "bigger" than females: that is, taller *and* heavier. So height and weight should both be bigger for males. So let's summarize height and weight by gender, and see what we get. This could be a numerical summary, such as comes out of `proc means`, or it could be a graphical one like a boxplot (boxplots are the best for comparing distributions).

Thus, one of these:

```
proc means;
  var height weight;
  class sex;
```

(you can also do two separate `proc means`)

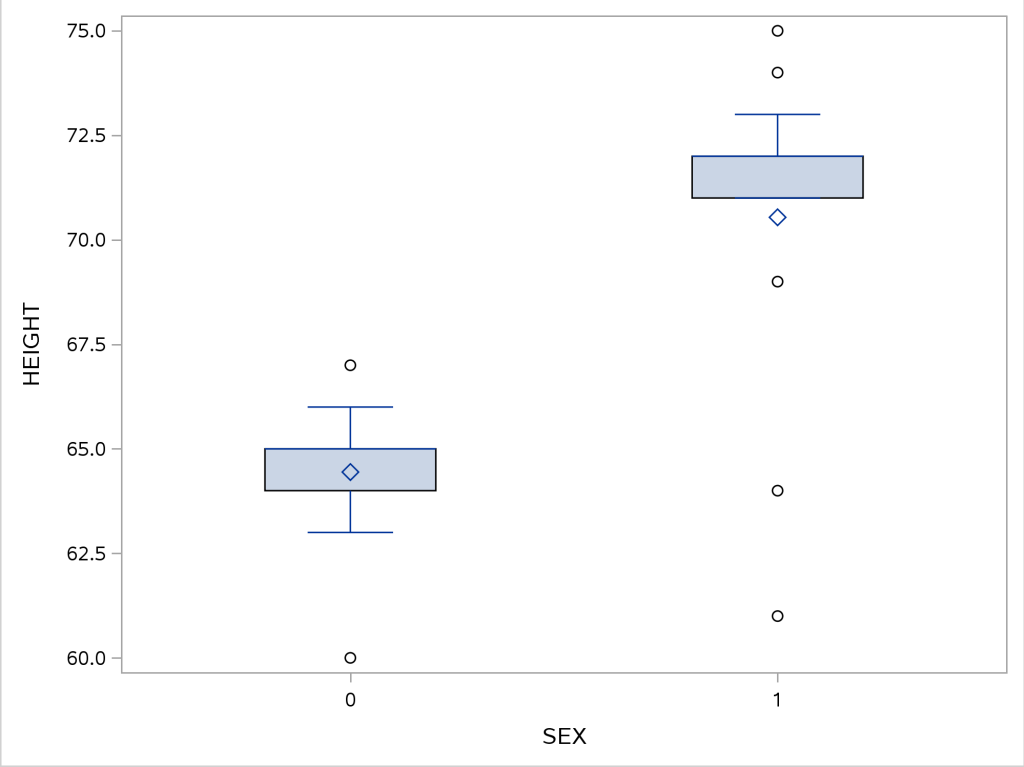
The MEANS Procedure

SEX	N Obs	Variable	N	Mean	Std Dev	Minimum
0	9	HEIGHT	9	64.4444444	2.0069324	60.0000000
		WEIGHT	9	134.8888889	23.9501798	106.0000000
1	13	HEIGHT	13	70.5384615	3.9075863	61.0000000
		WEIGHT	13	189.0000000	22.0340645	150.0000000

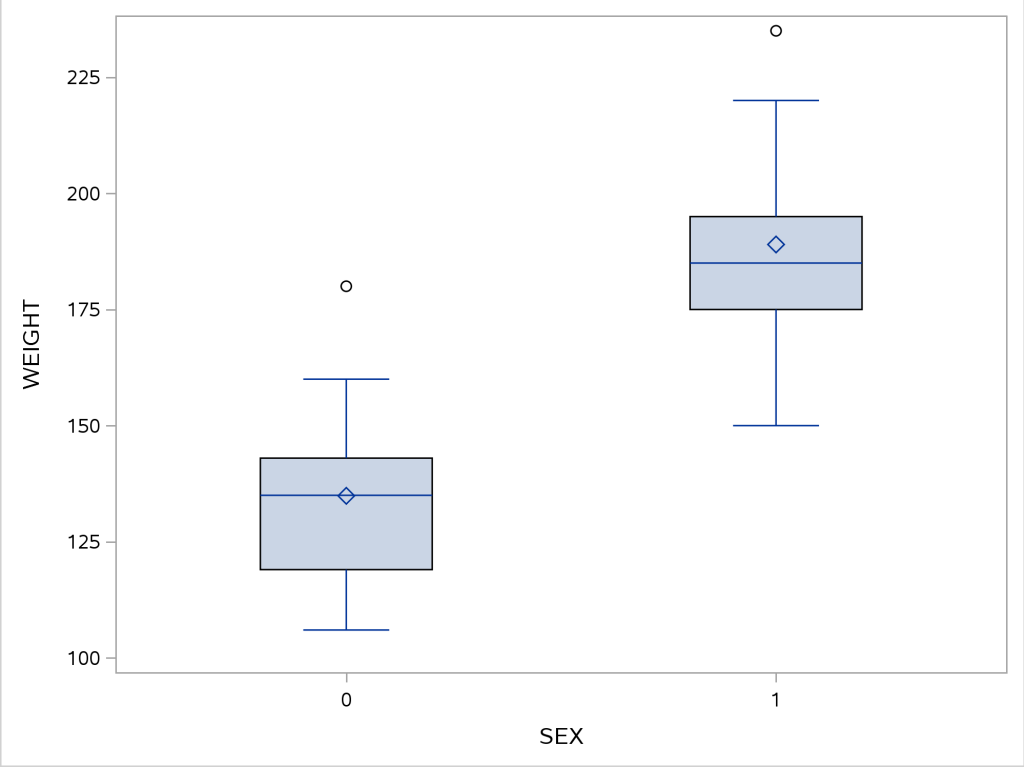
SEX	N Obs	Variable	Maximum
0	9	HEIGHT	67.0000000
		WEIGHT	180.0000000
1	13	HEIGHT	75.0000000
		WEIGHT	235.0000000

or (this one needs a plot each for height and weight if you do both):

```
proc sgplot;
  vbox height / category=sex;
```



```
proc sgplot;
  vbox weight / category=sex;
```



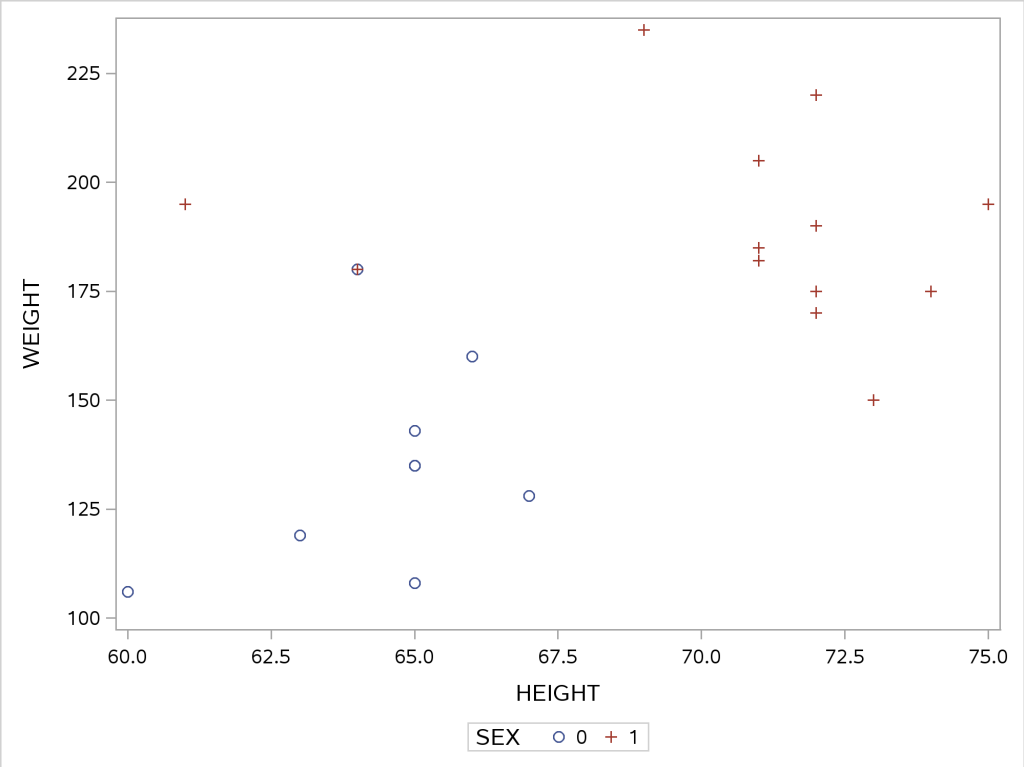
Whichever way you do it (and the minimum is to pick one variable, compare it somehow between genders, and then make a call), gender 1 is both taller on average and heavier on average than gender 0. So 1 is males and 0 is females.

I had an idea about getting both height and weight into one graph: plot height against weight as a *scatterplot*, and label the points differently according to whether they are gender 0 or gender 1. This is the idea: `proc sgplot` with `scatter` *and* `group=`:

```
proc sgplot;
  scatter x=height y=weight / group=sex;
```

This is, if you were keeping track, two quantitative variables and one categorical one.

There's no reason why the x and y-axes should be this way around; they could just as well be the other way around, since neither variable is a response to the other.



Gender 1, evidently the males, is up and to the right (taller *and* heavier), and gender 0 is down and to the left on the plot, evidently the females. There are a couple of males top left (short but heavier), but overall, the distinction is pretty clear.

4. Previously, we investigated whether children (aged 8–17) spend more time on electronic devices now than they did 10 years ago. Samples of 15 children aged 8–17 were taken in each of two years, 1999 and 2009, and the children (with their parents' help) were asked to keep a diary of the number of hours they spent using electronic devices on a certain day. The data are in the file `http://www.utsc.utoronto.ca/~butler/c32/pluggedin.txt`.

   (a) Why are these data two independent samples rather than matched pairs? (Think about the way the data were collected.)

Solution: Children that appeared in the 1999 sample would have been *too old* to be in the 2009 sample. So it must have been a different group of children in 2009 than it was in 1999. Thus, this is two independent samples (and a two-sample t-test is coming up).

For this to have been matched pairs, we would have had to have the *same* 15 children assessed both times, or at least we would have had to have some natural pairing-up. (Even having siblings of the 1999 children be the sample for 2009 would have been difficult to arrange.)

The fact that there were 15 children in each group was meant to confuse you a little: if they were matched pairs, there would *have to be* the same number of children both times, but with two independent samples, there might be the same number of children or there might not be.[3]

(b) Read the data into SAS, and list out the values. Make sure you have 30 values altogether, and two different years.

Solution: Look at the file to see that the data values are separated by spaces (the clue is in the file extension `.txt`), and then use the version of `proc import` that reads space-delimited files. Copy an old one. That's what I do:

```
filename myurl url "http://www.utsc.utoronto.ca/~butler/c32/pluggedin.txt";

proc import
  datafile=myurl
  dbms=dlm
  out=pluggedin
  replace;
  delimiter=' ';
  getnames=yes;


proc print;
```

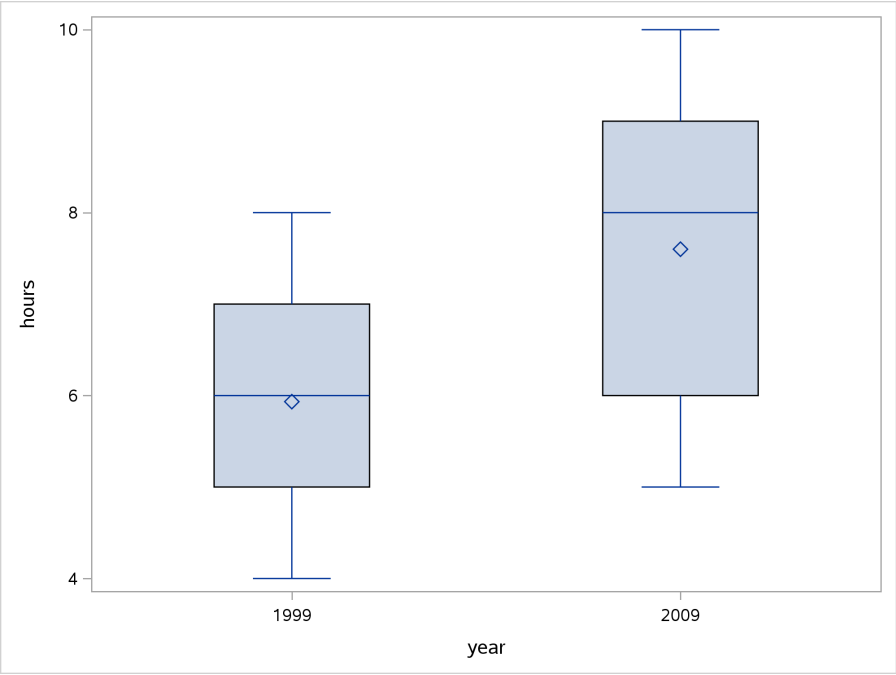
Obs	year	hours
1	1999	4
2	1999	5
3	1999	7
4	1999	7
5	1999	5
6	1999	7
7	1999	5
8	1999	6
9	1999	5
10	1999	6
11	1999	7
12	1999	8
13	1999	5
14	1999	6
15	1999	6
16	2009	5
17	2009	9
18	2009	5
19	2009	8
20	2009	7
21	2009	6
22	2009	7
23	2009	9
24	2009	7
25	2009	9
26	2009	6
27	2009	9
28	2009	10
29	2009	9
30	2009	8

$2 \times 15 = 30$ lines, years 1999 and 2009. Good. (Even though my data values ran off the bottom of the page.)

(c) Draw side-by-side boxplots of hours spent watching electronic devices for each year.

Solution: This is easy, unless you stop to think about it!

```
proc sgplot;
  vbox hours / category=year;
```



(d) What do your boxplots tell you? You're looking for a couple of things: how the means/medians compare, and whether you have any issues with skewness or outliers. Have a think about this before you look at my answer.

Solution: The mean and median for 2009 are quite a bit higher for 2009 compared to 1999, which suggests that the "average" number of hours really has increased.

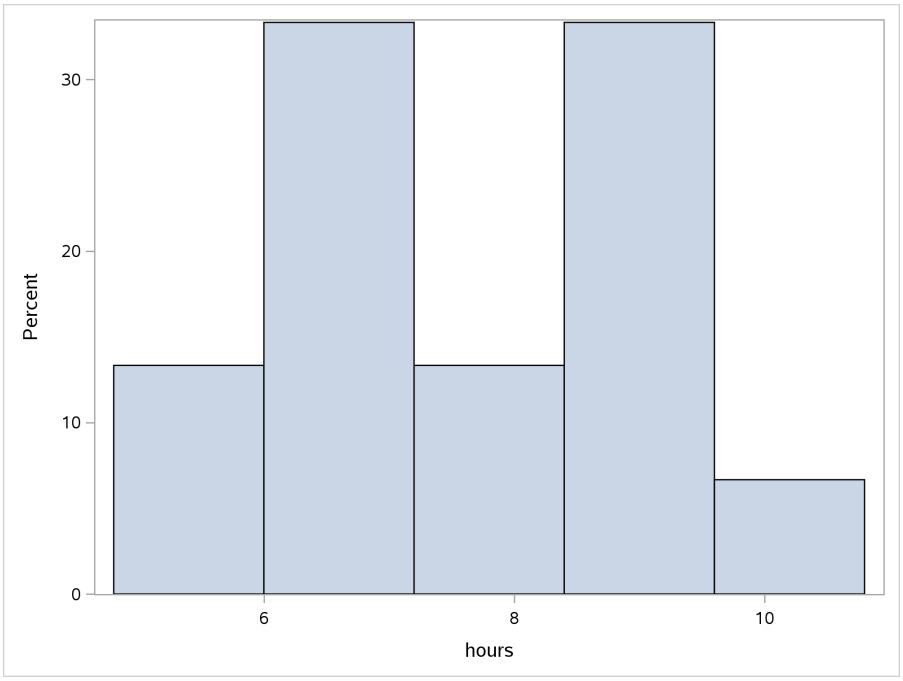
As for skewness and outliers, I really don't see any issues at all: the top and bottom whiskers are about the same length in both cases, and there are no outliers. You might be concerned about the median bar not being in the middle of the box for the 2009 data, but that doesn't really indicate a problem with skewness because that shows up in the *tails*: outliers if you have them, otherwise whisker length.[4]

This kind of thinking is to assess whether a t-test is the right thing to do, and for that we need data that are approximately normal within each group. By "approximately normal", it is generally enough to be able to say, as we have here, that the distributions are more or less symmetric and that we have no serious problems with outliers; in other words, the kind of things that boxplots will tell you.[5]

Extra: if you want to assess the 2009 data further, you can make a histogram via this trick:

```
proc sgplot;
  where year=2009;
  histogram hours;
```

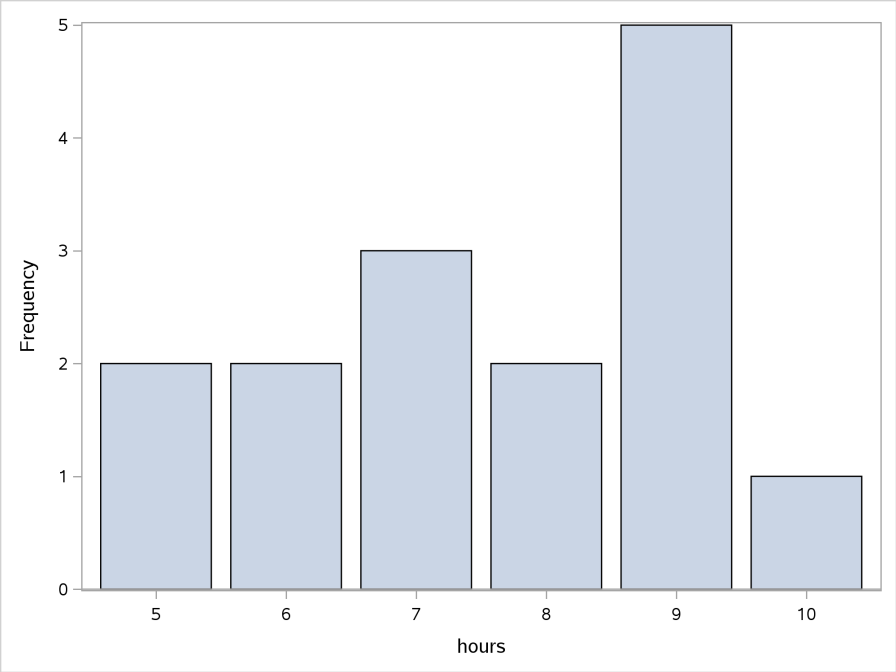
The `where` line says "only use data from year 2009 for this `proc`":


30
20
10
0
Percent
6
8
10
hours

The centre is indeed somewhere near 8, but look at the "hole" in the middle of the distribution (normally a histogram peaks in the middle). I suspect it's this that puts the median asymetrically in the box of the boxplot.

The numbers of hours are actually all integers, so you could reasonably think of the numbers of hours as being (ordered) categorical with a small number of categories:

```
proc sgplot;
  where year=2009;
  vbar hours;
```



This shows that the histogram is actually rather deceiving. The histogram bin between 8 and 10 has a lot of data because of all the 9s, but the histogram bin starting at 6 has a lot of data because that bin happens to contain *both* 6 and 7.

This came up before, and I might have to rewrite my advice about "one quantitative variable". Perhaps the story is "if the number of distinct values is not too much bigger than the number of bins you would use on a histogram, go with a bar chart".

Evidently the median here is 8, Q1 is 6 and Q3 is 9. Because of all the observations that are 9, the median is closer to Q3 than to Q1, and so the top "half" of the box is smaller than the bottom "half". Is this indicative of skewness? Well, maybe; it's enough to make the mean a bit smaller than the median. But it's not the kind of skewness that will cause trouble for the t-test; there is no *long* tail or lower-end outliers. One of those mushy ones that's not so clear.

(e) Carry out a test to determine if there is evidence that the mean number of hours spent per day watching electronic devices has *increased* since 1999. State your null and alternative hypotheses, and from your output, obtain a P-value and interpret it.

Solution: I like to start with the *alternative* hypothesis, since that is what we are trying to prove (which is usually the easiest thing to figure out). Here, that is that the mean in 1999 is *less* than the mean in 2009; in symbols that would be $H_a : \mu_{1999} < \mu_{2009}$. The null hypothesis is that the two means are the same, in symbols $H_0 : \mu_{1999} = \mu_{2009}$, or if this offends your logical sensibilities, $H_0 : \mu_{1999} \geq \mu_{2009}$. Either is good.

All right, getting some output. The only non-default thing here is the one-sidedness of the alternative. To figure out which `sides` you want, note that 1999 is before 2009 (alphabetically, actually, but numerically as well), so you have to express your alternative as how 1999 compares with 2009 *in that order*, as I did above. Or, you can try one of the `sides`, and if the answer makes no sense, try the other one. The intuition from the boxplots is that the P-value should be at least fairly small (since the story there is more in line with the alternative hypothesis than the null). I think the alternative is 1999 less than 2009, so we should have `sides=L`.[6] `side=` and `sides=` both work. The `var` and the `class` are the same as you would use on `proc means`:

```
proc ttest sides=L;
  var hours;
  class year;
```

```
year            N        Mean     Std Dev     Std Err     Minimum     Maximum

1999           15      5.9333      1.0998      0.2840      4.0000      8.0000
2009           15      7.6000      1.5946      0.4117      5.0000     10.0000
Diff (1-2)            -1.6667      1.3697      0.5002

   year          Method               Mean       95% CL Mean        Std Dev

   1999                               5.9333      5.3243   6.5424      1.0998
   2009                               7.6000      6.7169   8.4831      1.5946
   Diff (1-2)    Pooled              -1.6667    -Infty    -0.8158      1.3697
   Diff (1-2)    Satterthwaite       -1.6667    -Infty    -0.8121

                 year          Method             95% CL Std Dev

                 1999                             0.8052   1.7345
                 2009                             1.1675   2.5149
                 Diff (1-2)    Pooled             1.0870   1.8525
                 Diff (1-2)    Satterthwaite

           Method           Variances        DF    t Value    Pr < t

           Pooled           Equal            28      -3.33    0.0012
           Satterthwaite    Unequal      24.861      -3.33    0.0013

                            Equality of Variances

               Method      Num DF    Den DF    F Value    Pr > F

               Folded F        14        14       2.10    0.1768
```

With SAS, you have to choose whether to use the pooled or the Satterthwaite test. The choice is whether you believe the two samples come from populations with the same SD (pooled) or not (Satterthwaite). It often doesn't make much difference, as here. I think the interquartile range for the 2009 figures is a bit bigger, so (in the absence of outliers) I would expect its SD to be a bit bigger also.[7] Thus here I would choose the Satterthwaite test, though (as I said) it won't make much difference to your conclusion if you disagree with me (and say that the two IQRs are not different enough to be worth worrying about). In any case, the P-value is 0.0013 or 0.0012, smaller than 0.05, and so you *reject* the null hypothesis[8] and conclude that the 2009 mean is indeed larger, for *all* children, not just the ones that happened to be sampled.

The bottom test, the one labelled `Folded F`, is a test for whether the SDs (variances) in the two groups are equal (vs. the alternative that they are not). This null is not rejected, suggesting that we would be entitled to use the pooled test because the two group SDs are not significantly different. It is a mistake, though, to make a formal procedure out of this: to look at the Folded F test first and then decide which two-sample t-test to do. This is because doing it this way messes with the type I error probability. See for example `https://onlinelibrary.wiley.com/doi/pdf/10.1348/000711004849222`.[9]

Extra: if you don't like the skewness in the distribution of hours for 2009, we learned in R that the right test is Mood's median test. We'll get to that in SAS as well, but looking ahead:

```
proc npar1way median;
  var hours;
  class year;
```

```
                        The NPAR1WAY Procedure

      Median Scores (Number of Points Above Median) for Variable hours
                      Classified by Variable year

                       Sum of      Expected       Std Dev          Mean
 year        N         Scores      Under H0      Under H0         Score

 1999       15       4.428571          7.50      1.310717      0.295238
 2009       15      10.571429          7.50      1.310717      0.704762

                  Average scores were used for ties.
                       Median Two-Sample Test

                    Statistic              4.4286
                    Z                     -2.3433
                    One-Sided Pr <  Z      0.0096
                    Two-Sided Pr > |Z|     0.0191
                       Median One-Way Analysis

                    Chi-Square             5.4911
                    DF                          1
                    Pr > Chi-Square        0.0191
```

Later, I get to how this corresponds to what we do in R, but the important point for now is the one-sided P-value of 0.0096. This is not as small as for the t-tests, but it is still significant at $\alpha = 0.01$.

Extra extra bit for those of you that took STAB57 (the rest of you should probably skip ahead): you probably derived the pooled t-test, because the theory is the same as for the one-sample t-test. That, as a reminder, starts like this: suppose you have one sample $x_1, x_2, \ldots, x_n$ from a normal distribution with mean μ and variance σ^2. Then, *if you know* σ^2, $\bar{x} = (1/n)\sum_{i=1}^{n} x_i$ is exactly normal with mean μ and variance σ^2/n, or equivalently, this thing:

$$z = \frac{\bar{x} - \mu}{\sigma/\sqrt{n}}$$

is exactly standard normal. But if you don't know σ^2, you have to estimate it using the usual unbiased estimate $s^2 = (1/(n-1))\sum_{i=1}^{n}(x_i - \bar{x})^2$, and then this thing

$$t = \frac{\bar{x} - \mu}{s/\sqrt{n}}$$

has exactly a t-distribution with $n-1$ degrees of freedom. Strictly, $(n-1)s^2/\sigma^2$ has a chi-squared distribution with $n-1$ degrees of freedom, and a normal divided by a suitably scaled chi-squared has a t distribution with the same df. (What you have done is to introduce an extra source of variability in that s is probably not exactly equal to σ, so a t distribution has slightly larger variance[10] and longer tails than the normal.)

Now we think about two samples, the first of size n_1 from a normal distribution with mean μ_1 and variance σ^2, and the second of size n_2 from another independent normal distribution with mean μ_2 and variance σ^2. (The samples can be different sizes.) Thus, the means are possibly different but *the variances are the same.* The usual thing is to compare the two means, so you would estimate $\mu_1 - \mu_2$ via the unbiased estimate $\bar{x}_1 - \bar{x}_2$, the difference of the two sample means. If you knew σ^2, that would have a normal distribution (exactly) with mean $\mu_1 - \mu_2$ and variance $\sigma^2/n_1 + \sigma^2/n_2$. Since there is only one σ, you can also write the variance like this:

$$\sigma^2\left(\frac{1}{n_1} + \frac{1}{n_2}\right)$$

and thus this would have a standard normal distribution (exactly):

$$z = \frac{\bar{x}_1 - \bar{x}_2}{\sigma\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

In the usual case where you don't know σ^2, you have to estimate it. Since you have *one* population variance σ^2 to estimate (common to the two groups), you can estimate it by *one* sample variance

$$s_p^2 = \frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}$$

(a weighted average of the two sample variances, with the larger sample getting more weight). Thus you have this:

$$t = \frac{\bar{x}_1 - \bar{x}_2}{s_p\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

a normal thing with one σ^2 estimated by an s^2, which is therefore t with the right df, in this case $n_1 + n_2 - 2$.

When the two groups have *different* population SDs, the theory is a whole lot more complicated; in fact, there isn't even *any* exact answer.[11] What Satterthwaite and Welch did[12] was to say that you look at

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$$

In the second-last lecture of STAB22 this is introduced as "the two-sample t-test", since there they don't see the pooled test. This will have *approximately* a t-distribution. Welch and Satterthwaite independently looked at the thing on the bottom inside the square root:

$$W = \frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}$$

In the one-sample and pooled cases, the thing on the bottom can be scaled to have a chi-squared distribution, but not here, since there are two s^2 terms. The idea is to say that this sum is *approximately* a multiple of chi-squared with some unknown degrees of freedom: that is, you write it as $X = a\chi_b^2$. Since the mean and variance of a chi-squared distribution are both the degrees of freedom, we can work out the mean and variance of X: $E(X) = ab$, $var(X) = a^2 b$.

Now, we go back to the thing I called W above. What are its mean and variance? s_1^2 and s_2^2 are two independent sample variances, so we can get the mean and variance of W by finding the mean and variance of the two pieces and adding them together. So what are they? Well, $(n_i - 1)s_i^2/\sigma_i^2$ has a χ_{n-1}^2 distribution, so it has mean and variance $n_i - 1$. Thus

$$E((n_i - 1)s_i^2/\sigma_i^2) = n_i - 1$$

and

$$var((n_i - 1)s_i^2/\sigma_i^2) = n_i - 1$$

and thus

$$E\left(\frac{s_i^2}{n_i}\right) = \frac{\sigma_i^2}{n_i}$$

(the $n_i - 1$ terms cancelling; note that this also says that s_i^2 is an unbiased estimator of σ_i^2), and

$$var\left(\frac{s_i^2}{n_i}\right) = \frac{\sigma_i^4}{n_i^2(n_i - 1)}$$

(hence s_i^2 is also a consistent estimator of σ_i^2: look at the powers of n_i.)

Thus W has mean and variance

$$E(W) = \frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}$$

and

$$var(W) = \frac{\sigma_1^4}{n_1^2(n_1 - 1)} + \frac{\sigma_2^4}{n_2^2(n_2 - 1)}$$

To make W be a multiple of a chi-squared with some df, we match the mean and variance of W with the mean and variance of X (method of moments is what we're doing), to get

$$ab = \frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}$$

and

$$ab^2 = \frac{\sigma_1^4}{n_1^2(n_1-1)} + \frac{\sigma_2^4}{n_2^2(n_2-1)}$$

and "simply" solve for a and b. The business end of this is the degrees of freedom b, and if you compare the left-hand sides of the two equations above, you'll see that we can get b by dividing the two equations by each other and the a will cancel:

$$b = \frac{\frac{\sigma_1^4}{n_1^2(n_1-1)} + \frac{\sigma_2^4}{n_2^2(n_2-1)}}{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}$$

The final problem here is that the σ_i are not known, so we have to approximate *those* using the sample variances s_i. Then, as the very last stage, this df number is used as the df for the t-distribution.

It is likely that I have made some errors here, but that's the idea. There are, as you see, approximations upon approximations: we don't *know* that $s_1^2/n_1 + s_2^2/n_2$ has approximately a chi-squared distribution, and even if that's all right, we are estimating parameters by method of moments rather than maximum likelihood, and even then we are replacing σ_i^2 by s_i which is (we hope) somewhere close to it. If you think that statistics is nothing more than rigorous mathematics, you are likely to be in for a disappointment.

(f) Obtain a 99% confidence interval for the difference in means. You'll have to get some more output for this.

Solution: The reason that we have to do some more work is that a confidence interval is by its nature a *two-sided* thing, so we have to do a two-sided test to get it. (The confidence intervals in the previous part were *one-sided*: they started at minus-something and went all the way down to minus infinity.) So we have to take out the `sides` thing and put in something that will get us a 99% CI, namely `alpha=0.01`, since that's what SAS works with:

```
proc ttest alpha=0.01;
  var hours;
  class year;
```

year	N	Mean	Std Dev	Std Err	Minimum	Maximum
1999	15	5.9333	1.0998	0.2840	4.0000	8.0000
2009	15	7.6000	1.5946	0.4117	5.0000	10.0000
Diff (1-2)		-1.6667	1.3697	0.5002		

year	Method	Mean	99% CL Mean		Std Dev
1999		5.9333	5.0880	6.7786	1.0998
2009		7.6000	6.3743	8.8257	1.5946
Diff (1-2)	Pooled	-1.6667	-3.0487	-0.2846	1.3697
Diff (1-2)	Satterthwaite	-1.6667	-3.0615	-0.2719	

year	Method	99% CL Std Dev	
1999		0.7353	2.0386
2009		1.0662	2.9558
Diff (1-2)	Pooled	1.0150	2.0532
Diff (1-2)	Satterthwaite		

```
Method           Variances        DF    t Value    Pr > |t|

Pooled           Equal            28      -3.33      0.0024
Satterthwaite    Unequal      24.861      -3.33      0.0027

                   Equality of Variances

     Method      Num DF    Den DF    F Value    Pr > F

     Folded F        14        14       2.10    0.1768
```

The confidence interval goes from -3.06 to -0.27 (hours). Note that SAS did indeed label it as a 99% interval, so that we have the right thing. I pulled out the Satterthwaite interval, since that's what I thought was best; if you think the pooled test was OK, then use the pooled interval here. (As for the tests, they are not very different).

(g) What does your confidence interval tell you, in the context of the data?

Solution: It says that the time spent by a child per day in front of an electronic device has *increased* by between 0.3 and 3.1 hours between 1999 and 2009. (The negative numbers mean that the 1999 values were less on average than the 2009 ones.)

(h) Does your confidence interval contain zero? Does that surprise you? Why, or why not?

Solution: No, it doesn't: all the values in the interval are negative. This does not surprise me, because we said from the test that the mean number of hours had significantly increased, and we would therefore expect a CI all on one side of zero (negative, because of the way the numbers were).

This is strictly speaking not quite right (though I didn't need you to observe this), because the correspondence is between a *two-sided* test and a confidence interval (or between the one-sided test and the one-sided confidence interval, which didn't contain zero either). We got away with it here, though, because the P-value was so small that even the two-sided P-value was still safely less than 0.01 (to go with the 99% CI).

The confidence interval was quite wide. This is partly because it was a 99% one; a 90% CI would be narrower. This says that we know that there *was* an increase in the mean number of hours spent watching electronic devices, but that we don't have a very precise idea of how big that increase was. This is unfortunately all too common.

5. How long does it take students to get to school? A survey was done of British secondary school students, and a similar survey of Ontario high-school students, with 40 students in each (which, you may assume, are a random sample of their respective populations). In both surveys, the "typical" time taken to get to school was recorded. The question of interest is whether there is a difference in the time students take to get to school in Ontario and the UK. The data are in `http://www.utsc.utoronto.ca/~butler/c32/to-school.csv`.

   (a) Read the data into SAS. There should be two columns, `traveltime` and `location`. Obtain the mean travel time for each location. How many travel times do you have at each location?

Solution: The usual business for reading in a `.csv`:

```
filename myurl url 'http://www.utsc.utoronto.ca/~butler/c32/to-school.csv';

proc import
  datafile=myurl
  dbms=csv
  out=mydata
  replace;
  getnames=yes;
```

Normally you'd follow this with `proc print`, which you probably should for yourself, but there are 80 lines of data, a lot to hand in, so I asked you to summarize things, thus:

```
proc means;
  var traveltime;
  class location;
```

The MEANS Procedure

Analysis Variable : traveltime

location	N Obs	N	Mean	Std Dev	Minimum	Maximum
Ontario	40	40	17.0000000	9.6609178	2.0000000	47.0000000
UK	40	40	20.6500000	13.1276768	3.0000000	60.0000000

There are 40 travel times in each location; the mean for the Ontario students is 17 minutes, and the mean for the British students is 20.65 minutes. *You need to say this.*

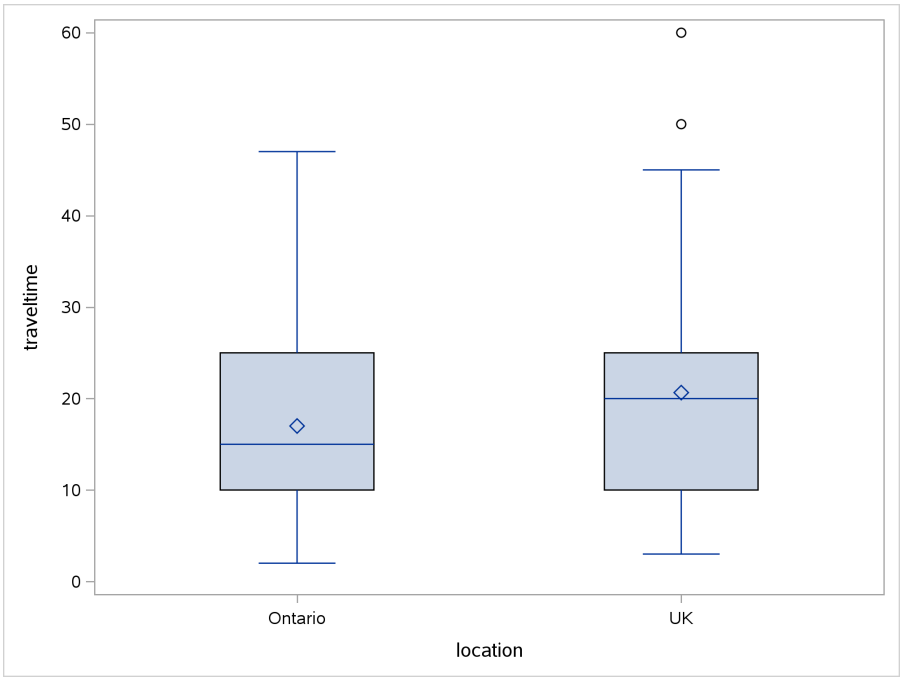
If something went astray with the reading in, it will probably show up here, and that would alert you to check what you did.

The first time I did this, I had the British times first in the data file, and the locations were labelled `UK` and `On` (it seems to take the maximum length from the first one it finds[13]). But I didn't want you to be dealing with that, so I switched things around in the data file.

(b) Make a suitable plot of travel times for each location. Describe the shapes of the distributions. Do they have similar spreads?

Solution: There is one quantitative variable here, travel time, and one categorical one, location, so the obvious thing is a side-by-side boxplot:

```
proc sgplot;
  vbox traveltime / category=location;
```


60
50
40
30
20
10
0
traveltime
Ontario
UK
location

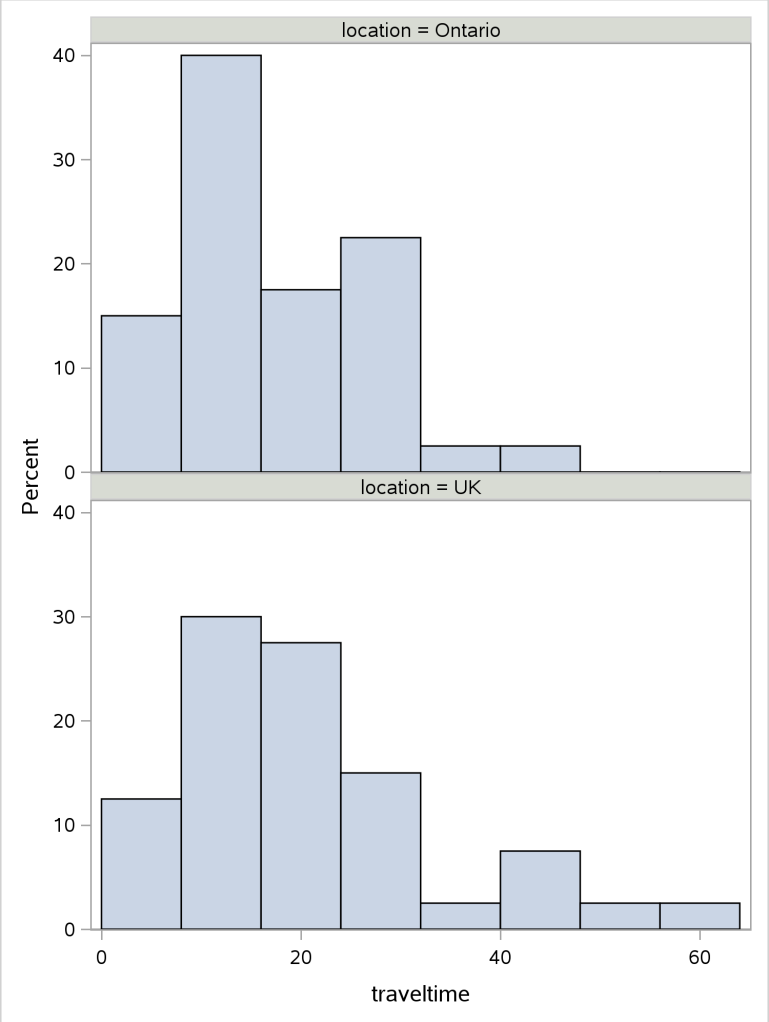
I think both of those distributions are skewed to the right: look at the longer upper whiskers, and the outliers on the UK distribution. However, the spreads, as measured by the heights of the boxes, look very similar to me.

Two points for a suitable graph, and one each for appropriate comment about shape and and about spreads.

Another possibility for a graph would be a paneled histogram, but this requires extra cajoling to come out above and below, like this:

```
proc sgpanel;
  panelby location / columns=1;
  histogram traveltime;
```

The `columns=1` arranges all the plots in one long vertical column, which is what we want:



The right-skewed shape is obvious enough (though less obvious than on the boxplots), but how are you going to assess spread well enough to conclude that it's about the same? Maybe looking at the bin from 40 to about 46 that has 3 observations in it for the UK data, and only one for the Ontario data. Or you can go back to the SDs in the output from `proc means`, where the SD for the UK measurements is a bit higher, but is that because of the outliers?

(c) Is there any evidence of a difference in mean travel time between the two locations? Run a suitable t-test. What do you conclude?

Solution: This code, the test being two-sided:

```
proc ttest;
  var traveltime;
  class location;
```

Note that the `var` and `class` are exactly as for the `proc means`.

location	N	Mean	Std Dev	Std Err	Minimum	Maximum
Ontario	40	17.0000	9.6609	1.5275	2.0000	47.0000
UK	40	20.6500	13.1277	2.0757	3.0000	60.0000
Diff (1-2)		-3.6500	11.5254	2.5772		

location	Method	Mean	95% CL Mean		Std Dev
Ontario		17.0000	13.9103	20.0897	9.6609
UK		20.6500	16.4516	24.8484	13.1277
Diff (1-2)	Pooled	-3.6500	-8.7807	1.4807	11.5254
Diff (1-2)	Satterthwaite	-3.6500	-8.7879	1.4879	

location	Method	95% CL Std Dev	
Ontario		7.9138	12.4050
UK		10.7537	16.8564
Diff (1-2)	Pooled	9.9662	13.6676
Diff (1-2)	Satterthwaite		

Method	Variances	DF	t Value	Pr > |t|
Pooled	Equal	78	-1.42	0.1607
Satterthwaite	Unequal	71.663	-1.42	0.1610

Equality of Variances

Method	Num DF	Den DF	F Value	Pr > F
Folded F	39	39	1.85	0.0590

You should make a choice between the pooled and Satterthwaite tests, based on whether you thought the spreads were similar or noticeably different. My choice was that the spreads (IQRs) were almost the same, so I would go with the pooled P-value 0.1607. If you thought the spreads were different (eg. by considering the SDs), you need to use the Satterthwaite P-value 0.1610. I don't mind which way you go, but you need to be *consistent* with what you said before. That's the key. In either case, you fail to reject the null, and therefore conclude that there is no evidence of any difference between the mean travel times in the two places.

If you look at the two confidence intervals for the difference in mean, you'll see that they both include 0, which is another way of seeing that you are likely not going to be rejecting a hypothesis that the difference in population means is zero.

I guess another way of choosing between the two tests is to note that the P-values are almost identical, so that it doesn't matter which one you choose. This is, perhaps, surprising, given that equality of variances (the bottom test) is almost rejected, perhaps because the SD of the UK measurements is larger, inflated, perhaps, by the upper outliers in that distribution.

(d) Do you have any concerns about the t-test you just did? Explain briefly why or why not.

Solution: The critical assumption here is of approximately normal distributions *within* each group. Equal spreads does not matter. Go back to the boxplots you drew earlier, and make a call: does it look as if *both* groups have approximately normal distributions? I would say not, because I think they're both skewed to the right. That's a good answer.

Having said that, the skewness probably matters (somewhat) less than you think, because you have the Central Limit Theorem ($n = 40$ in each group) working in your favour. Is the sample size large enough to overcome the skewness that you see? That's a hard question to answer, without resorting to something like bootstrapping, but it's reasonable to answer that

the skewness may not be causing as much of a problem as it appears because of the largish sample sizes.

Extra: there's a lot of hand-waving involved in all of this, and it may be difficult to get a definitive answer about whether we should do the two-sample t-test or something else (eg. Mood's median test, coming up later), but I want you to get at the issues in your answer: at least, that both distributions are skewed right, so that approximate normality fails, but possibly also that we have $n = 40$ in both groups so the Central Limit Theorem is in our favour (and the normality doesn't matter as much).

One other thing in among the infinity of issues here is that it helps if both groups are *skewed in the same direction*, as here, because whichever t-statistic you calculate, you subtract the sample means, and this allows the skewness to "cancel out", or at least get reduced. The idea is that you might get an unusually large travel time in either group, and those will *both* inflate the mean in that group upwards, so that when you subtract the means this effect is dampened down. (Compare if one group were skewed right and the other skewed *left*; then you'd have the sample means being pulled potentially opposite ways by unusual values and the difference could, if you were unlucky, be pulled a long way away from zero.)

One way to do less hand-waving, as I hinted above, is via the "bootstrap". The idea behind this is that you treat the observed data as populations, and then you sample from them *with* replacement.[14] You calculate the test statistic each time, and then this gives an idea of what the sampling distribution looks like. Since we're doing a test which is based on a null hypothesis, we can arrange things beforehand so that the means actually *are* equal, and then see how often we mistakenly reject. Here, though, I'm interested in the shape of the sampling distribution: if it's normal, using a t-test will have no problems.

As you might expect, R is the tool for this. Let's start with a small one where the distributions are skewed opposite ways, so we expect it to go wrong. But first:

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.6
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

```
## -- Conflicts ------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(broom)
library(rsample)
```

```
##
## Attaching package:  'rsample'
```

```
## The following object is masked from 'package:tidyr':
##
##     fill
```

and then some data to play with:

```
d=tribble(~value, ~gp,
          -3, "X",
          -2, "X",
          -1, "X",
          6, "X",
          -6, "Y",
          1, "Y",
          2, "Y",
          3, "Y")
d
```

```
## # A tibble: 8 x 2
##   value gp
##   <dbl> <chr>
## 1    -3 X
## 2    -2 X
## 3    -1 X
## 4     6 X
## 5    -6 Y
## 6     1 Y
## 7     2 Y
## 8     3 Y
```

The values in group X are skewed to the left, and the values in group Y are skewed to the right. Both groups have mean zero. So we know the means are actually equal, but the question is what kind of test statistic values we might get.

We know how to compare these actual values:

```
t.test(value~gp,data=d)
```

```
##
##  Welch Two Sample t-test
##
## data:  value by gp
## t = 0, df = 6, p-value = 1
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -7.063626  7.063626
## sample estimates:
## mean in group X mean in group Y
##               0               0
```

Package `broom` produces output that is itself a data frame, so that it can be handled tidily. That's why I loaded it above. For example, `tidy` summarizes the whole thing in one line:

```
tidy(t.test(value~gp,data=d))
```

```
## # A tibble: 1 x 10
##   estimate estimate1 estimate2 statistic p.value parameter conf.low
##      <dbl>     <dbl>     <dbl>     <dbl>   <dbl>     <dbl>    <dbl>
## 1        0         0         0         0       1        6.    -7.06
## # ... with 3 more variables: conf.high <dbl>, method <chr>,
## #   alternative <chr>
```

Here, the P-value is exactly 1 since both sample means are exactly the same. The column `statistic` is the t-statistic, and the columns labelled `estimate` are the sample means of the two groups (the numbered ones) and the difference of those means (the un-numbered one).

All right, down to bootstrapping. This is done with the `rsample` package (loaded above, that you'll probably have to install first). The function `bootstraps` does the actual bootstrap sampling. Let's draw a bunch of bootstrap samples now and then use them later. I'm setting my random number seed so that these come out the same when I run them again:

```
set.seed(457299)
boot1 = bootstraps(d, times=1000, strata="gp")
boot1
```

```
## # Bootstrap sampling using stratification
## # A tibble: 1,000 x 2
##    splits       id
##    <list>       <chr>
##  1 <S3: rsplit> Bootstrap0001
##  2 <S3: rsplit> Bootstrap0002
##  3 <S3: rsplit> Bootstrap0003
##  4 <S3: rsplit> Bootstrap0004
##  5 <S3: rsplit> Bootstrap0005
##  6 <S3: rsplit> Bootstrap0006
##  7 <S3: rsplit> Bootstrap0007
##  8 <S3: rsplit> Bootstrap0008
##  9 <S3: rsplit> Bootstrap0009
## 10 <S3: rsplit> Bootstrap0010
## # ... with 990 more rows
```

`boot1` has a column `id` that says which bootstrap sample we have information for, and a list-column `splits` that contains the information about each bootstrap sample. With a bit of struggle we can take a look at these:[15]

```
boot1 %>% mutate(sample=map(splits, ~analysis(.))) %>% unnest(sample)
```

```
## # A tibble: 8,000 x 3
##    id            value gp
##    <chr>         <dbl> <chr>
##  1 Bootstrap0001    -3 X
##  2 Bootstrap0001    -3 X
##  3 Bootstrap0001    -1 X
##  4 Bootstrap0001     6 X
##  5 Bootstrap0001     1 Y
##  6 Bootstrap0001     1 Y
##  7 Bootstrap0001     3 Y
##  8 Bootstrap0001     3 Y
##  9 Bootstrap0002    -3 X
## 10 Bootstrap0002    -3 X
## # ... with 7,990 more rows
```

The first eight rows are the first bootstrap sample. You can see the repeated values; you can also see that there are exactly four values for group `X` and four for group `Y`. This is what the `strata="gp"` thing did in the generation of the bootstrap samples; if I had omitted that, it would have sampled with replacement from the *whole* data frame, and I might have come out

with say five `X` and three `Y`, which I didn't want. (If you are doing this for one sample, you don't need to worry about this, since the bootstrapped samples will all be the same size as the original sample. Here, though, all that's guaranteed is that our bootstrapped samples are of size 8 altogether, without any guarantee that there will be four `X` and four `Y`.)

Now, for each of those bootstrap samples, we want to run the two-sample t-test on it, and get hold of the test statistic. The way I phrased it, you ought to be thinking `map` for the for-each, and `.` for "it", and so it proves. In a moment.

The thing that we have to do "for each" turns out to be rather complicated (usually, in this work), so we do better to write a little function to do the work for us. I'll take you through my function in a moment:

```
boot_t=function(split) {
    d=analysis(split)
    d.1=t.test(value~gp, data=d)
    d.2=tidy(d.1)
    d.2$estimate
}
```

The input to this function is one of the things from the `splits` column of the bootstrap data frame. This contains several things, but the result of running `analysis` on it is the actual bootstrapped sample. (That was the reason for the `analysis` above when I displayed the bootstrapped samples.) Then we run the t-test, then we get the tidy version of the output, and then we return the thing called `estimate` from it. (I could shorten the code for those last things, but I wanted you to be clear about what I was doing.) `estimate` is the difference between the two sample means.

Now we have to create a new column called `est` in `boot1` that is the result of running that function on each of our bootstrap samples:

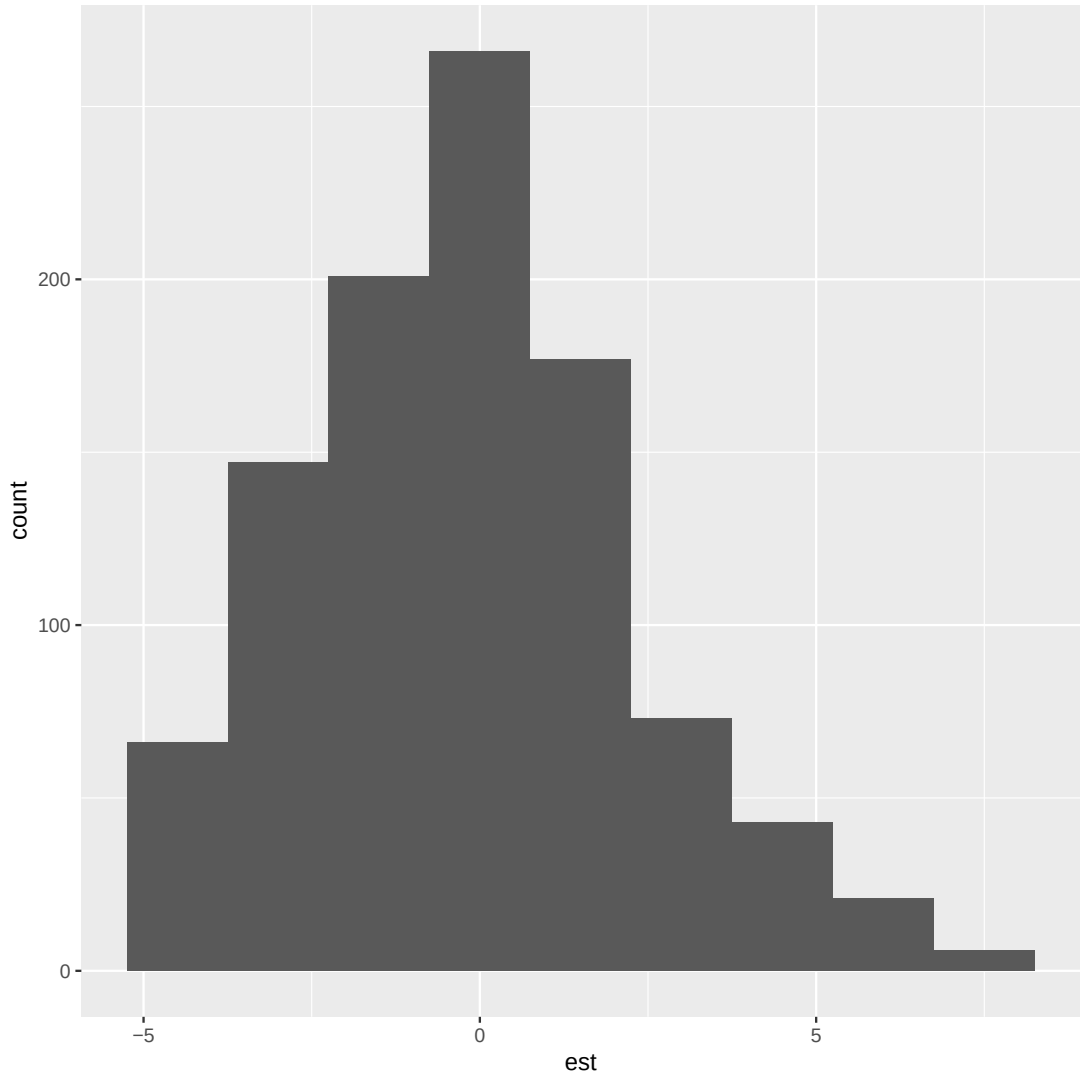
```
boot1 %>% mutate(est=map_dbl(splits, ~boot_t(.))) -> boot1_res
boot1_res
```

```
## # Bootstrap sampling using stratification
## # A tibble: 1,000 x 3
##    splits       id              est
##  * <list>       <chr>         <dbl>
##  1 <S3: rsplit> Bootstrap0001 -2.25
##  2 <S3: rsplit> Bootstrap0002 -2.25
##  3 <S3: rsplit> Bootstrap0003  2.25
##  4 <S3: rsplit> Bootstrap0004  0.5
##  5 <S3: rsplit> Bootstrap0005  0.75
##  6 <S3: rsplit> Bootstrap0006 -0.25
##  7 <S3: rsplit> Bootstrap0007  4
##  8 <S3: rsplit> Bootstrap0008 -0.25
##  9 <S3: rsplit> Bootstrap0009 -0.25
## 10 <S3: rsplit> Bootstrap0010  6
## # ... with 990 more rows
```

I have to say that I wasn't expecting this to work first time, but it did. (I had lots of practice playing with this last night, though.)

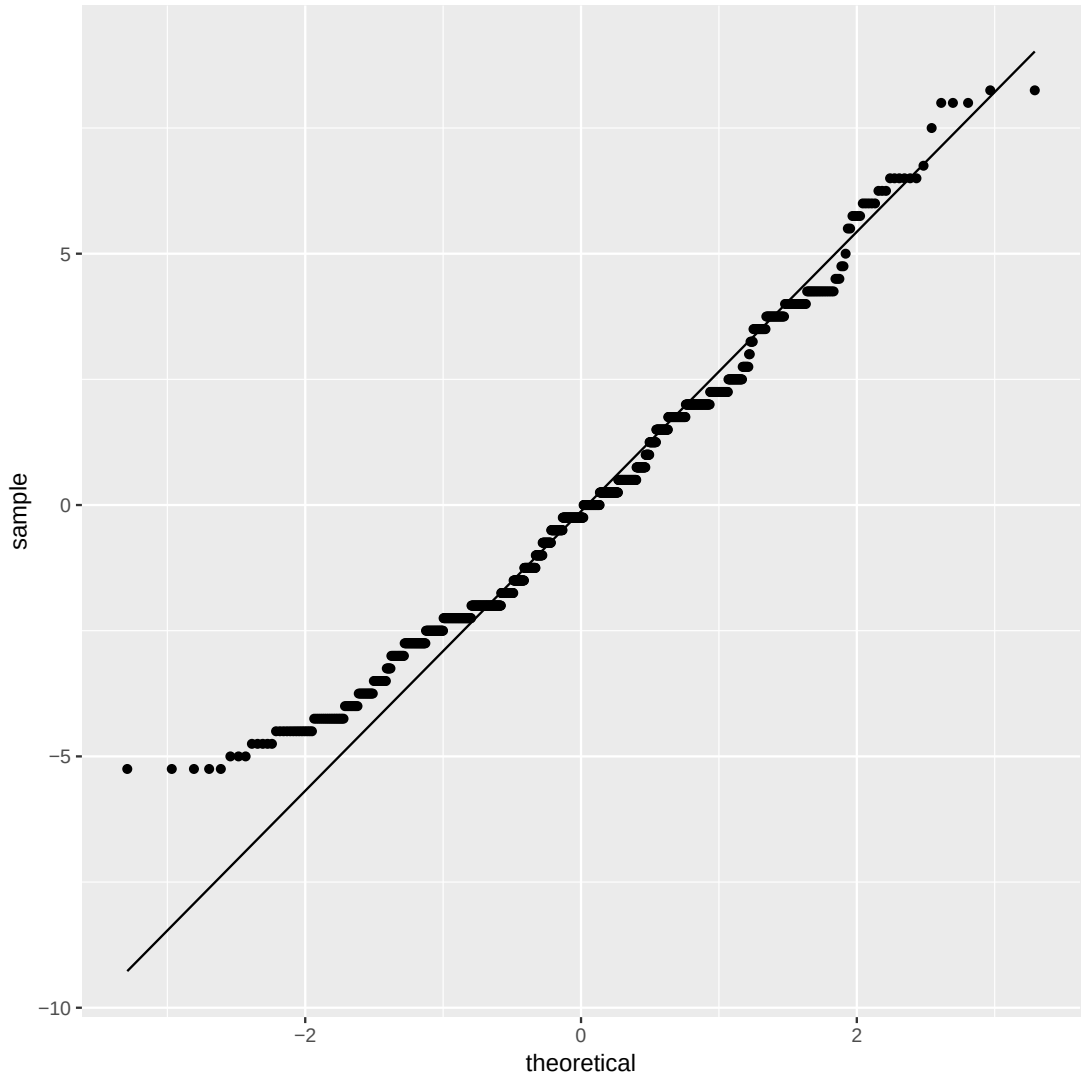
Now, how does that distribution of values in `est` look?

```
ggplot(boot1_res, aes(x=est))+geom_histogram(bins=10)
```



That looks right-skewed, which is about what we'd expect because we could get a very big value from `X` or a very small value from `Y`, which would make the mean of `X` minus the mean of `Y` large.

```
ggplot(boot1_res, aes(sample=est))+stat_qq()+stat_qq_line()
```



The reason for the right skewness is actually not that the right tail is too long, but that the left tail is too *short*: the values don't go negative enough. The horizontal patches of dots are discreteness; with only four values in each sample, there are not so many possible means for a bootstrapped sample. If the horizontal patches more or less followed the line, we'd be happy, but these don't.

Now for our actual data. This goes much the same way, reading in the data first:

```
my_url="http://www.utsc.utoronto.ca/~butler/c32/to-school.csv"
to_school=read_csv(my_url)
```

```
## Parsed with column specification:
## cols(
##  traveltime = col_integer(),
##  location = col_character()
## )
```

and then

```
boot2=bootstraps(to_school, times=1000, strata="location")
boot2
```

```
## # Bootstrap sampling using stratification
## # A tibble: 1,000 x 2
##    splits       id
##    <list>       <chr>
##  1 <S3: rsplit> Bootstrap0001
##  2 <S3: rsplit> Bootstrap0002
##  3 <S3: rsplit> Bootstrap0003
##  4 <S3: rsplit> Bootstrap0004
##  5 <S3: rsplit> Bootstrap0005
##  6 <S3: rsplit> Bootstrap0006
##  7 <S3: rsplit> Bootstrap0007
##  8 <S3: rsplit> Bootstrap0008
##  9 <S3: rsplit> Bootstrap0009
## 10 <S3: rsplit> Bootstrap0010
## # ... with 990 more rows
```

Now, we have to modify our function slightly since the columns now have different names:

```
boot_t=function(split) {
    d=analysis(split)
    d.1=t.test(traveltime~location, data=d)
    d.2=tidy(d.1)
    d.2$estimate
}
```

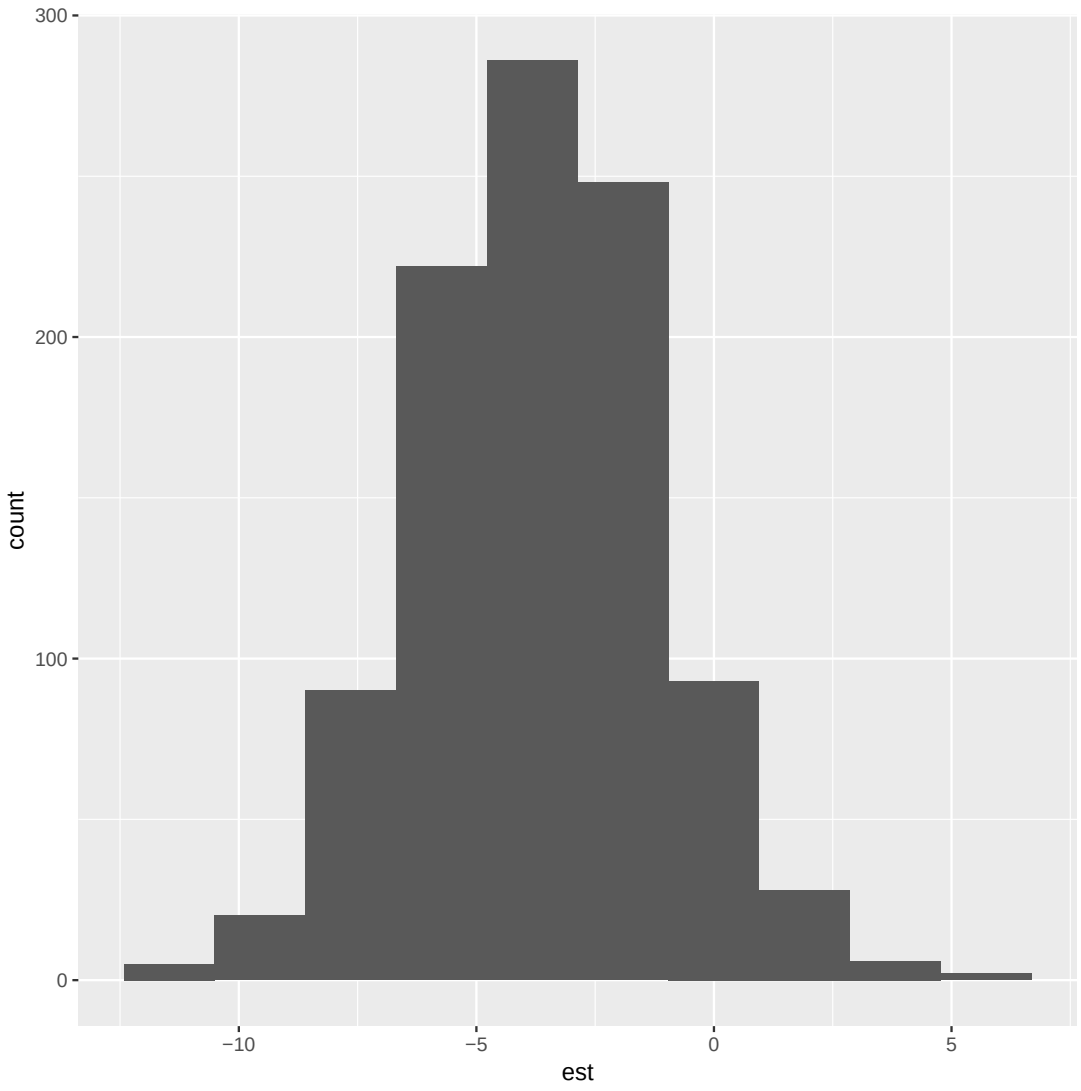
and then we get our bootstrapped distribution:

```
boot2 %>% mutate(est=map_dbl(splits, ~boot_t(.))) -> boot2_res
boot2_res
```

```
## # Bootstrap sampling using stratification
## # A tibble: 1,000 x 3
##    splits       id                est
##  * <list>       <chr>           <dbl>
##  1 <S3: rsplit> Bootstrap0001 -2.52
##  2 <S3: rsplit> Bootstrap0002 -4.93
##  3 <S3: rsplit> Bootstrap0003  1.02
##  4 <S3: rsplit> Bootstrap0004  1.70
##  5 <S3: rsplit> Bootstrap0005  0.1000
##  6 <S3: rsplit> Bootstrap0006 -1.68
##  7 <S3: rsplit> Bootstrap0007 -6.62
##  8 <S3: rsplit> Bootstrap0008  4.82
##  9 <S3: rsplit> Bootstrap0009 -1.4
## 10 <S3: rsplit> Bootstrap0010  1.32
## # ... with 990 more rows
```

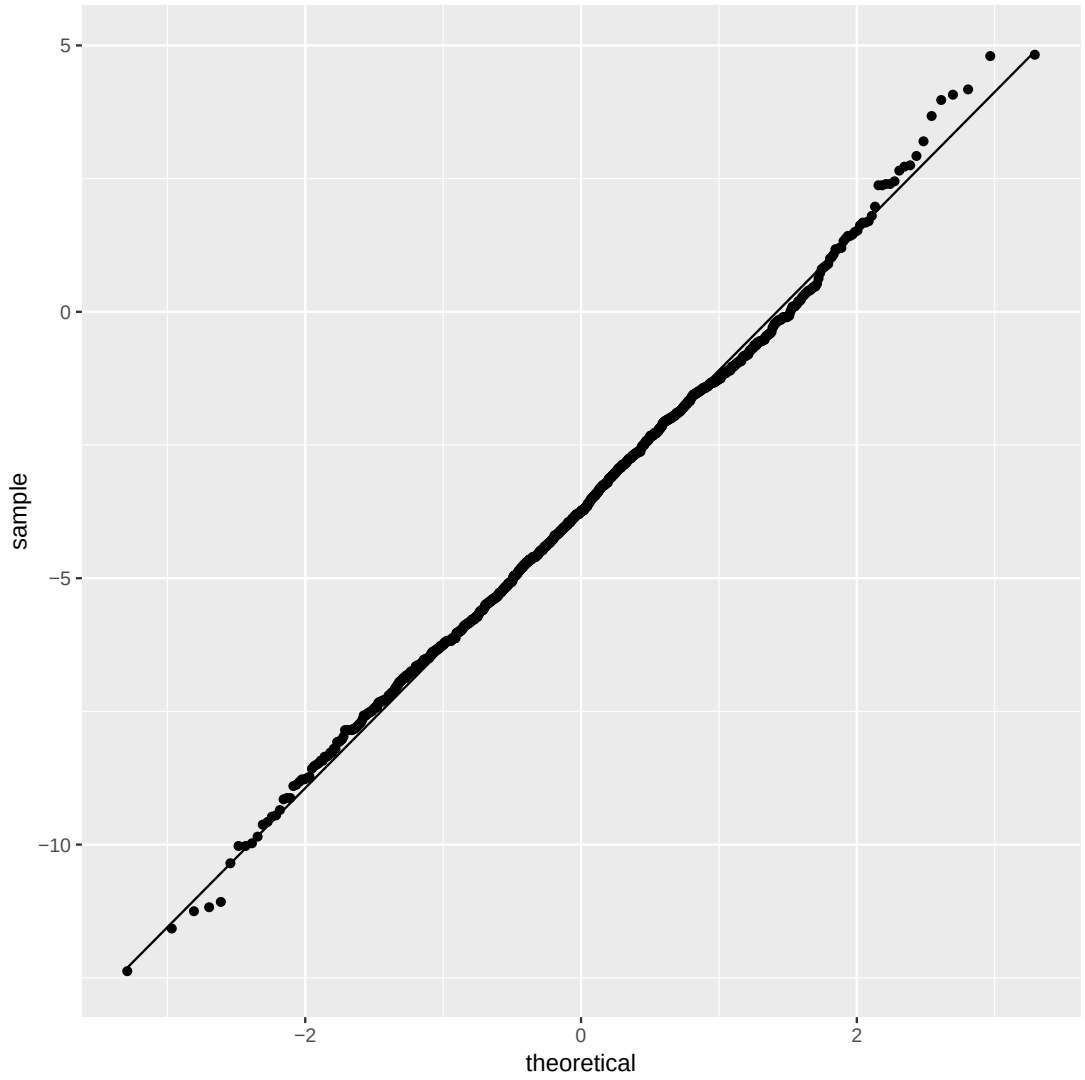
The sampling distribution won't be centred at zero, since the sample means are different, but we can still look at its shape:

```
ggplot(boot2_res,aes(x=est))+geom_histogram(bins=10)
```



That's nice and bell-shaped. Is it as normal as it looks here?

```
ggplot(boot2_res, aes(sample=est))+stat_qq()+stat_qq_line()
```



It's hard to improve on that. This means that the t-test for our data on journeys to school is entirely trustworthy, and we need look no further for a test.

You can (with sufficient organization) do a bootstrap in SAS also, but I have no intention of making you struggle with that.

6. We are planning a study to estimate a population mean. The population standard deviation is believed to be 20, and the population distribution is believed to be approximately normal. We will be testing the null hypothesis that the population mean is 100. Suppose the population mean is actually 110, and we want to determine how likely we are to (correctly) reject the null hypothesis in this case, using a two-sided (but one-sample) test with $\alpha = 0.05$.

   This is the same situation as we investigated before with R.

   (a) With a sample of size 30, what is the power of this test? You will need to specify the situation (`onesamplemeans`), the test `test=t`, the true mean (`mean`), the null mean `nullmean`, the population SD (`stddev`), the "total" sample size `ntotal` and the `power` that you are trying to find.

Solution: `proc power`, suitably modified. Guess or search for these:

```
proc power;
  onesamplemeans
  test=t
  mean=110
  nullmean=100
  stddev=20
  ntotal=30
  power=.;
```

The POWER Procedure
One-Sample t Test for Mean

Fixed Scenario Elements

Distribution	Normal
Method	Exact
Null Mean	100
Mean	110
Standard Deviation	20
Total Sample Size	30
Number of Sides	2
Alpha	0.05

Computed Power

Power
0.754

The power is the same 0.754 as R (to 3 decimals rather than R's 7).

(b) Use SAS to find the sample size necessary to obtain a power of (i) 0.80 and (ii) 0.90. (This is doable in one step.) What sample sizes do you get?

Solution: Use the same SAS code as (b), and modify it to specify the power values (with a space between) and leave the sample size blank:

```
proc power;
  onesamplemeans
  test=t
  mean=110
  nullmean=100
  stddev=20
  ntotal=.
  power=0.80 0.90;
```

The POWER Procedure
One-Sample t Test for Mean

Fixed Scenario Elements

Distribution	Normal
Method	Exact
Null Mean	100
Mean	110
Standard Deviation	20
Number of Sides	2
Alpha	0.05

Computed N Total

Index	Nominal Power	Actual Power	N Total
1	0.8	0.808	34
2	0.9	0.900	44

Sample sizes of 34 (for power 0.80) and 44 (for power 0.90).

The first of these is what we got from R.[16]

7. You are designing an experiment to compare a treatment with a control. The response variable you are measuring, called y, is expected to have a mean of 40 in the treatment group and 30 in the control group. The standard deviation of y is expected to be about 16 in both groups. The researchers plan to use the Satterthwaite test, since they suspect the spreads of the treatment and control groups to be different.

   Use SAS to answer the questions below. Bear in mind that you don't need any data; you just need to run the appropriate `proc` with the appropriate values.

   (a) If you have funding to collect a sample size of 25 within each group, how likely are you to (correctly) reject a null hypothesis that the treatment and control means are equal, in favour of a one-sided alternative that the treatment mean is *higher*, at $\alpha = 0.05$?

Solution: Feed `proc power` the appropriate things. This would be `twosamplemeans` with `test=diff_satt`, `sides=1` (one-sided test), `stddev=15` (the same for both groups), `ntotal=50` (a sample size of 25 within each group, so 50 altogether). Leave `power` "missing", since that's what we're trying to find. There are no semicolons until right at the end, because this (as far as SAS is concerned) is all one line:

```
proc power;
  twosamplemeans
    test=diff_satt
    sides=1
    meandiff=10
    stddev=16
    ntotal=50
    power=.;
```

My `meandiff` is 10 because $40 - 30 = 10$. How much power do we have to detect this kind of alternative with this kind of sample size?

The POWER Procedure
Two-Sample t Test for Mean Difference with Unequal Variances

Fixed Scenario Elements

Distribution	Normal
Method	Exact
Number of Sides	1
Mean Difference	10
Standard Deviation	16
Total Sample Size	50
Null Difference	0
Nominal Alpha	0.05
Group 1 Weight	1
Group 2 Weight	1

Computed Power

Actual Alpha	Power
0.0499	0.703

The power is 0.703.

There are actually several ways to do this. Another way is this:

```
proc power;
  twosamplemeans
    test=diff_satt
    sides=1
    groupmeans=40|30
    stddev=16
    ntotal=50
    power=.;
```

which gives the same result:

```
                          The POWER Procedure
        Two-Sample t Test for Mean Difference with Unequal Variances

                          Fixed Scenario Elements

                    Distribution                Normal
                    Method                       Exact
                    Number of Sides                  1
                    Group 1 Mean                    40
                    Group 2 Mean                    30
                    Standard Deviation              16
                    Total Sample Size               50
                    Null Difference                  0
                    Nominal Alpha                 0.05
                    Group 1 Weight                   1
                    Group 2 Weight                   1
                              Computed Power

                              Actual
                               Alpha    Power

                              0.0499    0.703
```

If you have done something plausible-looking, I am good with it. You can also specify the group SDs separately via `groupstddevs` (which you would want to do if they are different from each other), and you can also specify the sample size within each group via `npergroup`. (If you do the latter in the next part, SAS will tell you how many subjects to use in each group rather than altogether.) My reference for these is `https://support.sas.com/documentation/cdl/en/statug/63033/HTML/default/viewer.htm#statug_power_sect013.htm`.

About the `sides=1` thing, that being a number 1. When you are doing a test and getting a P-value, if your test is one-sided, you have to say *which* one side you want via `sides=U` or `sides=L`. You can also do that in `proc power`, but it seems not to be intuitive which way around it is. Sometimes it gives you a very small power, which is an indication that you were actually looking in the wrong tail. The safe way around this, *for `proc power` only*, is to use `sides=1`, which says that you're doing a one-sided test and you want the power *in the same direction as the effect*. In this case the true mean for group 1 was (expected to be) 40 and for group 2 was 30. The obvious thing to obtain power for here is the one-sided test with a null of equal means and an alternative that group 1's mean is bigger than group 2's, because the true mean for group 1 *is* bigger than for group 2. The advantage of using `sides=1` is that it will do this, whichever way around the groups are listed. That is to say, this works, as above:

```
proc power;
  twosamplemeans
    test=diff_satt
    sides=1
    meandiff=10
    stddev=16
    ntotal=50
    power=.;
```

```
                          The POWER Procedure
        Two-Sample t Test for Mean Difference with Unequal Variances

                          Fixed Scenario Elements

                    Distribution                Normal
                    Method                       Exact
                    Number of Sides                  1
                    Mean Difference                 10
                    Standard Deviation              16
                    Total Sample Size               50
                    Null Difference                  0
                    Nominal Alpha                 0.05
                    Group 1 Weight                   1
                    Group 2 Weight                   1
                              Computed Power

                              Actual
                               Alpha    Power

                              0.0499    0.703
```

but this also works, *without changing* `sides`:

```
proc power;
  twosamplemeans
    test=diff_satt
    sides=1
    meandiff=-10 /* switch the groups around */
    stddev=16
    ntotal=50
    power=.;
```

The POWER Procedure
Two-Sample t Test for Mean Difference with Unequal Variances

Fixed Scenario Elements

Distribution	Normal
Method	Exact
Number of Sides	1
Mean Difference	-10
Standard Deviation	16
Total Sample Size	50
Null Difference	0
Nominal Alpha	0.05
Group 1 Weight	1
Group 2 Weight	1

Computed Power

Actual Alpha	Power
0.0499	0.703

(b) Some extra research funding becomes available, and the principal investigator is curious how large a sample size would be needed to increase this power to 0.80, with everything else remaining the same. Find out how large a sample size would be needed, so that the principal investigator can determine whether the funding is available to support the collection of the larger sample.

Solution: Use the same code as before, but make two changes: set `ntotal` to missing, and fill in 0.80 for the power:

```
proc power;
  twosamplemeans
    test=diff_satt
    sides=1
    meandiff=10
    stddev=16
    ntotal=.
    power=0.80;
```

This gives

The POWER Procedure
Two-Sample t Test for Mean Difference with Unequal Variances

Fixed Scenario Elements

Distribution	Normal
Method	Exact
Number of Sides	1
Mean Difference	10
Standard Deviation	16
Nominal Power	0.8
Null Difference	0
Nominal Alpha	0.05
Group 1 Weight	1
Group 2 Weight	1

Computed N Total

Actual Alpha	Actual Power	N Total
0.0499	0.807	66

Read off "N Total" from the output, 66. Remember that this is the sample size needed *altogether*, so half of these should be in each group: that is, 33 in the treatment group and 33 in the control group. So, you tell the principal investigator that 33 subjects are needed in each group, and they can see whether they can get funding to support that.

As per the discussion above, this also works:

```
proc power;
  twosamplemeans
    test=diff_satt
    sides=1
    meandiff=10
    stddev=16
    npergroup=.  /* this was changed */
    power=0.80;
```

This gives

```
                          The POWER Procedure
        Two-Sample t Test for Mean Difference with Unequal Variances

                          Fixed Scenario Elements

                    Distribution                Normal
                    Method                       Exact
                    Number of Sides                  1
                    Mean Difference                 10
                    Standard Deviation              16
                    Nominal Power                  0.8
                    Null Difference                  0
                    Nominal Alpha                 0.05

                           Computed N per Group

                         Actual    Actual    N per
                          Alpha     Power    Group

                         0.0499     0.807       33
```

This gives you the number of observations required in each group, 33, so that you again need $33 + 33 = 66$ observations altogether.

You might notice that when SAS does a sample size calculation, it rounds up the sample size for you (unlike R, which gives you a decimal-number sample size that you have to round up yourself). SAS tells you about this using the words "nominal power" for the 0.8 you were aiming for, and the "actual power" of an experiment with 33 subjects in each group, which will be a little bigger than 0.8 (here it is 0.807). The implication is that 32 subjects per group gives power a little bit *less* than 0.8, and so we're erring on the side of caution, so to speak, by rounding the sample size up. Likewise, with 33 subjects per group, the actual probability of a type I error ("actual alpha") is a tiny bit less than the 0.05 we were aiming for ("nominal alpha").

Hand the next two questions in.

8. Random samples of healthy men and women were taken, and their heart rates measured under normal

conditions. The data are from an Excel spreadsheet, saved at `http://www.utsc.utoronto.ca/~butler/c32/heart-rates.csv`. We are interested in whether males and females differ in average heart rate.

(a) (2 marks) Read the data into SAS and display the data set.

Solution: This is a `.csv` file (it was created from a spreadsheet), so make sure the `dbms` line says that:

```
filename myurl url "http://www.utsc.utoronto.ca/~butler/c32/heart-rates.csv";

proc import
  datafile=myurl
  out=heartrates
  dbms=csv
  replace;
  getnames=yes;

proc print;
```

with output

Obs	gender	heartrate
1	male	74
2	male	80
3	male	75
4	male	69
5	male	58
6	male	76
7	male	78
8	male	78
9	male	86
10	male	84
11	male	71
12	male	80
13	male	75
14	female	75
15	female	66
16	female	57
17	female	87
18	female	89
19	female	65
20	female	69
21	female	79
22	female	85
23	female	59
24	female	65
25	female	80
26	female	74

Extra: this was originally an Excel spreadsheet called `heart-rates.xlsx`, which you can find at `http://www.utsc.utoronto.ca/~butler/c32/heart-rates.xlsx`.[17] I thought you could read an `.xlsx` file directly from a URL, but SAS has the same problem with that as `read_excel` does. So to read that you would have to do two extra steps:

1. *download* the spreadsheet from the URL (change the `.csv` at the end to `.xlsx`; it goes into `Downloads` or wherever it goes on your computer)

2. *upload* the spreadsheet from your computer to SAS Studio.

Then you read it in via the `/home/username` thing. This is how it looks for me (you'd need to replace my username with yours). Note that there is no `filename` line this way because you are going to put the file's name directly on the `datafile` line:

```
proc import
  datafile='/home/ken/heart-rates.xlsx'
  out=heartrates
  dbms=xlsx
  replace;
  getnames=yes;
  sheet=Sheet1;

proc print;
```

Note that I needed to specify the worksheet name (since there might have been more than one, though in this case there wasn't).

This gives the same output (since it's the same data):

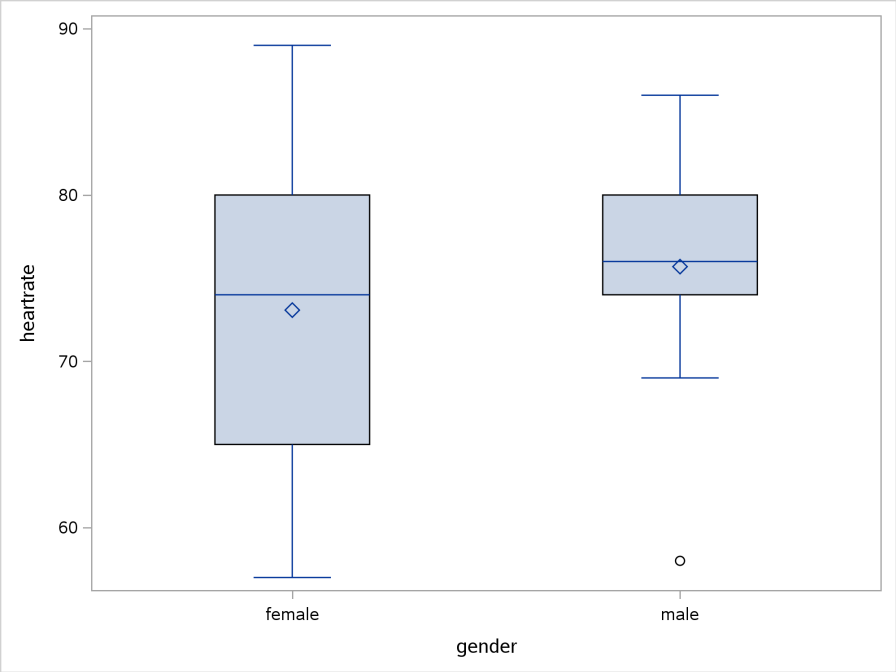
Obs	gender	heartrate
1	male	74
2	male	80
3	male	75
4	male	69
5	male	58
6	male	76
7	male	78
8	male	78
9	male	86
10	male	84
11	male	71
12	male	80
13	male	75
14	female	75
15	female	66
16	female	57
17	female	87
18	female	89
19	female	65
20	female	69
21	female	79
22	female	85
23	female	59
24	female	65
25	female	80
26	female	74

This is now the "most recently created" data set, so it's the one that will get used below, but it's the same as yours.

(b) (2 marks) Make a suitable plot of the two variables.

Solution: One quantitative and one categorical ought to suggest "boxplot" to you:

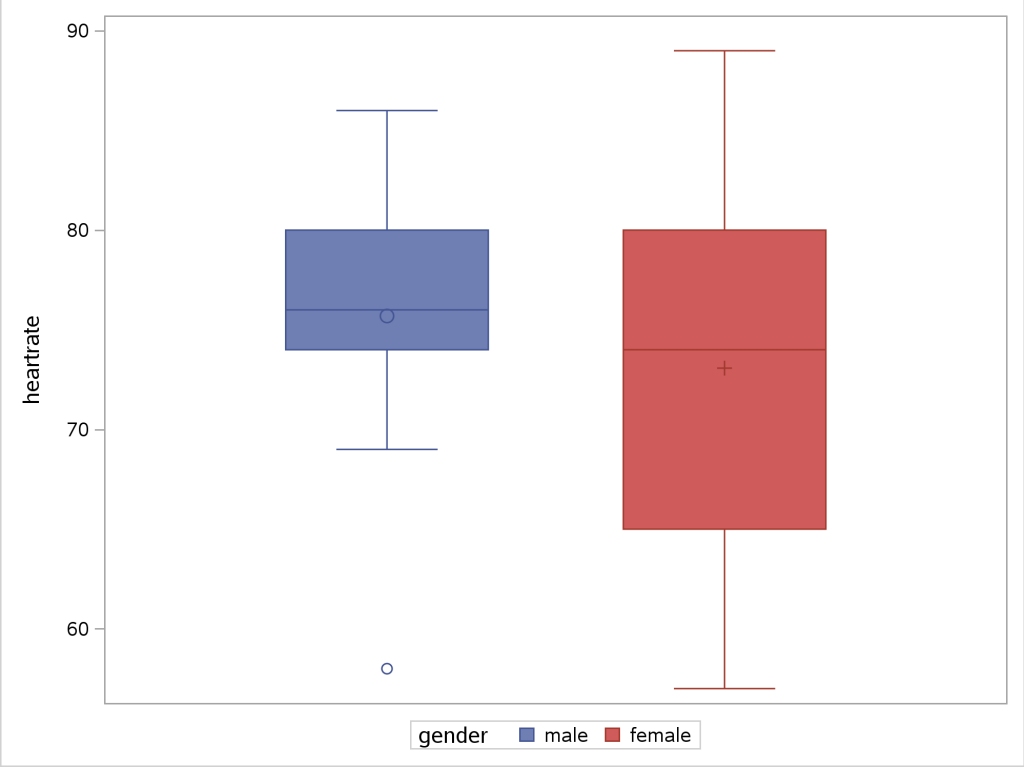
```
proc sgplot;
  vbox heartrate / category=gender;
```



I'm not asking you for comment yet, but looking ahead to a two-sample t-test, I hope you're looking for outliers, skewness and (in terms of which two-sample t-test to run) whether the genders have similar spread or not. That is coming up later.

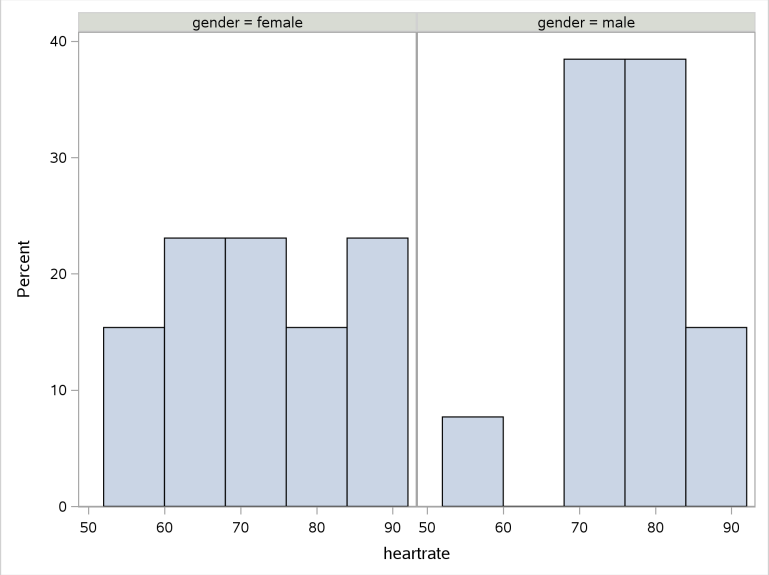
This gets you coloured boxplots, which it is up to you whether you prefer. For some reason, they also come out the other way around:

```
proc sgplot;
  vbox heartrate / group=gender;
```



Faceted histograms would also work:

```
proc sgpanel;
  panelby gender;
  histogram heartrate;
```



(c) (3 marks) Run the most appropriate t-test to compare the mean heart rate for males and females. What do you conclude, in the context of the data?

Solution: This is `proc ttest`, used to do a two-sample t-test. SAS's approach is to give you both the Satterthwaite-Welch and the pooled tests, and leave it to you to choose which one you want:

```
proc ttest;
  var heartrate;
  class gender;
```

Here's the (text part of the) output:

```
gender          N        Mean     Std Dev     Std Err     Minimum     Maximum

female         13     73.0769     10.5314      2.9209     57.0000     89.0000
male           13     75.6923      7.1108      1.9722     58.0000     86.0000
Diff (1-2)            -2.6154      8.9854      3.5244

   gender       Method               Mean       95% CL Mean         Std Dev

   female                         73.0769     66.7129  79.4410     10.5314
   male                           75.6923     71.3953  79.9893      7.1108
   Diff (1-2)   Pooled            -2.6154     -9.8893   4.6585      8.9854
   Diff (1-2)   Satterthwaite     -2.6154     -9.9434   4.7127

               gender        Method             95% CL Std Dev

               female                            7.5519  17.3845
               male                              5.0991  11.7381
               Diff (1-2)    Pooled              7.0160  12.5000
               Diff (1-2)    Satterthwaite
```

```
Method           Variances        DF    t Value    Pr > |t|

Pooled           Equal            24      -0.74      0.4652
Satterthwaite    Unequal      21.059      -0.74      0.4662
                   Equality of Variances

     Method      Num DF    Den DF    F Value    Pr > F

     Folded F        12        12       2.19    0.1881
```

I looked at the boxplot and said to myself "the females have a bigger spread than the males do", and so I chose the Satterthwaite test, with a P-value of 0.4662. This ignores the low outlier in the males, which will inflate the SD some, but if you look at the output from `proc ttest`, the females have SD 10.5 and the males 7.1 (smaller, even including the outlier), which are not very close to being equal in my opinion, though you are free to disagree. For instance, if you want to say "the group SDs are not all that different, so I use the pooled t-test and obtain a P-value of 0.4652", I'm good with that.

In summary, pick *one* of the P-values, say which one it is, and give some defensible reason why you chose it. If you say something like "the P-value is not less than 0.05", you are *wrong*, because there are two P-values (at least) and you haven't said which one you're looking at (or why). Saying *which* P-value you are using, if there is more than one, is *always* a good idea, on exams too. Otherwise it makes your work look sloppy. You might think that "none of the P-values are significant" would do it, but that includes the "folded F" that is a test for comparing *variances* as well, so that doesn't work. "Neither the Welch-Satterthwaite nor the pooled two-sample t-tests are significant" *would* do it, but if you're going to say that, you might as well pick *one* t-test and talk about that.

Either way, the P-value is not anywhere near small enough to reject with, so there is no evidence that males and females differ in mean heart rate.

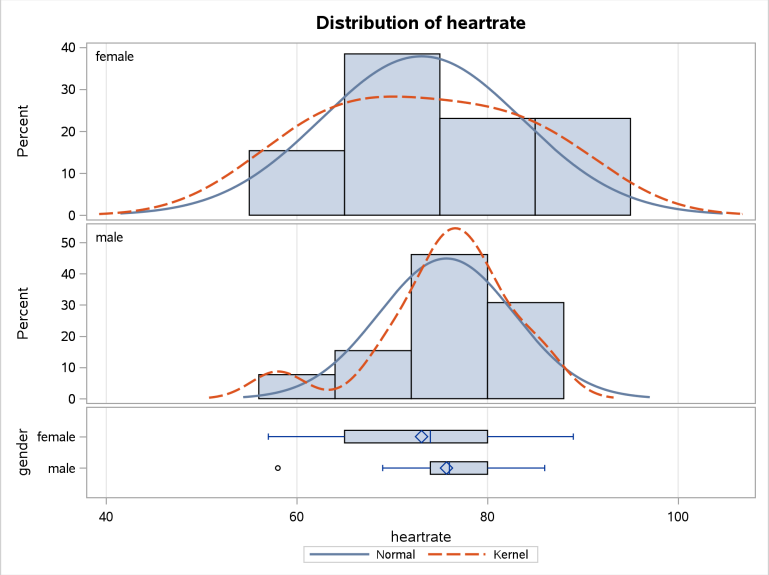
I carefully said "most appropriate t-test" in the question. If you don't think any kind of t-test is OK, I am asking you to choose the "least inappropriate" one, since the task here is to practice running a two-sample t-test, and *then* to think about what else you might do.

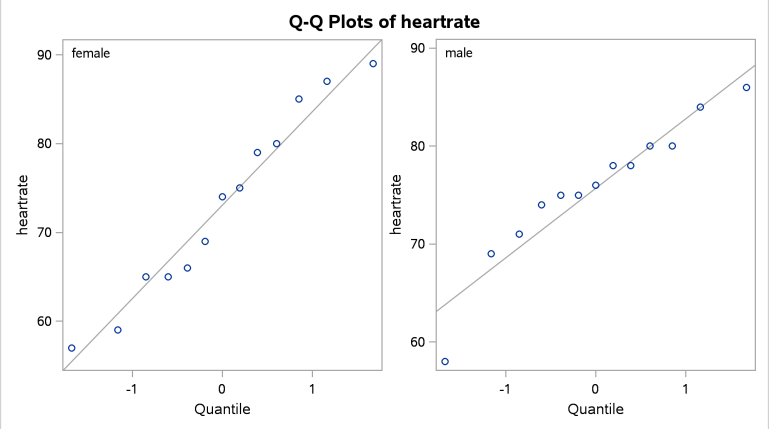
Extra: even though it looks that the groups are rather different in spread, the two P-values for the pooled and Satterthwaite tests are almost identical, as in almost all of the examples we've seen. I would even (here) accept "the two P-values are almost the same so it doesn't matter which test we do" (that is to say, Welch-Satterthwaite and pooled) as a defensible reason for *not* picking one over the other.

(d) (2 marks) Would you trust the result of your test? Explain briefly why or why not. Refer back to any of your previous output as necessary.

Solution: This is an invitation to go back to your graph and check for symmetry and outliers. Symmetry appears OK (all the whiskers are about the same length), but that low outlier on the males troubles me. That would be a reason to *not* trust a t-test here. Having said that, you could defend the t-test in a couple of ways: (i) despite the outlier, the mean is not pulled down very much compared to the median, (ii) the test is so far from being significant that abandoning the t-test and doing something like Mood's median test instead is very unlikely to change the P-value much. (I come back to this in a moment.)

Extra: you might have noticed that `proc ttest` produces graphs along with the text output. I would also (happily) accept a critique of your t-test based on one of these graphs, as long as you say which one(s) and why. (I'm looking for one reason, probably the outlier, why you don't like the t-test, but that might be supported by more than one of these graphs.)


Distribution of heartrate
female
male
Percent
gender
female
male
40
60
80
100
heartrate
Normal
Kernel


Q-Q Plots of heartrate
female
male
heartrate
Quantile
-1
0
1
60
70
80
90

The top plot shows histograms for the males and females. It is a little hard to assess these for normality with only 13 observations in each group, though the male histogram looks a bit skewed left. The red kernel density curves are pretty close to the blue normal curves in each case, though the kernel density curve for the males has a "bump" where the outlier appears to be. I think the decision about whether the smallest value is an outlier will be very dependent on the choice of bins for the histogram; the outlier looks a lot clearer on the boxplot, reproduced below the histograms. (I ran into another example like this, where the outlier was a lot less clear on a histogram, which makes me wonder whether this happens often. Unless you tweaked the bins to make it show up, once you'd found where it was. Tweaking the bins like this has, to my mind, crossed over the line into cheating.)

The bottom plot shows normal quantile plots for each group. I have to say that the normality looks pretty acceptable with these sample sizes; I was expecting the low outlier on the males to stand out more, when here it's only somewhat lower, about 5 units lower, than you would expect on a normal distribution. So maybe the boxplot was actually exaggerating the outlierness of that lowest observation in the males. (Or, the fact that the default SAS line uses the standard deviation, which is inflated by the outlier, and so the line on the normal quantile plot for the males would be flatter on R's normal quantile plot and thus make the outlier stand out more.)

You know the drill: make a call and defend it. I don't mind whether you think that value is an outlier, causing trouble, or not, as long as you have decent reasons for your opinion.

Extra extra: I mentioned Mood's median test above, and I said that I suspected that it wouldn't be anywhere near significant either:

```
proc npar1way median;
  var heartrate;
  class gender;
```

```
                        The NPAR1WAY Procedure

     Median Scores (Number of Points Above Median) for Variable heartrate
                     Classified by Variable gender

                       Sum of      Expected       Std Dev          Mean
    gender       N     Scores      Under H0      Under H0         Score

    female      13   5.333333          6.50      1.231530      0.410256
    male        13   7.666667          6.50      1.231530      0.589744

                   Average scores were used for ties.
                        Median Two-Sample Test

                     Statistic              5.3333
                     Z                     -0.9473
                     One-Sided Pr <  Z      0.1717
                     Two-Sided Pr > |Z|     0.3435
                        Median One-Way Analysis

                     Chi-Square             0.8974
                     DF                          1
                     Pr > Chi-Square        0.3435
```

The P-value is a bit smaller, 0.3435, but still not anywhere near small enough to reject with. (Yes, I know I haven't talked about this in class yet, but the principle is the same as in R.)

9. A study is being done to test whether a population mean is 100. The population standard deviation is

believed to be about 10, and the population is believed to be normal in shape. Use SAS for this question.

(a) (4 marks) If a sample of size 35 is taken, and the population mean is actually 103, how likely is it that the null mean of 100 will be correctly rejected, against a two-sided alternative?

Solution: This is a one-sample t-test (there will be only one sample of observations to estimate one mean), so you need this kind of thing:

```
proc power;
  onesamplemeans
  test=t
  mean=103
  nullmean=100
  sttdev=10
  ntotal=35
  power=.;
```

Leave the power missing, since that's what you're trying to find.

The POWER Procedure
One-Sample t Test for Mean

Fixed Scenario Elements

Distribution	Normal
Method	Exact
Null Mean	100
Mean	103
Standard Deviation	10
Total Sample Size	35
Number of Sides	2
Alpha	0.05

Computed Power

Power
0.407

You can also have `mean` be the difference 3 between the null and actual means, and omit `nullmean` completely. (You can also put in `sides=2`, but that is the default, so you don't need to worry about that.)

The power is 0.407, so that's the chance of correctly rejecting the null: not as big as you might like.

(b) (3 marks) Under the same conditions as the previous part, how big a sample size is needed to obtain power 0.75?

Solution: Two changes to your code: set `ntotal=.`, since you are trying to find a sample size, and put in the power value you are shooting for:

```
proc power;
  onesamplemeans
  test=t
  mean=103
  nullmean=100
```

```
        sttdev=10
        ntotal=.
        power=0.75;
```

This is the output:

```
                          The POWER Procedure
                      One-Sample t Test for Mean

                          Fixed Scenario Elements

                    Distribution                Normal
                    Method                       Exact
                    Null Mean                      100
                    Mean                           103
                    Standard Deviation              10
                    Nominal Power                 0.75
                    Number of Sides                  2
                    Alpha                         0.05

                             Computed N Total

                             Actual        N
                              Power    Total

                              0.755       80
```

A sample size of 80 is needed. (There is only one sample, so this is how big the one sample has to be.)

(c) (2 marks) SAS gave you an "actual power" that is slightly different from what you asked for. Explain briefly why it did that.

Solution: You can only get exactly the power you asked for by allowing a fractional sample size (which is what R does). But in real life the sample size must be a whole number, so you either get slightly more power than you wanted or slightly less, depending on whether you round up or down. The "safe" way to proceed is to round *up*, which gives you at least as much power as you asked for (usually a little more: here it's 0.755 vs. 0.75). SAS tells you (under `Actual Power`) exactly how much power you got. The line `Nominal Power` tells you what power you were aiming for.

The implication of this is that a sample size of 80 would give you power slightly greater than 0.75 and a sample size of 79 would give you power slightly *less* than 0.75. When we were working with R, this is why I told you always to round sample sizes up, even if the answer was something like 79.1: rounding down would give you power slightly less than your target.

"Because the sample size has to be an integer" is only one point because it's not enough insight. The second point is for noting the implication of this: that the actual power you get is either a bit too much or not quite enough.

(d) (3 marks) After further study, the researchers found that the population SD is about 12 rather than about 10. Calculate the sample size now required to get a power of 0.75, and explain briefly why the change in results from part (b) is not surprising.

Solution: For the calculation, repeat the previous part but with the `stddev` changed from 10 to 12, using another dose of copy and paste (or editing of what you had before):

```
proc power;
  onesamplemeans
  test=t
  mean=103
  nullmean=100
  sttdev=12
  ntotal=.
  power=0.75;
```

```
                          The POWER Procedure
                       One-Sample t Test for Mean

                          Fixed Scenario Elements

                    Distribution                Normal
                    Method                       Exact
                    Null Mean                      100
                    Mean                           103
                    Standard Deviation              12
                    Nominal Power                 0.75
                    Number of Sides                  2
                    Alpha                         0.05

                             Computed N Total

                             Actual        N
                              Power    Total

                              0.750      113
```

The sample size needed is now 113, about 40% bigger than before.

This makes sense because if the population SD is bigger, the data we get are likely to be more variable, and thus we will have a harder time rejecting the null hypothesis then before, all else being equal. (A little more precisely, a larger population SD will tend to mean a larger sample SD, and thus a *smaller* test statistic, not so far away from zero, and thus a less small P-value.) In order to keep the power the same, without changing anything else, we have to take a bigger sample size.

Extra: I was actually surprised it came out this much bigger, but it's not always easy to predict what will happen in a power calculation. If we had been thinking about confidence interval length instead, it would have been easier: increasing the SD by a factor of 1.2 (as here) would also increase the length of the confidence interval by a factor of 1.2, all else equal. So if we wanted to keep the confidence interval length the same, we'd have to increase the sample size by a factor of $1.2^2 = 1.44$. So maybe the 40% increase in sample size in our power calculation ought not to have surprised me.

Extra extra: it looks as if we hit our target power of 0.75 exactly this time, which we may have been lucky enough to do, but I think it is more likely that the "actual power" is something like 0.7501, a teeny bit bigger than 0.75, if enough decimal places were shown.

Notes

[1] "The default number of bins is determined by the system", it says, helpfully.

[2] In power calculations done with `proc power` later, the number `1` is used, since the results are the same whichever one side is used.

[3] For reasons of statistical power, which we study later, it is good to aim for the same number of individuals in each group. That is, if you have the same number in each group, you maximize your chances of finding a difference, if there really is one.

[4] It is possible to get outliers at both ends, and then you might be wondering about skewness. I would say in that case that your major problem is outliers at both ends, and leave it at that. Boxplots were designed for small-to-moderate data sets, which is typically what you saw in the 1950s when Tukey invented boxplots; with a bigger data set, you can happen to get a lot more outliers just by chance, so if your data set is big, it may not be worth worrying about a few outliers.

[5] You'll recall that the t-tests are "robust" to non-normality, which means that you can get away with the data not being all that normal, at least if you have large samples.

[6] A lowercase `l` also works, but I prefer not to use it because it looks too much like a number `1`. When you specify a one-sided alternative, you need to say *which* one side you want, lower or upper.

[7] Compare this boxplot to the R one from earlier, where the spreads of the two groups look a lot more similar. R and SAS have different default definitions for quartiles (there are actually a *lot* of possible definitions), so the boxplots look different. The consequence of this is that your choice of test might be different when using R vs. when using SAS, though the conclusion, as you see from the two P-values, is pretty much identical either way.

[8] Throwing your hat across the room.

[9] I think that is free to view. Find it via `library.utoronto.ca` if not.

[10] With ν degrees of freedom, the variance is $\nu/(\nu-2)$.

[11] Some people call this the "Behrens-Fisher problem". See that name Fisher again?

[12] R calls it the "Welch t-test".

[13] This is the issue around `guessingrows` which I talk about elsewhere.

[14] If you sample *without* replacement, you just get the original samples back, which is kind of pointless.

[15] This code won't make much sense yet.

[16] Rounded up: SAS has done the proper rounding for you.

[17] Actually, it was a Libre Office spreadsheet before that, but I am trying not to confuse you too much.