Assignment 8

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

Do *not* expect to get help on this assignment. The purpose of the assignments is for you to see how much *you* have understood. You will find that you also learn something from grappling with the assignments. The time to get help is after you watch the lectures and work through the problems from PASIAS, via tutorial and the discussion board, that is *before* you start work on the assignment. The only reasons to contact the instructor while working on an assignment are to report (i) something missing like a data file that cannot possibly be read, (ii) something *beyond your control* that makes it impossible to finish the assignment in time after you have started it.

There is a time limit on this assignment (you will see Quercus counting down the time remaining).

1. Is it possible to estimate the height of a person from the length of their foot? To find out, 33 (male) students had their height and foot length measured. The data are in http://ritsokiguess.site/STAC32/heightfoot.csv.

   (a) Read in and display (some of) the data. (If you are having trouble, make sure you have *exactly* the right URL. The correct URL has no spaces or other strange characters in it.)

Solution:

The usual:

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

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   height = col_double(),
##   foot = col_double()
## )
```

```
hf
```

```
## # A tibble: 33 x 2
##    height  foot
##     <dbl> <dbl>
##  1   66.5  27
##  2   73.5  29
##  3   70    25.5
##  4   71    27.9
```

```
##  5   73    27
##  6   71    26
##  7   71    29
##  8   69.5  27
##  9   73    29
## 10   71    27
## # ... with 23 more rows
```

Call the data frame whatever you like, but keeping away from the names `height` and `foot` is probably wise, since those are the names of the columns.

There are indeed 33 rows as promised.

Extra: my comment in the question was to help you if you copy-pasted the file URL into R Studio. Depending on your setup, this might have gotten pasted with a space in it, at the point where it is split over two lines. The *best* way to proceed, one that won't run into this problem, is to *right*-click on the URL and select Copy Link Address (or the equivalent on your system), and then it will put the whole URL on the clipboard in one piece, even if it is split over two lines in the original document, so that pasting it will work without problems.

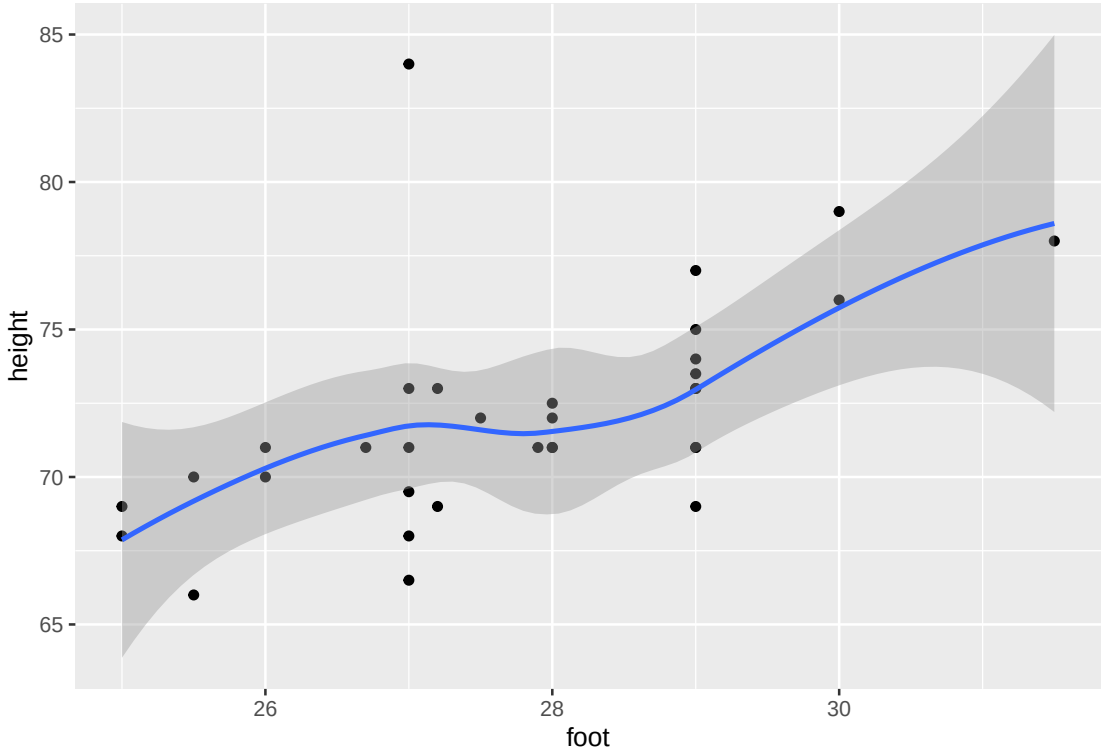
(b) Make a suitable plot of the two variables in the data frame.

Solution:

They are both quantitative, so a scatter plot is called for:

```
ggplot(hf, aes(y=height, x=foot)) + geom_point() + geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



I added a smooth trend, or you could just plot the points. (This is better than plotting a regression line at this stage, because we haven't yet thought about whether the trend is straight.)

Now that we've seen the scatterplot, the trend looks more or less straight (but you should take a look at the scatterplot first, with or without smooth trend, before you put a regression line on it). That point top left is a concern, though, which brings us to…

(c) Are there any observations not on the trend of the other points? What is unusual about those observations?

Solution:

The observation with height greater than 80 at the top of the graph looks like an outlier and does not follow the trend of the rest of the points. Or, this individual is much taller than you would expect for someone with a foot length of 27 inches. Or, this person is over 7 feet tall, which makes little sense as a height. Say something about what makes this person be off the trend.

(d) Fit a regression predicting height from foot length, *including* any observations that you identified in the previous part. For that regression, plot the residuals against the fitted values and make a normal quantile plot of the residuals.

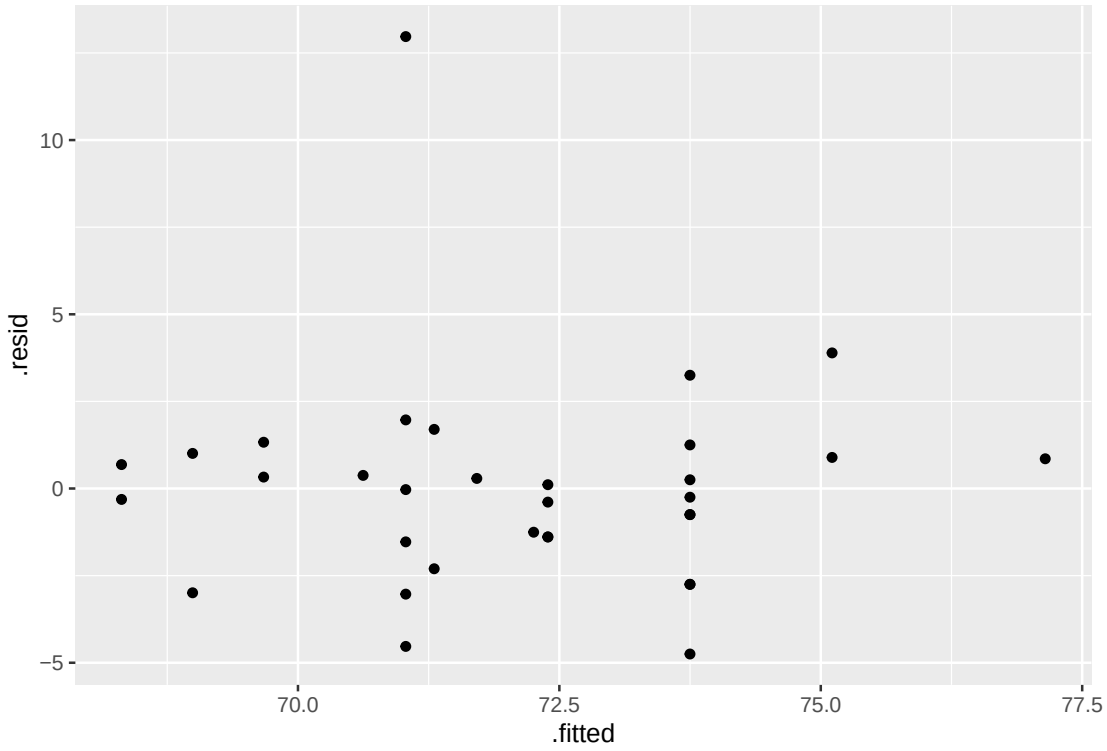
Solution:

These things. Displaying the `summary` of the regression is optional, but gives the grader an opportunity to check that your work is all right so far.

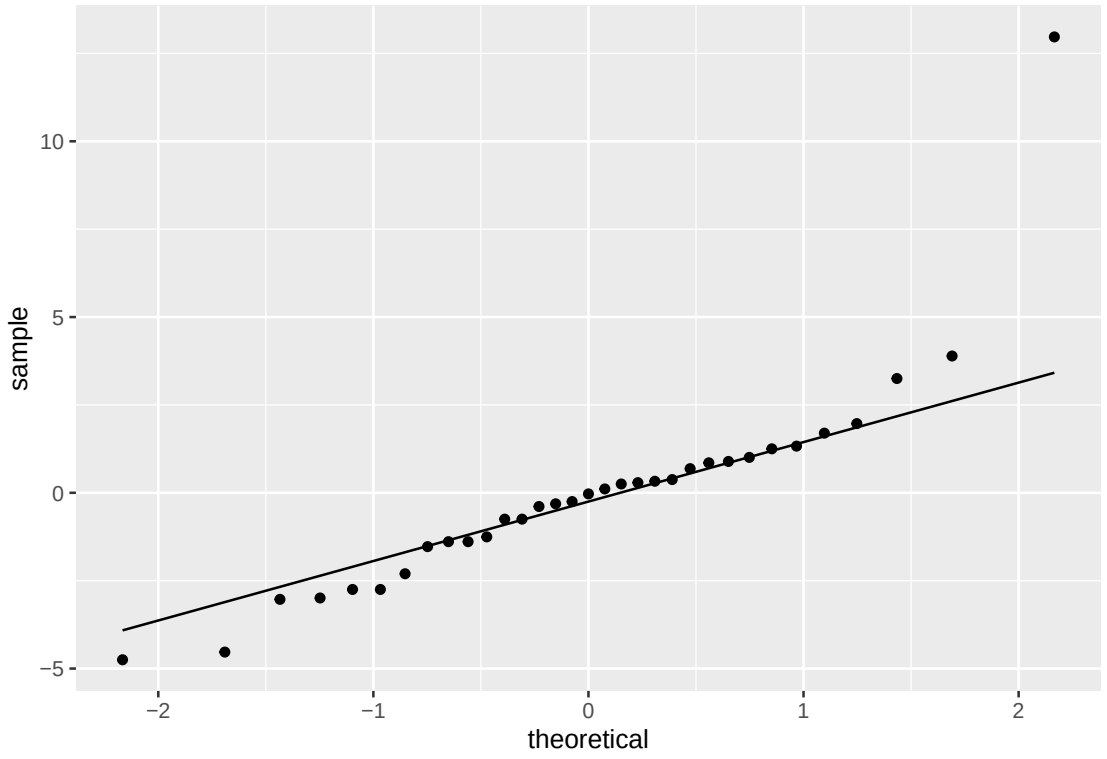
```
hf.1 <- lm(height~foot, data=hf)
summary(hf.1)
```

```
##
## Call:
## lm(formula = height ~ foot, data = hf)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.7491 -1.3901 -0.0310  0.8918 12.9690
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  34.3363     9.9541   3.449 0.001640 **
## foot          1.3591     0.3581   3.795 0.000643 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.102 on 31 degrees of freedom
## Multiple R-squared:  0.3173, Adjusted R-squared:  0.2952
## F-statistic: 14.41 on 1 and 31 DF,  p-value: 0.0006428
```

```
ggplot(hf.1, aes(x=.fitted, y=.resid)) + geom_point()
```




```
ggplot(hf.1, aes(sample=.resid)) + stat_qq() + stat_qq_line()
```



Note that we did not exclude the off-trend point. Removing points *because they are outliers* is

a **bad** idea. This is a good discussion of the issues.

(e) Earlier, you identified one or more observations that were off the trend. How does this point or points show up on each of the plots you drew in the previous part?

Solution:

On its own at the top in both cases; the large positive residual on the first plot, and the unusually large value at the top right of the normal quantile plot. (You need to say one thing about each graph, or say as I did that the same kind of thing happens on both graphs.)

Extra: in the residuals vs. fitted values, the other residuals show a slight upward trend. This is because the regression line for these data, with the outlier, is pulled (slightly) closer to the outlier and thus slightly further away from the other points, particularly the ones on the left, compared to the same data but with the outlier removed (which you will be seeing shortly). If the unusual point had happened to have an extreme x (foot length) as well, the effect would have been more pronounced.

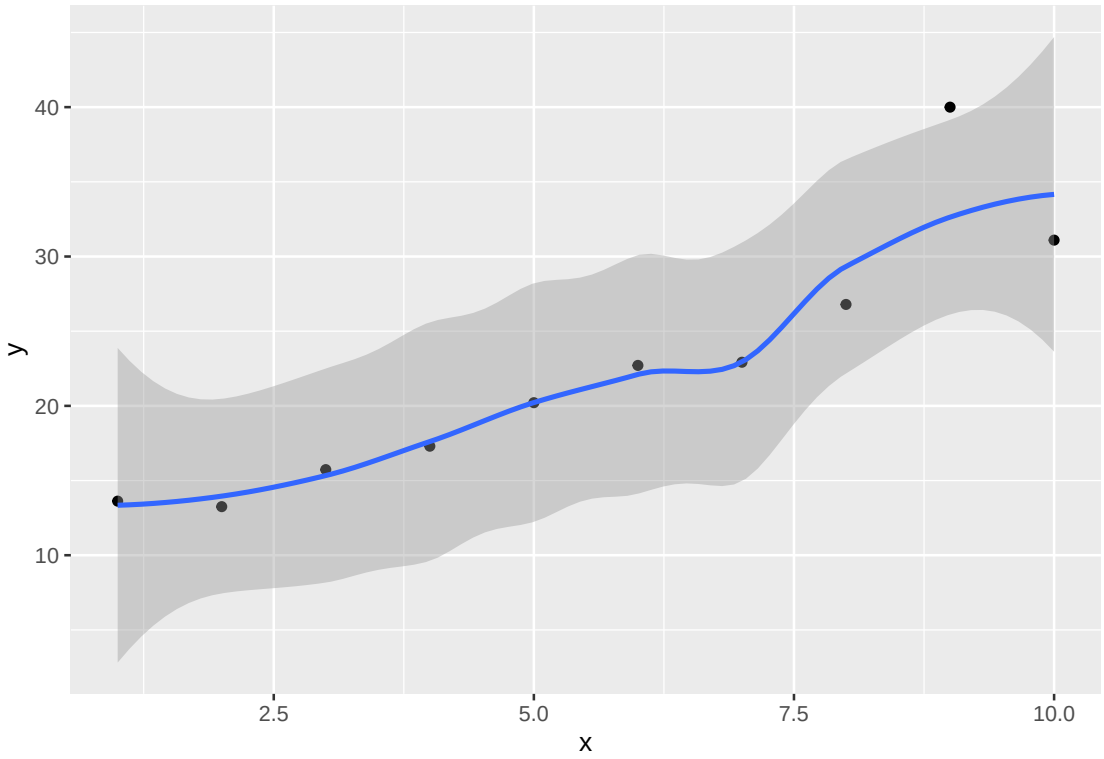
This is the kind of thing I mean (made-up data):

```
tibble(x = 1:10) %>%
  mutate(y = rnorm(10, 10+2*x, 1)) %>%
  mutate(y = ifelse(x == 9, 40, y)) -> madeup
madeup
```

```
## # A tibble: 10 x 2
##        x     y
##    <int> <dbl>
##  1     1  13.6
##  2     2  13.3
##  3     3  15.7
##  4     4  17.3
##  5     5  20.2
##  6     6  22.7
##  7     7  22.9
##  8     8  26.8
##  9     9  40
## 10    10  31.1
```

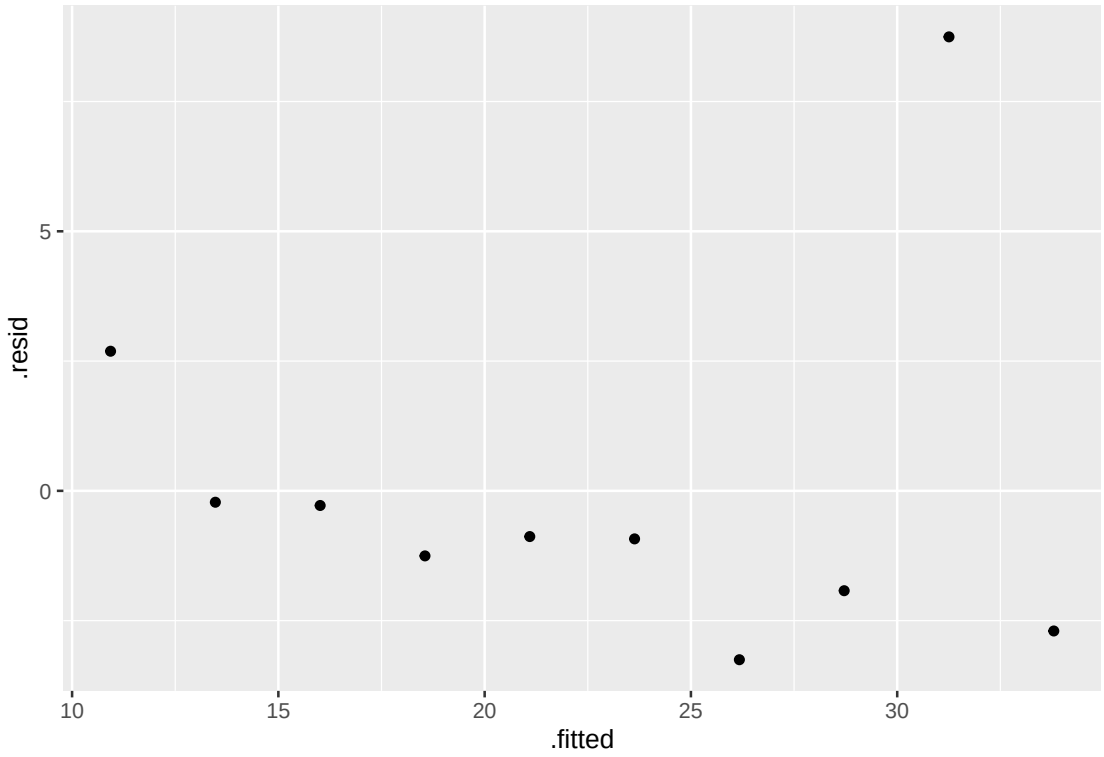
```
ggplot(madeup, aes(x=x, y=y)) + geom_point() + geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



The second-last point is off a clearly linear trend otherwise (the smooth gets "distracted" by the outlying off-trend point). Fitting a regression anyway and looking at the residual plot gives this:

```
madeup.1 <- lm(y~x, data = madeup)
ggplot(madeup.1, aes(x = .fitted, y = .resid)) + geom_point()
```



This time you see a rather more obvious downward trend in the other residuals. The problem is not with them, but with the one very positive residual, corresponding to the outlier that is way off the trend on the scatterplot.

The residuals in a regression have to add up to zero. If one of them is very positive (as in the one you did and the example I just showed you), at least some of the other residuals have to become more negative to compensate – the ones on the right just above and the ones on the left in the one you did. If you have done STAC67, you will have some kind of sense of why that is: think about the two equations you have to solve to get the estimates of intercept and slope, and how they are related to the residuals. Slide 6 of this shows them; at the least squares estimates, these two partial derivatives both have to be zero, and the things inside the brackets are the residuals.

(f) Any data points that concerned you earlier were actually errors. Create and save a new data frame that does not contain any of those data points.

Solution:

Find a way to not pick that outlying point. For example, you can choose the observations with height less than 80:

```
hf %>% filter(height<80) -> hfx
hfx
```

```
## # A tibble: 32 x 2
##    height  foot
##     <dbl> <dbl>
##  1   66.5  27
```

```
##  2   73.5  29
##  3   70    25.5
##  4   71    27.9
##  5   73    27
##  6   71    26
##  7   71    29
##  8   69.5  27
##  9   73    29
## 10   71    27
## # ... with 22 more rows
```

Only 32 rows left.

There are many other possibilities. Find one.

(g) Run a regression predicting height from foot length for your data set without errors. Obtain a plot of the residuals against fitted values and a normal quantile plot of the residuals for this regression.

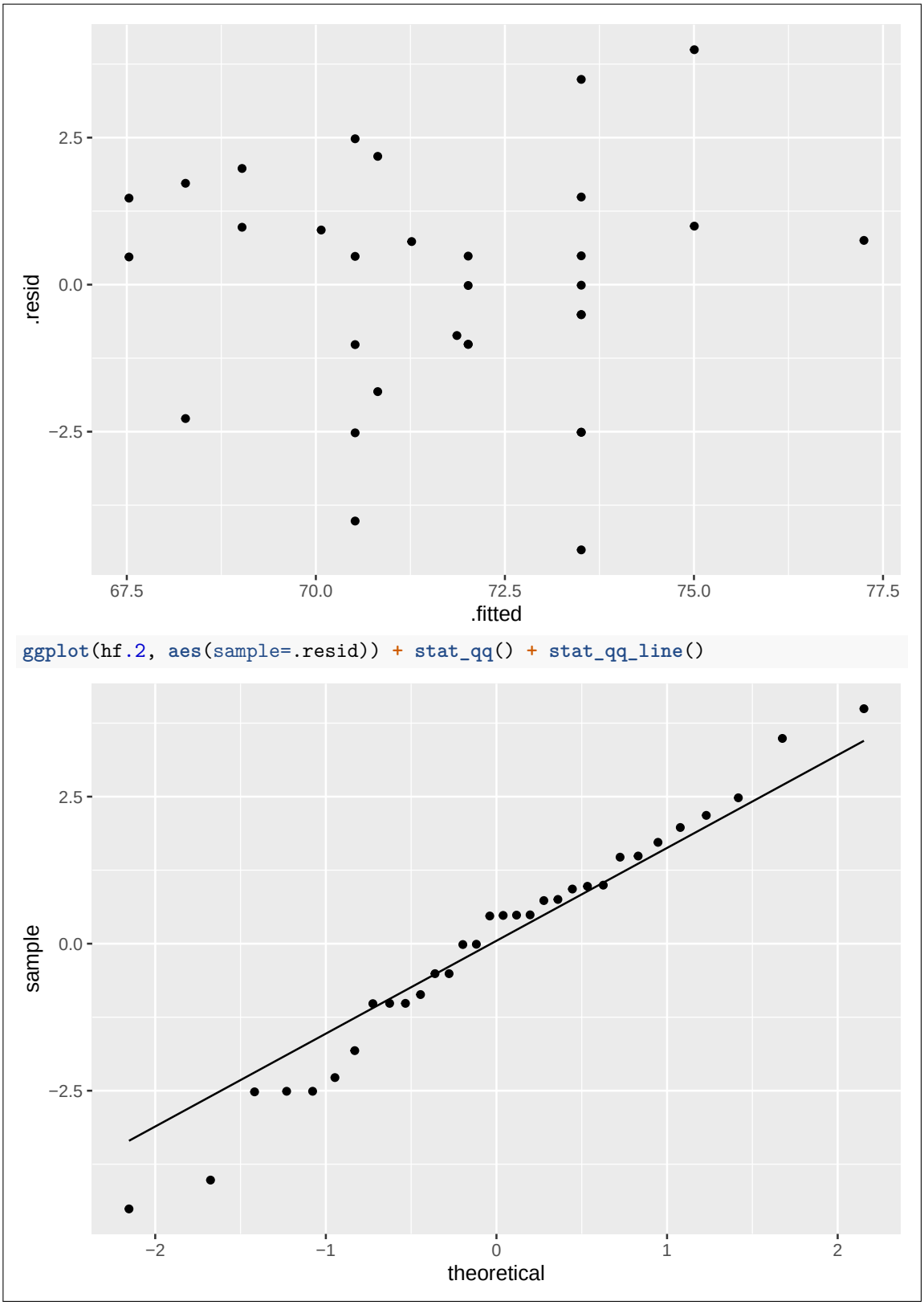
Solution:

Code-wise, the same as before, but with the new data set:

```
hf.2 <- lm(height~foot, data=hfx)
summary(hf.2)
```

```
##
## Call:
## lm(formula = height ~ foot, data = hfx)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5097 -1.0158  0.4757  1.1141  3.9951
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  30.1502     6.5411   4.609 7.00e-05 ***
## foot          1.4952     0.2351   6.360 5.12e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.029 on 30 degrees of freedom
## Multiple R-squared:  0.5741, Adjusted R-squared:  0.5599
## F-statistic: 40.45 on 1 and 30 DF,  p-value: 5.124e-07
```

```
ggplot(hf.2, aes(x=.fitted, y=.resid)) + geom_point()
```




```
ggplot(hf.2, aes(sample=.resid)) + stat_qq() + stat_qq_line()
```


(h) Do you see any problems on the plots you drew in the previous part? Explain briefly.

Solution:

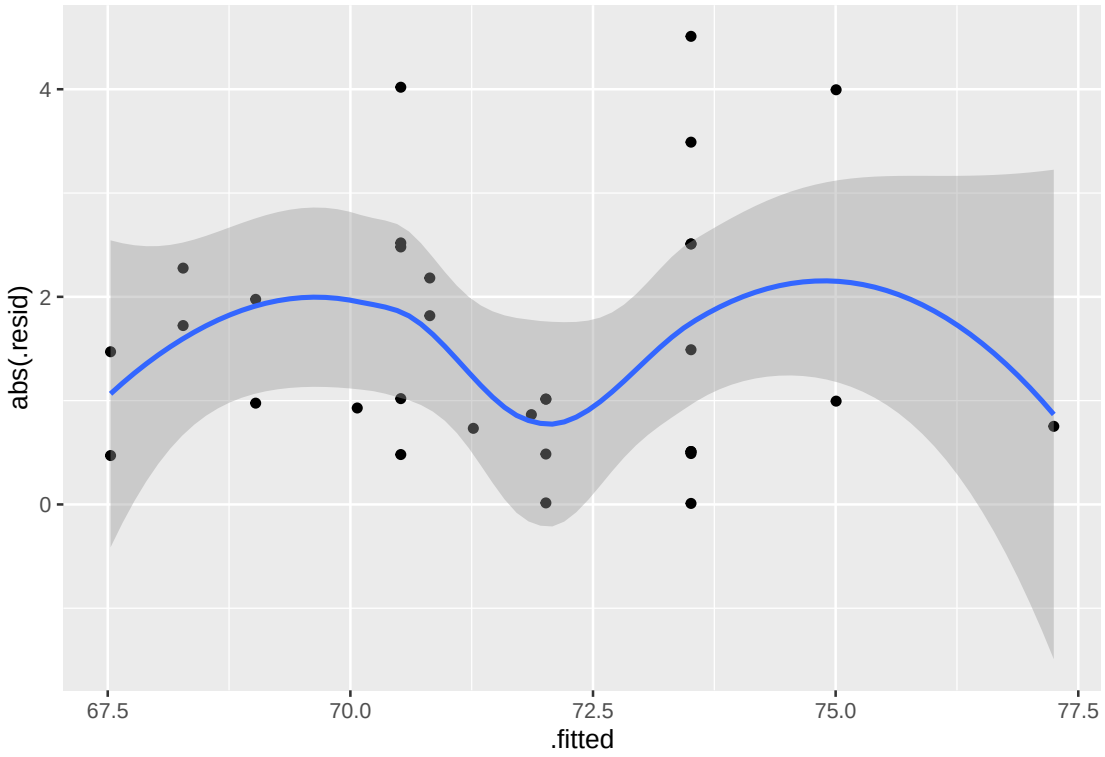
For myself, I see a random scatter of points on the first plot, and points close to the line on the second one. Thus I don't see any problems at all. I would declare myself happy with the second regression, after removing the outlier. (Remember that we removed the outlier *because it was an error*, not just because it was an outlier. Outliers can be perfectly correct data points, and if they are, they have to be included in the modelling.)

You might have a different point of view, in which case you need to make the case for it. You might see a (very mild) case of fanning out in the first plot, or the two most negative residuals might be a bit too low. These are not really outliers, though, not at least in comparison to what we had before.

Extra: a standard diagnostic for fanning-out is to plot the *absolute values* of the residuals against the fitted values, with a smooth trend. If this looks like an increasing trend, there is fanning-out; a decreasing trend shows fanning-in. The idea is that we want to see whether the residuals are changing in *size* (for example, getting more positive *and* more negative both):

```
ggplot(hf.2, aes(x=.fitted, y=abs(.resid))) + geom_point() + geom_smooth()
```

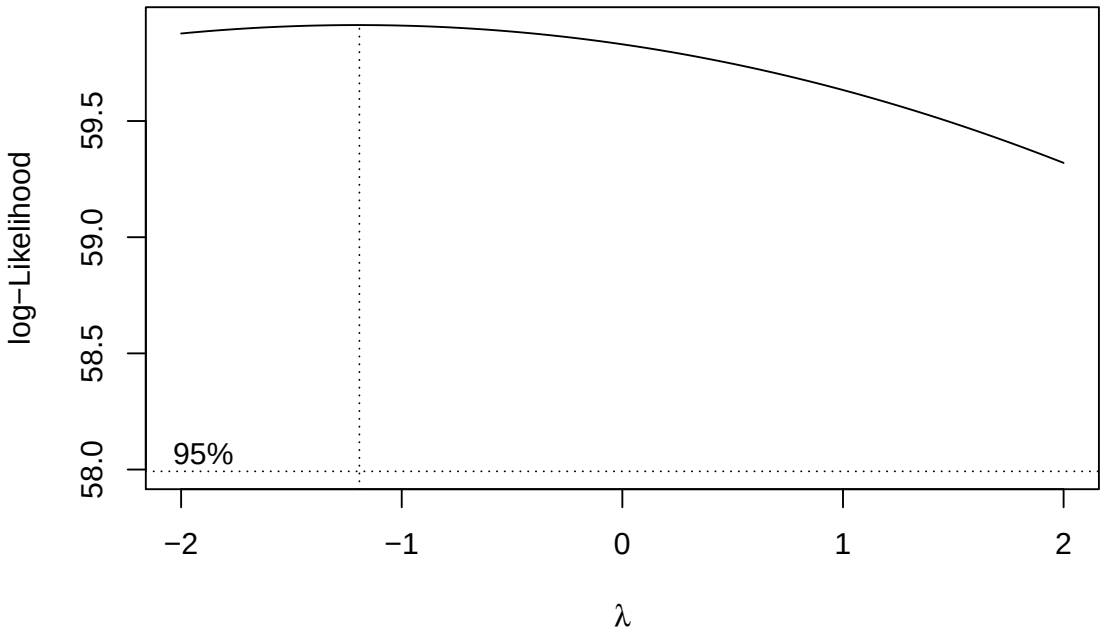
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



No evidence of fanning-out at all. In fact, the residuals seem to be smallest in size *in the middle*.

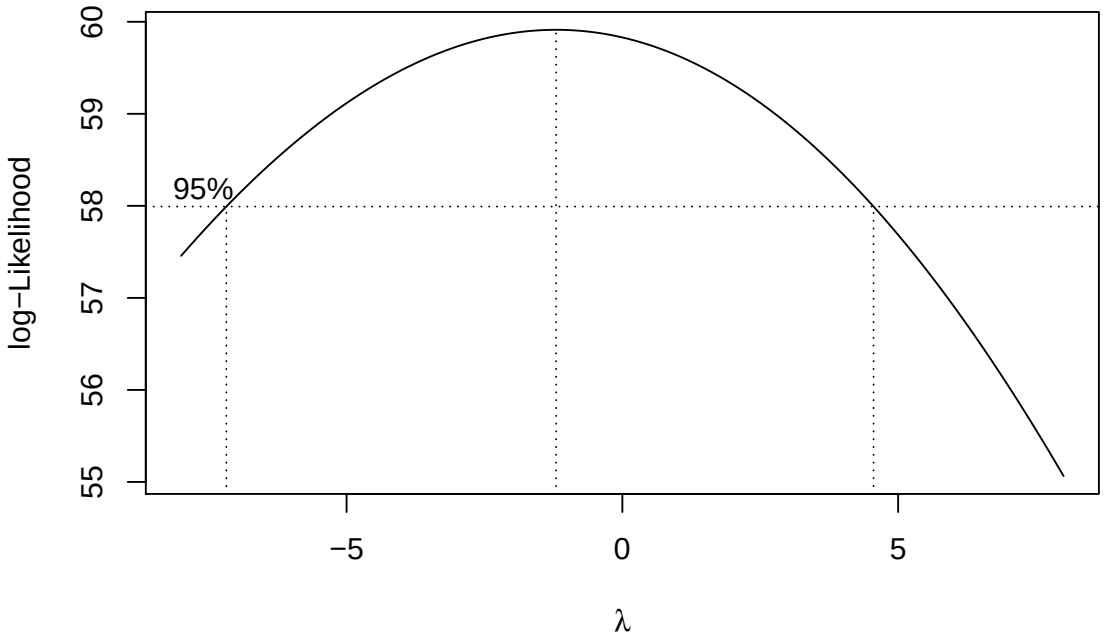
Another thing you might think of is to try Box-Cox:

```
boxcox(height~foot, data=hfx)
```



It looks as if the best λ is -1, and we should predict one over height from foot length. But this plot is deceiving, since it doesn't even show the whole confidence interval for λ! We should zoom out (a lot) more:

```
boxcox(height~foot, data=hfx, lambda = seq(-8, 8, 0.1))
```



This shows that the confidence interval for λ goes from about -7 to almost 5: that is, *any* value of λ in that interval is supported by the data! This very definitely includes the do-nothing $\lambda = 1$, so there is really no support for any kind of transformation.

(i) Find a way to plot the data and *both* regression lines on the same plot, in such a way that you can

see which regression line is which. If you get help from anything outside the course materials, cite your source(s).

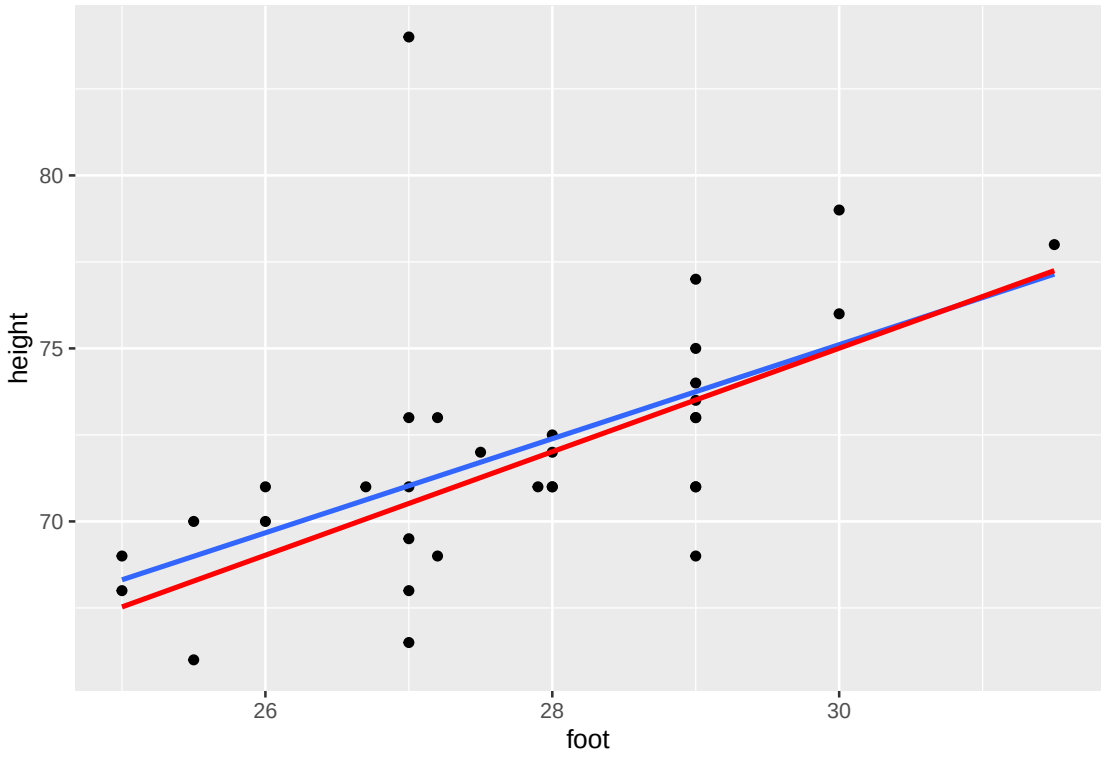
Solution:

This is the same idea as with the windmill data, page 22, though this one is a bit easier since everything is linear (no curves).

The easiest way is to use `geom_smooth` twice, once with the original data set, and then on the one with the outlier removed:

```
ggplot(hf, aes(y=height, x=foot)) + geom_point() + geom_smooth(method = "lm", se=F) +
  geom_smooth(data=hfx, method="lm", colour="red", se=F)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```

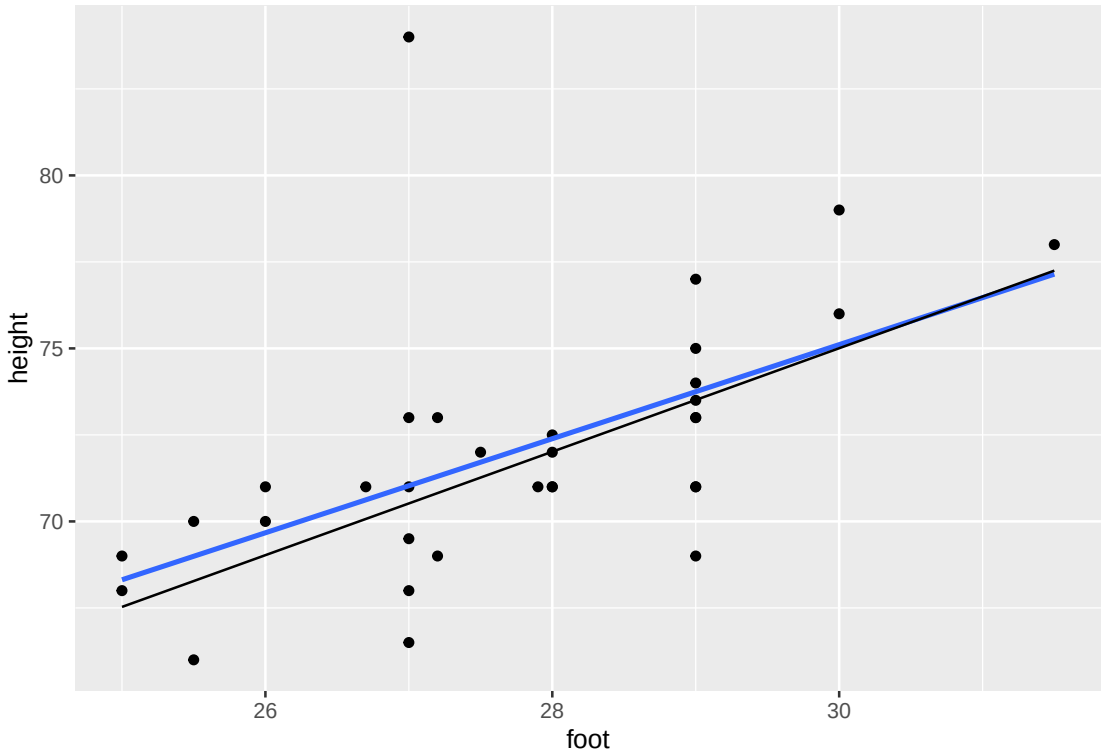


I would use the original data set as the "base", since we want to plot its points (including the outlier) as well as its line. Then we want to plot just the line for the second data set. This entails using a `data=` in the second `geom_smooth`, to say that we want to get *this* regression line from a different data set, and also entails drawing this line in a different colour (or in some way distinguishing it from the first one). Putting the `colour` *outside* an `aes` is a way to make the *whole line* red. (Compare how you make points different colours according to their value on a third variable.)

This is, I think, the best way to do it. You can mimic the idea that I used for the windmill data:

```
ggplot(hf, aes(y=height, x=foot)) + geom_point() + geom_smooth(method = "lm", se=F) +
  geom_line(data=hf.2, aes(y = .fitted))
```

```
## `geom_smooth()` using formula 'y ~ x'
```



but this is not as good, because you don't need to use the trickery with `geom_line`: the second trend is another regression line not a curve, and we know how to draw regression lines with `geom_smooth` without having to actually fit them. (Doing it this way reveals that you are copying without thinking, instead of wondering whether there is a better way to do it.)

The idea of using a different data set in different "layers" of a plot is quite well-known. For example, the idea is the one in here, though used for a different purpose there (plotting different sets of points instead of different lines).

(j) Discuss briefly how removing the observation(s) that were errors has changed where the regression line goes, and whether that is what you expected.

Solution:

The regression line for the original data (my blue line) is pulled up compared to the one with outliers removed (red).

This is not very surprising, because we know that regression lines tend to get pulled towards outliers. What was surprising to me was that the difference wasn't very big. Even at the low end, where the lines differ the most, the difference in predicted height is only about one inch. Since regression lines are based on means, I would have expected a big outlier to have moved the line a lot more.

Say something about what you expected, and say something insightful about whether that was what you saw.

Extra: the regression lines are very similar, but their R-squared values are not: 32% and 57%

respectively. Having a point far from the line has a big (downward) impact on R-squared.