Assignment 5

1. An auditor is, from here, "a person authorized to review and verify the accuracy of financial records and ensure that companies comply with tax laws". Auditors have to be regularly re-trained on the latest tax laws, to be sure that their work is accurate. In a study, 30 auditors were about to be re-trained. They were randomly assigned to one of three methods to learn the material, learning it at home (online) by themselves, at a local training session with a facilitator, or at a national training session, also with a facilitator. This is in the column `method`. There are three other columns. Each auditor was tested on the material in the training before the training happened, to see how much understanding they had already (`pretest`) and each auditor was also tested again afterwards (`posttest`). The two tests are not directly comparable with each other, but on each test a higher score indicates greater understanding. The remaining column, called `grad`, is a code for how long ago each auditor graduated from university, A being the most recent and J being the least recent (longest ago). We will not be using the `grad` column in this question. The data are in http://ritsokiguess.site/STAD29/auditor.csv.

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

Solution:

Nothing surprising here:

```
my_url <- "http://ritsokiguess.site/STAD29/auditor.csv"
auditor <- read_csv(my_url)
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   posttest = col_double(),
##   grad = col_character(),
##   method = col_character(),
##   pretest = col_double()
## )
```

```
auditor
```

```
## # A tibble: 30 x 4
##    posttest grad  method   pretest
##       <dbl> <chr> <chr>      <dbl>
##  1       73 A     home          93
##  2       81 A     local         98
##  3       92 A     national      91
##  4       76 B     home          94
##  5       78 B     local         93
##  6       89 B     national      94
##  7       75 C     home          89
##  8       76 C     local         91
##  9       87 C     national      92
## 10       74 D     home          86
## # ... with 20 more rows
```

Extra: there is of course a "how I got the data to this point" story again. This is how it looked originally:

```
my_url <- "http://ritsokiguess.site/STAD29/auditor.txt"
auditor0 <- read_table(my_url)
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   posttest = col_double(),
##   grad = col_double(),
##   method = col_double(),
##   pretest = col_double()
## )
```

```
auditor0
```

```
## # A tibble: 30 x 4
##    posttest  grad method pretest
##       <dbl> <dbl>  <dbl>   <dbl>
##  1       73     1      1      93
##  2       81     1      2      98
##  3       92     1      3      91
##  4       76     2      1      94
##  5       78     2      2      93
##  6       89     2      3      94
##  7       75     3      1      89
##  8       76     3      2      91
##  9       87     3      3      92
## 10       74     4      1      86
## # ... with 20 more rows
```

In `grad` and `method`, numeric codes were used, with a description elsewhere of what those codes mean. This is easier for data entry (getting the data from the experimenter's notes into a spreadsheet, say), but it is less easy for the statistician because those number codes will come out in graphs and analysis, instead of what those codes actually mean. It is also more difficult for the reader of the final report to understand. In addition, *you* have a harder time, because those two "numeric" variables `method` and `grad` are actually categorical rather than quantitative (the numbers have no meaning except as labels). I like my categorical variables to look categorical. Let's make them so.

For `method`, this would work perfectly well with `case_when` (try it and see). But I wanted to show you another way, "recoding categories":

```
auditor0 %>%
  mutate(method=factor(method)) %>%
  mutate(method=fct_recode(method,
                           home="1",
                           local="2",
                           national="3"))
```

```
## # A tibble: 30 x 4
##    posttest  grad method   pretest
##       <dbl> <dbl> <fct>      <dbl>
##  1       73     1 home          93
```

```
##  2       81     1 local         98
##  3       92     1 national      91
##  4       76     2 home          94
##  5       78     2 local         93
##  6       89     2 national      94
##  7       75     3 home          89
##  8       76     3 local         91
##  9       87     3 national      92
## 10       74     4 home          86
## # ... with 20 more rows
```

`fct_recode` comes from `forcats` (part of the `tidyverse`). It takes a genuine categorical `factor` and changes some or all of its levels. So the first thing to make is an actual `factor`, and, having done that, we can change the numbers into more descriptive text. It's up to you whether you like this better than using `case_when`; this way is a little bit less code, once you remember how it works (you don't have to keep saying `method==`, for example). I have to stop and think about how this goes; it's new name (no quotes) equals old name (in quotes), which, if you are like me, might seem backwards. `rename`, for giving whole columns new names, works the same way: new name equals old name.

`grad` is categorical as well, not numerical, so I decided to use letters for this one. R has a built-in vector called `LETTERS`:

```
LETTERS
```

```
##  [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J" "K" "L" "M" "N" "O" "P" "Q" "R" "S"
## [20] "T" "U" "V" "W" "X" "Y" "Z"
```

and correspondingly

```
letters
```

```
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o" "p" "q" "r" "s"
## [20] "t" "u" "v" "w" "x" "y" "z"
```

If you want a particular letter, say the sixth one, use square brackets:

```
LETTERS[6]
```

```
## [1] "F"
```

So in `grad`, I turned the consecutive numbers into consecutive letters. Here's the whole process:

```
auditor0 %>%
  mutate(method=factor(method)) %>%
  mutate(method=fct_recode(method,
                           home="1",
                           local="2",
                           national="3")) %>%
  mutate(grad=LETTERS[grad]) -> auditor
auditor
```

```
## # A tibble: 30 x 4
##    posttest grad  method   pretest
##       <dbl> <chr> <fct>      <dbl>
##  1       73 A     home          93
##  2       81 A     local         98
```

```
##  3       92 A     national      91
##  4       76 B     home          94
##  5       78 B     local         93
##  6       89 B     national      94
##  7       75 C     home          89
##  8       76 C     local         91
##  9       87 C     national      92
## 10       74 D     home          86
## # ... with 20 more rows
```

This I then saved in a `csv` for you.

(b) Make a suitable graph of the three variables of interest. Add suitable regression lines if appropriate.

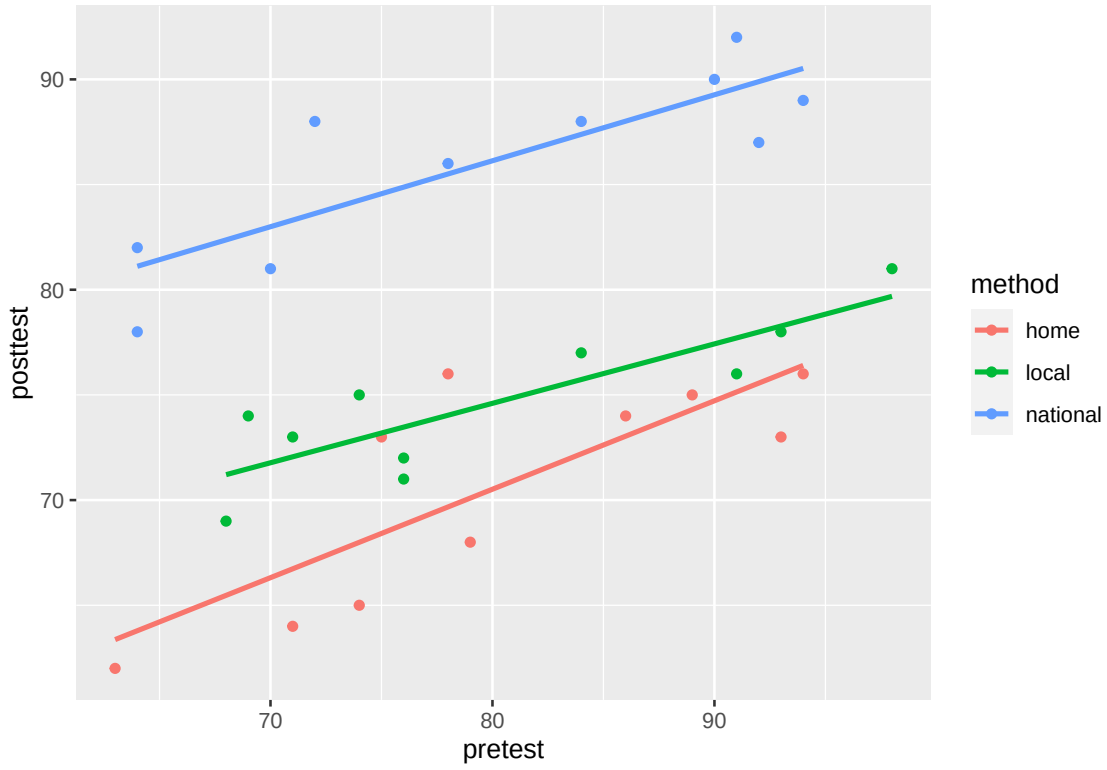
Solution:

I would do this by thinking about the kind of graph that would make sense, then see what happens if I add regression lines to it, then see if I like that.

There are two quantitative variables (the two test scores) and one categorical variable (`method`), remembering that we are not thinking about `grad` at the moment. Two quantitative variables and one categorical suggests a scatterplot with the levels of the categorical variable distinguished by colour. In principle at least, adding a regression to a scatter plot makes sense, so let's try it and see what happens:

```
ggplot(auditor, aes(x=pretest, y=posttest, color=method)) +
  geom_point() + geom_smooth(method="lm", se=FALSE)
```

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



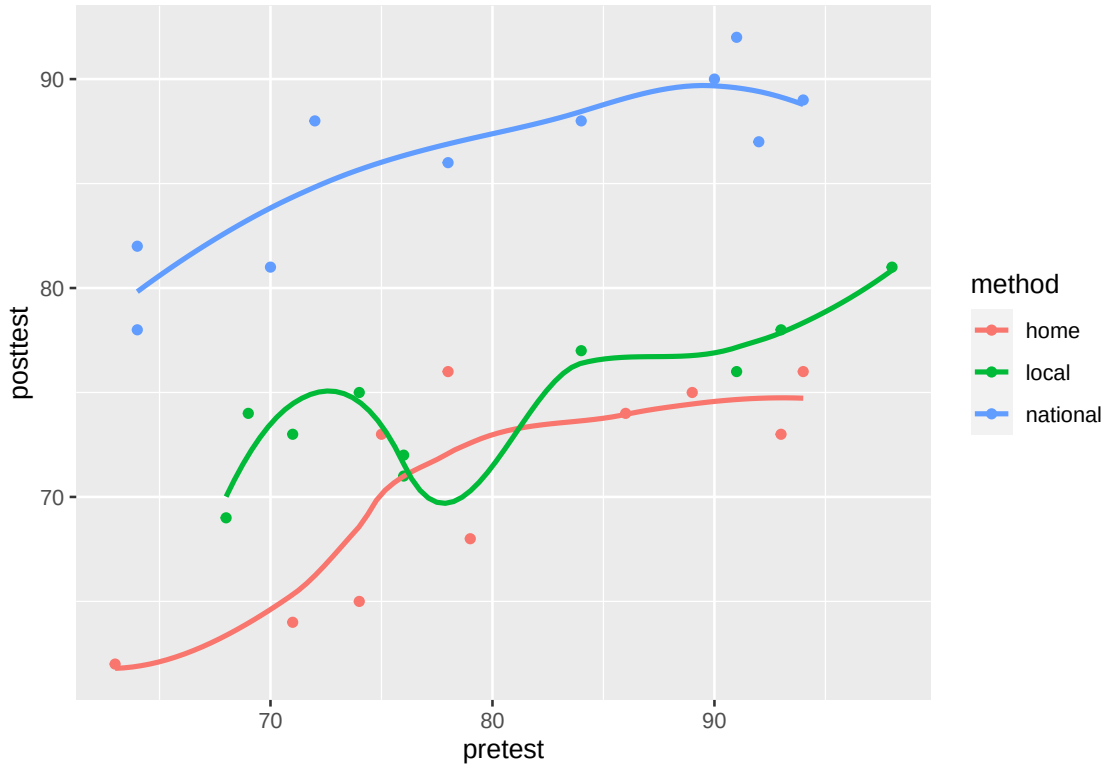
The clouds of red, green, and blue points look more or less linear, so I think the regression lines make sense. There are three lines, one for each method, because your `ggplot` had a `colour` in it, and so anything else you plot, points or lines, respects those colours.

`posttest` is the response, so that has to go on the y-axis.[1]

Extra 1: if you want to see what not-necessarily-linear smooths look like, take out the `method="lm"`:

```
ggplot(auditor, aes(x=pretest, y=posttest, color=method)) +
  geom_point() + geom_smooth(se=FALSE)
```

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

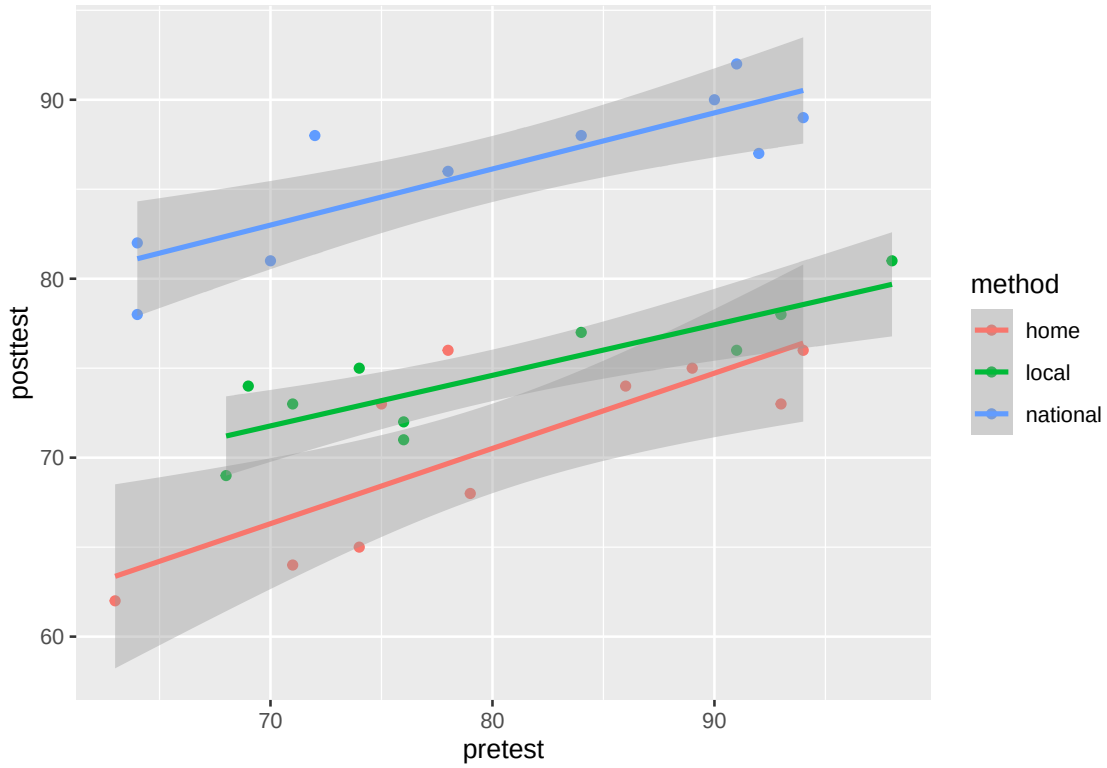


The green curve has a wiggle in it (you can decide whether you think that's more than chance), but the red and blue trends don't look far from linear. I think the green wiggle is mainly driven by those two observations with pretest score of 76 that have a posttest score lower than you would have guessed. There isn't much else to suggest a curve.

Also, make a call as to whether you want those grey envelopes. Here's how it looks with them on:

```
ggplot(auditor, aes(x=pretest, y=posttest, color=method)) +
  geom_point() + geom_smooth(method="lm")
```

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



This gives an idea about likely significance, so you might find it useful to keep the envelopes.

Extra 2: I think regression lines are all right here, because I took a look at the plot without them and decided that the coloured clouds of points were more or less straight. Another way you could think through this is to first add the non-linear smooths, on the basis that you don't know yet what kind of trends you have, then decide that they are not that far off straight (something I rationalized for the green one as well), then draw the regression lines. Starting with regression lines is not really the best way to go until you know that the trends *are* more or less straight. I simplified things for you a bit in this question to keep it shorter.

(c) Discuss briefly what your graph says about the main effects of and the interaction between your explanatory variables, as they relate to the response variable.

Solution:

Think first about the effects of your explanatory variables singly:

- The three regression lines are going uphill, so as pretest score increases, posttest score increases, and this is true for each of the three methods. (This is not really much of a surprise; an auditor who does better before the training has a better baseline knowledge and would be expected to master the new information in the training more readily.)
- the blue line is higher than the other two lines, for any pretest score. This means that the national training is the most effective. For two auditors whose pretest score was the same, the auditor doing the national training would be expected to have a higher posttest score than the auditor[2] doing the local or the at-home training. Whether you think the other two forms of training are likely to be significantly different is up to you (say what

you think and how you know). If you put the grey envelopes on your plot, the local and home ones overlap a little, but maybe they will still be different. (You might expect that the national training would be bigger, and the facilitator might be better than the one at a local training.)

Then, the interaction. This is a matter of looking at the slopes of your lines.

- For me, the red and blue lines have about the same slope, and the issue is whether you think the green line is going to be significantly less steep than the others. My take is that the difference in slopes is not enough to be significant, because the points are somewhat variable about the lines. Feel free to disagree, but you will need to use words like "important" or "meaningful" to describe the difference in slopes: even if the underlying population slopes are equal, the sample slopes (the ones you see here) will be different, at least a little, just by chance. Whatever you conclude, say something that indicates that you have thought about this issue.

A meaningful interaction means that the effect of a one-point difference in pretest score on posttest score is *different* depending on what the training method was. In this case, you might say that a one-point increase in pretest score is *less* important, has less impact on the posttest score, for auditors that did the local training, because the slope is a bit less.

(d) Develop an appropriate analysis of covariance model for these data. This may mean fitting one model, modifying it, and then fitting a second model. Display the output for at least your final model.

Solution:

The place to start is to fit the model with interaction first, and then see where to go from there:

```
auditor.1 <- lm(posttest~pretest*method, data = auditor)
```

Here you need to be careful. It is tempting to do this:

```
summary(auditor.1)
```

```
##
## Call:
## lm(formula = posttest ~ pretest * method, data = auditor)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.1146 -2.0706  0.1112  1.1927  6.3250
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)              36.88034    7.23221   5.099 3.23e-05 ***
## pretest                   0.42044    0.08954   4.696 9.01e-05 ***
## methodlocal              15.11430    9.90658   1.526   0.1402
## methodnational           24.16708    9.50828   2.542   0.0179 *
## pretest:methodlocal      -0.13788    0.12274  -1.123   0.2724
## pretest:methodnational   -0.10690    0.11777  -0.908   0.3731
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 2.712 on 24 degrees of freedom
## Multiple R-squared:  0.9041, Adjusted R-squared:  0.8841
## F-statistic: 45.24 on 5 and 24 DF,  p-value: 1.915e-11
```

There are two interaction terms here (because there are three `methods` and one of them is the baseline), and this doesn't tell you whether the interaction term *as a whole* is significant. For that, you need `drop1`, doing an F-test because we are in standard linear-model territory here:

```
drop1(auditor.1, test = "F")
```

```
## Single term deletions
##
## Model:
## posttest ~ pretest * method
##                Df Sum of Sq    RSS    AIC F value Pr(>F)
## <none>                      176.56 65.175
## pretest:method  2    10.144 186.71 62.851  0.6894 0.5115
```

The interaction term is nowhere near significant, so it can be removed. (*Say this*, rather than going straight ahead and fitting the model without it. I want to be clear that you know what you are doing.) This means that those lines were actually *not* significantly different from being parallel.

If you don't think of this, fit a model without the interaction here, instead of just below, and use `anova` to compare the two models:

```
auditor.2 <- lm(posttest~pretest+method, data = auditor)
anova(auditor.2, auditor.1)
```

```
## Analysis of Variance Table
##
## Model 1: posttest ~ pretest + method
## Model 2: posttest ~ pretest * method
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     26 186.71
## 2     24 176.56  2    10.144 0.6894 0.5115
```

The test and conclusion is exactly the same.

The next step is `update` or copy-and-paste, as you prefer:

```
auditor.2 <- lm(posttest~pretest+method, data = auditor)
drop1(auditor.2, test="F")
```

```
## Single term deletions
##
## Model:
## posttest ~ pretest + method
##         Df Sum of Sq     RSS     AIC F value    Pr(>F)
## <none>                186.71  62.851
## pretest  1    358.99  545.70  93.026  49.991 1.653e-07 ***
## method   2   1309.59 1496.30 121.287  91.183 1.778e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The main effects are both significant, so here we stop. If you want to use `anova` again, you can,

but it's more work (and thus not the best). To do that, fit models without each of `pretest` and `method`, and compare each of them with what I called `auditor.2`.

The last step is to display the `summary`:

```
summary(auditor.2)
```

```
##
## Call:
## lm(formula = posttest ~ pretest + method, data = auditor)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.5294 -2.1181  0.4576  1.4472  6.1347
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    43.81517    3.88192  11.287 1.61e-11 ***
## pretest         0.33398    0.04724   7.070 1.65e-07 ***
## methodlocal     4.06680    1.19847   3.393  0.00222 **
## methodnational 15.60019    1.19851  13.016 6.77e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.68 on 26 degrees of freedom
## Multiple R-squared:  0.8986, Adjusted R-squared:  0.8869
## F-statistic: 76.77 on 3 and 26 DF,  p-value: 4.792e-13
```

(e) Explain briefly how the values in the Estimate column of the output from your best model correspond to your plot (or do not, if you think they do not).

Solution:

In the Estimate column:

- the slope for `pretest` is 0.33, positive. This is an ordinary slope, so it means that, holding `method` constant, as the pretest score increases, the posttest score also increases. This is consistent with the upward-sloping lines on the graph.
- the Estimate for `methodlocal` is 4.07. This means that the mean posttest score for someone who did the local training is about 4 points greater than someone who had the same `pretest` score, but who did the at-home training instead. This is consistent with the `local` line on the graph being slightly above the at-home line.
- the Estimate for `methodnational` is 15.60. This means that the mean posttest score for someone who did the national training is about 16 points greater than someone who had the same `pretest` score, but who did the at-home training instead. This is consistent with the `national` line on the graph being far above the at-home line.

You need to say something about what each estimate means, and then relate that to your graph (the lines are going uphill, or this line is above that one, as appropriate).

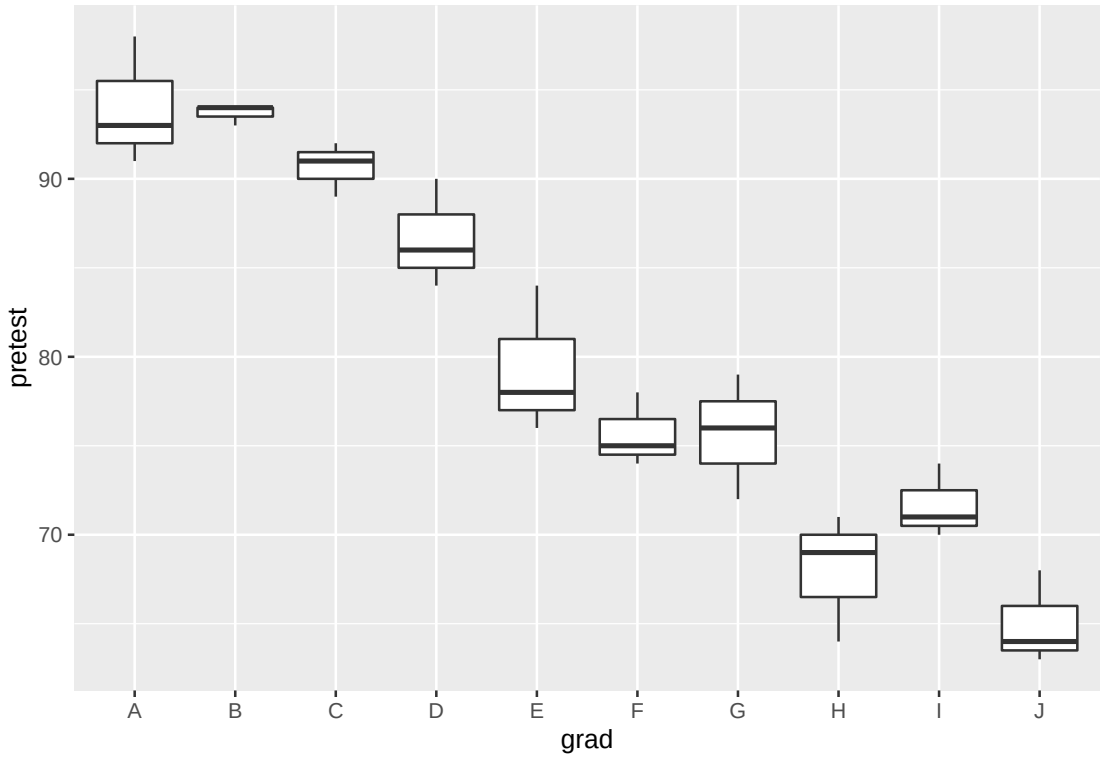
A rather long and rambling Extra:[3] I originally allowed you the choice to talk about P-values instead of Estimates, but that didn't survive the editing process: I decided to make it clearer

what I wanted you to do. The interesting conclusion there is that the at-home post-test scores are significantly lower than both of the other two learning methods, even though the red and green lines on my plot didn't look all that far apart.

In our dataframe, we also had another explanatory variable `grad`. How is it associated with `pretest`?

The other explanatory variable `grad` is categorical, so the right graph with the quantitative `pretest` is a side-by-side boxplot:

```
ggplot(auditor, aes(x=grad, y=pretest)) + geom_boxplot()
```



There is a clear downward trend with the boxplots: as graduation from university was longer ago (a later letter in the alphabet), the pretest score was lower. This suggests that `grad` and `pretest` are really telling the same story, and that having either one of them in the model (for predicting `posttest`) ought to be about equally good. (This we come back to later.) But, having them both in the model is probably going to be a waste of time, since there is really no information in `grad` that was not also in `pretest`.

There is exactly one observation per `grad-method` combination (by design). If you ignore the `pretest`, what you have is a *randomized blocks design*, that you may have met in B27 or C53, shortly before you got to two-way ANOVA for the first time. Because there is only one observation per combination, there is not enough data to estimate and test an interaction (as we see shortly), but there is still enough to estimate the main effects. In a randomized block design, we *assume* that there is no interaction, because there is no way to test whether there is one or not. (If you want to do *that*, you need replicate observations,[4] for example two auditors for each `grad-method` combination.)

Next, we fit a model, using `lm`, predicting the `posttest` score from the training `method` and how long ago the auditors' university graduation was, and display the results. Is it better to

use grad or pretest as an explanatory variable, or does it not matter much which one we use?

The model-fitting part is straightforward, just this:

```
auditor.4 <- lm(posttest~grad+method, data = auditor)
summary(auditor.4)
```

```
##
## Call:
## lm(formula = posttest ~ grad + method, data = auditor)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -4.833 -1.125 -0.500  1.500  4.167
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       75.500      1.580  47.786  < 2e-16 ***
## gradB             -1.000      2.040  -0.490 0.629872
## gradC             -2.667      2.040  -1.307 0.207544
## gradD             -1.667      2.040  -0.817 0.424554
## gradE             -3.667      2.040  -1.798 0.089033 .
## gradF             -4.000      2.040  -1.961 0.065533 .
## gradG             -6.000      2.040  -2.942 0.008725 **
## gradH             -8.667      2.040  -4.249 0.000483 ***
## gradI             -9.000      2.040  -4.412 0.000336 ***
## gradJ            -12.333      2.040  -6.047 1.02e-05 ***
## methodlocal        4.000      1.117   3.580 0.002139 **
## methodnational    15.500      1.117  13.874 4.72e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.498 on 18 degrees of freedom
## Multiple R-squared:  0.939,  Adjusted R-squared:  0.9017
## F-statistic: 25.18 on 11 and 18 DF,  p-value: 1.107e-08
```

To decide whether to prefer this model over the one with pretest, look at how well the two models fit. They have a different number of parameters, so the best answer is to compare the *adjusted* R-squared (and to say why you are doing so). Here, the adjusted R-squared is 90%, and in the model with pretest it is 88.7%. It is also reasonable, though not quite as good, to compare the ordinary R-squared values, 93.9% here and 89.9% before.

Then make a call about whether the two values you looked at are different enough to be worth calling different. My take is that they are very similar, especially the adjusted R-squared values, which are really the best ones to be comparing, so that it doesn't really matter whether you use grad or pretest as an explanatory variable. Either is good.

I don't want you to get too hung up on small differences in R-squared values, so if you want to prefer the model with grad in it, you will need to assert that its (adjusted or not) R-squared value is "much" higher, or "clearly" higher, or a word like that. That would be a fair reason for preferring one model over the other; it's not one that I agree with here, but if that's what you think, your logic is sound. (It's not an ANCOVA then, which is why I had you go the other way in the question up to this point.)

You might have tried to include an interaction term, thus:

```
auditor.3 <- lm(posttest~grad*method, data = auditor)
drop1(auditor.3, test="F")
```

```
## Warning: attempting model selection on an essentially perfect fit is nonsense
```

```
## Single term deletions
##
## Model:
## posttest ~ grad * method
##             Df Sum of Sq    RSS  AIC F value Pr(>F)
## <none>                     0.00 -Inf
## grad:method 18    112.33 112.33   64
```

"The model fits perfectly" is a warning sign in Statistics, because statistical models might fit very well, but they don't fit perfectly. This is a sign that we have tried to fit a more complicated model than the data can support:

```
anova(auditor.3)
```

```
## Warning in anova.lm(auditor.3): ANOVA F-tests on an essentially perfect fit are
## unreliable
```

```
## Analysis of Variance Table
##
## Response: posttest
##             Df  Sum Sq Mean Sq F value Pr(>F)
## grad         9  433.37   48.15
## method       2 1295.00  647.50
## grad:method 18  112.33    6.24
## Residuals    0    0.00
```

There are *no* degrees of freedom for error! This is a randomized block design, so we can't estimate interactions. We have to be content with a simpler model, the one with main effects only.

What happens if we include both explanatory variables? `grad` and `pretest` are not really correlated explanatory variables, since one of them is categorical, but they certainly have something to do with each other, so you would be right to expect trouble:

```
auditor.5 <- lm(posttest~pretest+grad+method, data=auditor)
drop1(auditor.5, test="F")
```

```
## Single term deletions
##
## Model:
## posttest ~ pretest + grad + method
##         Df Sum of Sq     RSS     AIC F value    Pr(>F)
## <none>                112.33  65.608
## pretest  1      0.00  112.33  63.608  0.0001    0.9921
## grad     9     74.38  186.71  62.851  1.2507    0.3298
## method   2   1292.18 1404.52 137.388 97.7771 4.735e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Neither `pretest` nor `grad` is significant, and as in regression with correlated x-variables, the solution is to remove one of them (it doesn't really matter which one), and then the one that's left is significant.

2. Obsessive-compulsive disorder (OCD) is an anxiety disorder in which people have recurring, unwanted thoughts, ideas or sensations (obsessions) that make them feel driven to do something repetitively (compulsions). The repetitive behaviors, such as hand washing, checking on things or cleaning, can significantly interfere with a person's daily activities and social interactions. (This description from here.)

   A common treatment for OCD is cognitive behavioral therapy (CBT). In the case of OCD, this can consist of exposure to situations that lead to obsessive behaviours (under the guidance of a therapist), along with training in new behaviours to use in response to these situations. The aim is that the new behaviours will replace the obsessive ones.

   A study compared CBT, another form of behavioral therapy (labelled BT) and a control (no therapy). Thirty people with OCD were randomly allocated to one of the three treatment groups. At the end of the therapy, each person recorded the number of obsession-related Thoughts and obsession-related Actions they had over a certain time period. (A good therapy will lead to low numbers on both.)

   The data are in https://gaopinghuang0.github.io/assets/Rdata/OCD.dat, with the data values separated by *tabs*. (Hint: `read_tsv`.)

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

Solution:

As per the hint, `read_tsv` will read in values delimited by tabs:

```
my_url <- "https://gaopinghuang0.github.io/assets/Rdata/OCD.dat"
ocd <- read_tsv(my_url)
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Group = col_character(),
##   Actions = col_double(),
##   Thoughts = col_double()
## )
```

```
ocd
```

```
## # A tibble: 30 x 3
##    Group Actions Thoughts
##    <chr>   <dbl>    <dbl>
##  1 CBT         5       14
##  2 CBT         5       11
##  3 CBT         4       16
##  4 CBT         4       13
##  5 CBT         5       12
##  6 CBT         3       14
##  7 CBT         7       12
##  8 CBT         6       15
##  9 CBT         6       16
## 10 CBT         4       11
```

```
## # ... with 20 more rows
```

These are the people who did CBT; scroll down to see the people who did behavioral[5] therapy (BT) and the control group.

For a change, I didn't need to do anything to the data. This is a direct link to somebody else's Github page.

(b) What is it about this data set that makes MANOVA an appropriate method to consider for the analysis?

Solution:

There are two response variables, the number of Actions and the number of Thoughts. These (might) both depend on which treatment group a person was in. (The key thing here is how *many* response variables there are; yes, they need to be quantitative, but the crucial thing is how many of them there are.)

(c) Carry out a suitable MANOVA for these data, displaying the results.

Solution:

Two steps: make a response out of the right columns out of the dataframe, and then run the MANOVA. There are two choices for each of the steps; either choice is fine.

For making the response variable, you need to pick out the `Actions` and `Thoughts` columns from the dataframe (looking out for the Capital Letters) and make a matrix of them, either the `cbind` way:

```
response <- with(ocd, cbind(Actions, Thoughts))
head(response)
```

```
##      Actions Thoughts
## [1,]       5       14
## [2,]       5       11
## [3,]       4       16
## [4,]       4       13
## [5,]       5       12
## [6,]       3       14
```

or the `tidyverse` way:

```
ocd %>% select(Actions, Thoughts) %>% as.matrix() -> response
head(response)
```

```
##      Actions Thoughts
## [1,]       5       14
## [2,]       5       11
## [3,]       4       16
## [4,]       4       13
## [5,]       5       12
## [6,]       3       14
```

Displaying some of `response` is a good idea, but there are 30 observations, so `response` has

30 rows, which is rather a lot to display. Hence using `head` is a good idea. (You could display a few more than six rows if you wanted, by adding a number after the matrix in `head`, which is the number of rows to display.) Remember that a matrix will display all of itself unless you take steps to stop that happening.

From here, run the MANOVA using either the base-R way:

```
ocd.1 <- manova(response ~ Group, data = ocd)
summary(ocd.1)
```

```
##           Df  Pillai approx F num Df den Df  Pr(>F)
## Group      2 0.31845   2.5567      4     54 0.04904 *
## Residuals 27
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

or using the `car` capital-M `Manova` way:

```
ocd.2 <- lm(response ~ Group, data = ocd)
Manova(ocd.2)
```

```
##
## Type II MANOVA Tests: Pillai test statistic
##       Df test stat approx F num Df den Df  Pr(>F)
## Group  2   0.31845   2.5567      4     54 0.04904 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

(d) What do you conclude, in the context of the data?

Solution:

The P-value of 0.049 is (just) less than 0.05, so the treatment has an effect on the number of Thoughts, the number of Actions, or a combination of both. (This is inevitably going to be a bit vague.) Or think about the null hypothesis, which would be that all three treatments have the same mean number of Actions and also they have the same mean number of Thoughts. This is rejected, but all we know at this stage is that the null is false, somehow (so all you can say is to conclude that the null is not true somehow).

Be careful not to go overboard here. All you can really say is that something is going on, but not anything about what it is.

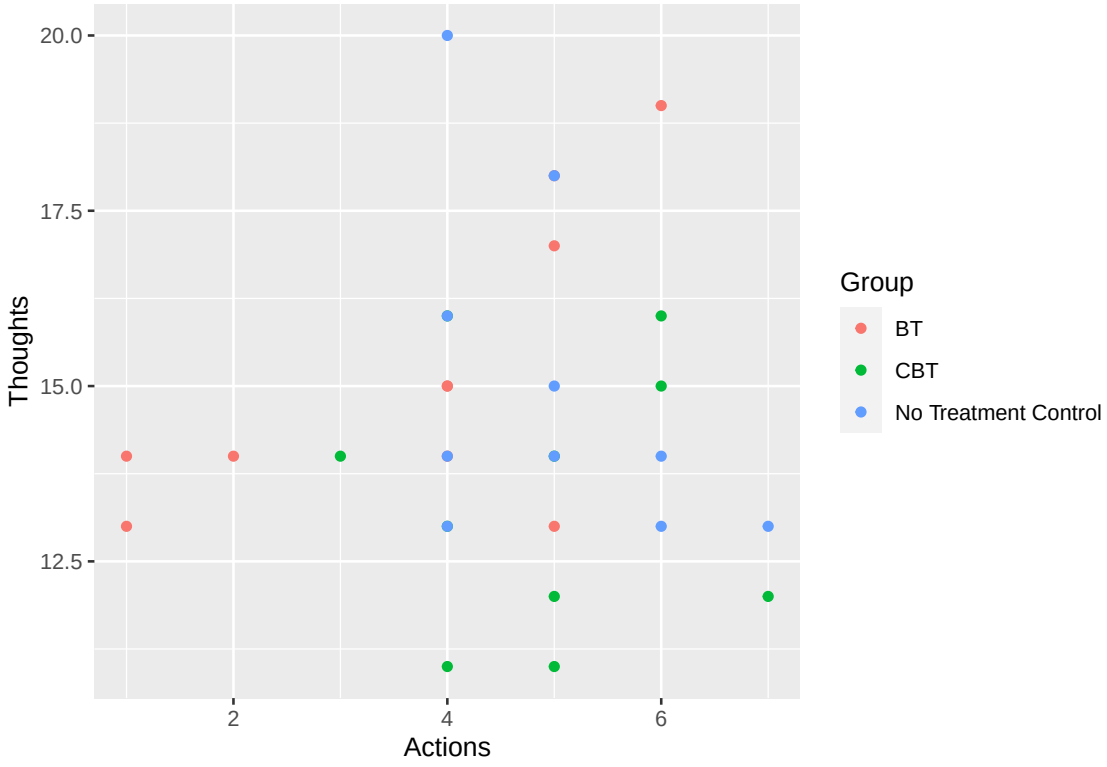
(e) Make one or two suitable graphs of this dataset.

Solution:

I think you have a couple of ways to go here: either you can plot all three variables at once, or you can plot each response against the treatment. Both ways work here, because there are exactly two responses; if there had been three (or more), the first option would not have been on the table.

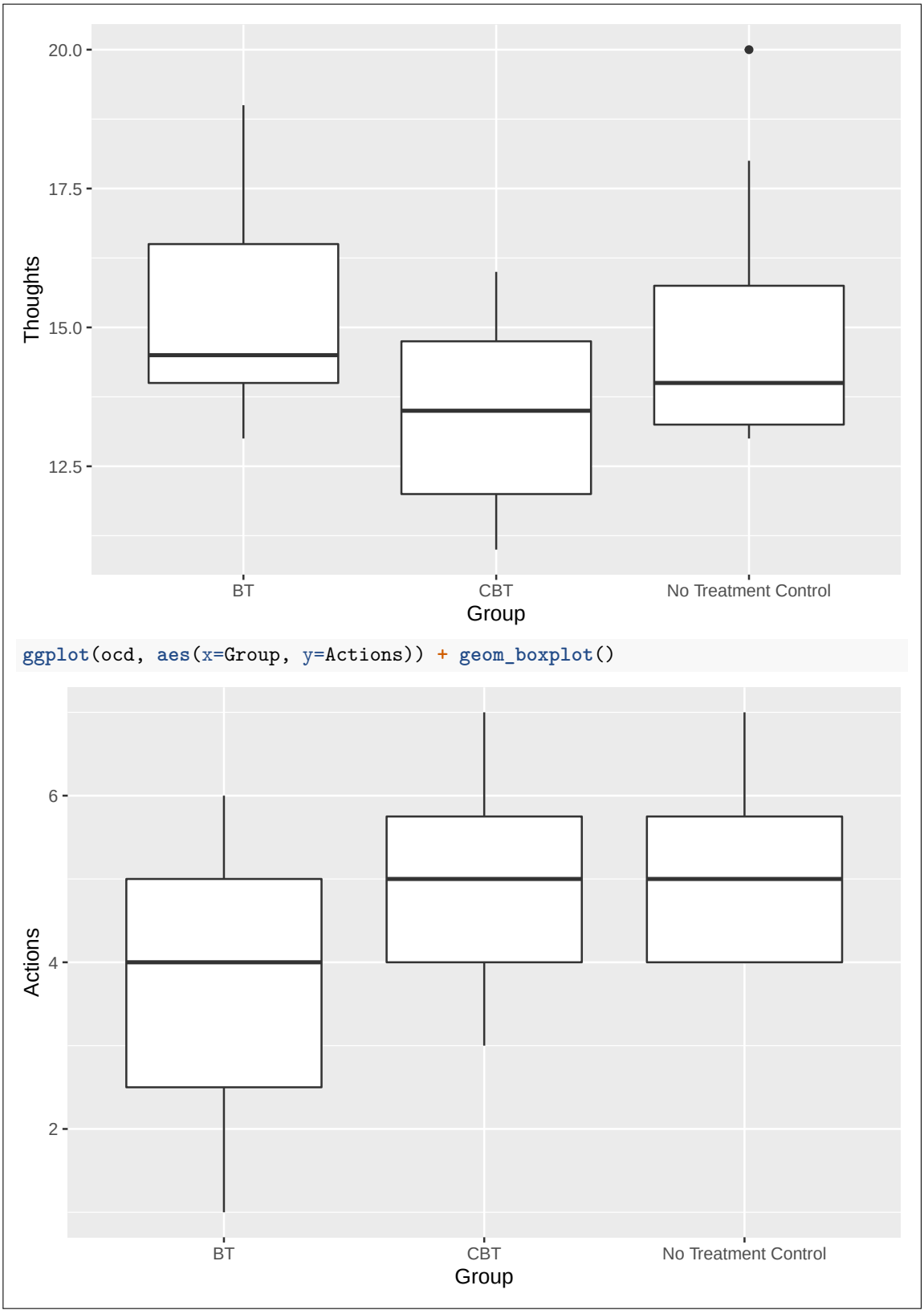
To plot all three variables at once: there are two quantitative variables (the two responses) and one categorical (the treatment), so plot one of the responses against the other one (either way around), labelling the treatment groups by colour:

```
ggplot(ocd, aes(x = Actions, y = Thoughts, colour = Group)) + geom_point()
```



Two graphs, one for each response, would be boxplots, because you then have one quantitative variable (the response) and one categorical (the treatment):

```
ggplot(ocd, aes(x=Group, y=Thoughts)) + geom_boxplot()
```




```
ggplot(ocd, aes(x=Group, y=Actions)) + geom_boxplot()
```


(f) Comment briefly on your graphs. Do they offer any explanation for the significance of your

MANOVA?

Solution:

The scatterplot is hard to say much about. I'll come back to that in a minute.

As for the boxplots:

- The people who had fewest obsessive Thoughts were those in the CBT group (the people who did BT had the most)
- The people who had fewest obsessive Actions were those in the BT group, with the other two groups being the same (in terms of median).

On the scatterplot, the three colours of points look at first glance to be intermingled with not much pattern. But if you look more closely (which you're allowed to do, since this is not a residual plot), you might see that the green dots are slightly more likely to be near the bottom (on my graph) and the red ones are slightly more likely to be towards the left. Decoding that, people on CBT have slightly fewer obsessive Thoughts than the others, and people on BT have slightly fewer obsessive Actions than the others. This is, with more work, the same conclusion that came out of the boxplots.

The null hypothesis for the MANOVA is that all three treatments had the same mean on both Thoughts and Actions. This null hypothesis was (just) rejected, and the reason for that seems to be that CBT is lowest on Thoughts and BT is lowest on Actions. (Those small differences on the boxplots, or the slight arrangement of the coloured points on the scatterplot, are apparently large enough, taken together, to be significant.)

There's nothing wrong with drawing all the graphs for yourself and choosing whatever is easier to interpret. The point of a graph, after all, is that it tells you something about your data.

Extra 1: thinking about OCD, presumably a good treatment is one that goes with a smaller number on both Thoughts and Actions. The story we are getting here, though, is that there isn't a treatment that goes with a small number on *both*; if you want to minimize Thoughts, go with CBT, but if you want to minimize Actions, go with BT. This is something that I'll have you explore on another assignment with a discriminant analysis.

Extra 2: there are about four different tests that the group means are all the same for each of the response variables. This is because there is actually a 2×2 matrix of things to make the test statistic out of,[6] and there are different ways to boil that down into a one-number test statistic. `Anova` (uppercase A) from `car` will show you all of them via its `summary`:

```
summary(Anova(ocd.2))
```

```
##
## Type II MANOVA Tests:
##
## Sum of squares and products for error:
##          Actions Thoughts
## Actions       51       13
## Thoughts      13      122
##
## ------------------------------------------
##
## Term: Group
##
## Sum of squares and products for the hypothesis:
```

```
##            Actions  Thoughts
## Actions  10.466667 -7.533333
## Thoughts -7.533333 19.466667
##
## Multivariate Tests: Group
##                   Df test stat approx F num Df den Df   Pr(>F)
## Pillai             2 0.3184546 2.556658      4     54 0.049037 *
## Wilks              2 0.6985090 2.554546      4     52 0.049665 *
## Hotelling-Lawley   2 0.4073352 2.545845      4     50 0.050798 .
## Roy                2 0.3347974 4.519764      2     27 0.020272 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The four P-values are clustered around 0.05, and it really isn't clear whether we should be rejecting or not. As they say, clear as mud.

Notes

1. There is only one response here. Compare the other question, where you have the same three kinds of variables, two quantitative and one categorical, but they play different roles here: pretest is explanatory here.

2. I keep typing that as "auditer"; I am typing this on a Dvorak keyboard, where o and e are next to each other, and I quite often confuse e (middle finger of left hand) and o (ring finger). Plus, a lot more words end in -er than -or, so this is where my fingers seem to go first. I was a very undisciplined typist on a regular keyboard, so a covid project of mine has been to learn Dvorak, which, I hope, has enabled me to start again and this time to build better habits. In case you are wondering, my keyboard still looks the same, and I ignore the letters on the keys, because most of the time I am not looking at the keys anyway. My daughter is now thoroughly confused by my keyboard!

3. These were originally more parts to the question, but I decided to shorten things up.

4. This means repeat observations taken under the *same* experimental conditions, so that the only reason they differ is random error.

5. Even if you spell "behaviour" with a u, this is the correct spelling of the adjective meaning "related to behaviour". "Labour" and "laborious" works the same way.

6. Because there are two response variables.