University of Toronto Scarborough
Department of Computer and Mathematical Sciences
STAC32 (K. Butler), Final Exam
December 9, 2022

Aids allowed (on paper, no computers):

- My lecture overheads (slides)
- Any notes that you have taken in this course
- Your marked assignments
- My assignment solutions
- Non-programmable, non-communicating calculator

This exam has xxx numbered pages of questions plus this cover page.
In addition, you have an additional booklet of Figures to refer to during the exam.
The maximum marks available for each part of each question are shown next to the question part.
If you need more space, use the last page of the exam. Anything written on the back of the page will not be graded.
You may assume throughout this exam that the code shown in Figure 1 of the booklet of Figures has already been run.
The University of Toronto's Code of Behaviour on Academic Matters applies to all University of Toronto Scarborough students. The Code prohibits all forms of academic dishonesty including, but not limited to, cheating, plagiarism, and the use of unauthorized aids. Students violating the Code may be subject to penalties up to and including suspension or expulsion from the University.

1. The crab species *Leptograpsus variegatus* has two colour forms, blue and orange. Fifty crabs of each colour form and both sexes were collected at Fremantle, Western Australia, and various measurements were taken on each crab. The variables in the data file are as follows:

   - `sp` species (B is blue and O is orange, a letter O)
   - `sex` (male, M or female, F)
   - `index` serial number of each crab within species and sex
   - `FL` frontal lobe size
   - `RW` rear width
   - `CL` carapace length
   - `CW` carapace width
   - `BD` body depth

   All five of the measurements are in millimetres.

   Some of the data file is shown in Figure 2, and the data file is stored in `crabs.txt` in the same folder that R Studio is running in.

   (a) [2] What code would read the data in the file into a dataframe called `crabs` and display it?

My answer:

In Figure 2, the data values are each rather oddly separated by a single colon, so `read_delim` with that as separator:

```
crabs <- read_delim("crabs.txt", ":")
```

```
## Rows: 200 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ":"
## chr (2): sp, sex
## dbl (6): index, FL, RW, CL, CW, BD
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
crabs
```

```
## # A tibble: 200 x 8
##    sp    sex   index    FL    RW    CL    CW    BD
##    <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 B     M         1   8.1   6.7  16.1  19     7
##  2 B     M         2   8.8   7.7  18.1  20.8   7.4
##  3 B     M         3   9.2   7.8  19    22.4   7.7
##  4 B     M         4   9.6   7.9  20.1  23.1   8.2
##  5 B     M         5   9.8   8    20.3  23     8.2
##  6 B     M         6  10.8   9    23    26.5   9.8
##  7 B     M         7  11.1   9.9  23.8  27.1   9.8
##  8 B     M         8  11.6   9.1  24.5  28.4  10.4
##  9 B     M         9  11.8   9.6  24.2  27.8   9.7
## 10 B     M        10  11.8  10.5  25.2  29.3  10.3
## # ... with 190 more rows
```

or, without the separator, but with explanation:

```
read_delim("crabs.txt")
```

```
## Rows: 200 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ":"
## chr (2): sp, sex
## dbl (6): index, FL, RW, CL, CW, BD
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
## # A tibble: 200 x 8
##    sp    sex   index    FL    RW    CL    CW    BD
##    <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 B     M         1   8.1   6.7  16.1  19     7
##  2 B     M         2   8.8   7.7  18.1  20.8   7.4
##  3 B     M         3   9.2   7.8  19    22.4   7.7
##  4 B     M         4   9.6   7.9  20.1  23.1   8.2
##  5 B     M         5   9.8   8    20.3  23     8.2
##  6 B     M         6  10.8   9    23    26.5   9.8
##  7 B     M         7  11.1   9.9  23.8  27.1   9.8
##  8 B     M         8  11.6   9.1  24.5  28.4  10.4
##  9 B     M         9  11.8   9.6  24.2  27.8   9.7
## 10 B     M        10  11.8  10.5  25.2  29.3  10.3
## # ... with 190 more rows
```

This will work because (you may assert) `read_delim` will guess that the data values are separated by colons. My output reveals that this assertion is correct (in the third line).

One point for using `read_delim` without the delimiter and without an explanation of why it will work. One point off for a large mistake and half a point per small one, down to a total of 1 if you got something substantial correct and a total of 0.5 if you showed any progress towards working code (at the grader's discretion).

(b) [3] What code would obtain the number of observations in each group, and the mean and SD of carapace length, for each combination of species and sex?

My answer:

A `group_by` and `summarize`, with both species and sex going into the `group_by`, and an `n()` in the summarize to count the number of observations in each group:

```
crabs %>%
  group_by(sp, sex) %>%
  summarize(n = n(), mean_length = mean(CL), sd_length = sd(CL))
```

```
## `summarise()` has grouped output by 'sp'. You can override using the `.groups`
## argument.
```

```
## # A tibble: 4 x 5
## # Groups:   sp [2]
##   sp    sex       n mean_length sd_length
##   <chr> <chr> <int>       <dbl>     <dbl>
## 1 B     F        50        28.1      5.92
## 2 B     M        50        32.0      7.31
## 3 O     F        50        34.6      5.84
## 4 O     M        50        33.7      7.61
```

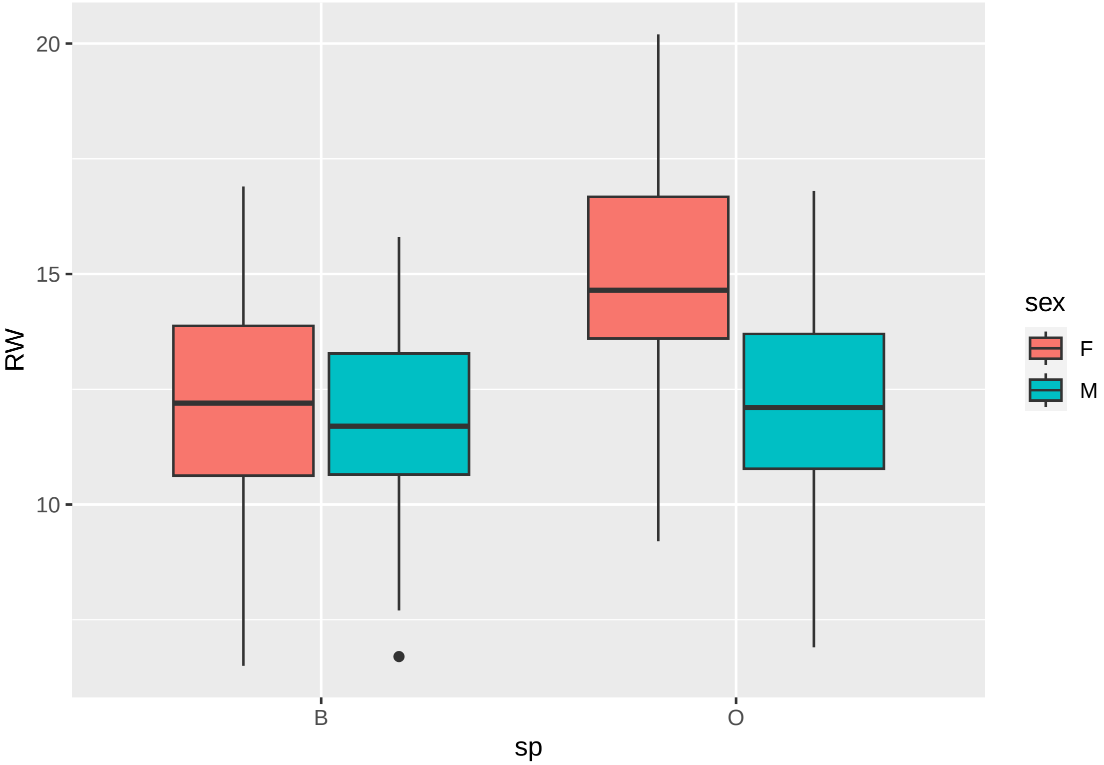
One point for getting the `group_by` line correct, and two points for getting the `summarize` line correct; minus a half point for any errors.

(c) [4] What code would make a suitable graph of rear width for each species and sex? What is the name of this type of graph?

My answer:

Two categorical variables and one quantitative one, so a **grouped boxplot**. I'm not particular about which categorical variable is `x` and which is `fill`, so they can be either way around:

```
ggplot(crabs, aes(x = sp, fill = sex, y = RW)) + geom_boxplot()
```

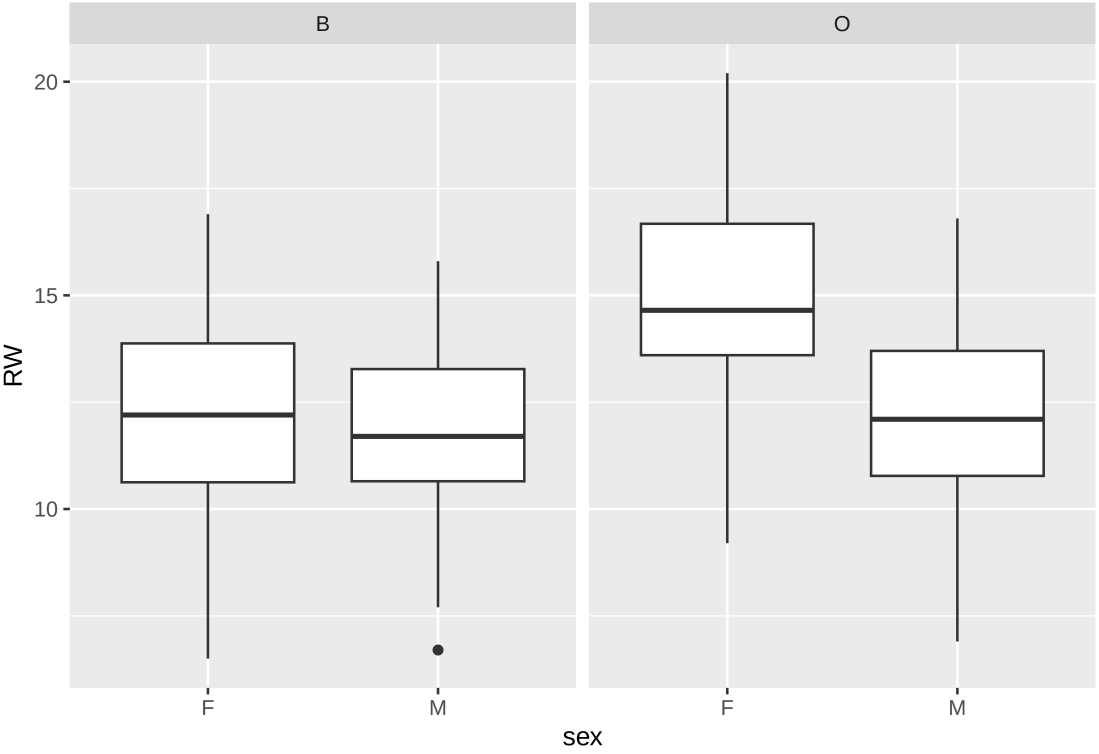


The word “grouped” needs to be there; it’s not just a simple side-by-side boxplot.

Three points for the code (minus one per error), one point for “grouped boxplot”. In the code, minus a half point for errors. I suppose half a point if you say “boxplot” without saying “grouped”.

The point of any kind of boxplot is to *compare* groups, so the four individual boxes need to be side by side. If you are careful, you can do it with facets:

```
ggplot(crabs, aes(x = sex, y = RW)) + geom_boxplot() + facet_wrap(~sp)
```



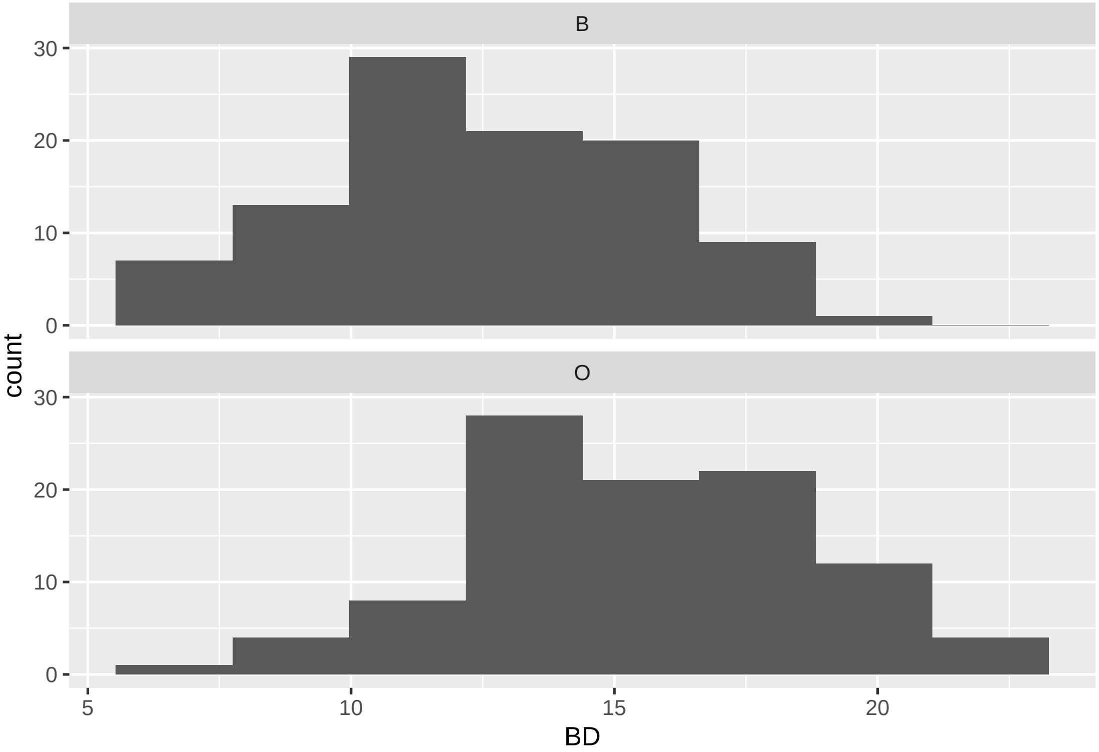
Since the idea is to compare groups, you need to have the *same* vertical scale for both facets, ie. `scales = "free"` is an error here. But code for the graph I just made, or with `sex` as facets and `sp` as `x`, is full credit as long as you describe it as a grouped boxplot.

(d) [3] What code would produce histograms of body depth for the two species separately, laid out above and below each other so that you can compare the distributions?

My answer:

This one can only be facetted:

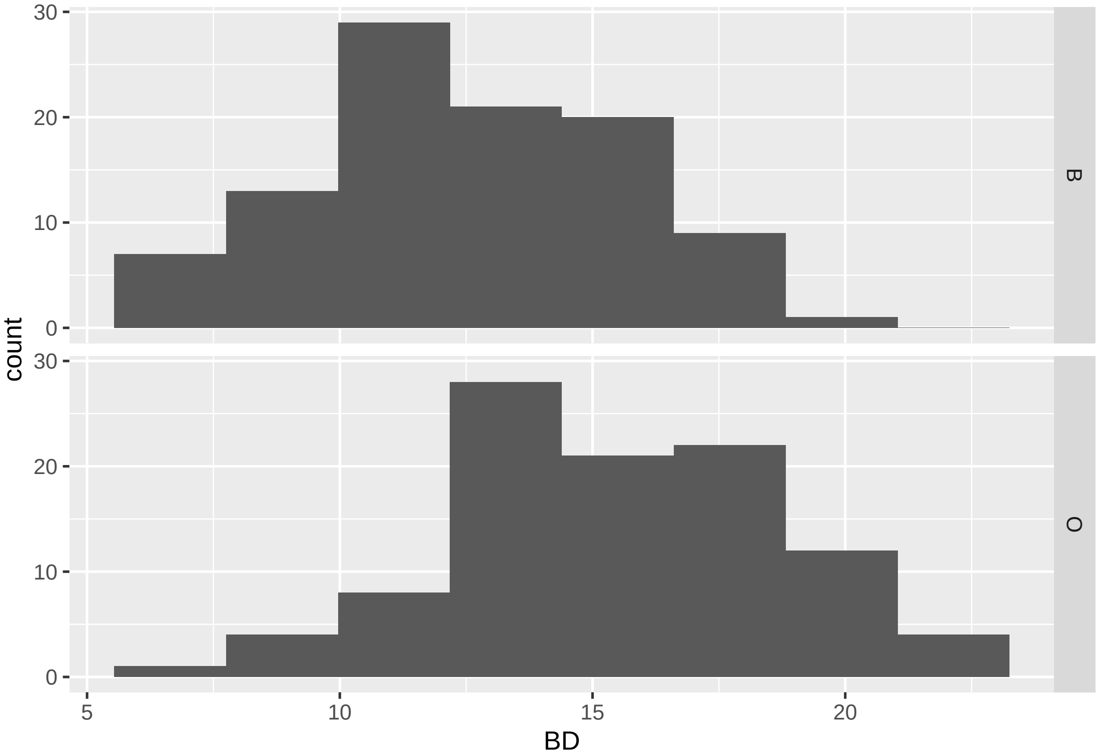
```
ggplot(crabs, aes(x = BD)) + geom_histogram(bins = 8) + facet_wrap(~sp, ncol = 1)
```



Minus one per error. The `ncol` is necessary because otherwise the two graphs will be left and right (and you won't be able to compare the distributions). Likewise, as in my variation in the previous part, you do *not* want a `scales = "free"`, for the same reason as there. You need some number of bins in the `geom_histogram`; you can actually use a few more than normal because the number you specify is *shared* between the two histograms. So any number of bins between, say, 5 and 20 is fine by me, but have one.

If you have mastered `facet_grid`, that also works, but the coding is a bit different:

```
ggplot(crabs, aes(x = BD)) + geom_histogram(bins = 8) + facet_grid(sp ~ .)
```



In the `facet_grid`, the first thing is the categorical variable that makes the y direction, and the second is the one that makes the x-direction. If either of these is empty (here the x), use a dot.

(e) [2] How many crabs are there with carapace length greater than 40 for each species? What code would determine this?

My answer:

```
crabs %>%
  filter(CL>40) %>%
  count(sp)
```

```
## # A tibble: 2 x 2
##   sp        n
##   <chr> <int>
## 1 B         8
## 2 O        19
```

There can be some credit for something else that will work, such as this:

```
crabs %>%
  count(sp, CL>40)
```

```
## # A tibble: 4 x 3
##   sp    `CL > 40`     n
##   <chr> <lgl>     <int>
## 1 B     FALSE        92
## 2 B     TRUE          8
## 3 O     FALSE        81
## 4 O     TRUE         19
```

but this is not as good because it also displays the counts of crabs with carapace length *less* than 40, which you don't want. One point for this. If you rescue yourself, you can get both points, eg:

```
crabs %>%
  count(sp, CL>40) %>%
  filter(`CL > 40`)
```

```
## # A tibble: 2 x 3
##   sp    `CL > 40`     n
##   <chr> <lgl>     <int>
## 1 B     TRUE          8
## 2 O     TRUE         19
```

remembering to put the thing inside the `filter` in backticks because it is not actually a legal column name (minus a half point if you get this without the backticks). But it is easier to do the `filter` first.

(f) [3] Out of the crabs with carapace length greater than 40, what is the smallest rear width, categorized by species? What code would calculate this?

My answer:

```
crabs %>%
  filter(CL>40) %>%
  group_by(sp) %>%
  summarize(small_rw = min(RW))
```

```
## # A tibble: 2 x 2
##   sp    small_rw
##   <chr>    <dbl>
## 1 B         13.5
## 2 O         13.7
```

A point for each line. If you do the same thing as in the previous part, replacing the `count` there with a `group_by` and `summarize`, the grader will try to give you credit here (the guidelines are whether you have made any additional errors here, and whether by making an error earlier you have made this part easier).

2. Hemophilia is a bleeding disorder in which the blood does not clot properly. This can lead to bleeding that does not stop. Blood contains many proteins called clotting factors that can help to stop bleeding.

One type of hemophilia is called Hemophilia A. In a study, 75 women were classified according to whether they do not have hemophilia A (30 women; "normal") or whether they do have hemophilia A (45 women; "carrier"). This is in the column called `gr`. Two variables were measured, called `AHFactivity` and `AHFantigen`, and the research question was whether either of them have any diagnostic value (meaning, whether either of them in any way distinguish women with hemophilia A from women without hemophilia A.)

Some of the data are shown in Figure 3.

(a) [2] Describe *in words* a graph that would help to determine whether either of the measured variables help to distinguish women with hemophilia A from women without.

My answer:

With two quantitative variables and one categorical, the graph we know is a scatterplot, with the points distinguished by colour. Here, that means a scatterplot of `AHFactivity` against `AHFantigen` (either way around, since neither is a response), with the points coloured according to the value of `gr`.

Points: a half point for saying "scatterplot", a half for saying which variables go on the scatterplot (by name), a half for saying "distinguished by colour", a half for saying (by name) which variable is used for distinguishing by colour.

A second-best example would call for boxplots for each measured variable against `gr` individually. This is second best because the distinction between "normal" and "carrier" women might depend on a combination of the two measured variables (eg. them both being high together), which wouldn't show up (or not as well) on individual boxplots. One point for a suitably complete description of the boxplots you would draw (complete enough that I could code it from your description). Half a point for "boxplot" plus some non-trivial description of the boxplot(s) you would draw.

It is perhaps a good idea to read ahead to the next part and think about how you are going to answer that based on the graph you are proposing.

Note that the question asks for a description in words, *not* for code. This was on purpose, since the previous question asks for code, so this question is getting at whether you can *describe* what you think is appropriate.

(b) [2] Explain briefly how you would use your graph of the previous part to determine whether either of the two measured variables distinguish women with hemophilia A from women without hemophilia A. If you find it helpful, use an example or a sketch of the kind of graph you might get to support your explanation.

My answer:

Your scatterplot will have a collection of red and blue points. If they are all mixed up, then neither of the two measured variables distinguish the two groups of women, but if (say) the red points are mostly in one part of the graph, then the part of the graph they are in tells you about which variable(s) distinguish them. For example, if the red points are mostly in the top right, that would mean that this group is distinguished by both variables being high. (If they are mostly on the right, that would mean that the variable on the x-axis is high for them, and so on.)

You could draw a scatterplot (axes labelled) with some Os and Xs in different parts of the graph to achieve the same purpose. Some students used colour (I think) to distinguish the two groups, but the exam was scanned in black and white, so I couldn't be sure. If you seem to have drawn an example plot that seems to have different coloured points in different places, along with some sensible explanation, you should have two points.

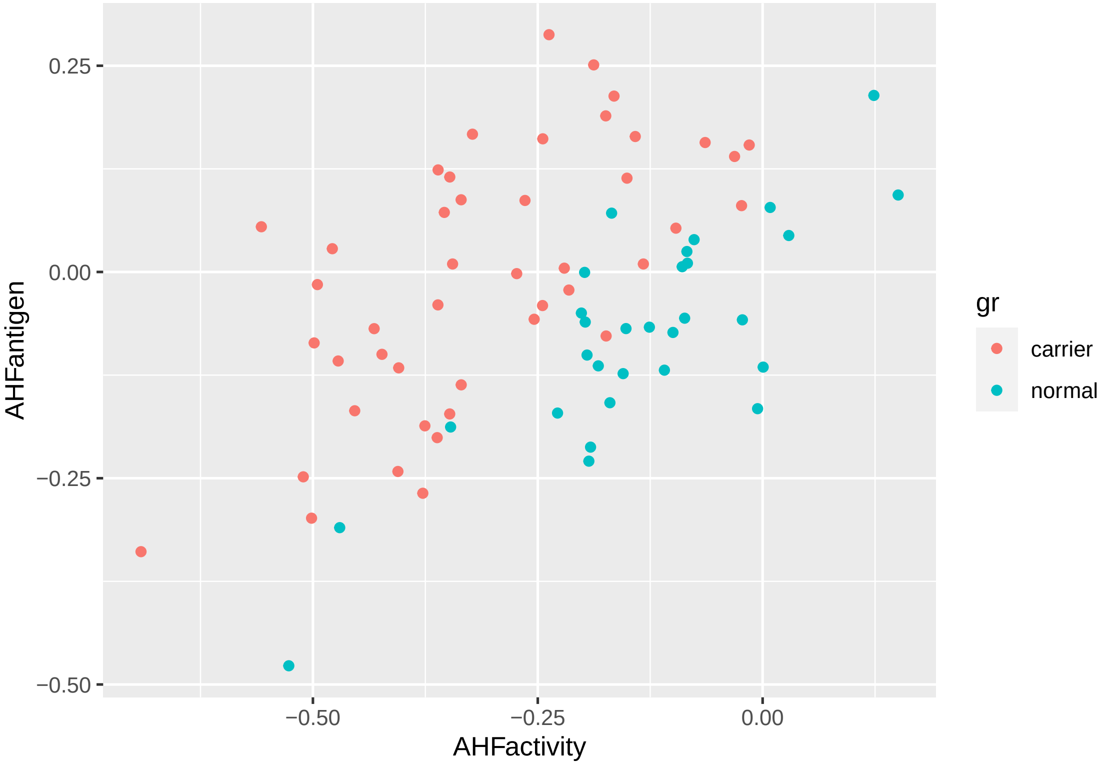
You don't have all the data, so you will have to make a guess about what the plot might look like. Any guess is fine, to illustrate what you are looking for.

If you drew boxplots, you can draw some example plots to illustrate the sort of thing that might happen, eg. pick one of the variables to be on the y-axis of your plot (doesn't matter which, but label it) and draw two boxplots with different medians. In that case, you have it a bit easier here, but you will have lost at least a point above, because a scatterplot is better than a pair of boxplots (see above).

Points: two points for anything that completely answers the question (there are many possibilities), one for an answer that gets part of the way but is missing important details.

Extra: the scatterplot actually looks like this:

```
ggplot(hemophilia, aes(x = AHFactivity, y = AHFantigen, colour = gr)) + geom_point()
```



The `normal` women are actually at the bottom right, so are distinguished by having high `AHFactivity` and low `AHFantigen`. Of course, you won't know this, so I am asking you to imagine the kind of thing that might happen. This plot also shows you why having the second dimension is valuable: the normal women are a bit lower on `AHFantigen` and a bit higher an `AHFactivity`, but there is a lot of overlap in both cases, so neither of the two boxplots will show much of a difference. On the scatterplot, though, it's pretty clear where the blue dots are relative to the red ones, most of the time. What has happened is that within each `gr` group, the two measured variables are associated, so if you guessed to add regression lines to each cloud of points (a red line and a blue line), that will also show the difference between the two groups. You might phrase that differently, though: something like "for any `AHFactivity`, `AHFantigen` is higher for the carrier women than for the normal women".

This kind of data responds well to discriminant analysis (which we will see in D29). This is the technique of choice when you want to find out what makes groups different, particularly when you have several quantitative variables, and the answer might have to do with combinations of them. But here, I wanted to give you a two-sample question, so I made one out of this data.

(c) [3] The researchers decided to use AHFactivity to compare the two groups of women. A graph is shown in Figure 4. Two possible analyses are shown in Figures 5 and 6. Which analysis do you prefer and why? Explain briefly.

My answer:

The two tests in the Figures are a (Welch) two-sample t-test and a Mood's median test, respectively.

To decide which one to look at, start with the boxplot in Figure 4. This indicates that AHFactivity for the carrier group is pretty close to normal, but that for the normal group it has outliers in both tails.

To run a two-sample t-test, we need both groups to be close enough to normal given the sample sizes. Make a call on this; either answer is justifiable, so the points are for your justification. I don't think there is any problem with the carrier group (with a sample size of 45). With the normal group (sample size 30), you can say either:

- the sample size is not large enough to overcome the effect of the outliers. Therefore it is not true that both groups are normal enough in shape, and we should use Mood's median test (Figure 6). Your answer must mention the sample size; it is not enough to say that the AHFactivity values are not normally distributed (for the normal group of women), and stop.
- the sample size of 30 is large enough to overcome the effect of the outliers, because the outliers are not serious and we will be helped by the Central Limit Theorem. Therefore, both groups are normal enough and we should use the t-test, Figure 5.

Another possible consideration is that the two tests in Figures 5 and 6 have rather similar P-values. From that point of view, it doesn't matter which test we do, since the conclusion is the same either way. (The usual thing in that case is to prefer the t-test, since it uses the data more efficiently (the exact values rather than whether or not a value is above the grand median), but it would be better to say at least something about sufficient normality before asserting that.) Any other discussion of P-values here is premature, because that belongs in the next part.

An answer without a justification looks like a guess, and is a fast zero. The reasoning is the purpose of the question.

One point each for a discussion of normality and sample size for each group of women (a half point in each case for discussing normality only), and one point for using that to make a reasoned choice of test. Alternatively, if you decided that the normal group wasn't normally distributed enough and you say something like "showing that one group is not normal enough rules out using a t-test", you don't need to consider the carrier group at all.

(d) [2] What do you conclude from your preferred test, in the context of the data?

My answer:

If you preferred the Mood's median test, your P-value is 5.7×10^{-7} and you reject the null hypothesis, which says that the `normal` and `carrier` women have the same median level of `AHFactivity`, in favour of a two-sided alternative (no alternative stated) that the two groups of women have different medians.

If you preferred the *t*-test, your P-value is 5.7×10^{-6} and again you reject the null hypothesis, which says that the `normal` and `carrier` women have the same *mean* level of `AHFactivity`, in favour of a two-sided alternative (no alternative stated) that the two groups of women have different means.

Make sure that the population parameters that you are making an inference about match the test you are doing. Only one point if they don't and you are otherwise correct. Also, make sure that you say that it's `AHFactivity` you're talking about, not the other one `AHFantigen`; we don't know anything about that.

Another possible angle is to think back to what we are trying to find out: does `AHFactivity` level distinguish the `normal` and `carrier` women? The small P-value says that it does. If you word it this way, that's fine as far as I am concerned.

(e) [2] Figure 7 shows bootstrap sampling distributions of the sample mean for each group of women. Do these plots support your conclusions about which test to do, or do they cause you to change your mind? Explain briefly.

My answer:

The two bootstrap sampling distributions are both very close to normal (the points are very close to the line), which means that *both* sample sizes are large enough to overcome any non-normality in the population distributions. (In the case of the `normal` group, you may find that surprising, but that's the way it is.) Hence the *t*-test is actually just fine.

If you preferred the *t*-test before, this supports your decision; if you preferred the Mood median test, it should cause you to change your mind. Again, the points are for the explanation.

Your answer, for full credit, needs to say something about the sample sizes actually being big enough to make a *t*-test viable, since that is the purpose of looking specifically at the bootstrap sampling distributions of the sample means. Some discussion about these distributions being close to normal, or not, without taking it further, is one point. (In my opinion, with the masses of points close to the lines, there is no problem with the normality here and therefore the sample sizes are big enough, but if you want to disagree with me, fine, if you can take your argument where it needs to go. If you think the normality here is no good, then these plots support your decision to use Mood's median test, or make you change your mind from using a *t*-test, as appropriate.) There was also 1.5 for getting close but not completely showing that you knew what the Figure was telling you.

My intended purpose here was to get you to realize just how powerful the Central Limit Theorem is. As I see it, the boxplot for the `normal` group is nothing like normally distributed, and somewhere, either earlier or here, I wanted you to be able to say that because the sample is large enough, the *t*-test is actually just fine. If you said that earlier, then the normal quantile plots of the bootstrap sampling distributions of the sample mean will probably confirm your earlier call that the sample sizes are big enough, and all you have to do is say that.

(f) [2] Why did we not do a one-sided test, despite the evidence in Figure 4?

My answer:

The alternative hypothesis cannot be chosen by looking at the data; it has to come from some theory or expectations *outside* of the data. Here, we didn't know ahead of time whether we expected `AHFactivity` to be higher or lower for the `carrier` group, because we had, or I gave you, no hints about that. The clue is in the last sentence of the question (before I tell you which Figure shows the data): "whether either of them in any way distinguish women with hemophilia A from women without hemophilia A." The key words are "in any way": it might be making the variable in question higher or lower, and which way it goes is part of the research question.

In the class example of the children learning to read, we wanted to know whether the new reading method was *better*, because that would indicate that it was useful and worth investigating further. The interest in "better" came from outside the data we had, not by looking at the boxplot.

I was wondering whether many people would get sidetracked by the very obvious direction of the differences in medians in the boxplot, but most people seemed to recognize in some fashion that the "research hypothesis" was what we needed the alternative hypothesis to reflect. Some of the answers, though, got a bit close to "we did a two-sided test because we did a two-sided test", which I don't want to be giving marks to.

3. This question is all about tidying data: that is, rearranging it to display the data in a different layout that might be more convenient for graphing, display, etc.

(a) [3] Figure 8 shows a dataframe `d1` that needs to be rearranged as the dataframe `d2` shown in Figure 9. What code would rearrange `d1` into `d2`?

My answer:

This is making a wider dataframe longer, so `pivot_longer` will do it:

```r
d1 %>%
  pivot_longer(-id, names_to = "treatment", values_to = "score")
```

```
## # A tibble: 9 x 3
##   id    treatment score
##   <chr> <chr>     <dbl>
## 1 A     g1           10
## 2 A     g2           21
## 3 A     g3           29
## 4 B     g1           11
## 5 B     g2           20
## 6 B     g3           28
## 7 C     g1            12
## 8 C     g2            22
## 9 C     g3            31
```

There is no obligation to save the result. The focus in this question is on the tidying. Note that you need to get the columns in the long dataframe to have the same names as mine, so that the `names_to` and `values_to` have to be the same as mine. (This differs from the usual tidying that you will do in the workplace, where you get to choose the names of the new columns.) Any way of specifying the columns to pivot longer is good, for example:

```
d1 %>%
  pivot_longer(g1:g3, names_to = "treatment", values_to = "score")
```

```
## # A tibble: 9 x 3
##   id    treatment score
##   <chr> <chr>     <dbl>
## 1 A     g1           10
## 2 A     g2           21
## 3 A     g3           29
## 4 B     g1           11
## 5 B     g2           20
## 6 B     g3           28
## 7 C     g1           12
## 8 C     g2           22
## 9 C     g3           31
```

Two points if you get the `pivot_longer` right, but have different names for one or both of the new columns. Minus one point per error otherwise, with 1 if you have something substantial correct (in the grader's estimation). You might lose only half a point for something the grader considers to be "very small".

(b) [2] Suppose, instead, you had been given the dataframe `d2` shown in Figure 9 and were asked to rearrange it into `d1`. What code would do this?

My answer:

I tried to not advertise the fact that this is the exact opposite of the previous part, and so `pivot_wider` is what you need:

```
d2 %>%
  pivot_wider(names_from = treatment, values_from = score)
```

```
## # A tibble: 3 x 4
##   id       g1    g2    g3
##   <chr> <dbl> <dbl> <dbl>
## 1 A        10    21    29
## 2 B        11    20    28
## 3 C          12    22    31
```

This time, the names of the new columns will come from the *values* in the `treatment` column, so there is no need to worry about what the names will be.

One point if it's not right, but you have something substantial correct.

(c) [4] A dataframe `dd` is shown in Figure 10. The numerical values are all of a variable `y` observed under different conditions: a `level` that takes the values `Hi` and `Lo`, and a `size` that takes the values `Large` and `Small`. There are two replicate values of `y` observed at each level-size combination; these are labelled `R1` and `R2` in the column `rep`.

It is desired to arrange the dataframe so that all the values of `y` are in one column, with columns indicating the `level`, `size`, and `rep` at which that value of `y` was observed. What is the most concise code that would do this?

My answer:

This is the variant of `pivot_longer` where the current column names encode two things: both a level and a size. To handle this, use *two* things in `names_to`, and also use a `names_sep` to say what they are separated by. The answer to this is often an underscore, but here they are not separated by anything at all. However, the level is always two characters long, so separating after the second character, that is, with `names_sep` being 2, will work. Make sure to use the names `level` and `size` for your new columns:

```
dd %>%
  pivot_longer(-rep, names_to = c("level", "size"), names_sep = 2,
               values_to = "y")
```

```
## # A tibble: 8 x 4
##   rep   level size      y
##   <chr> <chr> <chr> <dbl>
## 1 R1    Hi    Large    16
## 2 R1    Hi    Small    17
## 3 R1    Lo    Large    19
## 4 R1    Lo    Small    18
## 5 R2    Hi    Large    18
## 6 R2    Hi    Small    20
## 7 R2    Lo    Large    22
## 8 R2    Lo    Small    21
```

There is no need to do anything special with the `rep` column; this will be repeated as necessary to match the values of the other variables.

The size is also always five characters long, so it's also possible to split before the fifth character from the end with `names_sep = -5`.

Points: one for identifying the columns to pivot-longer (`HiLarge:LoSmall` also works), one for the two-part `names_to`, one for a suitable `names_sep`, one for `values_to`. A `pivot_longer` without anything else correct is 1 in total. (There has to be a `pivot_longer` to get anything.)

The second-best answer (worth 3 points if you get it right) is the two-step process with the basic `pivot_longer` followed by `separate`. This needs you to invent a name for the variable that contains the original column names, and then, in `separate`, use a `sep` that is the same as the `names_sep` I used above:

```
dd %>%
  pivot_longer(-rep, names_to = "var", values_to = "y") %>%
  separate(var, into = c("level", "size"), sep = 2)
```

```
## # A tibble: 8 x 4
##   rep   level size      y
##   <chr> <chr> <chr> <dbl>
## 1 R1    Hi    Large    16
## 2 R1    Hi    Small    17
## 3 R1    Lo    Large    19
## 4 R1    Lo    Small    18
## 5 R2    Hi    Large    18
## 6 R2    Hi    Small    20
## 7 R2    Lo    Large    22
## 8 R2    Lo    Small    21
```

One point for a correctly-done `pivot_longer` (this is really the same idea as the previous part, so this is rather a giveaway), and two for the `separate` with the right `into` and `sep`. Expect only one for the `separate` part if you made any mistakes in it but had a `separate` there.

(d) [4] A dataframe `ddd` is shown in Figure 11. Some code is shown in Figure 12. What output will that code produce?

My answer:

This is one of those cases where pivot-wider may not work as you expect:

```
ddd %>%
  pivot_wider(names_from = id, values_from = y)
```

```
## # A tibble: 2 x 5
##   g         A     B     C     D
##   <chr> <dbl> <dbl> <dbl> <dbl>
## 1 lo       20    NA    23    NA
## 2 hi       NA    22    NA    24
```

The column `id` gives names to the new columns, and the column `y` contains the values that go in them. The other column `g` not named in the pivot-wider determines the row where each value goes. (This is a slightly different arrangement than the example in class.) Thus there are four columns, `A` through `D`, and two rows, one where `g` is `lo` and the other where `g` is `hi`.

Then:

- the 20 goes in the `lo` row and the `A` column
- the 22 goes in the `hi` row and the `B` column
- the 23 goes in the `lo` row and the `C` column
- the 24 goes in the `hi` row and the `D` column.

There are two rows, four columns and only four values of `y` to go in the eight cells, so the other four cells will be empty and will get an `NA` (missing) in them, such as the `hi` row of the `A` column.

Two points for the right thing in the right column, but with only one row (or more than two rows). Likewise two for the right *rows* but the wrong columns. One for any progress towards the right answer (in the grader's judgement). Getting the right rows and columns but a small error in what goes in them (a wrong value or switching of two values) is three.

4. 164 men took part in an experiment to see whether the drug cholostyramine lowered blood cholesterol levels. The men were supposed to take six packets of cholostyramine per day, but many actually took much less. We want to investigate whether men that better adhered to the instructions had a greater improvement in blood cholesterol levels.

   Some of the data are shown in Figure 13. There are two columns:

   - `compliance`: the percentage of all the packets of cholostyramine given to that man that were actually taken
   - `improvement`: the blood cholesterol was measured at the beginning and at the end of the study. This column is the percentage decrease, compared to the initial value. A negative `improvement` means that blood cholesterol was worse (higher) at the end of the study than it was at the beginning.

   (a) [4] A scatterplot is shown in Figure 14. Interpret this scatterplot in terms of the form, direction, and strength of the relationship. ("Form" means whether it is linear or curved or something else, "direction" is up or down, "strength" is strong, weak, or something in between.) Explain briefly.

My answer:

This is an upward trend (direction) and weak to moderate in strength (there is a lot of scatter, but not so much as to hide the trend). As for form, it is mostly linear, but improvement for the highest-compliance men (say, above a compliance rate of about 90) is both higher and more spread out than a line through the rest of the points would suggest.

One point for "upward", one for a properly reasoned description of the strength, one for saying that it is mostly linear, and one for saying something relevant about the high-compliance men. Somebody used the term "ramp up" which is a nice description of what is happening at the high end. If you think it's curved, say something about the kind of curve you think it is. You could also say that there is fanning out on the right side because the variability seems to increase there.

(b) [2] The output from a linear regression is shown in Figure 15. Do you think there is there a real relationship between compliance and improvement, on the basis of this output? Explain briefly.

My answer:

This is asking whether the relationship is significant: that is, if the slope is significantly different from zero, which it is (P-value less than 2.2×10^{-16}). There is definitely a real relationship between compliance and improvement. (In this case, people that take more of their medication improve more.)

One point for saying that the slope. 0.58, is "clearly not zero" or similar, without actually getting to the P-value. Likewise, asserting that the R-squared is "reasonably high" gets about the same distance to the answer, so is also one point.

I don't want you to get too much credit for a "kitchen sink" answer in which you write down everything you can think of. My take is if you lead with the P-value, then discuss how the R-squared is not very high (or whatever you think it is), that's actually good (because you can then say that it is a real relationship but not a very strong one). But if you lead off with something else and mention the P-value as what looks like an afterthought, expect only one, because you have not shown that you understand what is important here.

(c) [2] A plot of residuals from the regression `cholost.1` is shown in Figure 16. What do you conclude from this plot? Explain briefly.

My answer:

This is a normal quantile plot of the residuals. The residuals have a normal distribution, because the points are close to the line. If the regression is appropriate, the residuals will be close to normal like this. Or, this particular plot is OK as far as the regression is concerned.

Note the logic: if the regression is OK, the residuals will be normal, but if the residuals are normal, the regression may not be OK because there might be a problem with the plot of residuals vs fitted values (see next part).

As usual, if you conclude something defensible, even not what I did, I am fine with that. The major point is to show that you know it to be a normal quantile plot, and to have an opinion about the normality it displays. (The question tells you that it's a plot of the residuals.) I am not wild keen on an answer of "skewness", because the endmost points are close to the line (it's the ones further in that are off the line), but I can be swayed by an answer that seems well thought out.

You need to say something showing that you know this is a normal quantile plot (like "the points are close to the line so the residuals are close to normal"), rather than making me think that you are looking at a scatterplot ("there is an upward trend").

(d) [2] Another plot of residuals is shown in Figure 17. What do you conclude from this plot? Explain briefly.

My answer:

This is a plot of residuals against fitted values. For the regression to be appropriate, this should be a random scatter of points. However, it is not quite that: the points on the right extend further up than the scatter of the other residuals (the most positive four or five residuals go with a fitted value greater than 50). That is to say, if you know the fitted value is among the largest ones, the residual is more likely to be very positive (or, another way to say it, is more likely to be positive than negative). This is different from the residuals to the left of a fitted value of about 50, which form a more or less horizontal band around zero.

You can also describe this as "fanning-out" as long as you are careful to say where the fanning out is happening: only for the very largest fitted values, because the variability of the residuals is about the same the rest of the way across.

There are several ways to say something like this. Find one and I'm happy. There are also two apparently outlying negative residuals with fitted values around 35, but this is to my mind less of an issue than what is happening with the fitted values over 50. One point for naming this as a problem (and not talking about the highest fitted values). One point also for saying that everything is OK, and 1.5 for saying that something is amiss on the right without being precise enough about what that something is.

(e) [3] One of the researchers believes that a compliance over 95 (percent) is associated with better improvement, and fitted the model whose code and output are shown in Figure 18. Is there a significant benefit to a compliance over 95 percent, over and above the greater improvement that we have already seen goes with a greater compliance? How big is this benefit? How can you tell? (Note that in a regression model, a true-false variable is treated the same as a categorical variable with levels `TRUE` and `FALSE`).

My answer:

The best answer notes that the new column `bonus` is TRUE for men with compliance 95 or greater and `FALSE` otherwise. In the summary table, the P-value for `bonus` is 0.00032, which is definitely significant, so there is clearly an additional impact of having compliance over 95 percent. The Estimate reveals that having a compliance of 95 or more adds a predicted 18 percentage points to the improvement, over and above the higher improvement that would be predicted for a higher compliance anyway. (Give the P-value, because there are several in the output and I want to make sure you have the right one, or at least tell me that you are looking at the P-value for `bonusTRUE`. "The P-value is small" doesn't really say anything much, because some of them are and some aren't.)

A less insightful answer looks at the R-squared from before, 0.46, and notes that it has increased to 0.50. This, it is true, indicates that the regression fits better than it did before, but it is not on the scale of changes in `improvement`, which is what we wanted to know about. You might be thinking that this is a rather modest increase in R-squared to be so significant, but this is because we have a lot of data. A perhaps better indicator is the *adjusted* R-squared. This also goes up from the first model to the second one, which indicates a real improvement in fit.

The effect of the extra term in `bonus` is to add a "jump" to the relationship at a compliance of 95 percent, so that men with that compliance or higher are predicted to have a higher improvement than before. This doesn't improve the fanning-out that you may have seen before, but it *does* do something about the majority of residuals for a high compliance (and thus high fitted value) being positive.

5. Hospitals can charge different amounts of money, even to patients requiring the same treatment. Are there systematic reasons why they do so? A doctor collected data on 49 patients with the same diagnosis, as follows:

- `Sex` M (Male) or F (Female).
- `MD` which doctor they were treated by (there are three different doctors)
- `Svty` severity of illness, from 1 (lowest severity) to 4 (highest), which we treat as quantitative.
- `Chrg` total amount charged by the hospital, in dollars (response)
- `Age` of patient in years

Some of the data is shown in Figure 19.

(a) [2] Plots of charges against each of the explanatory variables are shown in Figures 20 and 21. Why are some of the plots boxplots?

My answer:

Because the explanatory variables in question are categorical, and the right plot for quantitative vs. categorical is a boxplot.

The issues here are two:

- the explanatory variables in question are categorical (most important)
- the response variable is quantitative

so in principle you need to mention both. (If we had a categorical response, which of course we can't for a regression, we would need something like a grouped bar chart. So as long as you get "the explanatory variables that are gender and doctor are categorical, you are good.)

I could have treated severity as categorical also, and drawn a boxplot there as well (with four boxes, one for each level of severity), but I treated that as quantitative to stop things getting too messy. Hence talking about severity here won't help you.

(b) [2] There was one patient whose charge was much higher than for any of the other patients. What do Figures 20 and 21 tell you about that patient?

My answer:

This is the upper outlier on all four of the plots. Thus, reading across the plots, it is a person of age about 70 with severity 4 (the highest), who was treated by doctor MD1021, and who was female. Half a point each. Get reasonably close to these (50–75 is not close enough for age).

(c) [2] Ignoring the upper outlier, what do Figures 20 and 21 tell you about when charges are higher, for each of the four explanatory variables? (Four very short answers.)

My answer:

Charges are higher for:

- older patients (with higher age)
- patients with severity 4 (the highest)
- patients treated by doctor MD1021 (and lowest for those treated by MD730)
- female patients, as opposed to males.

Half a point for each of those.

If you want to say that the doctors are all about the same, say that (I am fine with it), or something else relevant about how the doctors compare, but say *something* about the doctors. You could also say that there is not much difference between the genders (something that comes out of the next part).

This is a good question to answer with bullet points, which makes it easy to check that you have said four things.

(d) [2] A regression model is fitted, with output shown in Figure 22. Assuming that the residual plots look appropriate, what would you do next, and why is the `drop1` output better to decide this from?

My answer:

We have categorical explanatory variables, so it is better to look at the `drop1` output to decide what to do next (eg. whether to remove the categorical variables).

This says that we should remove the non-significant `Sex` from the model, and fit a model with only `MD`, `Svty` and `Age`.

One point for each of those. 0.5 for the first if you only get as far as saying what to do without saying that `Sex` is the thing to be removed (you have the information to determine this, from Figure 22).

(e) [2] In Figure 22, do the positive Estimates for severity and age make practical sense, in the context of the data? Explain briefly.

My answer:

The positive estimates indicate that the charge is predicted to be greater for an older patient and for a patient with higher severity. Both of these are cases where there are more things to go wrong: an older patient may have more of other health issues, and a more severe case may require more work to treat satisfactorily.

One point for saying or implying what the positive Estimates imply for charges, and the second for saying something sensible about why that's what you'd expect. If you get the second point, it will generally imply that you said enough to get the first one as well.

The P-values are not relevant here.

(f) [2] Based on Figure 22, which doctor has the highest predicted charges, all else equal? Explain briefly.

My answer:

The two Estimates for the doctors shown are both negative, so the doctor with the highest predicted charges is the baseline doctor MD1021 (go back to the data listing or the boxplot to find out what this doctor is called, or at least what their ID is.)

This is the doctor with the very large outlying charge, but that doctor's average (median) charge is higher than for the other doctors as well (the boxplots are telling the same story as the regression).

Another reasonable answer is that the drop1 output says that there is no significant difference in charges between the three doctors (P-value 0.098), so that there is no reason to choose between them on the basis of charges. (Looking at the P-values in the summary output only compares with the baseline doctor MD1021, which does not tell the whole story about comparing doctors. So the P-value in the drop1 table is the only one of value, not the ones in the summary output.)

Have an explanation if you want any points.

(g) [2] What is the precise meaning of the P-value 0.0368 in Figure 22?

My answer:

This means that there is a significant difference in (predicted) charges between doctor MD499 and *the baseline doctor* MD1021, all else equal.

"The doctor MD499 is significant" is not enough. A small P-value means "significantly different *from* something", and the identity of the something is important. A generous one point for an answer like this, that at least identifies which doctor the P-value is talking about.

6. The function rnorm generates random normal data. It has three inputs: the sample size, the population mean, and the population standard deviation. Let's suppose that you will be generating a lot of normal random numbers with various sample sizes and various means, but the population standard deviation will always be half as big as the mean. (You are building a very niche application.) You want to streamline your process by writing a function called my_random that has as input a sample size n and a population mean mu, and generates and returns a normal random sample according to the specifications.

(a) [3] What code would you use to write your function?

My answer:

Get the layout of it right; the function itself is not very complicated. This is how I would do it:

```
my_random <- function(n, mu) {
  sigma <- 0.5 * mu
  rnorm(n, mu, sigma)
}
```

This function calls rnorm, and you literally pass the input n and mu straight into rnorm; you don't have to do anything else with those.

Variations: save the output from rnorm and then return it by putting the name of the saved sample on the next line; do the same but wrap it in return; directly use 0.5 * mu as the third input to rnorm without calculating sigma first. This last variant makes the function a one-liner, and thus you don't actually need the curly brackets in that case. You can also use mu / 2 rather than 0.5 * mu to get the same result.

By some means, you have to pass the right sample size, population mean and population SD into rnorm, based on the inputs to your function.

Points for this one are the grader's judgement of how far you got: 2 means "most of the way" and 1 means "some progress, but not much, towards a solution".

(b) [2] How would you use your function to obtain 7 random normal values with mean 10 and SD 5?

My answer:

The function has two inputs, the sample size and the mean, so you don't input the SD (this is, in any case, already half as big as the mean):

```
my_random(7, 10)
```

```
## [1]  6.861400  5.327836  5.983675 13.635171  5.527759 18.783726  7.723223
```

(of course, you don't know what the output is going to be. Mine is to show that it works.)

Other ways to call your function are to name the inputs, which you can then give in either order (since R then matches them by name):

```
my_random(n = 7, mu = 10)
```

```
## [1]  7.662627 11.947027 18.443977 13.876961  8.664886  7.454273  5.054414
```

or

```
my_random(mu = 10, n = 7)
```

```
## [1]  7.365263  9.893543  2.694266 18.843294  4.377927  7.004633  9.760273
```

These are different because they are random samples, but they are both consistent with a mean of 10 and an SD of 5 (and a sample size of 7).

I can't see much beyond 2 if you have something that will work, and 0 otherwise. If you find a way to get close (in the grader's judgement) but not close enough, you can get 1.

(c) [3] For the next little while, you are told that the sample size will be 10. How would you *change* your function to avoid having to enter the sample size if it is 10, and how could you most concisely use your new function to obtain 10 random normal numbers with mean 20 (and SD 10)? (You only need to give what *changes* you would make.)

My answer:

This can be done by making a sample size of 10 be the default, on the top line of the function, as shown:

```
my_random <- function(n = 10, mu) {
  sigma <- 0.5 * mu
  rnorm(n, mu, sigma)
}
```

You just need to tell me how the top line changes; we will assume that the rest of it is the same.

To use the new function, the only input it now needs is `mu`, but you'll need to name it, since `n` is first:

```
my_random(mu = 20)
```

```
##  [1] 39.840441 12.127610 16.257106 11.132222 24.686321  2.580573 27.053426
##  [8] 15.919922 18.770792  7.052527
```

There are, as you can count, 10 values here, which was the default sample size in our rewritten function (and we didn't specify a sample size).

If you don't name the input, R will think that you are trying to give a value for `n` rather than `mu`, and will complain that you are not supplying a value for `mu`:

```
my_random(20)
```

```
## Error in my_random(20): argument "mu" is missing, with no default
```

(d) [3] Figure 23 shows a dataframe `d` containing some population means. How would you use your modified function to make a list-column called `sample_data` containing random samples of size 10 from normal distributions with the appropriate means (and standard deviations that are half as big as the means)?

My answer:

This is meant to be an exercise in `mutate` with `map` (just plain `map` since the output from our function is several numbers rather than just one):

```
d %>% mutate(sample_data = map(the_mean, ~my_random(mu = .)))
```

```
## # A tibble: 3 x 2
##   the_mean sample_data
##      <dbl> <list>
## 1        4 <dbl [10]>
## 2        8 <dbl [10]>
## 3       24 <dbl [10]>
```

Another equally good way is to use `rowwise`, thus:

```
d %>%
  rowwise() %>%
  mutate(sample_data = list(my_random(mu = the_mean)))
```

```
## # A tibble: 3 x 2
## # Rowwise:
##   the_mean sample_data
##      <dbl> <list>
## 1        4 <dbl [10]>
## 2        8 <dbl [10]>
## 3       24 <dbl [10]>
```

If you didn't realize that we were using the default sample size again, it's easier to go rowwise with this:

```
d %>%
  rowwise() %>%
  mutate(sample_data = list(my_random(10, the_mean)))
```

```
## # A tibble: 3 x 2
## # Rowwise:
##   the_mean sample_data
##      <dbl> <list>
## 1        4 <dbl [10]>
## 2        8 <dbl [10]>
## 3       24 <dbl [10]>
```

although the `map` way is not actually much harder, since the sample size is always 10. If the dataframe `d` had had two columns, sample sizes and population means, the map would have been much messier (we would have had to use `map2` because we would have been for-eaching over 2 varying things). As it is, though:

```
d %>% mutate(sample_data = map(the_mean, ~my_random(10, .)))
```

```
## # A tibble: 3 x 2
##   the_mean sample_data
##      <dbl> <list>
## 1        4 <dbl [10]>
## 2        8 <dbl [10]>
## 3       24 <dbl [10]>
```

(e) [2] How would you arrange it so that you could see the actual random data that had been generated?

My answer:

Add `unnest(sample_data)` onto the end of your pipeline. This is all you need to say; no need to write the whole thing out again.

To demonstrate that it works:

```
d %>% mutate(sample_data = map(the_mean, ~my_random(mu = .))) %>%
  unnest(sample_data)
```

```
## # A tibble: 30 x 2
##    the_mean sample_data
##       <dbl>       <dbl>
##  1        4       5.66
##  2        4       4.39
##  3        4      -0.770
##  4        4       2.37
##  5        4       2.57
##  6        4       2.22
##  7        4       2.49
##  8        4       4.95
##  9        4       4.63
## 10        4       3.97
## # ... with 20 more rows
```

Extra: the 68-95-99.7 rule says (for example) that from a normal distribution, 95% of values sampled from it are within 2 SDs of the mean. In this question, the SD is always half of the mean, so 95% of the sampled values are between $\mu - 2\sigma = \mu - 2(0.5\mu) = 0$ and $\mu + 2\sigma = \mu + 2(0.5)\mu = 2\mu$. You can check the values above to see how well it worked; for example, for a mean of 4, almost all the values should be between 0 and 8.

Use the rest of this page if you need more space. Be sure to label any answers here with the question and part they belong to.

Figures

```
library(tidyverse)
library(readxl)
library(smmr)
```

Figure 1: Packages

```
sp:sex:index:FL:RW:CL:CW:BD
B:M:1:8.1:6.7:16.1:19:7
B:M:2:8.8:7.7:18.1:20.8:7.4
B:M:3:9.2:7.8:19:22.4:7.7
B:M:4:9.6:7.9:20.1:23.1:8.2
B:F:3:9.1:8.1:18.5:21.6:7.7
B:F:4:9.1:8.2:19.2:22.2:7.7
B:F:5:9.5:8.2:19.6:22.4:7.8
B:F:6:9.8:8.9:20.4:23.9:8.8
O:M:10:13.7:11:27.5:30.5:12.2
O:M:11:14:11.5:29.2:32.2:13.1
O:M:12:14.1:10.4:28.9:31.8:13.5
O:M:13:14.1:10.5:29.1:31.6:13.1
O:F:36:19.7:16.7:39.9:43.6:18.2
O:F:37:19.9:16.6:39.4:43.9:17.9
O:F:38:19.9:17.9:40.1:46.4:17.9
O:F:39:20:16.7:40.4:45.1:17.7
```

Figure 2: Crabs data (some)

```
hemophilia %>% slice_sample(n = 20)
```

```
##    AHFactivity AHFantigen      gr
## 46     -0.5573     0.0548 carrier
## 54     -0.2205     0.0046 carrier
## 37     -0.3608     0.1237 carrier
## 56     -0.3447     0.0097 carrier
## 70     -0.3352     0.0875 carrier
## 2      -0.1698    -0.1585  normal
## 29     -0.1972    -0.0607  normal
## 13     -0.0225    -0.0580  normal
## 27     -0.2280    -0.1710  normal
## 68     -0.2642     0.0867 carrier
## 63     -0.0312     0.1400 carrier
## 17     -0.4702    -0.3099  normal
## 41     -0.4719    -0.1079 carrier
## 19      0.0006    -0.1153  normal
## 73     -0.4055    -0.2418 carrier
## 1      -0.0056    -0.1657  normal
## 18     -0.1519    -0.0686  normal
## 40     -0.3539     0.0722 carrier
## 72     -0.1744     0.1892 carrier
## 30     -0.0867    -0.0560  normal
```

Figure 3: Hemophilia data (20 randomly chosen rows)

```
ggplot(hemophilia, aes(x = gr, y = AHFactivity)) + geom_boxplot()
```

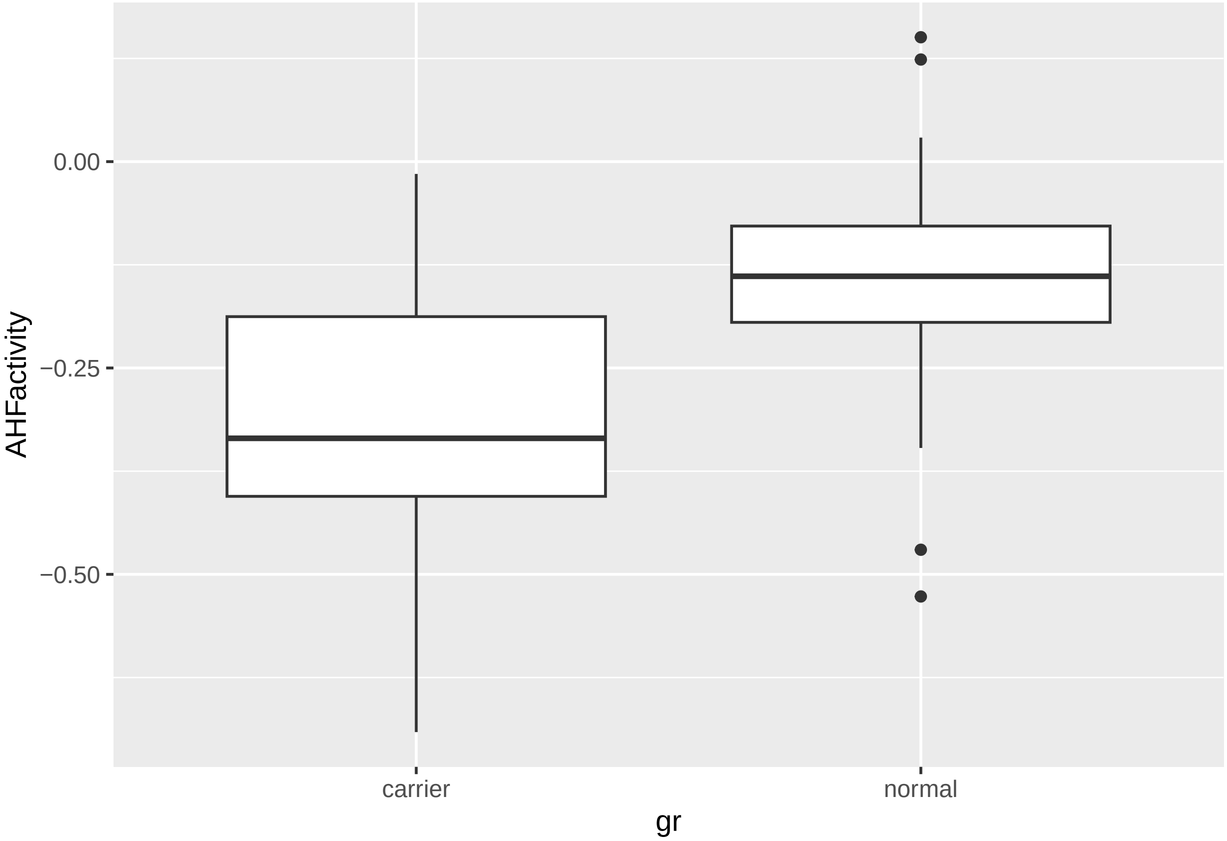


Figure 4: Graph of `AHFactivity` for each group of women

```
t.test(AHFactivity ~ gr, data = hemophilia)
```

```
##
##  Welch Two Sample t-test
##
## data:  AHFactivity by gr
## t = -4.9448, df = 65.029, p-value = 5.655e-06
## alternative hypothesis: true difference in means between group carrier and group normal is not equal
## 95 percent confidence interval:
##  -0.2429789 -0.1031744
## sample estimates:
## mean in group carrier  mean in group normal
##            -0.3079467            -0.1348700
```

Figure 5: Test 1 for hemophilia data

```
median_test(hemophilia, AHFactivity, gr)
```

```
## $table
##          above
## group     above below
##   carrier    12    33
##   normal     25     4
##
## $test
##        what        value
## 1 statistic 2.500690e+01
## 2        df 1.000000e+00
## 3   P-value 5.712562e-07
```

Figure 6: Test 2 for hemophilia data

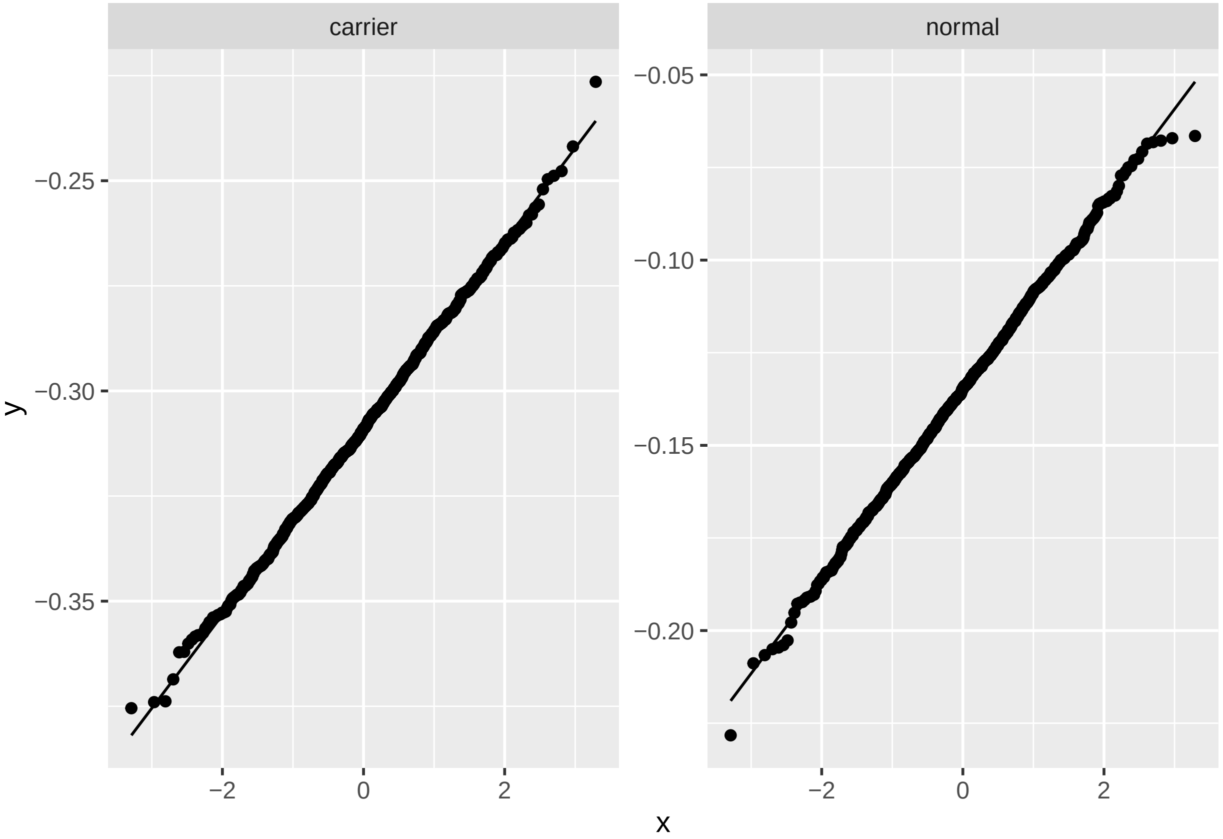


Figure 7: Bootstrap sampling distributions of sample means for hemophilia data, normal quantile plots

```
d1
```

```
## # A tibble: 3 x 4
##   id       g1    g2    g3
##   <chr> <dbl> <dbl> <dbl>
## 1 A        10    21    29
## 2 B        11    20    28
## 3 C        12    22    31
```

Figure 8: Dataframe `d1`

```
d2
```

```
## # A tibble: 9 x 3
##   id    treatment score
##   <chr> <chr>     <dbl>
## 1 A     g1           10
## 2 A     g2           21
## 3 A     g3           29
## 4 B     g1           11
## 5 B     g2           20
## 6 B     g3           28
## 7 C     g1           12
## 8 C     g2           22
## 9 C     g3           31
```

Figure 9: Dataframe `d2`

```
dd
```

```
## # A tibble: 2 x 5
##   rep   HiLarge HiSmall LoLarge LoSmall
##   <chr>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 R1         16      17      19      18
## 2 R2         18      20      22      21
```

Figure 10: Dataframe `dd`

```
ddd
```

```
## # A tibble: 4 x 3
##   id    g         y
##   <chr> <chr> <dbl>
## 1 A     lo       20
## 2 B     hi       22
## 3 C     lo       23
## 4 D     hi       24
```

Figure 11: Dataframe `ddd`

```
ddd %>%
  pivot_wider(names_from = id, values_from = y)
```

Figure 12: Code to run on dataframe `ddd`

```
cholost %>% slice(1:20)
```

```
##    compliance improvement
## 1           0       -5.25
## 2          27       -1.50
## 3          71       59.50
## 4          95       32.50
## 5           0       -7.25
## 6          28       23.50
## 7          71       14.75
## 8          95       70.75
## 9           0       -6.25
## 10         29       33.00
## 11         72       63.00
## 12         95       18.25
## 13          0       11.50
## 14         31        4.25
## 15         72        0.00
## 16         95       76.00
## 17          2       21.00
## 18         32       18.75
## 19         73       42.00
## 20         95       75.75
```

Figure 13: Cholostyramine data (some)

```
ggplot(cholost, aes(x = compliance, y = improvement)) + geom_point()
```

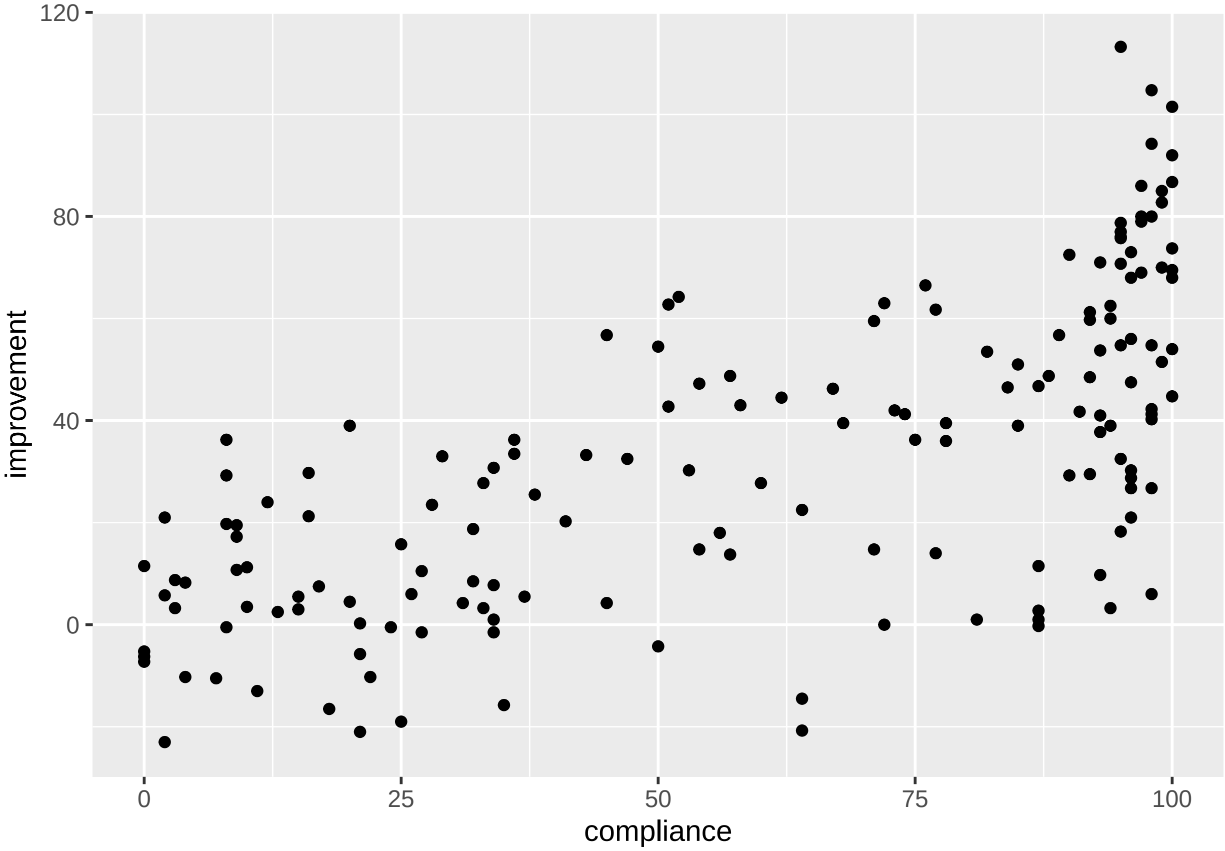


Figure 14: Cholostyramine scatterplot

```
cholost.1 <- lm(improvement ~ compliance, data = cholost)
summary(cholost.1)
```

```
##
## Call:
## lm(formula = improvement ~ compliance, data = cholost)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -55.83 -13.69   0.15  15.59  60.07
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.30725    3.44903  -0.669    0.504
## compliance   0.58410    0.04967  11.760   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 22.11 on 162 degrees of freedom
## Multiple R-squared:  0.4605, Adjusted R-squared:  0.4572
## F-statistic: 138.3 on 1 and 162 DF,  p-value: < 2.2e-16
```

Figure 15: Cholostyramine regression 1

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

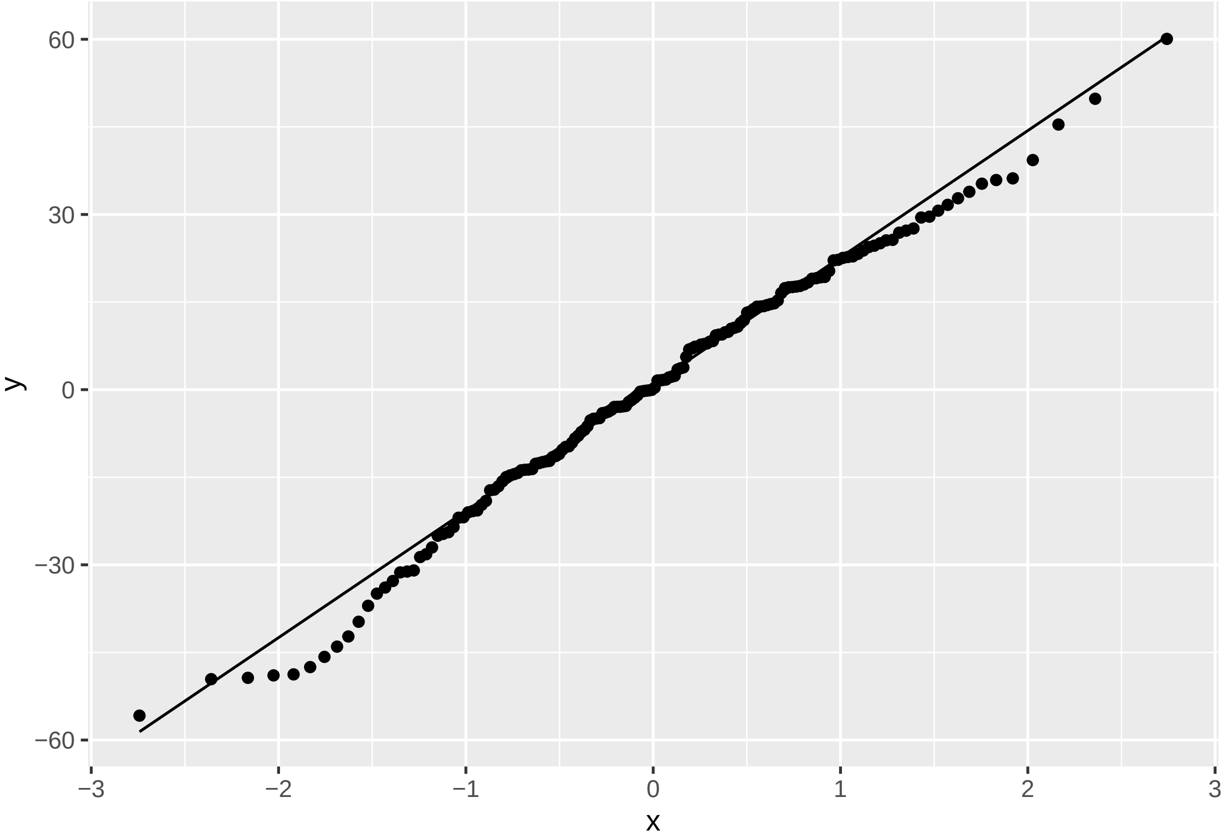


Figure 16: Residual plot 1 for cholostyramine data

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

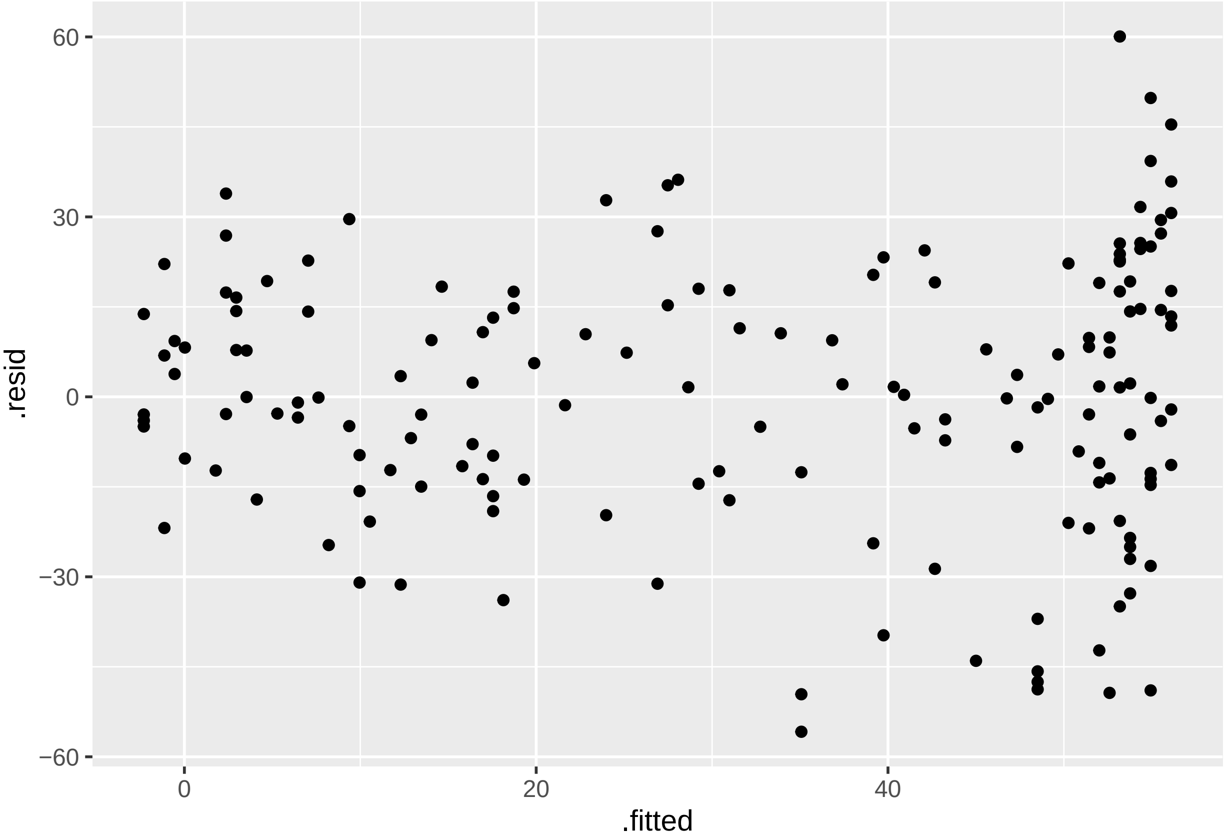


Figure 17: Residual plot 2 for cholostyramine data

```
cholost %>% mutate(bonus = (compliance >= 95)) -> cholost_bonus
cholost.2 <- lm(improvement ~ compliance + bonus, data = cholost_bonus)
summary(cholost.2)
```

```
##
## Call:
## lm(formula = improvement ~ compliance + bonus, data = cholost_bonus)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -56.951 -12.987   3.153  15.667  51.625
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.63250    3.49127   0.468  0.64071
## compliance   0.44178    0.06154   7.179 2.45e-11 ***
## bonusTRUE   18.02349    4.89995   3.678  0.00032 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 21.3 on 161 degrees of freedom
## Multiple R-squared:  0.5023, Adjusted R-squared:  0.4962
## F-statistic: 81.26 on 2 and 161 DF,  p-value: < 2.2e-16
```

Figure 18: Another regression for the cholostyrine data

```
charges %>% slice_sample(n = 20)
```

```
##    Sex     MD Svty  Chrg Age
## 3    M  MD730    1  1487  17
## 30   F  MD499    1  2499  39
## 28   M  MD499    3 15600  72
## 24   M  MD499    2  3535  20
## 12   F  MD730    2 14111  85
## 6    M  MD730    3 20280  61
## 18   F  MD730    3 24809  73
## 8    M  MD730    3 22382  90
## 37   F MD1021    4 64465  71
## 11   F  MD730    4 22642  77
## 33   M  MD499    3 15969  60
## 27   F  MD499    3 24121  86
## 5    M  MD730    2 18823  61
## 44   M MD1021    2  8759  56
## 38   F MD1021    3 17506  71
## 14   F  MD730    2 13343  65
## 7    F  MD730    1  4360  44
## 43   F MD1021    3 22734  66
## 31   M  MD499    3 12423  69
## 1    M  MD730    2  8254  57
```

Figure 19: Hospital charges data (20 randomly chosen rows)

```
charges %>%
  pivot_longer(c(Svty, Age)) %>%
  ggplot(aes(x = value, y = Chrg)) + geom_point() +
  facet_wrap(~name, scales = "free")
```

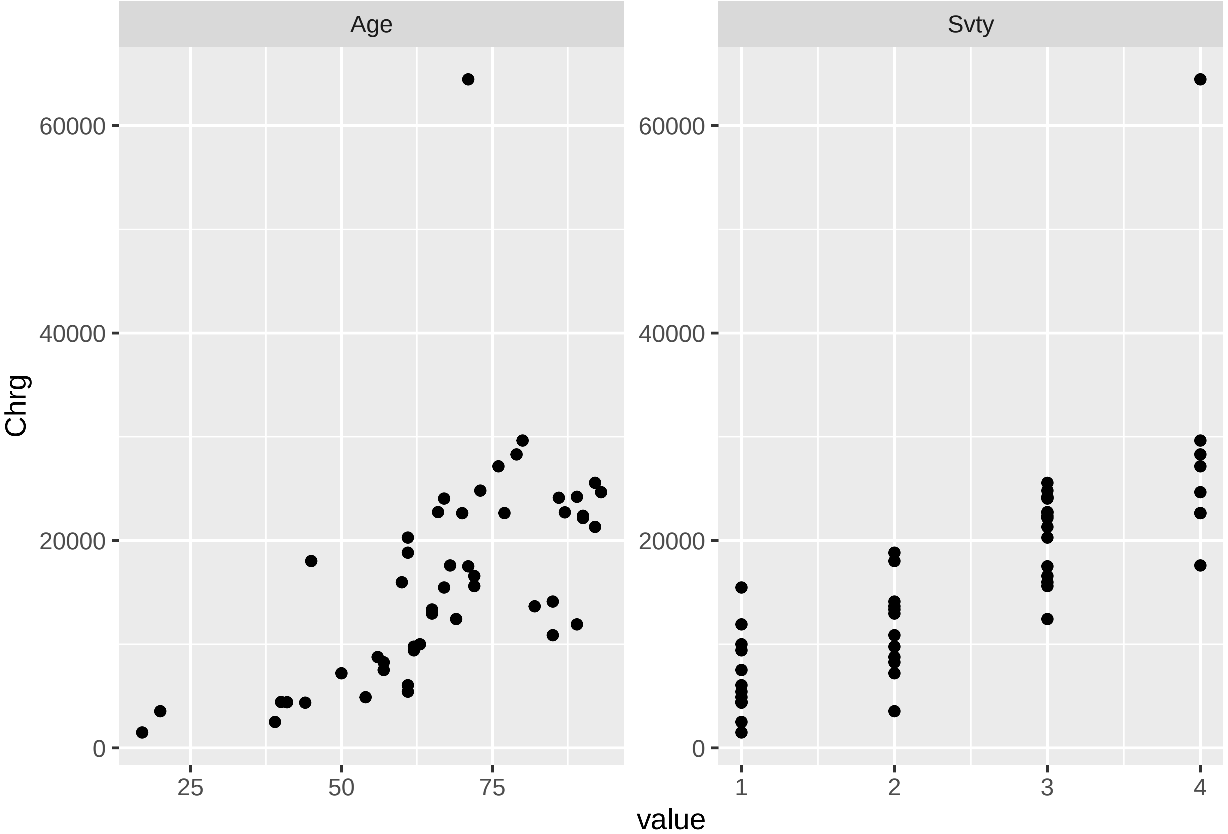


Figure 20: Plot of hospital charges against explanatory variables part 1

```
charges %>%
  pivot_longer(c(Sex, MD)) %>%
  ggplot(aes(x = value, y = Chrg)) + geom_boxplot() +
    facet_wrap(~name, scales = "free")
```

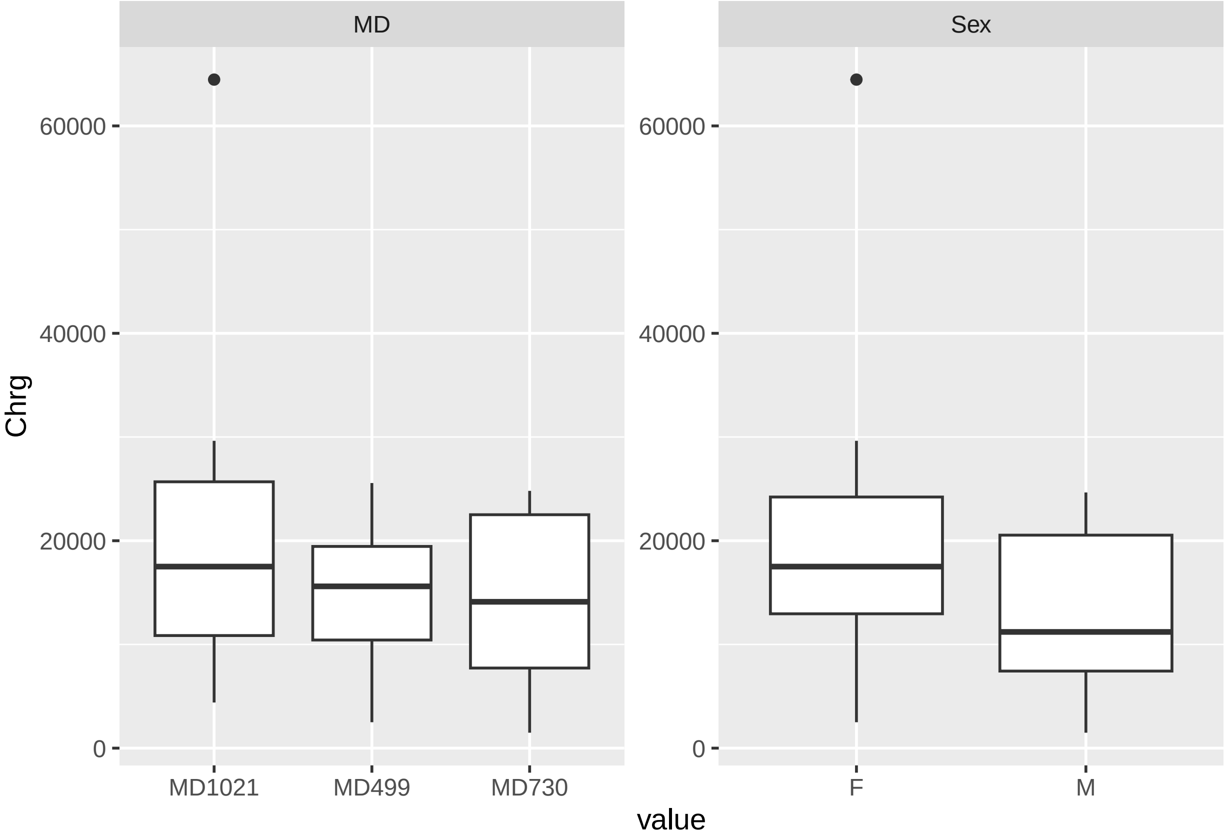


Figure 21: Plot of hospital charges against explanatory variables part 2

```
charges.1 <- lm(Chrg ~ Sex + MD + Svty + Age, data = charges)
summary(charges.1)
```

```
##
## Call:
## lm(formula = Chrg ~ Sex + MD + Svty + Age, data = charges)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -7435  -3094   -924   1661  33883
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3556.67    4211.82  -0.844   0.4031
## SexM        -1178.13    2076.91  -0.567   0.5735
## MDMD499     -5176.48    2402.16  -2.155   0.0368 *
## MDMD730     -3878.69    2389.86  -1.623   0.1119
## Svty         6292.14    1054.71   5.966  4.1e-07 ***
## Age           126.34      65.95   1.916   0.0621 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6405 on 43 degrees of freedom
## Multiple R-squared:  0.6684, Adjusted R-squared:  0.6299
## F-statistic: 17.34 on 5 and 43 DF,  p-value: 2.273e-09
```

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

```
## Single term deletions
##
## Model:
## Chrg ~ Sex + MD + Svty + Age
##        Df  Sum of Sq        RSS    AIC F value    Pr(>F)
## <none>                1763818288 864.55
## Sex     1   13198805 1777017093 862.91  0.3218   0.57349
## MD      2  201004856 1964823144 865.84  2.4501   0.09824 .
## Svty    1 1459873008 3223691295 892.10 35.5901 4.101e-07 ***
## Age     1  150508850 1914327138 866.56  3.6692   0.06209 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 22: Regression model and output

```
d
```

```
## # A tibble: 3 x 1
##   the_mean
##      <dbl>
## 1        4
## 2        8
## 3       24
```

Figure 23: Population means to use with your function for generating random normal data