MA 354: Data Analysis I – Fall 2021
Homework 4:

Complete the following opportunities to use what we've talked about in class. These questions will be graded for correctness, communication and succinctness. Ensure you show your work and explain your logic in a legible and refined submission.

0. **Complete weekly diagnostics.**

1. On its website, Ozempic, a medication for lowering the risk of major cardiovascular events (e.g., heart attack, stroke, etc.), states that

   - 66% of people taking 0.5 mg Ozempic
   - 73% of people taking 1 mg Ozempic
   - 40% of people taking 100 mg Januvia

   reached an A1C under 7%, noting higher A1C is indicative of higher risk of heart disease.

   (a) Explain why this statement alone isn't enough to conclude whether there is a statistically significant difference among the treatments.

   If we want to assess the statistically significant difference among the treatments, a t-sample proprotion test would be out best option. However, we don't have quite enough information to run it, since we have only the percents on our hands. We'd require more information (sample size, for example) before we can make any conclusions.

   (b) The statement on Ozempic's website comes from a phase 3a randomized double-blind study. Ahrén et al. (2017) reports that 409 received Ozempic (0.5 mg), 409 received Ozempic (1 mg), and 407 received Januvia (100 mg).

      i. Determine whether there is sufficient evidence of a difference in rates of attaining an A1C under 7% across treatments.

      Now, when we have more information about the treatments, we can use t-sample proportion test. However, before we do that, it would be highly advisable to check the assumptions first.
      — Data is representative: check. — Two possible outcomes (lowered/did not lower): check. — There are more than 10 instances of success and failure: check.

$$H_0 : \hat{p}_1 = \hat{p}_2 = \hat{p}_3$$
$$H_a : \hat{p}_1 \neq \hat{p}_2 \neq \hat{p}_3$$

```
# Successfu/total cases for 0.5 mg Ozempic
x1 = round(0.66*409)
n1 = 409

# Successful/total cases for 1 mg Ozempic
x2 = round(0.73*409)
n2 = 409

# Successful/total cases for 100 mg Januvia
x3 = round(0.4*407)
n3 = 407

answer1<-prop.test(x = c(x1, x2, x3), n = c(n1, n2, n3))
p.value1<-answer1$p.value
decision<-ifelse(p.value1<0.05, "is", "is not")
paste("The p-value: (", p.value1,")",
      sep="")
```

```
## [1] "The p-value: (5.01023071720164e-23)"
```

P-value provides us with statistically significant evidence to reject the null hypothesis in favor of the alternative.

ii. Perform a follow-up analysis for comparing treatments. If you were at high risk for cardiovascular events, which medication would you want to take.

```
answer2<-pairwise.prop.test(x = c(x1, x2, x3), n = c(n1, n2, n3))
answer2
##
##  Pairwise comparisons using Pairwise comparison of proportions
##
## data:  c(x1, x2, x3) out of c(n1, n2, n3)
##
##   1       2
## 2 0.033   -
## 3 3.6e-13 < 2e-16
##
## P value adjustment method: holm
```

The closer analysis of the pairwise proportion test provides us with the insight that Ozempic treatment is significantly better than the Januvia treatment (with $p < 0.001$). However, when it comes to the differences between Ozempic treatments, it's highly advisable to opt for 0.5mg treatment since it provided statistically better results compared to its 1mg alternative.

2. Is the ANOVA really robust to Normality? Equal sample size? Equal variance? To assess this we'll check the ability of ANOVA to detect differences in a sample and retain the $\alpha = 0.05$ across different settings. This homework question was motivated by Blanca et al. (2017) who published a simulation study about ANOVA.

   Remark: My professor in graduate school always told me that I didn't have to memorize any results, I could just derive them. The data analysis analog to this is that if we have any questions about how a model works under a given condition (or broken assumption) we can just simulate it!

   (a) Plot the Laplace distribution with $m = 0$ and $s = 2$; the PDF of this distribution is cataloged in `R` as `dlaplace()` in the rmutil package, which you'll need to install and load. Superimpose the graph of the Gaussian distribution with $\mu = 0$ and $\sigma = 2$. Comment on the differences you see and what you think might happen if the data are Laplace distributed instead of the Gaussian distribution.

```
library(rmutil)
library(tidyverse)

ggdat <- data.frame(x=seq(-5, 5, length.out=5000))%>%
mutate(f=dlaplace(x, m=0, s=2),
       f1=dnorm(x, mean=0, sd=2))

#set the legend
```

```
ggplot(ggdat, aes(x=x, y=f))+
  geom_line(color="blue")+
  geom_line(aes(y=f1), color="red")
```

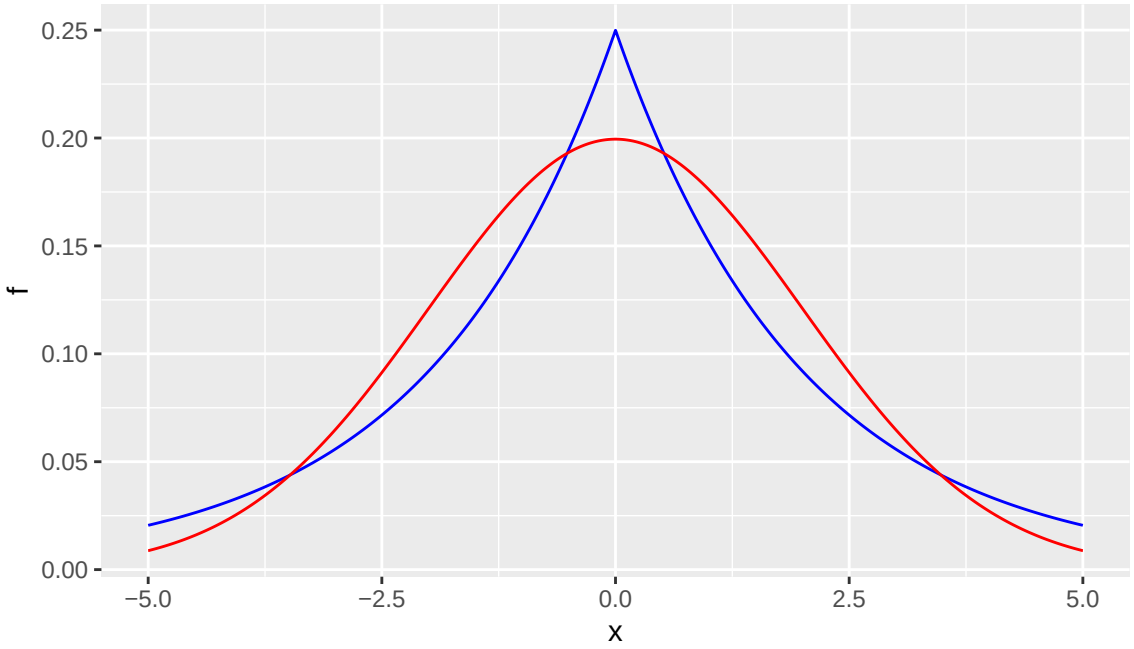

Figure 1: Laplace distribution with superimposed Gaussian distribution

Compared to the Guassian distribution, the Laplace one has higher kurtosis. In other terms, with the same mean, it has a higher peak. Moreover, we can observe fatter tails within the Laplae distribution, compared to its Gaussian counterpart.
I might be wrong here, the variance of the Laplace distribution reminds me of the two exponential distributions that face two opposite directions. I assume that if the data is following this distribution, we'd see more values in the tails of the distribution and at its mean.

(b) Conduct a simulation study using the Laplace distribution. To do so, complete the following 1000 times and report the proportion of times the data lead to a rejection of the null hypothesis.

The most efficient way to complete this question (including the other parts) is to write a function that completes the following.

- **Input:**
  - `rand.n=FALSE` – a logical object denoting whether the sample size is random or not. False by default. See part (f).
  - `rand.s=FALSE` – a logical object denoting whether the dispersion equal or not not. False by default. See part (g).
  - `equal.m=TRUE` – a logical object denoting whether the location parameters should be equal (part b) or different (part c). TRUE by default.
  - `n=5` – the desired sample size if not random. Five by default.
- **Loop the following tasks 1000 times:**
  - Generate $t = 4$ samples of size n drawn independently from the laplace distribution with m and s which can be done using `rlaplace()` function from the rmutil package (Swihart and Lindsey, 2020). Specify n, m, and s based on the values of the logical variables described above.
  - Perform the ANOVA procedure on these generated data.
  - Store whether the test resulted in a rejected null hypothesis or not.
- **Return:**
  - Your function should return the proportion of the 1000 ANOVA tests that resulted in a rejected null hypothesis.

Comment on the results of this simulation completed in the default case where $m_1 = m_2 = m_3 = m_4 = 0$, $s_1 = s_2 = s_3 = s_4 = 2$, and $n_1 = n_2 = n_3 = n_4 = 5$

```r
library(rstatix)

#Body of the function
aovFunc <- function(rand.n=FALSE, rand.s=FALSE, equal.m=TRUE, n=5, loop=0,
                    welch=FALSE){
  alpha<-0.05
  count=0
  #if random number of n is true, we'd use random number
  #from uniform distribution
  if(rand.n){
    n1=round(runif(1, min=5, max=100),0)
    n2=round(runif(1, min=5, max=100),0)
    n3=round(runif(1, min=5, max=100),0)
    n4=round(runif(1, min=5, max=100),0)
  }
  #else: use the number passed on within the function (default: 5)
  else{
    n1=n
    n2=n
    n3=n
    n4=n
  }

  if(rand.s){
    s1=rgamma(1, 2, 1)
    s2=rgamma(1, 2, 1)
    s3=rgamma(1, 2, 1)
    s4=rgamma(1, 2, 1)
  }else{
    s1=2
```

```
  s2=2
  s3=2
  s4=2
}
for(i in 1:loop){
  if(equal.m){
    #if mean is equal, pass 0 to the function
    t1<-rlaplace(n=n1, m=0, s=s1)
    label1<-"T1"

    t2<-rlaplace(n=n2, m=0, s=s2)
    label2<-"T2"

    t3<-rlaplace(n=n3, m=0, s=s3)
    label3<-"T3"

    t4<-rlaplace(n=n4, m=0, s=s4)
    label4<-"T4"
  }

  else{
    #else: pass 1 to the last function
    t1<-rlaplace(n=n1, m=0, s=s1)
    label1<-"T1"

    t2<-rlaplace(n=n2, m=0, s=s2)
    label2<-"T2"

    t3<-rlaplace(n=n3, m=0, s=s3)
    label3<-"T3"

    t4<-rlaplace(n=n4, m=1, s=s4)
    label4<-"T4"
  }


  #passing the data with labels
  dat<-data.frame(value=c(t1, t2, t3, t4),
                  group=c(rep(c("T1", "T2", "T3", "T4"),
                                                        times=c(length(t1),
                                                                length(t2),
                                                                length(t3),
                                                                length(t4)))))
  #if we're not using Welch's ANOVA, go here
  if(!welch){
    anova<-summary(aov(value~group, data=dat))
    sum_test <- unlist((anova))
    p.value<-sum_test["Pr(>F)1"]
    #print(p.value) #bugtest
  }
  #Use Welch's ANOVA to assess its results when S is different
  #NOT USED BUT STILL PRETTY COOL
  else{
    anova_w<-welch_anova_test(value~group, data=dat)
    p.value<-anova_w$p
  }
```

```
    if(p.value<0.05){
     count=count+1
    }
  }
  count/loop
}
```

```
aovFunc(loop=1000)
```

```
## [1] 0.041
```

On average, it would appear there were only less than 2 percent of cases when the function had enough evidence to reject the null hypothesis. No wonder: the sample size was really small and the means were equal, so that's exactly what we'd expect to see.

(c) Repeat the simulation study in (b-d), except with different means; i.e., $m_1 = m_2 = m_3 = 0$, and $m_4 = 1$. Comment on the results of this simulation.

```
aovFunc(equal.m=FALSE, loop=1000)
```

```
## [1] 0.075
```

Yet again, we're using a terribly small sample size, so, on average, only in less than 10 perecent of cases there was enough evidence to reject the null hypothesis. This suggests that ANOVA test majorly depends on its sample size. However, if we want to be certain in it, it would be highly advisable to simulate the same experiment with a bigger sample size.

(d) Repeat the simulation study in (b-c), except with $n = 15$. Comment on the results of this simulation.

```
aovFunc(equal.m=TRUE, loop=1000, n=15)
```

```
## [1] 0.039
```

```
aovFunc(equal.m=FALSE, loop=1000, n=15)
```

```
## [1] 0.148
```

Now, when we increased the sample size to 15, we can see certain changes within our function. When the means are not equal, ANOVA is able to reject every second test. However, that is equal to a coin toss, so $n = 15$ is still not enough to be certain in our results.

(e) Repeat the simulation study in (b-c), except with $n = 50$. Comment on the results of this simulation.

```
aovFunc(equal.m=TRUE, loop=1000, n=50)
```

```
## [1] 0.056
```

```
aovFunc(equal.m=FALSE, loop=1000, n=50)
```

```
## [1] 0.406
```

With a sample size as big as 50, on average, ANOVA is able to reject only less than 50-60 percent of hypothesis when the means are not equal. It implies that ANOVA is highly sensative to the sample size, and it's imperative for researchers to maintain it high!

(f) Repeat (b-e), except randomly select the sample size for each group by selecting n from the uniform(5,100) distribution. This will help us assess the robustness of the equal sample size assumption in the Laplace population distribution case. Comment on the results of this simulation.

```
aovFunc(equal.m=TRUE, rand.n=TRUE, loop=1000)
```

```
## [1] 0.041
```

```
aovFunc(equal.m=FALSE, rand.n=TRUE, loop=1000)
```

```
## [1] 0.622
```

Throughout various runs of this function, I failed to notice consistently high results from ANOVA. It is true that, on average, ANOVA managed to reject 40-50%+ of null hypothesis when means were not equal. However, it also failed to show consistently high results. I observed 20-30% of rejected hypothesis as many times as I saw 50-70% of rejected hypothesis. However, in the case of unequal sample size, I noticed that ANOVA did not show consistently worse results.

(g) Repeat (b-f), except randomly select the dispersion for each group by selecting s from the gamma(2,1) distribution. This will help us assess the robustness of the equal variance assumption in the Laplace population distribution case. Comment on the results of this simulation.

```
aovFunc(equal.m=TRUE, rand.n=TRUE, rand.s=TRUE, loop=1000)

## [1] 0.132

aovFunc(equal.m=FALSE, rand.n=TRUE, rand.s=TRUE, loop=1000)

## [1] 0.121
```

When dispersion is not equal (in addition to unequal sample size), ANOVA test manages to reject, on average, more less 50% of the hypothesis when the means are not equal. So while in the previous case this result might have been higher than 50%, an additional difference in variance greatly undermined the power of ANOVA test.

(h) Write a loop that conducts this simulation when (1) s is fixed and (2) when s is random, for the case where the means are unequal. The loop should be with respect to n, and should run for $n = 5$ to $n = 200$. **Note:** This can take some computation time, you'll want to run it and save the image as a .pdf so you can load it instead of rerunning the code.

```
results<-c()
n<-10:200
y1<-c()
y2<-c()

for(i in 10:200){
  print(i)
  unequal<-aovFunc(equal.m=FALSE, rand.s=TRUE, loop=1000, n=i)
  y1<-c(y1, unequal)
  equal<-aovFunc(equal.m=FALSE, rand.s=FALSE, loop=1000, n=i)
  y2<-c(y2, equal)
}
ggdat<-data.frame(x=n, equal.s=y2, unequal.s=y1)
```

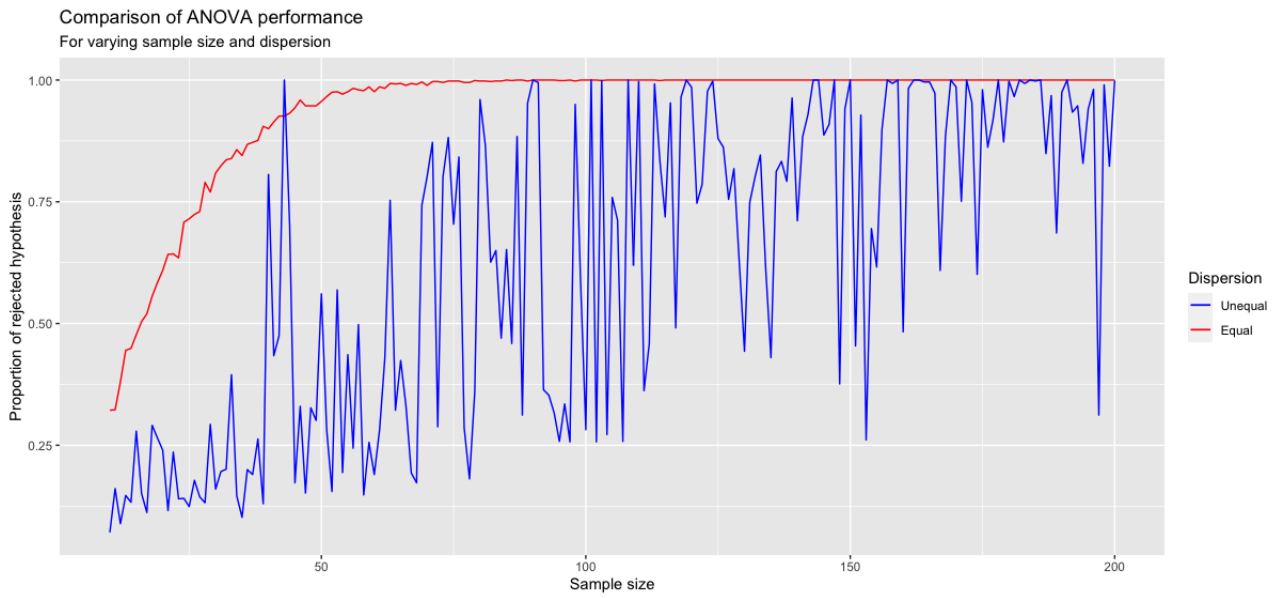

Figure 2: The assessment of ANOVA performance on Laplace distribution

This graph provides enough evidence to make a conclusion that the variance assumption is far more important for ANOVA than the sample size assumption. We can observe that even with massive sample size the function with unequal variance keeps underpeforming, compared to the with the equal variance. Hence, the reason of using Welch's ANOVA — it will eliminate the assumption about the equal variance.

3. Complete the following parts. This will lead you through the simulation of data, fitting regression lines and evaluating the assumptions.

   (a) Fit a model to the following simulated data. Make observations about the model equation and the Pearson correlation.

```
n=500
x<-sample(x = seq(0,5,0.01), size=n, replace=T)
y<-5*x + 3
```

```
ggdat<-data.frame(x=x, y=y)
ggplot(ggdat, aes(x=x, y=y))+
  geom_smooth(color="blue",
              method="lm",
              formula=y~x)+
  geom_point(shape=1,
             alpha=.3)+
  theme_bw()+
  annotate("text", x=1, y=22,
           label=paste("Pearson correlation:",
                       cor(x,y,method="pearson"),"\nPerfect!"))
```

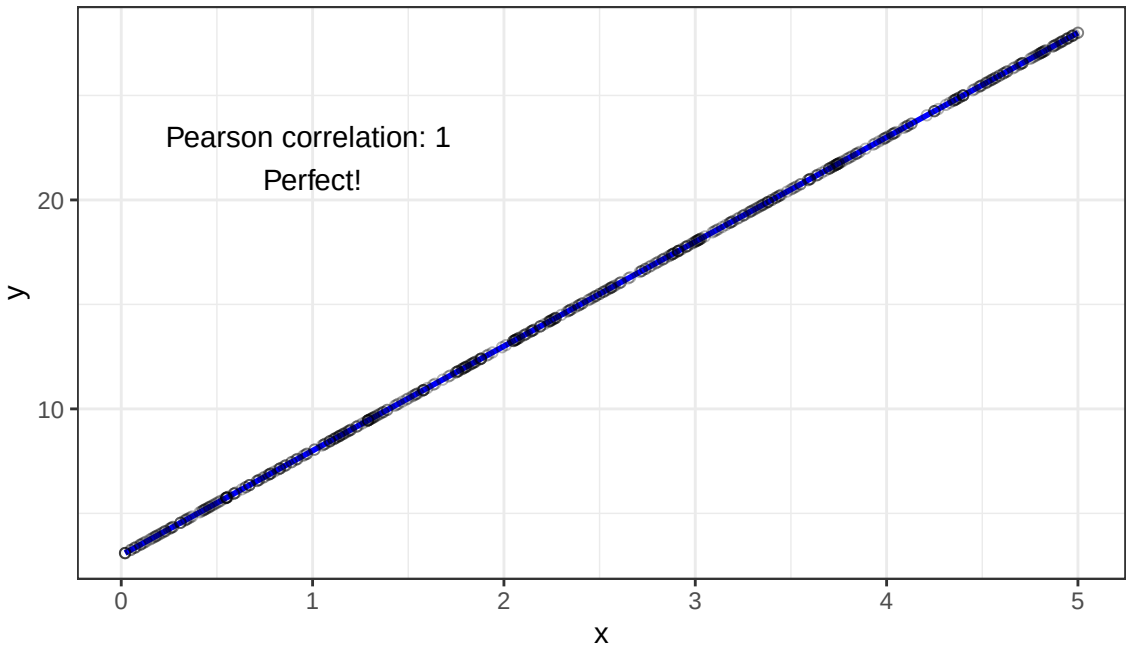

Figure 3: Fitted model for `x` and `y`

We have the Pearson correlation of one! It means that our linear regression perfectly aligns with the data that we have!

   (b) Fit a model to the following simulated data, now with added Normal error. Make observations about the model equation and the Pearson correlation in relation to (a).

```
e<-rnorm(n=n,mean=0,sd=3)
y2<-5*x + 3 + e
```

```
ggdat<-data.frame(x=x, y=y2)
ggplot(ggdat, aes(x=x, y=y))+
  geom_smooth(color="blue",
              method="lm",
              formula=y~x)+
  geom_point(shape=1,
             alpha=.3)+
  theme_bw()+
  annotate("text", x=1, y=25,
           label=paste("Pearson correlation:",
                       round(cor(x,y2,method="pearson"),2),"\nHighly correlated!"))+
  labs(title="Fitting the linear regression model",
       subtitle="With simulated data with added Normal error")
```

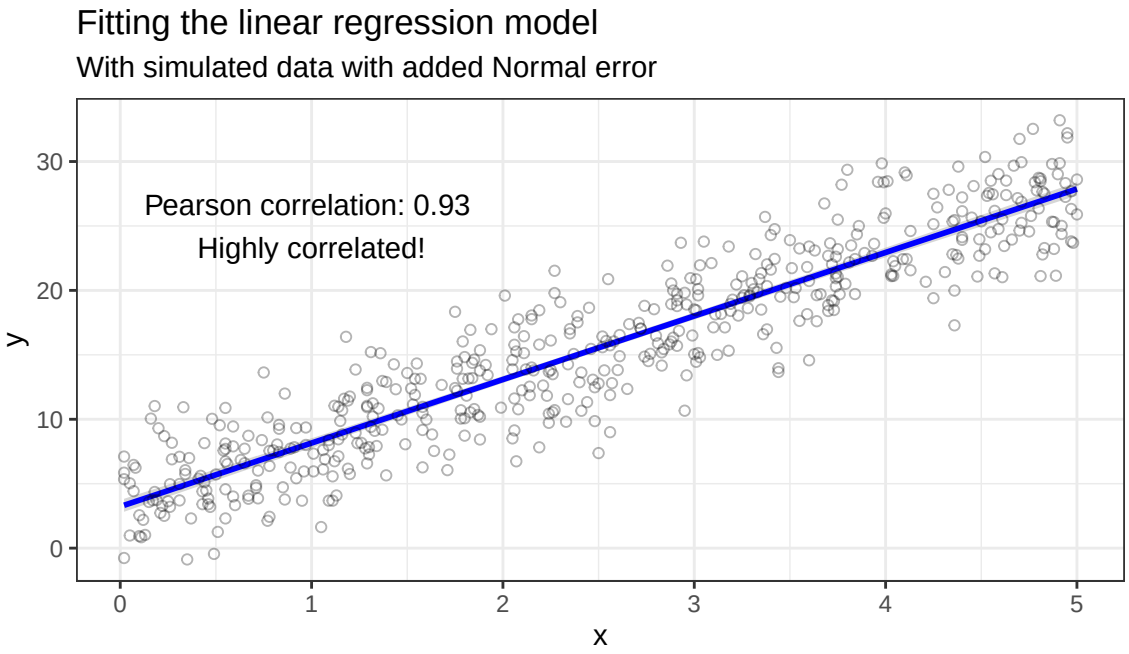

Figure 4: Fitted model for `x` and `y2`

Based on the visual inspection, the data in the figure 4 is homoscedastic (equal variance), and the Pearson correlation is 0.93! However, in order to be completely sure in our assessment, we ought to dig deeper and build additional graphs!

(c) In the model of part (b), evaluate the normality and homogeneity of error terms. Note that we know both of these items to be true since we've taken $\epsilon \sim \mathrm{N}(\mu = 0, \sigma = 3)$.

```
#Preparing the functions!
library(patchwork)
source("https://cipolli.com/students/code/plotResiduals.R")
source("https://cipolli.com/students/code/plotInfluence.R")
```

We'll start off with building `lm()` model for our data!

```
model1 <- lm(y2~x)
```

```
plotResiduals(model1)
```

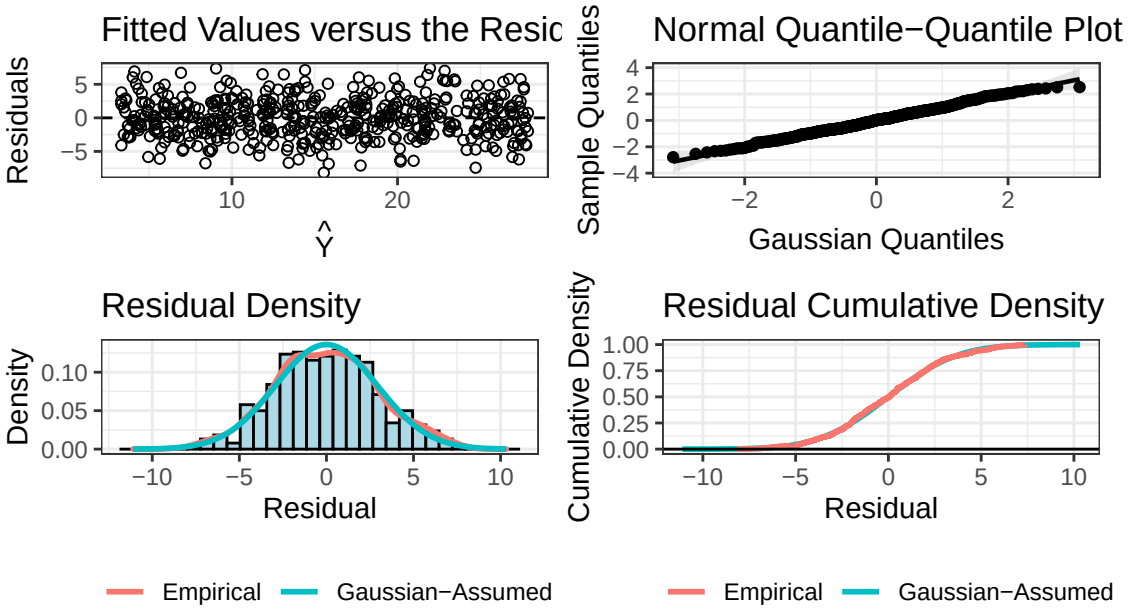

Figure 5: Simulated data: **Top Left:** A fitted versus residual scatterplot. **Top Right:** A normal q-q plot of the residuals. **Bottom Left:** A histogram of the residuals with a density curve. **Bottom Right:** A cumulitive density plot with an empirical density plot.

Looking at plot 5, it's obvious that residuals are Gaussian: q-q plot looks normal and assumed Gaussian distribution follows closely the observed distribution. In the fitted values versus residuals plot, we can observe that the data has equal variance.

(d) Fit a model to the following simulated data, now with added exponential error. Make observations about the model equation and the Pearson correlation in relation to the model of part (b).

```
e<-rexp(n=n,rate = 1/2)
y3<-5*x + 3 + e
```

```r
ggdat<-data.frame(x=x, y=y3)
ggplot(ggdat, aes(x=x, y=y3))+
  geom_smooth(color="blue",
              method="lm",
              formula=y~x)+
  geom_point(shape=1,
             alpha=.3)+
  theme_bw()+
  annotate("text", x=1, y=25,
           label=paste("Pearson correlation:",
                       round(cor(x,y3,method="pearson"),2),"\nHighly correlated!"))+
  labs(title="Fitting the linear regression model",
       subtitle="With simulated data with added exponential error")
```

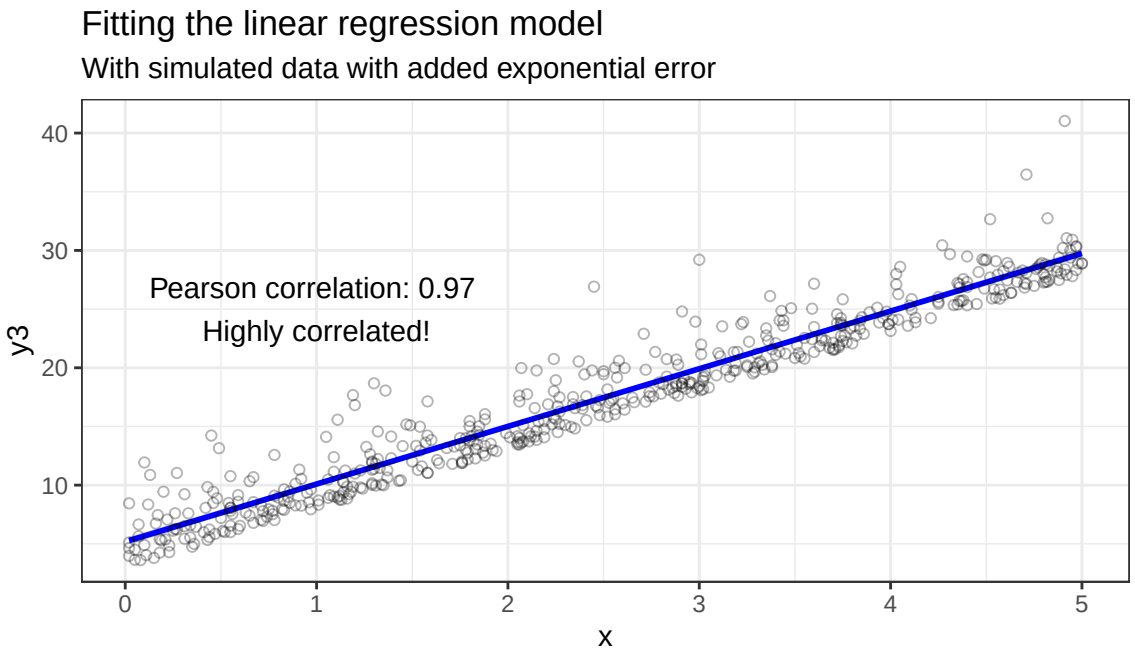

Figure 6: Fitted model for `x` and `y3`

The Pearson correlation tells us that values `x` and `y3` are highly correlated. Through visual inspection, it's easier to notice that the data apepars skewed towards one side. I would assume that the constant variance assumption holds true though! However, yet again, we ought to build more plots to assess the validity of my assumptions.

(e) In the model of part (d), evaluate the normality and homogeneity of error terms. Note that we know that common variance is true but we've taken $\epsilon \sim \exp(\beta = 2)$.

```r
model2 <- lm(y3~x)
#It's a skewed normal distribution, but it's homoscedastic!
```

```
plotResiduals(model2)
```

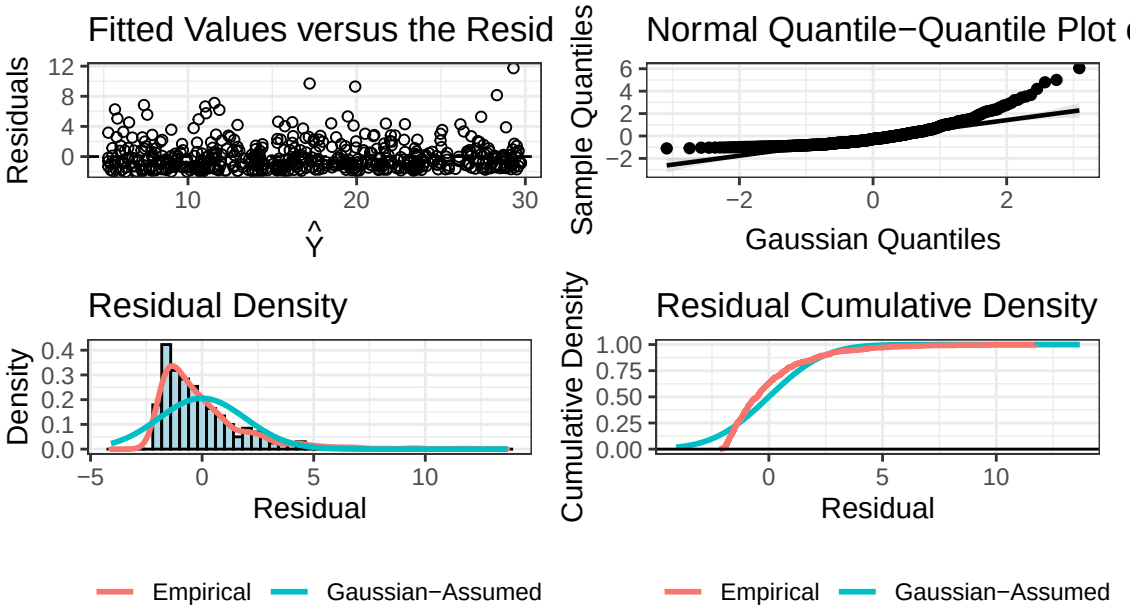

Figure 7: Simulated data: **Top Left:** A fitted versus residual scatterplot. **Top Right:** A normal q-q plot of the residuals. **Bottom Left:** A histogram of the residuals with a density curve. **Bottom Right:** A cumulitive density plot with an empirical density plot.

The first thing that's worth noting is that my visual inspection was correct. The Residual density histogram shows the skewed data. Moreoever, the ends of the q-q plot support this observation. However, it's worth noting that data appears to be homoscedastic! These differences lead to the observed differences in the cumulative density. We can be confident that errors do not follow the normal distribution!

(f) Fit a model to the following simulated data, now with added Heteroskedastic normal error. Make observations about the model equation and the Pearson correlation in relation to the model of part (b).

```
x4<-x[order(x)]
e<-rnorm(n=n,mean=0,sd=c(rep(1,n/2),rep(3,n/2)))
y4<-5*x4 + 3 + e
```

```r
ggdat<-data.frame(x=x4, y=y4)
ggplot(ggdat, aes(x=x, y=y))+
  geom_smooth(color="blue",
              method="lm",
              formula=y~x)+
  geom_point(shape=1,
             alpha=.3)+
  theme_bw()+
  annotate("text", x=1, y=25,
           label=paste("Pearson correlation:",
                       round(cor(x4,y4,method="pearson"),2),"\nHighly correlated!"))+
  labs(title="Fitting the linear regression model",
       subtitle="With simulated data with added heteroskedastic normal error")
```

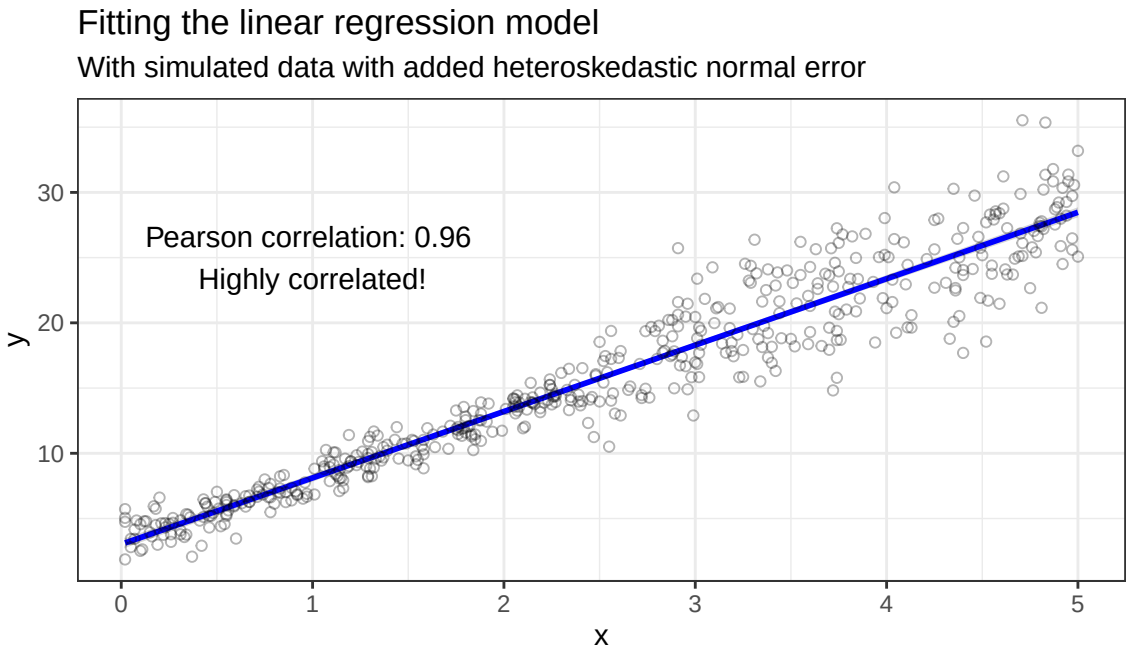

Figure 8: Fitted model for `x` and `y4`

The Pearson correlation keeps showing that values of `x` and `y` are highly correlated. However, it's important to stress that even the brief visual inspection allows us to notice the heteroskedastic nature of the data that we are working with! It would appear than variance is different for the values that are lower than 2.5 and the ones that are higher than that.

(g) In the model of part (f), evaluate the normality and homogeneity of error terms. Note that we know that normality of error terms is true, but $\epsilon \sim \mathrm{N}(\mu = 0, \sigma = 1)$ for $x < \widehat{m}$ and $\epsilon \sim \mathrm{N}(\mu = 0, \sigma = 3)$ for $x > \widehat{m}$.

```r
model3 <- lm(y4~x4)
```

```
plotResiduals(model3)
```

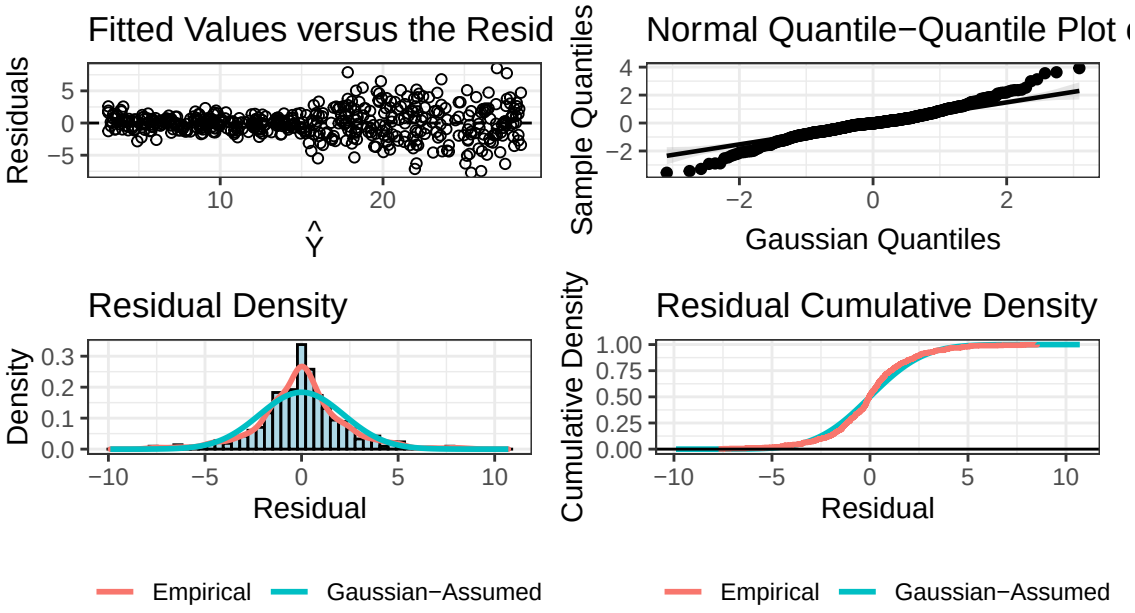

Figure 9: Simulated data: **Top Left:** A fitted versus residual scatterplot. **Top Right:** A normal q-q plot of the residuals. **Bottom Left:** A histogram of the residuals with a density curve. **Bottom Right:** A cumulitive density plot with an empirical density plot.

Unlike the plot7 that we've been working on before, the plot9's errors follow actually follow the normal distribution. The assumed Gaussian distribution fits incredibly well to the observed empirical distribution. However, this time we ought to pay attention to the the fitted values versus residuals plot. It shows the change in variance, implying that the data is heteroskedastic!

4. Consider the following simulation.

   (a) Plot the data simulated below. Assess the linear relationship.

```
library(tidyverse)
set.seed(7272)
n<-50
ggdat <- data.frame(x=sample(x=seq(0,100,0.01),size=n,replace=TRUE)) %>%
  mutate(y=3.5+2.1*x+rnorm(n=n,mean=0,sd=5))
```

   (b) Write out the population model.

$$\hat{Y} = 3.5 + (2.1 * x) + \epsilon$$

   It's also crucial to note that the population model's error is going to be normal.

   (c) Fit the model based on the sample data and write out the sample model below.

```
four.model<-lm(y~x, data=ggdat)
answer3<-summary(four.model)
answer3$coefficients
```

```
##             Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 5.222861 1.44260988  3.620425 7.068524e-04
## x           2.060557 0.02344535 87.887657 1.089004e-54
```

```
#Prediction = 5.22+2.06(x) + (error)
```

$$\hat{Y} = 5.22 + (2.06 * x) + \epsilon$$

(d) Add the regression line to the plot in black.

```
ggplot(ggdat, aes(x=x, y=y))+
  geom_smooth(color="black",
              method="lm",
              formula=y~x)+
  geom_point(shape=1,
             alpha=.3)+
  theme_bw()+
  labs(title="Fitting the linear regression model",
       subtitle="For simulated data")
```

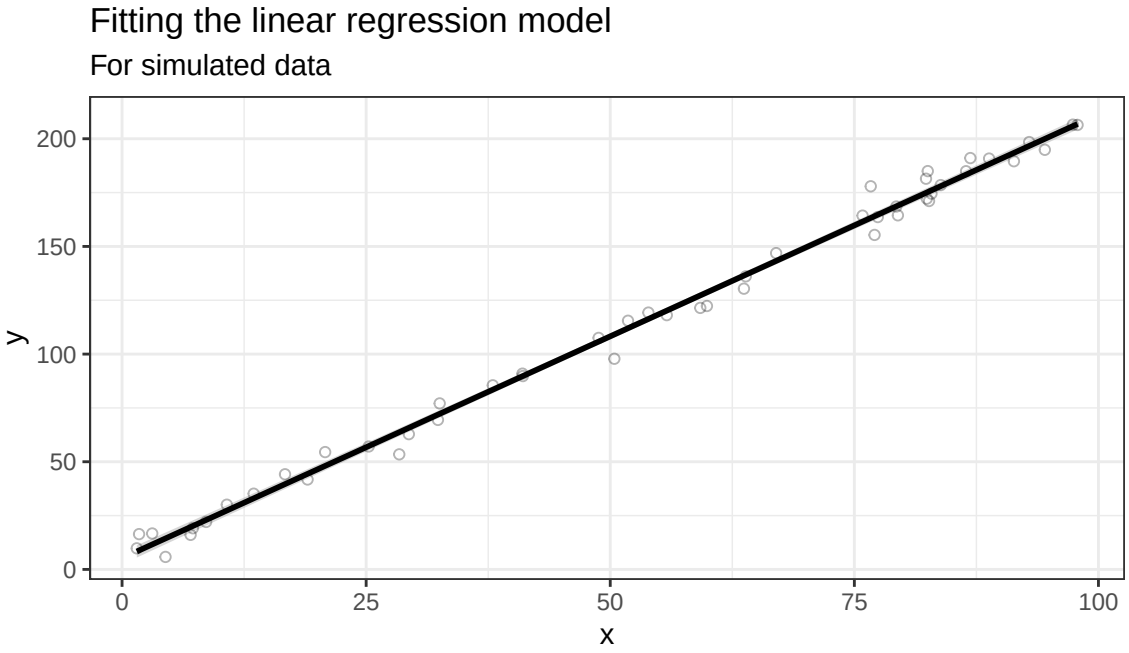

Figure 10: Fitted model for x and y

It would appear that plot10 manages to approximate our data reasonably well. In order to verify its ability to explain the observation that we have, I suggest using R^2 value!

(e) Interpret the R^2 of the model.

```
paste("Adjusted R-squared:", answer3$adj.r.squared)
```

```
## [1] "Adjusted R-squared: 0.993695511395507"
```

This implies that 99% of variance in data can be explained with the model that we've created!

(f) Interpret the overall F test of the model.

```
answer3$fstatistic
```

```
##   value   numdf   dendf
## 7724.24    1.00   48.00
```

We have an incredibly high F-statistic which implies that our model has great predictive capability. Moreover, F-statistic shows (with $p < 0.0001$) that our model provided us with the statistically significant results!

(g) Interpret the coefficients of the model; are they what you would expect? They're exactly what I expected them to be. The intercept begins at 5.2, which is approximately the first data point in our dataset. Then, with each additional value of x, the value of y increases by 2.06. It seems to approximate the data rather well!

(h) Now, let's add a bad datapoint to the data.

```
ggdat <- rbind(ggdat,      # original data
               c(100,25)) # bad observation
```

i. Briefly summarize how adding this data point affects parts (a)-(g). Before giving any assessments, I would rather check with BFFITs if the added observation is an outlier.

```
four.model<-lm(y~x, data=ggdat)
```

```
plotInfluence(four.model)
```

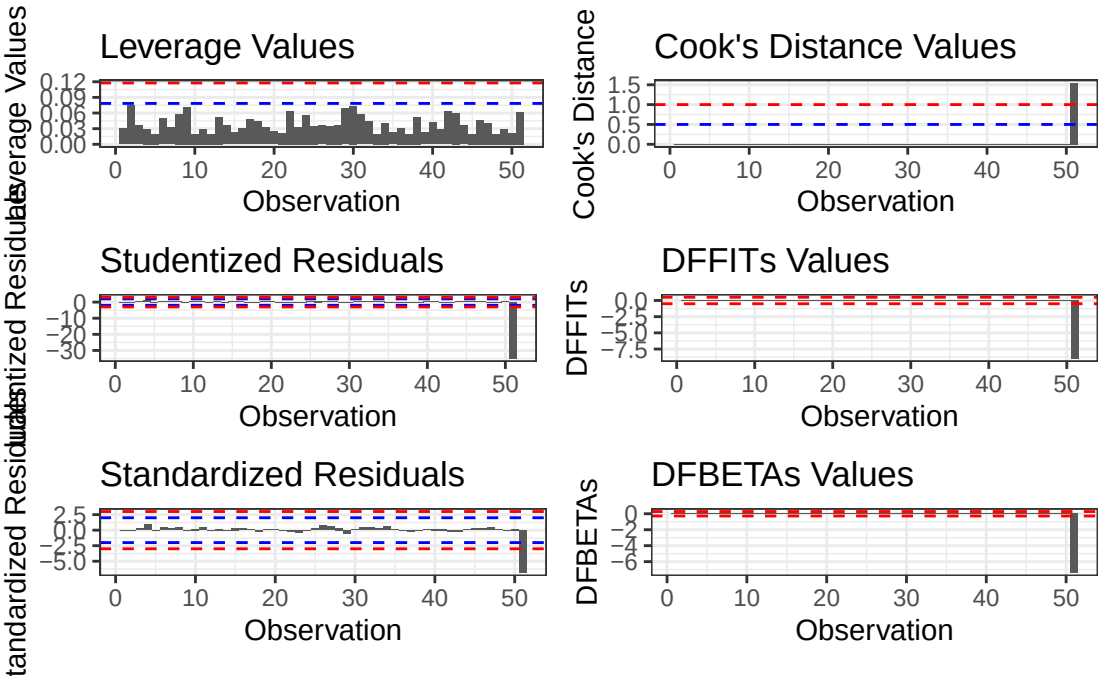

Figure 11: Plots for asessing the influence of outliers on the linear regression model

Based on both Cook's Distance values and DFFITs values, we have a massive outlier that has an oversized influence on our regression model, compared to the previous datapoints. Therefore, it would be highly advisable to come up with a way to deal with it.

ii. Add the resulting regression line to the plot in part (d) in blue.

```
#5.2+(2.06*x)
#10.742+(1.891*x)
ggdat.final <- ggdat %>%
  mutate(beforeAdding = 5.2+(2.06*x),
     afterAding = 10.742+(1.891*x))%>%
  pivot_longer(cols=ends_with("Adding"),
               names_to="Plot",
               values_to="Pred")
```

```
ggplot(ggdat.final, aes(x=x, y=Pred))+
  geom_smooth(aes(color=Plot),
              method="lm",
              formula=y~x)+
  geom_point(aes(y=y), shape=1,
             alpha=.3)+
  scale_color_manual(values=c("black", "blue"),
                     label=c("After",
                             "Before"))+
  labs(title="Fitting the linear regression models",
       subtitle="For simulated data with an outlier",
       color="Adding an outlier",
       y="y")
```

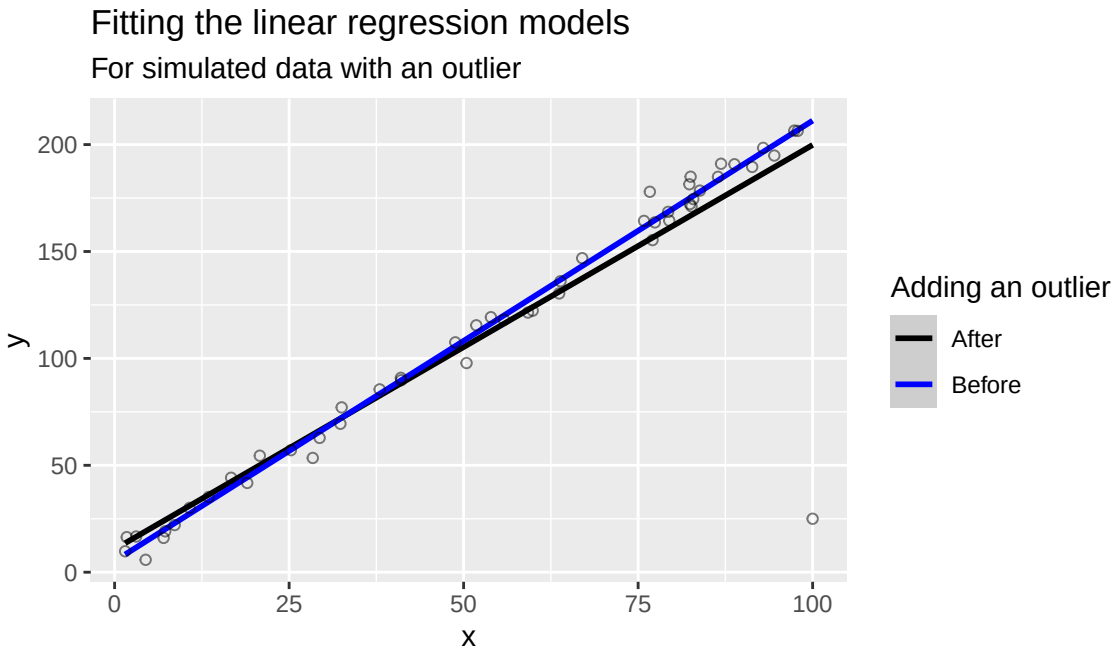

Figure 12: Plot assessing the influence of an outlier on the linear regression model

It would appear that the outlier pulls our linear model down. Therefore, we can expect it to undervalue its predictions regarding our data. Before we go on to using various techniques of dealing with the bad observation, it's important to stress that in the scientific environment we ought to speak with the researchers first. Based on the field and the research context, it might be reasonable to leave those observations in the model.

iii. Refit this model using several robust techniques for dealing with the bad observation. Create a plot that summarizes all the approaches taken, and use a metric to select the best model.
The first model we approach we are going to use is iteratively reweighted least squares.

```
library(MASS)
library(sfsmisc)
#Hubert
#6.0391 + (2.0421 * x) + e
mod.hubert <- rlm(y ~ x, data=ggdat,
                  psi=psi.huber)
summary(mod.hubert)
```

```
##
## Call: rlm(formula = y ~ x, data = ggdat, psi = psi.huber)
## Residuals:
```

```
##       Min        1Q    Median        3Q       Max
## -185.2474   -3.1410    0.5295    3.2745   15.3087
##
## Coefficients:
##             Value   Std. Error t value
## (Intercept)  6.0391  1.4224     4.2457
## x            2.0421  0.0228    89.7473
##
## Residual standard error: 4.747 on 49 degrees of freedom
```

Result:

$$Huber = 6.0391 + (2.0421 * x) + \epsilon$$

The second approach we're going to explore is the Bisquare iteratively reweighted least squares model.

```
#5.7016 + (2.0504 * x)
mod.bisquare<-rlm(y ~ x, data=ggdat,
                  psi=psi.bisquare)

summary(mod.bisquare)
##
## Call: rlm(formula = y ~ x, data = ggdat, psi = psi.bisquare)
## Residuals:
##      Min       1Q   Median       3Q      Max
## -185.739   -3.265    0.056    3.033   15.011
##
## Coefficients:
##             Value   Std. Error t value
## (Intercept)  5.7016  1.4342     3.9755
## x            2.0504  0.0229    89.3722
##
## Residual standard error: 4.607 on 49 degrees of freedom
f.robftest(mod.bisquare, var="x")
##
##  robust F-test (as if non-random weights)
##
## data:  from rlm(formula = y ~ x, data = ggdat, psi = psi.bisquare)
## F = 7888.7, p-value < 2.2e-16
## alternative hypothesis: true x is not equal to 0
```

Result:

$$Bisquare = 5.7016 + (2.0504 * x) + \epsilon$$

Since we are dealing with an outlier, it's also reasonable to use the quantile regression model. It's working with the median, so it won't be affected by the odd observation.

```
library(quantreg)
#5.52 + (2.05*x)
mod.quant <- rq(y~x, data=ggdat)
summary(mod.quant, se = "ker")
##
```

```
## Call: rq(formula = y ~ x, data = ggdat)
##
## tau: [1] 0.5
##
## Coefficients:
##             Value    Std. Error t value  Pr(>|t|)
## (Intercept)  5.52886  2.88708    1.91504  0.06133
## x            2.05271  0.04751   43.20164  0.00000
```

Result:

$$Quantile = 5.52 + (2.05 * x) + \epsilon$$

Finally, let's compare the results we've got!

```
#6.0391 + (2.0421 * x) | Huber
#5.7016 + (2.0504 * x) | Bisquare
#5.52 + (2.05*x) | Quantile

ggdat.compare <- ggdat %>%
  mutate(original_reg = 5.2+(2.06*x),
         OLS_reg = 10.742+(1.891*x),
         Huber_reg = 6.0391 + (2.0421 * x),
         Bisquare_reg = 5.7016 + (2.0504 * x),
         Quantile_reg = 5.52 + (2.05*x),
         Population_reg = 3.5+(2.1*x))%>%
  pivot_longer(cols=ends_with("_reg"),
               names_to="Plot",
               values_to="Pred")
```

```
ggplot(ggdat.compare, aes(x=x, y=Pred))+
  geom_smooth(aes(color=Plot),
              method="lm",
              formula=y~x)+
  geom_point(aes(y=y), shape=1,
             alpha=.3)+
  scale_color_manual(values=c("black", "blue", "green", "red", "yellow", "purple"),
                     labels=c("Bisquare", "Huber", "OLS", "Before outlier", "Population",
                              "Quantile"))+
  labs(title="Fitting the linear regression models",
       subtitle="For simulated data with an outlier",
       color="Technique",
       y="y")
```

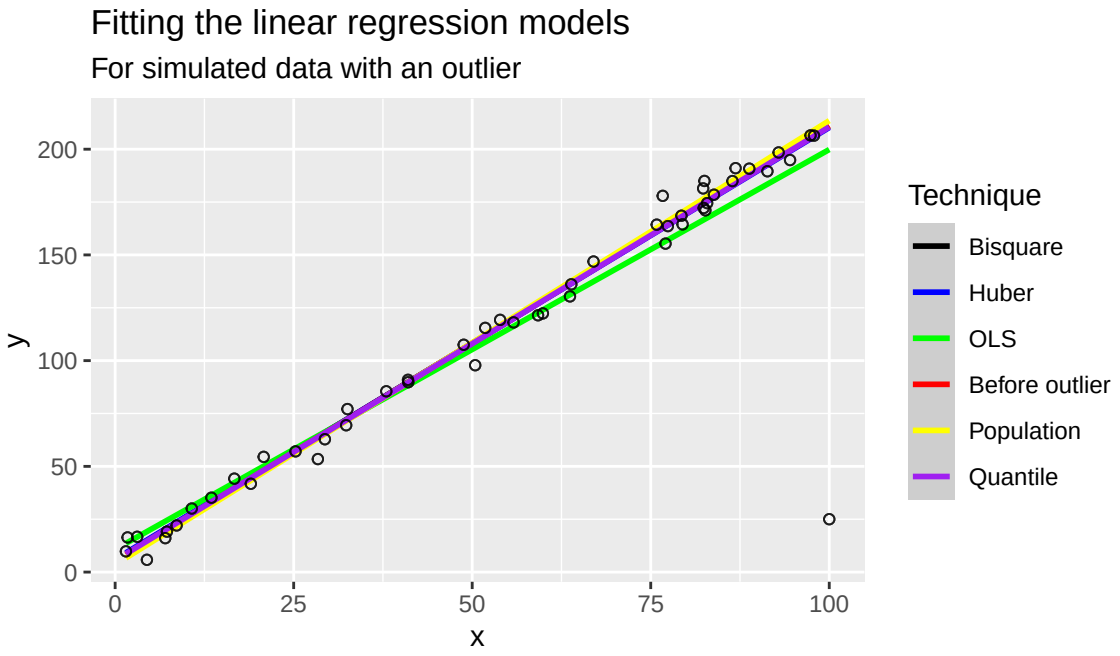

Figure 13: Plot assessing the effect of robust techniques of dealing with bad observations

It would appear that most of the robust models that we've built essentially overlap with each other. I suggest that we can use root mean squared error to assess the model that fits data the most! Based on the visual inspection alone, I suggest that we can drop the OLS model, since it's too influenced by the outlier.

```
library(Metrics)

rmse(ggdat$y, predict(mod.quant))
## [1] 26.49428
rmse(ggdat$y, predict(mod.hubert))
## [1] 26.42363
rmse(ggdat$y, predict(mod.bisquare))
## [1] 26.48649
```

It would appear that Huber's IRWLS gives back the best results, so I suggest that we should use this model to predict the values. However, it's still important to stress the point that I've made earlier: before making any decisions, it's highly advisable to talk to the research partners to clarify their opinion on it. We might not have to even adjust our linear regression model!
Communication is the key!

Thank you for this semester, it was so much fun!

References

Ahrén, B., Masmiquel, L., Kumar, H., Sargin, M., Karsbøl, J. D., Jacobsen, S. H., and Chow, F. (2017). Efficacy and safety of once-weekly semaglutide versus once-daily sitagliptin as an add-on to metformin, thiazolidinediones, or both, in patients with type 2 diabetes (sustain 2): a 56-week, double-blind, phase 3a, randomised trial. *The lancet Diabetes & endocrinology*, 5(5):341–354.

Blanca, M. J., Alarcón, R., Arnau, J., Bono, R., and Bendayan, R. (2017). Non-normal data: Is anova still a valid option? *Psicothema*, 29(4):552–557.

Swihart, B. and Lindsey, J. (2020). *rmutil: Utilities for Nonlinear Regression and Repeated Measurements Models*. R package version 1.1.5.