Lab 1

Week 2: Simple Linear Regression

Simran Johal

January 12, 2024

Introduction

In this lab, we will cover the following topics:

1. Simple Linear Regression with a Continuous Predictor
2. Simple Linear Regression with a Binary Predictor
3. Simple Linear Regression with Centering
4. Regression Assumptions

Packages

The following code checks whether the packages needed for today are already installed, if not, they will be installed.

Load required packages

```
library(dplyr)      # Data handling
library(foreign)    # Reading in SPSS data
library(ggplot2)    # Graphing
library(ggpubr)     # Graphing (arranging plots)
library(psych)      # For correlation test
library(DescTools)  # Summary statistics
library(kableExtra) # Table outputs for Markdown
```

Read in Lab Data

Today we will use one data set, described below.

The phobia data set is saved as an SPSS data file, so we will need to use read.spss() from the `foreign` package.

Note: In this data file there is hardly any missing data. However, there is one person with one missing value on the variable *SPAI_SP*. For ease of demonstration, we are going to exclude this person from the data set for this lab. We will review later how to handle data with missing values.

The other two are saved as a CSV files, so we will read it them in with the read.csv() function.

```
# To read in the SPSS data file:
phobia <- read.spss("phobiasocial.sav", to.data.frame = TRUE)
phobia <- phobia[which(!is.na(phobia$SPAI_SP)), ]
```

Social Phobia

The social phobia data set is from the following study:

Olivares, J., García-López, L. J., Hidalgo, M. D., & Caballo, V. (2004). Relationships Among Social Anxiety Measures and Their Invariance: A Confirmatory Factor Analysis. *European Journal of Psychological Assessment, 20*(3), 172–179. https://doi.org/10.1027/1015-5759.20.3.172

The `phobia` dataset contains the following variables:

- *SPAI_SP* is the total score on the Social Phobia and Anxiety Inventory: Social Phobia Subscale
- *SAS_FNE* is the total score on the Social Anxiety Scale for Adolescents: Fear of worries of negative evaluations from peers subscale
- *SAS_N* is the total score on the Social Anxiety Scale for Adolescents: Social avoidance and distress in new social situations subscale
- *SAS_G* is the total score on the Social Anxiety Scale for Adolescents: Social avoidance and distress generalized social inhibition subscale
- *FNE* is the total score on the Fear of Negative Evaluation Scale
- *SAD* is the total score on the Social Avoidance and Distress Scale
- *ADIS_IV* is the Anxiety Disorders Interview Schedule for DSM-IV (coded as 0 if no social phobia and 1 if social phobic).

```
# Lets look at the data set now:

phobia
```

Simple Linear Regression

"Regression" is a family of related analyses that focus on predicting outcomes (usually one at a time) based on one or more predictors. We will learn about many types of regression analyses in this class.

Today we will start with simple linear regression. A regression is considered "simple" if it only involves one outcome and one predictor. A regression is considered "linear" when the predictive linking functions assume some type of linear relationship between the predictor(s) and outcome.

Simple Regression with a Continuous Predictor

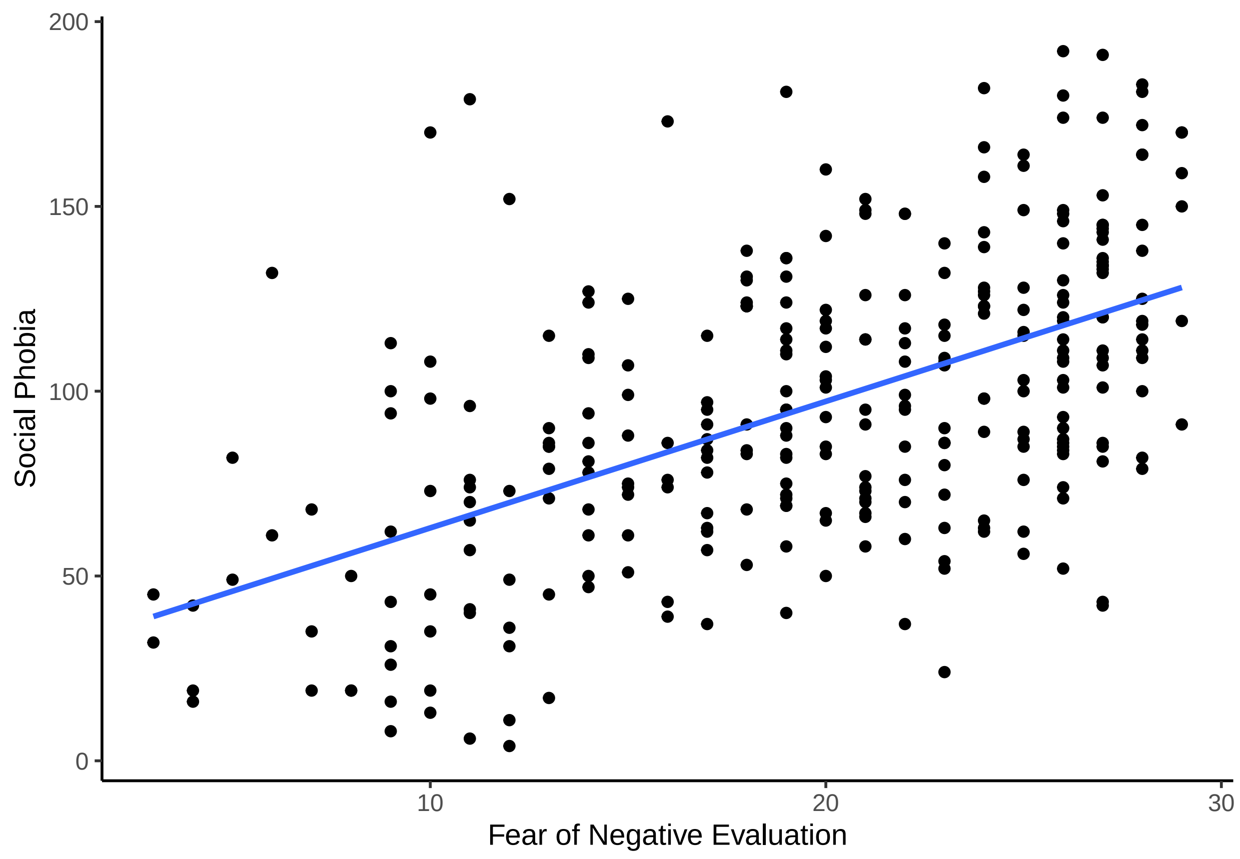
We can specify a simple linear regression model in R using the `lm()` function (which stands for "linear model").

The general format for this function is `lm(outcome ~ predictor)`.

In the previous example, we assessed the association between fear of negative evaluation and social phobia. Let's return to that example and do the analysis as a simple linear regression. We will test whether fear of negative evaluation (*FNE*) predicts levels of social phobia (*SPAI_SP*).

Before we dive into the analysis, lets start by looking at a scatter plot of the data. We typically put the predictor on the x axis and the outcome on the y axis.

```
ggplot(data = phobia, aes(y = SPAI_SP, x = FNE)) +
  geom_point() +
  geom_smooth(method = 'lm', se = F) +
  xlab('Fear of Negative Evaluation') +
  ylab('Social Phobia') +
  theme_classic()
```



It looks like there is a positive association between the two variables.

Now, lets do the regression.

1.
   ```r
   # Step 1: Save the linear model as an object
   model1 <- lm(SPAI_SP ~ FNE, data = phobia)

   # Step 2: Ask for a summary
   summary(model1)
   ```

   ```
   ##
   ## Call:
   ## lm(formula = SPAI_SP ~ FNE, data = phobia)
   ##
   ## Residuals:
   ##    Min     1Q Median     3Q    Max
   ## -83.52 -25.27  -0.49  21.88 112.57
   ##
   ## Coefficients:
   ##             Estimate Std. Error t value Pr(>|t|)
   ## (Intercept)  28.7652     6.3271   4.546 7.93e-06 ***
   ## FNE           3.4241     0.3047  11.236  < 2e-16 ***
   ## ---
   ## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
   ##
   ## Residual standard error: 33.73 on 300 degrees of freedom
   ## Multiple R-squared:  0.2962, Adjusted R-squared:  0.2938
   ## F-statistic: 126.2 on 1 and 300 DF,  p-value: < 2.2e-16
   ```

Lets walk through the output:

- **Call**: This tells you the regression equation that you specified.
- **Residuals**: This gives you a little more information about the residuals (how "off" each point was from the predicted score). We will return this a lot more later, but in general it is good to see a symmetrical distribution (in this case it seems like we have a more or less symmetrical distribution, but we would need to visualize it to check - more on this later).
- **Coefficients**: These are the coefficients (sometimes called estimates) from the regression equation (see below for how to interpret these numbers). The standard error is an error term in each coefficient, and the t-value is calculated as the estimate divided by the standard error. The p-value is obtained from the t-distribution, testing whether the Estimate is significantly different from 0.

We could use these coefficients to write the regression equation. We will write it in this form:

$$\widehat{y_i} = intercept + \beta x_{predictor}$$

This equation says "the predicted value of each data point of y is a function of the intercept plus x times beta". Here are a few key definitions that will help interpret what these values mean:

- $\widehat{y_i}$ The "hat" means "the predicted value of". The subscript i indicates "each value of". Hence, $\widehat{y_i}$ is the predicted value of each element of y.
- *intercept* is "the value of y when all other predictors in the model are equivalent to 0"
- β is the slope. It is how much you multiply each value of x by.
- $x_{predictor}$ is the value of the predictor variable, x.

Let's write the coefficients from the regression equation above in formula format.

$$\widehat{SPAI.SP_i} = 28.77 + 3.42x_{FNE}$$

How would we interpret the output and parameters?

1. ***(Intercept)*:** This is the model intercept. In conceptual terms, the intercept is the start point of the model. In statistical terms, **the intercept is the expected value of y when x is 0**. In our example, the expected value of social phobia is 28.77 when fear of negative evaluation is 0. This predicted value, 28.77, is significantly different from 0. For now, we don't really care too much about the intercept, as the value of "0 fear of negative evaluation" is not particularly meaningful (this is why we'll talk about centering later). We will soon cover situations when we would care about the intercept, though!
2. ***FNE***: This is the expected change in *SPAI_SP* for a one-unit change in *FNE*. In our example, a one-unit increase in fear of negative evaluation is associated with a 3.42 unit increase in social phobia (the units would be whatever metric *SPAI_SP* is on). This slope is significantly different from 0, indicating that Fear of Negative Evaluation is significantly associated with Social Phobia.

Here is an explanation of the remaining parts of the output above:

- **Residual Standard Error**: A measure of how well the model fits the data. It is a measure of variance among the residuals (how "off" each predicted value is from the observed value - more on this later). In general, smaller residual standard error is better. However, this measure is not on a standardized scale, it is on the metric of the original data, so it is not very meaningful right now.
- **Multiple R Squared**: How much total variance in the outcome the model accounts for. In this case, *FNE* (since that is the only predictor in our model) accounts for 29.62% of the variance in *SPAI_SP*.
- **F-statistic**: A statistic summarizing how well the overall model fits the data, with df and and a p value. Degrees of Freedom are calculated as the number of rows in the analysis minus number of predictors

in the model (one in this case) minus one. In this case, the overall model was significant (F(1, 300) = 126.1, $p < 0.001$). However, in a simple regression model this is not a very useful measure, as the overall model will always be significant if the one predictor in the model is significant. This measure will be more important when we learn about multiple regression, as it is possible for the overall model to be significant but to only have some predictors be significant, or for some predictors to be significant but not the overall model.

Using the regression equation above, we could make predictions about specific scores. For example:

```
# The predicted Social Phobia Score for someone with:

# A score of 10 FNE:
28.77 + (3.42*10)
```

```
## [1] 62.97
```

```
# A score of 15 FNE
28.77 + (3.42*15)
```

```
## [1] 80.07
```

```
# Someone with the highest level of FNE
28.77 + (3.42*max(phobia$FNE))
```

```
## [1] 127.95
```

```
# Someone with average levels of FNE
28.77 + (3.42*mean(phobia$FNE))
```

```
## [1] 96.35464
```

```
## Note: We could get the same results as above by extracting the coefficients
## directly from our regression object. To do this, we could use coef().

## This returns only the coefficients for the model
coef(model1)
```

```
## (Intercept)         FNE
##    28.76519     3.42408
```

```
b0 = coef(model1)[1] # The intercept
b1 = coef(model1)[2] # The slope

## Substituting these values in:
b0 + b1*10 # A score of 10 FNE
```

```
## (Intercept)
##      63.006
```

```
b0 + b1*15 # A score of 15 FNE
```

```
## (Intercept)
##     80.1264
```

```
b0 + b1*max(phobia$FNE) # The highest value of FNE
```

```
## (Intercept)
##    128.0635
```

```
b0 + b1*mean(phobia$FNE) # The average level of FNE
```

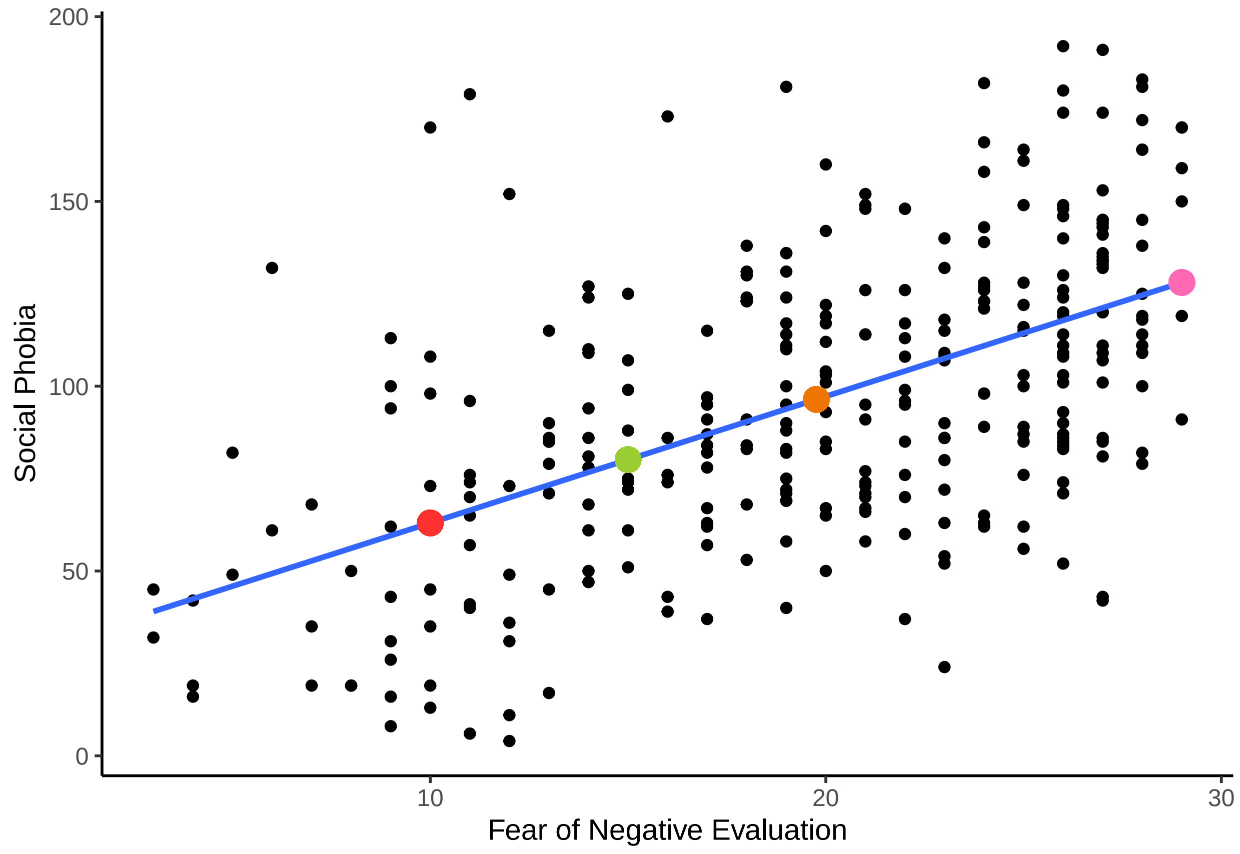
```
## (Intercept)
##    96.43046
```

An import point about regression is that these predicted values define the line of best fit on the graph. For example, lets see what the above predicted points look like on the original scatter plot:

```
ggplot(data = phobia,
       aes(y = SPAI_SP, x = FNE)) +

  # The basic scatter-plot
  geom_point() +
  geom_smooth(method = 'lm', se = F) + # This is what adds the line of best fit
  xlab('Fear of Negative Evaluation') +
  ylab('Social Phobia') +
  theme_classic()  +

  # To highlight specific points on the graph
  geom_point(aes(x = 10,
                 y = coef(model1)[1] + (coef(model1)[2]*10) ),
             color = "firebrick1",
             size = 4) +
  geom_point(aes(x = 15,
                 y = coef(model1)[1] + (coef(model1)[2]*15) ),
             color = "yellowgreen",
             size = 4) +
  geom_point(aes(x = mean(FNE),
                 y = coef(model1)[1] + (coef(model1)[2]*mean(FNE)) ),
             color = "darkorange2",
             size = 4) +
  geom_point(aes(x = max(FNE),
                 y = coef(model1)[1] + (coef(model1)[2]*max(FNE)) ),
             color = "hotpink",
             size = 4)
```



You will see that at each value of x that we considered, the predicted score of y falls exactly on the line of best fit.

We can get confidence intervals for the estimates using the `conf.int` function.

```
confint(model1)
```

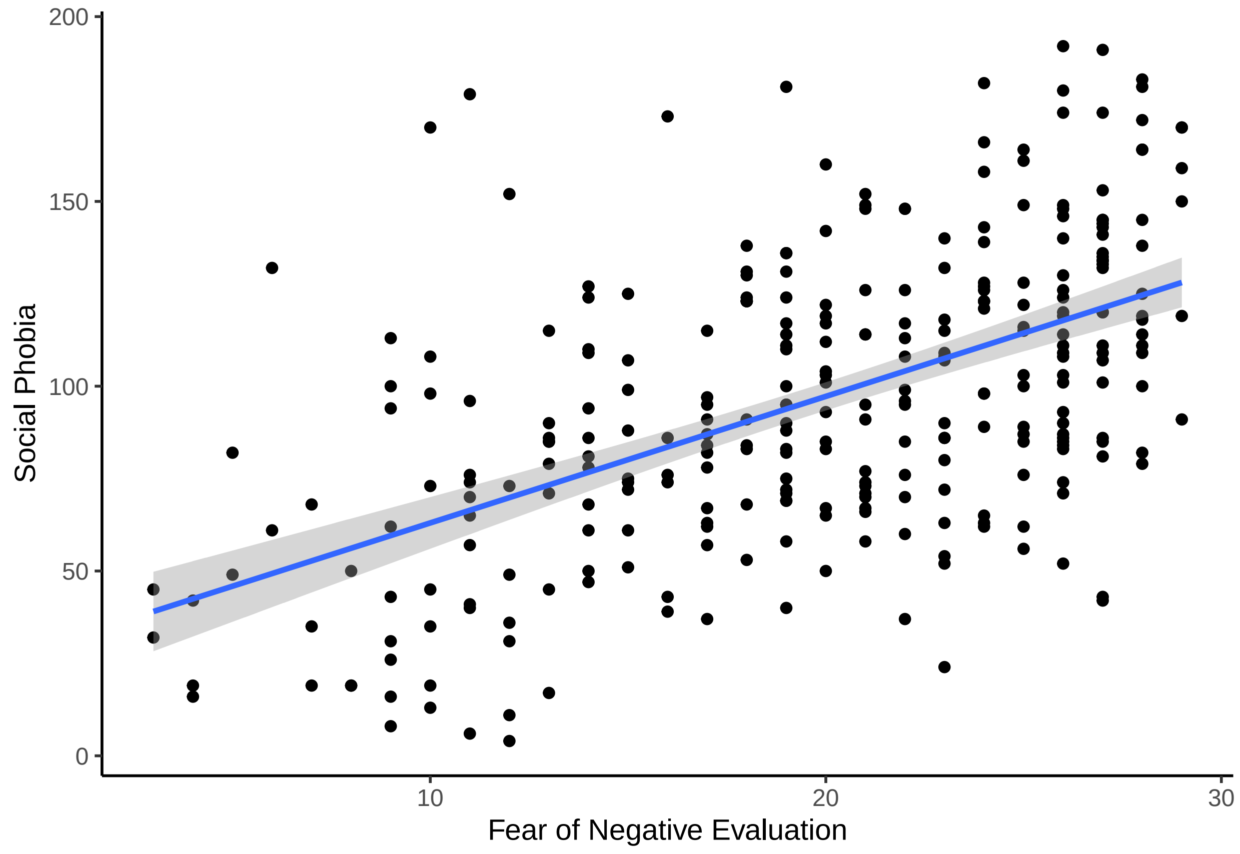
```
##                 2.5 %    97.5 %
## (Intercept) 16.313996 41.216391
## FNE          2.824383  4.023778
```

Standard Error of Prediction

Thus far, in the graphs that I have shown you, I did not include the standard error of prediction around the fitted line. It looks like this:

```
ggplot(data = phobia, aes(y = SPAI_SP, x = FNE)) +
  geom_point() +
  geom_smooth(method = 'lm', se = T) +
  xlab('Fear of Negative Evaluation') +
  ylab('Social Phobia') +
  theme_classic()
```

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



```
# Note: in the geom_smooth() function that adds the fit line, se = T is the default
# and doesn't need to be directly specified. I specified it here for demonstration.
```

The grey region represents the standard error of prediction. It tells us the region where the "true" linear prediction line might fall. We will learn more about how this is calculated in later classes. But for now, just understand that this is a region when the predicted line might fall, and that it is not linearly related to the prediction line (if it was, the grey region would not expand at different parts of the graph). For example, notice that the parts of the graph that have the widest grey region are the same parts of the graph with fewer data points. This is because when there are fewer data points, our prediction is less certain (because the less information we use to make a prediction, the less certain we are!).

Note: This grey region is not the same as the individual SE estimates for the slope and estimate, which indicate the standard error ("the standard deviation of the sampling distribution") for each estimate.

Centering Predictors

Lets return to the continuous regression that we did above, using *FNE* to predict *SPAI_SP*. As a reminder, this is what the analysis output was:

```
summary(model1)
```

```
##
## Call:
## lm(formula = SPAI_SP ~ FNE, data = phobia)
##
## Residuals:
##    Min      1Q Median      3Q     Max
## -83.52 -25.27  -0.49  21.88 112.57
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  28.7652     6.3271   4.546 7.93e-06 ***
## FNE           3.4241     0.3047  11.236  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 33.73 on 300 degrees of freedom
## Multiple R-squared:  0.2962, Adjusted R-squared:  0.2938
## F-statistic: 126.2 on 1 and 300 DF,  p-value: < 2.2e-16
```

In the analysis above, our intercept (28.77) refers to the expected value of *SPAI_SP* when *FNE* is 0. But what does it mean to have "zero Fear of Expected Negative Evaluation"? Does it mean "no fear"? Probably not (is this even possible from a conceptual stand point?). What about from a methodological stand point? Is it even possible to have a score of 0 on this scale? Note, the lowest observed value of *FNE* in our data is 3, and because *FNE* is likely a composite measure from summed survey items, my guess is that zero is not a possible score.

To make the intercept more meaningful, we can center the predictor variable. There are various options for how we center variables (e.g., centering at the mean, maximum, minimum values, etc.).

For this example, let's mean-center *FNE*, and run our analysis again. Now, to have "zero Fear of Expected Negative Evaluation" means you have average fear of negative evaluation.

```r
# First, create the centered variable in the data
phobia$FNE_c <- phobia$FNE - mean(phobia$FNE, na.rm = T)

# Now run the regression and create a new regression object
model1_c <- lm(SPAI_SP ~ FNE_c, data = phobia) # Using FNE_c
summary(model1_c)
```

```
##
## Call:
## lm(formula = SPAI_SP ~ FNE_c, data = phobia)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -83.52 -25.27  -0.49  21.88 112.57
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  96.4305     1.9408   49.69   <2e-16 ***
## FNE_c         3.4241     0.3047   11.24   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 33.73 on 300 degrees of freedom
## Multiple R-squared:  0.2962, Adjusted R-squared:  0.2938
## F-statistic: 126.2 on 1 and 300 DF,  p-value: < 2.2e-16
```

How do our interpretations change after centering *FNE*?

The intercept is now interpreted as the expected value of *SPAI_SP* when *FNE* is at its mean value. In our example, the intercept indicates that the expected value of social phobia for a person with an average level of fear of negative evaluation is 96.43. Notice that the t value, SE, and p values also changed. Because $p < 0.05$, we would conclude that people with average levels of FNE have significantly non-zero levels of SPAI_SP.

This could potentially be meaningful, depending on what "zero *SPAI_SP*" means.

Note that the interpretation and value of the slope did not change when we mean centered the data. That is because when you center data (or do any other kind of linear transformation), you do not change the linear relationship between x and y.

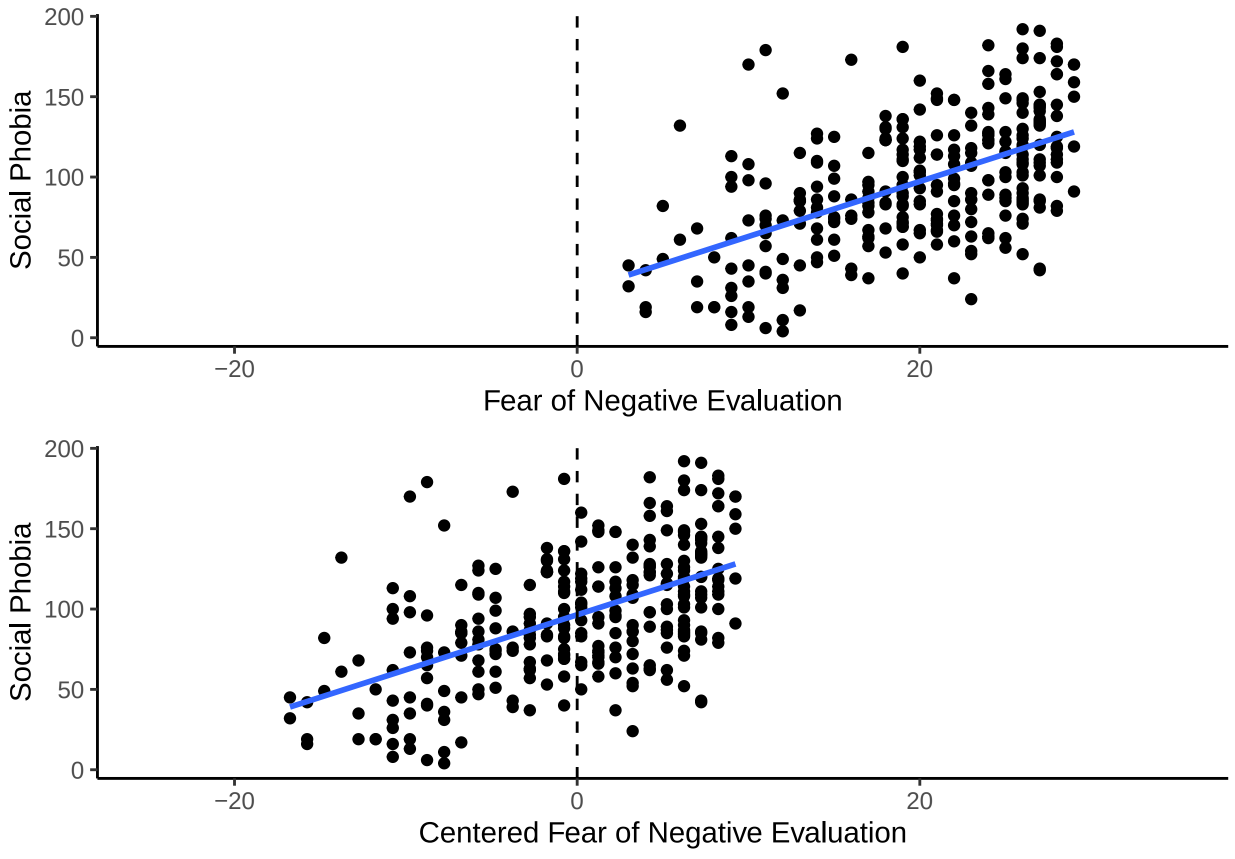
Lets graph the centered and un-centered regression to see how this works.

```r
# Original, uncentered Data
g1 <- ggplot(data = phobia,
       aes(y = SPAI_SP, x = FNE)) +
  geom_point() +
  theme_classic() +
  xlab('Fear of Negative Evaluation') +
  ylab('Social Phobia') +
  geom_smooth(method = 'lm', se = F) +
  geom_vline(xintercept =  0, linetype = "dashed")  +
  coord_cartesian(xlim = c(-25, 35))

# Mean-centered FNE
g2 <- ggplot(data = phobia,
       aes(y = SPAI_SP, x = FNE_c)) +
  geom_point() +
  theme_classic() +
  xlab('Centered Fear of Negative Evaluation') +
  ylab('Social Phobia') +
  geom_smooth(method = 'lm', se = F) +
  geom_vline(xintercept =  0, linetype = "dashed") +
  coord_cartesian(xlim = c(-25, 35))

ggarrange(g1, g2, nrow = 2)
```

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



```
# Note: To make this graph, I created two ggplot graphs and saved them as objects,
# and then I used the ggarrange function from the ggpubr package to create a single
# graph with the two graphs I just made.
```

Notice that the slope of the line is the same in both graphs; the only thing that changed was where 0 intersects.

Standardizing Predictors (and beta weights)

If we wanted to take a step further beyond centering, you could standardize (z-score) your data. When you do a simple regression with z-scored data, the Estimate that is output will be on a standardized scale, and will be equal to the Pearson r correlation (this only applies for simple regression, not multiple regression). We call the standardized slope a "beta weight".

Lets redo the above analysis using standardized values. Note: All variables in the regression must be standardized to obtain beta-weights.

```
phobia$FNE_z <- scale(phobia$FNE)
phobia$SPAI_SP_z <- scale(phobia$SPAI_SP)

model1_z <- lm(SPAI_SP_z ~ FNE_z, data = phobia)
summary(model1_z)
```

```
##
## Call:
## lm(formula = SPAI_SP_z ~ FNE_z, data = phobia)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -2.08094 -0.62960 -0.01222  0.54503  2.80477
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.866e-16  4.836e-02    0.00        1
## FNE_z        5.442e-01  4.844e-02   11.24   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8403 on 300 degrees of freedom
## Multiple R-squared:  0.2962, Adjusted R-squared:  0.2938
## F-statistic: 126.2 on 1 and 300 DF,  p-value: < 2.2e-16
```

```
# Note: you could get the same output all in one line and without creating new
# variables in your data frame, if you prefer:

# summary(lm(scale(SPAI_SP) ~ scale(FNE), data = phobia))
```

Lets walk through the output.

- **Intercept**: The value is essentially 0 (the t-test shows this value is not significantly different from 0). In other words, people with average *FNE* are predicted to have average *SPAI_SP*.
- *Slope*: The Estimate is 0.54. This means that "for every one unit increase in *FNE_z* (i.e., for every 1 SD increase in *FNE*, because we are on the z-scale), there is a predicted 0.54 unit increase in *SPAI_SP* (i.e., a 0.54 SD inrease in *SPAI_SP*)". In other words, for every 1 SD change in *FNE*, there is a 0.54 SD increase in *SPAI_SP*. Note, this would be a technical interpretation, and we normally wouldn't use that language in a formal manuscript. We normally would just talk about them in terms of Cohen's guidelines for effect sizes. So, we would consider a beta weight of 0.54 as a moderately strong correlation.

Notice that even though the Estimate and SE changed for the FNE estimate (because we changed the scale of the variables), the t-value and p-value did not change (because we transformed the data linearly, and did not change the degree of association between the data points).

Also notice that the beta-weight above is equal to the Pearson r correlation:

```
cor(phobia$SPAI_SP, phobia$FNE, use = "complete.obs")
```

```
## [1] 0.5442308
```

Binary Predictors

In the previous example, we used a continuous variable to predict a continuous outcome. In this section, we will demonstrate that we can also use binary variables to predict continuous outcomes using simple regression. This would be the equivalent of a t-test. Note, in a simple regression, the outcome variable needs to be continuous (we will cover how to predict binary outcomes using logistic regression in a later lab).

Binary predictors work best with regression when they are coded as 0 and 1. If you have a binary categorical variable that is not coded as 0 and 1, in most cases, R will automatically re-code it for you to be 0 and 1.

Usually when we have two-level grouping data, we think in terms of average differences between groups. For example, lets say that we wanted to know if average Fear of Negative Evaluation (*FNE*) differed between people with and without a social phobia diagnosis according to formal diagnostic criteria (*ADIS_IV*, where 0 = No and 1 = Yes).

One of the easiest ways to answer this question would be with a `t.test`. For example:

```r
t.test(phobia$FNE ~ phobia$ADIS_IV, var.equal = T)
```

```
##
##  Two Sample t-test
##
## data:  phobia$FNE by phobia$ADIS_IV
## t = -12.461, df = 300, p-value < 2.2e-16
## alternative hypothesis: true difference in means between group 0 and group 1 is not equal to 0
## 95 percent confidence interval:
##  -9.128137 -6.638180
## sample estimates:
## mean in group 0 mean in group 1
##        14.51485        22.39801
```

The results of this t-test show that there is a significant difference between the groups. Those with no clinical social phobia (Group 0) have less Fear of Negative Evaluation than those with a clinical diagnosis (Group 1).

We could also do this analysis as a simple regression to "predict" a dependent variable based on a grouping variable. In this case we would "predict" fear of negative evaluation (*FNE*) using clinical social phobia diagnosis (*ADIS_IV*; 0 = no clinical social phobia; 1 = yes clinical social phobia).

```r
model2 <- lm(FNE ~ ADIS_IV, data = phobia)
summary(model2)
```

```
##
## Call:
## lm(formula = FNE ~ ADIS_IV, data = phobia)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -13.398  -3.486   0.602   3.602  13.485
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  14.5149     0.5161   28.12   <2e-16 ***
## ADIS_IV       7.8832     0.6326   12.46   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.187 on 300 degrees of freedom
## Multiple R-squared:  0.341,  Adjusted R-squared:  0.3389
## F-statistic: 155.3 on 1 and 300 DF,  p-value: < 2.2e-16
```

Lets walk through the relevant portions of the output (focusing mainly on the coefficients).

- **Intercept**: This tells us that the predicted value of y (*FNE*) is 14.51 when x (*ADIS_IV*) is 0. In this case, someone with "0 ADIS_IV" would be someone with no social phobia diagnosis. In other words, 14.51 it is the predicted score of someone with no clinical social phobia. Notice that 14.51 is the average level of FNE for people in this group. From a conceptual standpoint, this makes sense: If I wanted to guess someone's *FNE*, and all I knew about them was that they were in "Group 0", my best guess would be the mean of that group.
- **Slope for *ADIS_IV***: The Estimate of the ADIS_IV (7.88) is the predicted increase in y (*FNE*) for every one-unit increase in x (*ADIS_IV*). In this case, a "one-unit increase" in *ADIS_IV* means going from 0 to 1, which means going from "No clinical diagnosis" to "clinical diagnosis". Hence, the predicted difference between group 0 and group 1 is 7.88. In other words, people with a clinical diagnosis are predicted to have 7.88 more units of *FNE* than people with no clinical diagnosis. In this case, the

slope is significantly different from 0, which means that we could conclude that there was a significant difference in *FNE* between group.

Notice that this regression analysis matches up with the t-test done above. The t-value for *ADIS_IV* is the t-value obtained from the independent t-test. The Estimate for the intercept matches the mean of Group 0. The Estimate for *ADIS_IV* matches the difference between Group 0 and Group 1 (the t-test output doesn't give this to you directly, but you can subtract the means of the two groups).

Here is how we would write the regression equation:

$$\widehat{FNE_i} = 14.51 + 7.88x_{ADIS.IV}$$

Regression Assumptions

We will discuss regression assumptions in much greater depth in a later lab. For now, we will just touch on assumptions briefly.

The three assumptions that we can check are:

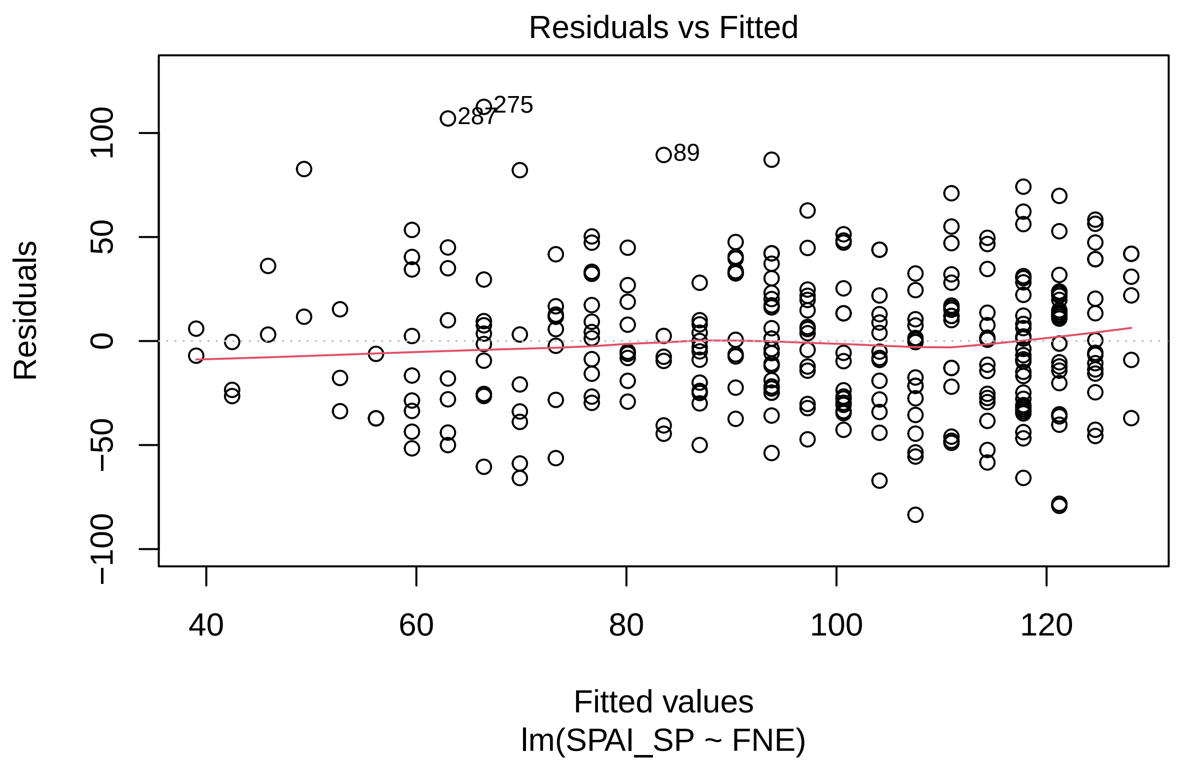
1. Linearity
2. Homoscedasticity
3. Normality of residuals

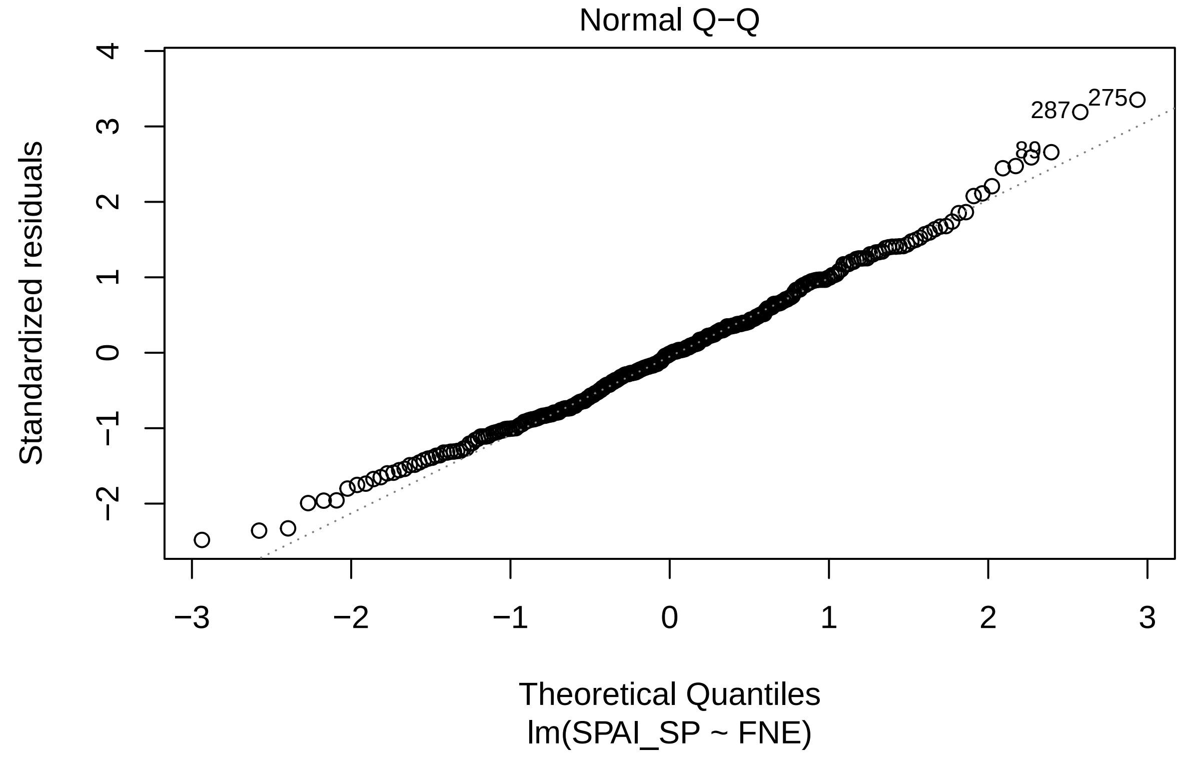
The assumptions of linearity and homoscedasticity go hand in hand. These assumptions are that the predicted values are not associated with residuals (linearity) and that the bi-variate distribution of the residuals and predicted values is homogeneous (homoscedasticity). Why are these important assumptions? When we generate a regression analysis, we are generating a model fit that is supposed to generalize to all of the data that we fed into the model. But if there is a correlation between what the predicted value is and how "off" the predicted value is, this means that the model "works better" for some of the data than it does for other parts of the data. The same is true if there is heteroscedasticity; this means that some predicted values are systematically "off" for some parts of the data.

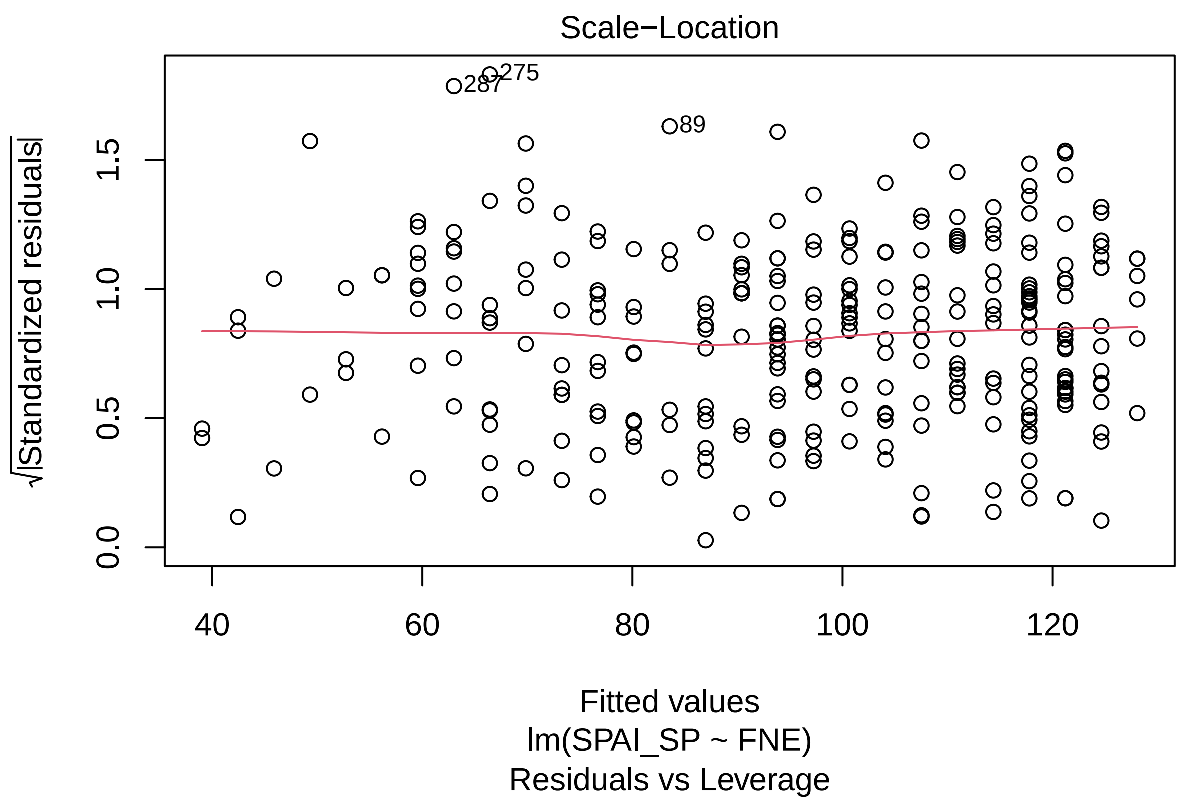
The assumption of normality relates to how the residuals are distributed: We assume that the residuals will be normally distributed (note: This assumption is NOT about the predictor or outcome variables in the regression!). This allows us to do statistical inference for our model estimates.

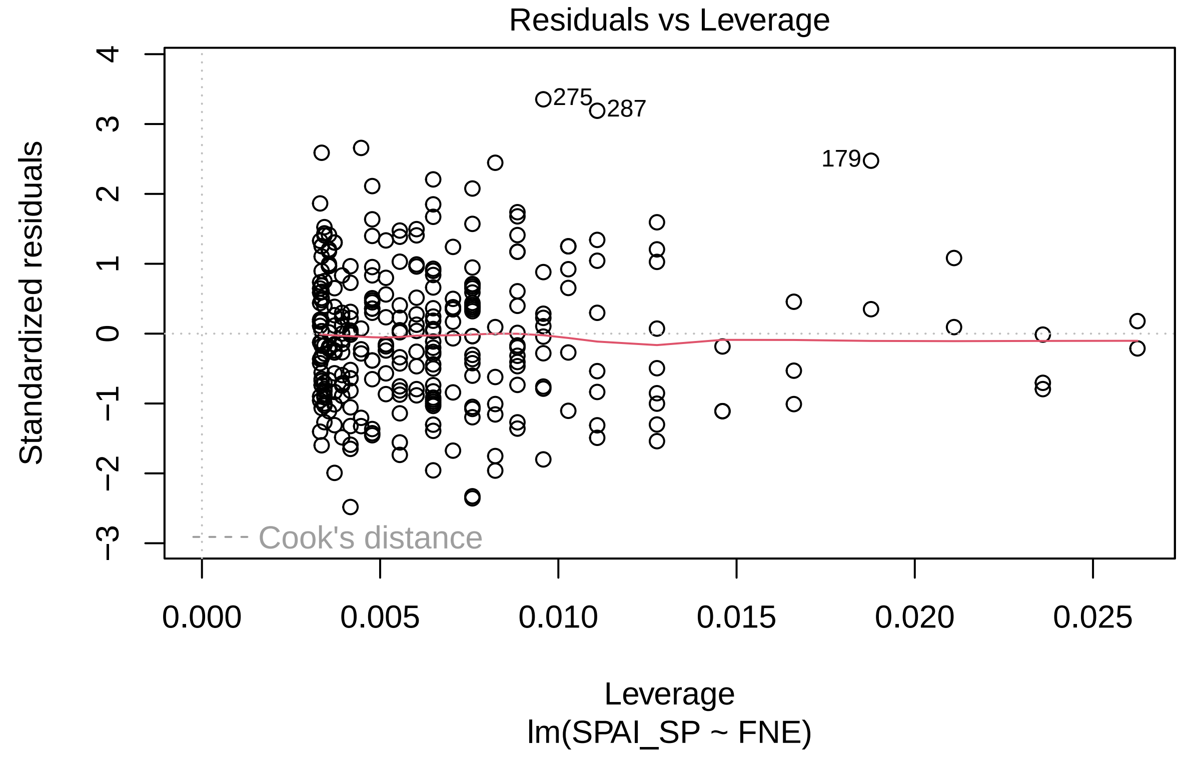
One easy way to quickly check your assumptions is to use the plot() function. This will generate some graphs that will help you evaluate assumptions.

```
plot(model1)
```


Residuals vs Fitted
Residuals
Fitted values
lm(SPAI_SP ~ FNE)
275
287
89
100
50
0
−50
−100
40
60
80
100
120


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(SPAI_SP ~ FNE)
275
287
89
4
3
2
1
0
−1
−2
−3
−2
−1
0
1
2
3





We will focus on the first two graphs for now.

1. Fitted vs Residuals

The first plot (Fitted vs Residuals) tells us about the assumption of linearity and the assumption of homogeneity.

The flatter the red line is, the better the assumption of linearity is met. This is because if there was a perfect

0 correlation between the predicted and residual values, we would expect a flat line at 0 (the dashed line). The red line shows a predicted line (not fitted linearly, though). Overall the assumption of linearity looks pretty well met.

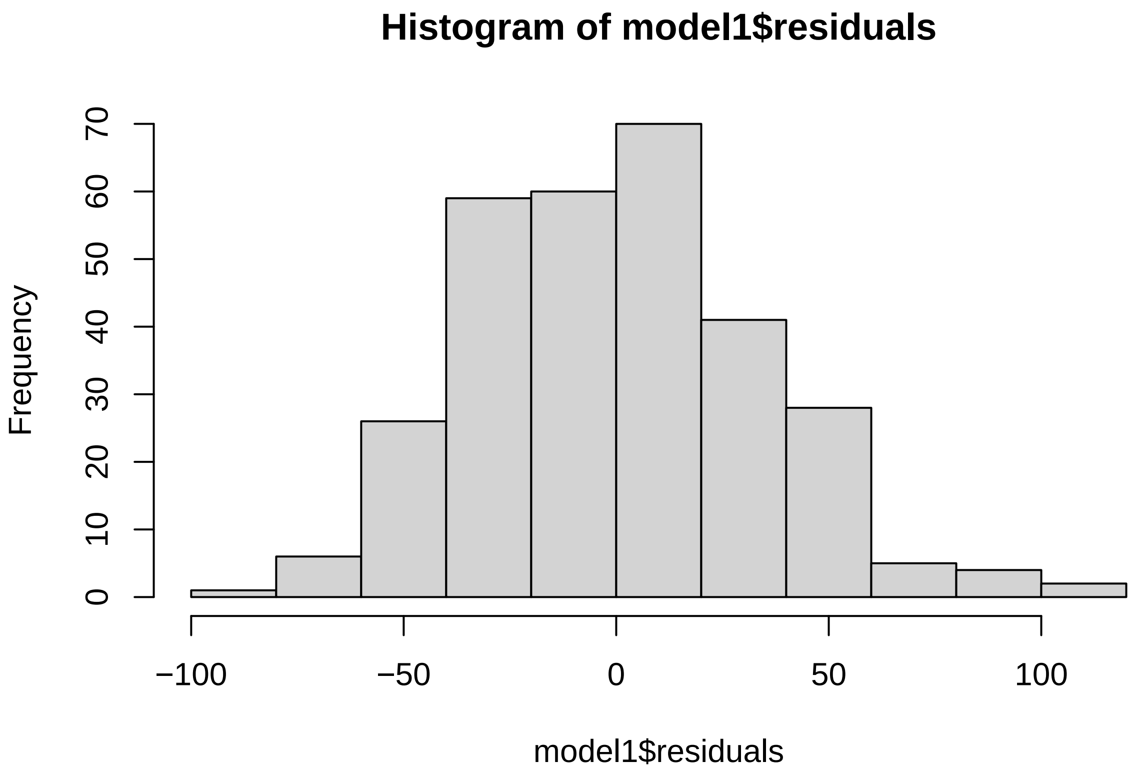
This plot can also tell us something about homogeneity of variance of the residuals across the predicted values. We should see a cloud of data points, rather than a specific pattern related to fitted values.

2. QQ Plot

The normal Q-Q plot shows the standardized residuals (y-axis) plotted against residuals we would expect if they were perfectly normally distributed. If our residuals were perfectly normally distributed, all of points would fall on the dotted diagonal one-to-one line. Departures from the dotted line suggest that there is some non-normality.

You could also look at a histogram of the residuals if you prefer.

```
hist(model1$residuals)
```



If assumptions are not met, one option is to transform your data (we will go over this in more detail in later labs). You might also consider doing a different type of analysis (again, more on this in later labs).