STAT 420: Homework 09

Fall 2020, D. Unger

Due: Tuesday, November 10 by 11:30 PM CT

Contents

Directions

Students are encouraged to work together on homework. However, sharing, copying or providing any part of a homework solution or code is an infraction of the University's rules on Academic Integrity. Any violation will be punished as severely as possible.

Final submissions must be uploaded to our Compass 2g site on the Homework page. No email, hardcopy, or late submissions will be accepted.

- Your assignment must be submitted through the submission link on **Compass 2g.** You are required to attach one `.zip` file, named `hw09_yourNetID.zip`, which contains:
  - Your RMarkdown file which should be saved as `hw09_yourNetID.Rmd`. For example `hw09_dunger.Rmd`.
  - The result of knitting your RMarkdown file as `hw09_yourNetID.html`. For example `hw09_dunger.html`.
  - Any raw data supplied by me for the assignment. For example `nutrition.csv`.
- Your resulting `.html` file will be considered a "report" which is the material that will determine the majority of your grade. Be sure to visibly include all `R` code and output that is relevant to answering the exercises. (You do not need to include irrelevant code you tried that resulted in error or did not answer the question correctly.)
- You are granted an unlimited number of submissions, but only the last submission *before* the deadline will be viewed and graded.
- If you use this `.Rmd` file as a template, be sure to remove the directions section. Consider removing `eval = FALSE` from any code chunks provided in the template, if you would like to run that code as part of your assignment.
- Your `.Rmd` file should be written such that, if it is placed in a folder with any data your are asked to import, it will knit properly without modification.

- Unless otherwise stated, you may use `R` for each of the exercises.
- Be sure to read each exercise carefully!
- Include your Name and NetID in the final document, not only in your filenames.

Assignment

Exercise 1 (Writing Functions)

(a) Write a function that takes as input a model object (variable) fit via `lm()` and outputs a fitted versus residuals plot. Also, create arguments `pointcol` and `linecol`, which control the point and line colors, respectively. Code the plot to add a horizontal line at $y = 0$, and label the x-axis "Fitted" and the y-axis "Residuals".

(b) Write a function that takes as input a model fit via `lm()` and plots a Normal Q-Q plot of the residuals. Also, create arguments `pointcol` and `linecol`, which control the point and line colors, respectively. Code the plot to add the line from `qqline()`.

(c) Test your two functions above on the `test_fit` model. For both functions, specify point and line colors that are not black.

```
set.seed(114)
test_data = data.frame(x = runif(n = 20, min = 0, max = 10),
                       y = rep(x = 0, times = 20))
test_data$y = with(test_data, 5 + 2 * x + rnorm(n = 20))
test_fit = lm(y ~ x, data = test_data)
```

Exercise 2 (Swiss Fertility Data)

For this exercise we will use the `swiss` data, which can be found in the `faraway` package. After loading the `faraway` package, use `?swiss` to learn about this dataset.

```
library(faraway)
```

(a) Fit an additive multiple regression model with `Fertility` as the response and the remaining variables in the `swiss` dataset as predictors. Output the estimated regression coefficients for this model.

(b) Check the constant variance assumption for this model. Do you feel it has been violated? Justify your answer.

(c) Check the normality assumption for this model. Do you feel it has been violated? Justify your answer.

(d) Check for any high leverage observations. Report any observations you determine to have high leverage.

(e) Check for any influential observations. Report any observations you determine to be influential.

(f) Refit the additive multiple regression model without any points you identified as influential. Compare the coefficients of this fitted model to the previously fitted model.

(g) Create a data frame that stores the observations that were "removed" because they were influential. Use the two models you have fit to make predictions with these observations. Comment on the difference between these two sets of predictions.

Exercise 3 (Concrete, Again)

Return to the concrete data from the ANOVA homework. Recall, we chose the additive model. Now that we see how ANOVA can be framed as a linear model, check for any violation of assumptions for this model.

Exercise 4 (Why Bother?)

Why do we care about violations of assumptions? One key reason is that the distributions of the parameters that we have used are all reliant on these assumptions. When the assumptions are violated, the distributional results are not correct, so our tests are garbage. **Garbage In, Garbage Out!**

Consider the following setup that we will use for the remainder of the exercise. We choose a sample size of 50.

```
n = 50
set.seed(1)
x_1 = runif(n, 0, 10)
x_2 = runif(n, -5, 5)
```

Consider the model,

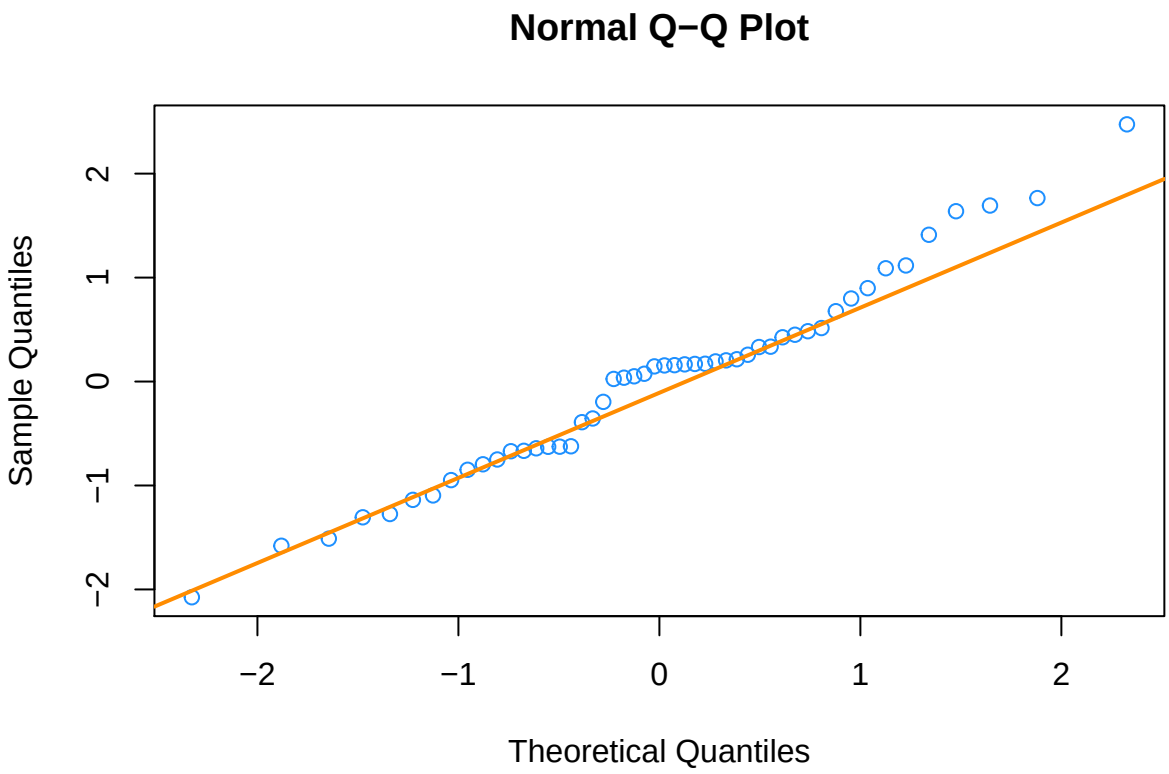
$$Y = 2 + 1x_1 + 0x_2 + \epsilon.$$

That is,

- $\beta_0 = 2$
- $\beta_1 = 1$
- $\beta_2 = 0$

We now simulate `y_1` in a manner that does not violate any assumptions, which we will verify. In this case $\epsilon \sim N(0, 1)$.

```
y_1 = 2 + x_1 + 0 * x_2 + rnorm(n = n, mean = 0, sd = 1)
fit_1 = lm(y_1 ~ x_1 + x_2)
qqnorm(resid(fit_1), col = "dodgerblue")
qqline(resid(fit_1), col = "darkorange", lwd = 2)
```

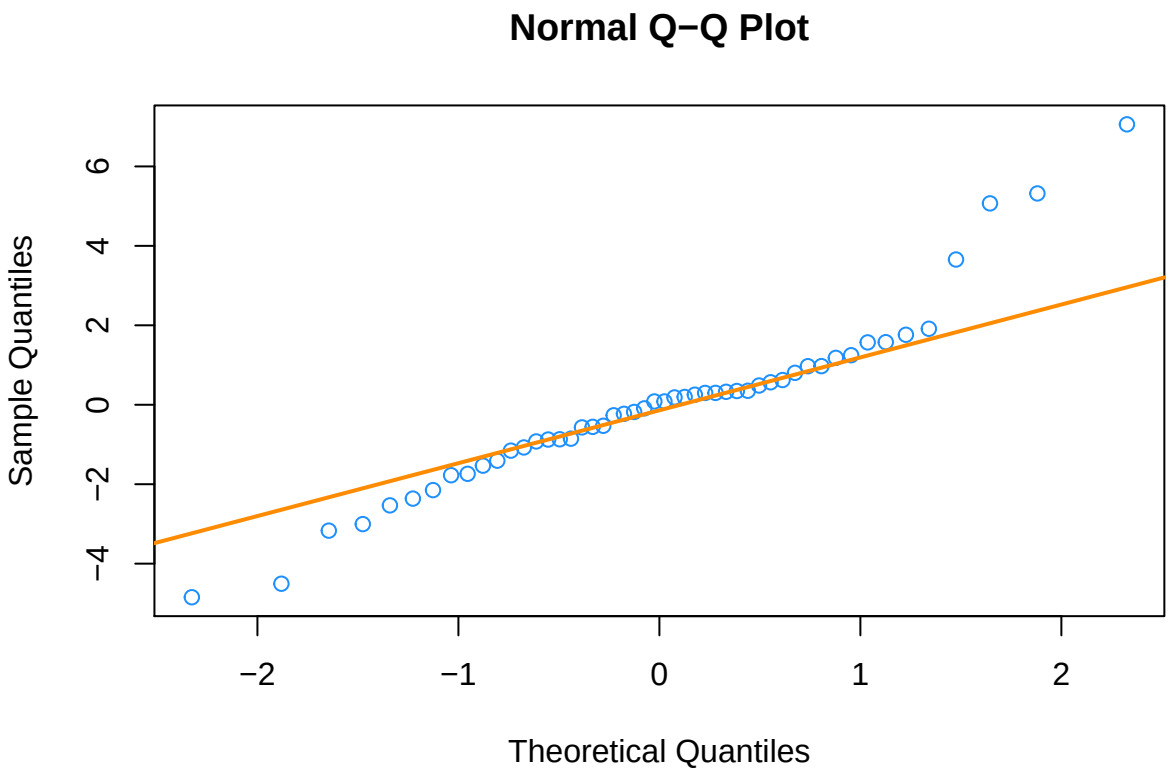



```
shapiro.test(resid(fit_1))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(fit_1)
## W = 0.98049, p-value = 0.5727
```

Then, we simulate `y_2` in a manner that **does** violate assumptions, which we again verify. In this case $\epsilon \sim N(0, \sigma = |x_2|)$.

```
y_2 = 2 + x_1 + 0 * x_2 + rnorm(n = n, mean = 0, sd = abs(x_2))
fit_2 = lm(y_2 ~ x_1 + x_2)
qqnorm(resid(fit_2), col = "dodgerblue")
qqline(resid(fit_2), col = "darkorange", lwd = 2)
```




```
shapiro.test(resid(fit_2))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(fit_2)
## W = 0.92848, p-value = 0.004843
```

(a) Use the following code after changing `birthday` to your birthday.

```
num_sims = 1000
p_val_1 = rep(0, num_sims)
p_val_2 = rep(0, num_sims)
birthday = 19081014
set.seed(birthday)
```

Repeat the above process of generating `y_1` and `y_2` as defined above, and fit models with each as the response `1000` times. Each time, store the p-value for testing,

$$\beta_2 = 0,$$

using both models, in the appropriate variables defined above. (You do not need to use a data frame as we have in the past. Although, feel free to modify the code to instead use a data frame.)

(b) What proportion of the `p_val_1` values are less than 0.05? Less than 0.10? What proportion of the `p_val_2` values are less than 0.05? Less than 0.10? Briefly explain these results.