Week 8 - Homework

STAT 420, Summer 2017, Dalpiaz

Exercise 1 (Writing Functions)

(a) Write a function named `diagnostics` that takes as input the arguments:

- `model`, an object of class `lm(), that is a model fit via`lm()‘
- `pcol`, for controlling point colors in plots, with a default value of `black`
- `lcol`, for controlling line colors in plots, with a default value of `black`
- `alpha`, the significance level of any test that will be performed inside the function, with a default value of `0.05`
- `plotit`, a logical value for controlling display of plots with default value `TRUE`
- `testit`, a logical value for controlling outputting the results of tests with default value `TRUE`

The function should output:

- A list with two elements when `testit` is `TRUE`:
  - `p_val`, the p-value for the Shapiro-Wilk test for assesing normality
  - `decision`, the decision made when performing the Shapiro-Wilk test using the `alpha` value input to the function. “Reject” if the null hypothesis is rejected, otherwise “Fail to Reject”.
- Two plots, side-by-side, when `plotit` is `TRUE`:
  - A fitted versus residuals plot that adds a horizontal line at $y = 0$, and labels the x-axis “Fitted” and the y-axis “Residuals”. The points and line should be colored according to the input arguments. Give the plot a title.
  - A Normal Q-Q plot of the residuals that adds the appropriate line using `qqline()`. The points and line should be colored according to the input arguments. Be sure the plot has a title.

Solution:

```r
diagnostics = function(model, pcol = "black", lcol = "black", alpha = 0.05,
                       plotit = TRUE, testit = TRUE) {

  if (plotit == TRUE) {

    # side-by-side plots (one row, two columns)
    par(mfrow = c(1, 2))

    # fitted versus residuals
    plot(fitted(model), resid(model),
         col = pcol, pch = 20, cex = 1.5,
         xlab = "Fitted", ylab = "Residuals",
         main = "Fitted versus Residuals")
    abline(h = 0, col = lcol, lwd = 2)

    # qq-plot
    qqnorm(resid(model), col = pcol, pch = 20, cex = 1.5)
    qqline(resid(model), col = lcol, lwd = 2)
  }

  if (testit == TRUE) {
    # p-value and decision
    p_val = shapiro.test(resid(model))$p.value
    decision = ifelse(p_val < alpha, "Reject", "Fail to Reject")
    list(p_val = p_val, decision = decision)
  }

}
```

(b) Run the following code.

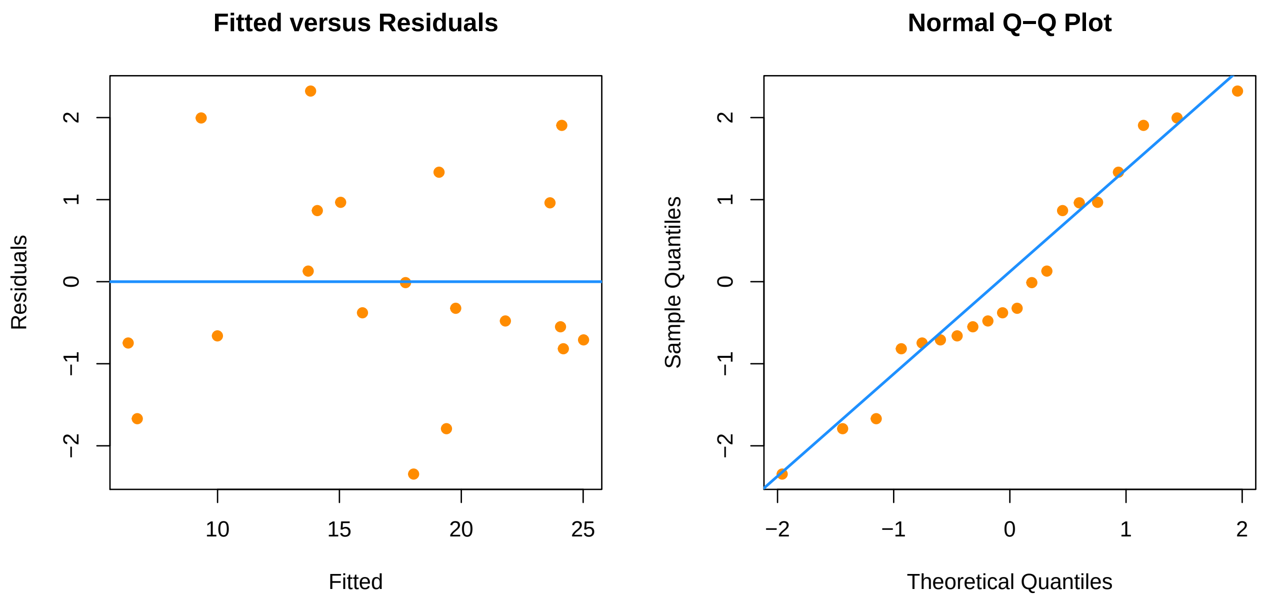
```r
set.seed(42)
data1 = data.frame(x = runif(n = 20, min = 0, max = 10),
                   y = rep(x = 0, times = 20))
data1$y = with(data1, 5 + 2 * x + rnorm(n = 20))
fit1 = lm(y ~ x, data = data1)

data2 = data.frame(x = runif(n = 30, min = 0, max = 10),
                   y = rep(x = 0, times = 30))
data2$y = with(data2, 2 + 1 * x + rexp(n = 30))
fit2 = lm(y ~ x, data = data2)

data3 = data.frame(x = runif(n = 40, min = 0, max = 10),
                   y = rep(x = 0, times = 40))
data3$y = with(data3, 2 + 1 * x + rnorm(n = 40, sd = x))
fit3 = lm(y ~ x, data = data3)

diagnostics(fit1, plotit = FALSE)$p_val
diagnostics(fit1, testit = FALSE, pcol = "darkorange", lcol = "dodgerblue")

diagnostics(fit2, plotit = FALSE)$decision
diagnostics(fit2, testit = FALSE, pcol = "grey", lcol = "green")

diagnostics(fit3)
```

Solution:

```r
set.seed(42)
data1 = data.frame(x = runif(n = 20, min = 0, max = 10),
                   y = rep(x = 0, times = 20))
data1$y = with(data1, 5 + 2 * x + rnorm(n = 20))
fit1 = lm(y ~ x, data = data1)

data2 = data.frame(x = runif(n = 30, min = 0, max = 10),
                   y = rep(x = 0, times = 30))
data2$y = with(data2, 2 + 1 * x + rexp(n = 30))
fit2 = lm(y ~ x, data = data2)

data3 = data.frame(x = runif(n = 40, min = 0, max = 10),
                   y = rep(x = 0, times = 40))
data3$y = with(data3, 2 + 1 * x + rnorm(n = 40, sd = x))
fit3 = lm(y ~ x, data = data3)

diagnostics(fit1, plotit = FALSE)$p_val
```

```
## [1] 0.518
```

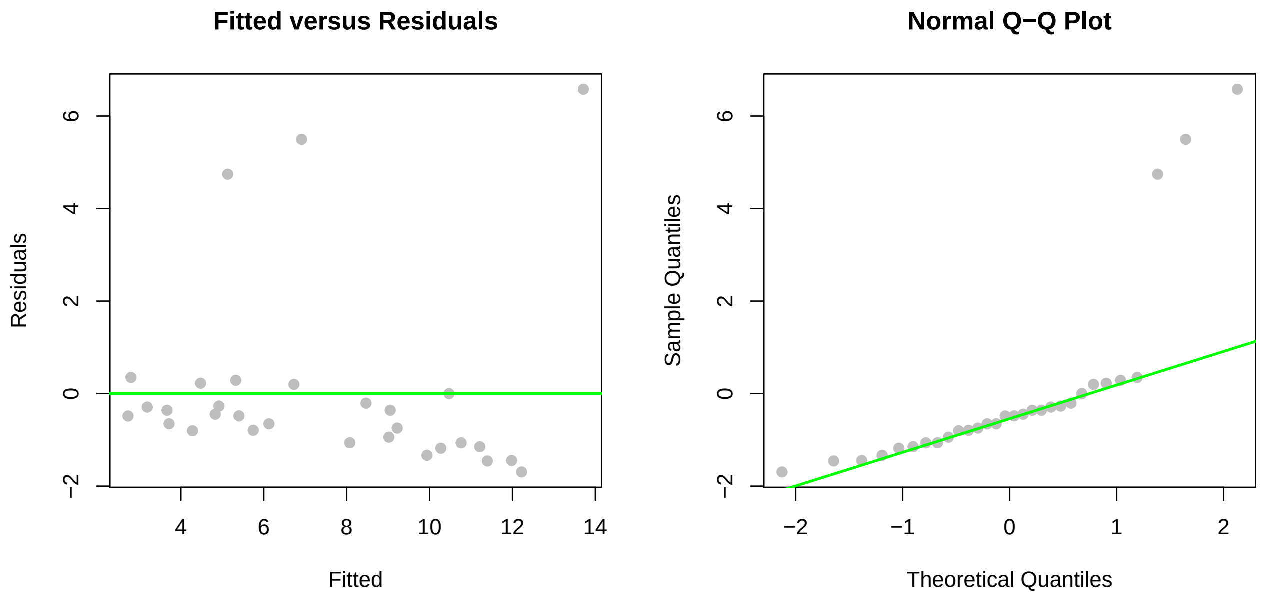
```r
diagnostics(fit1, testit = FALSE, pcol = "darkorange", lcol = "dodgerblue")
```



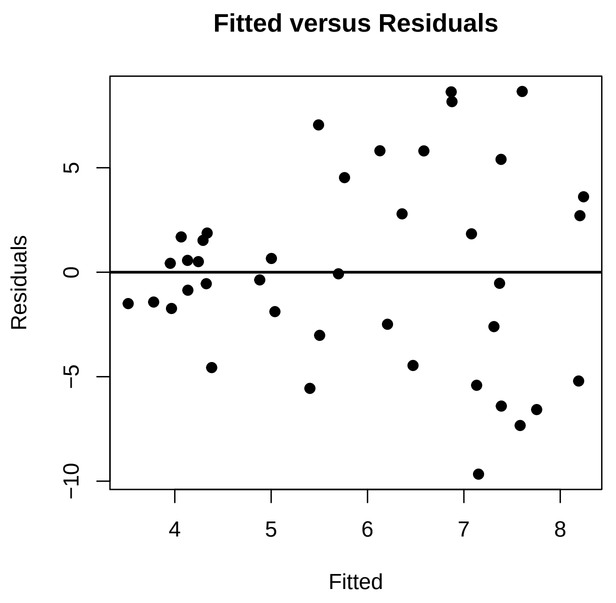
```r
diagnostics(fit2, plotit = FALSE)$decision
```

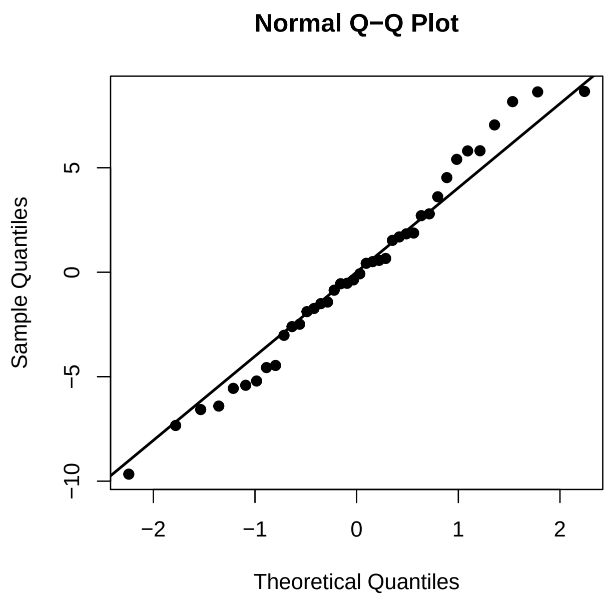
```
## [1] "Reject"
```

```r
diagnostics(fit2, testit = FALSE, pcol = "grey", lcol = "green")
```



```r
diagnostics(fit3)
```





```
## $p_val
## [1] 0.7288
##
## $decision
## [1] "Fail to Reject"
```

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. Report the R^2 for this model.

Solution:

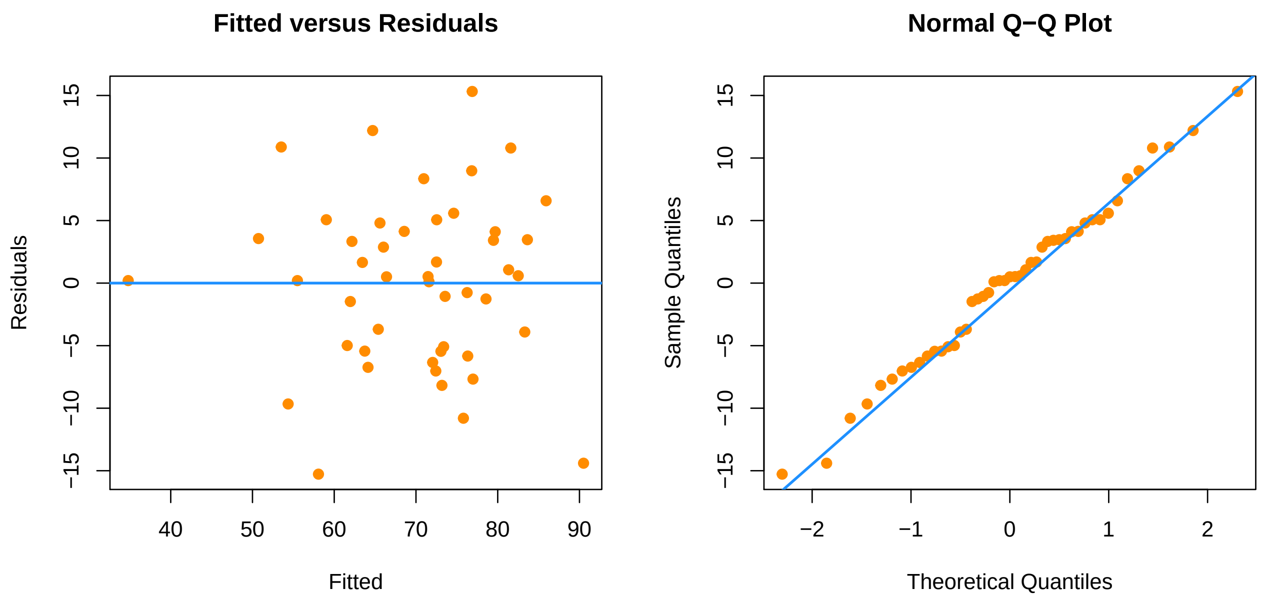
```
swiss_mod = lm(Fertility ~ ., data = swiss)
summary(swiss_mod)$r.squared
```

```
## [1] 0.7067
```

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

Solution:

```
diagnostics(swiss_mod, testit = FALSE, pcol = "darkorange", lcol = "dodgerblue")
```



```r
library(lmtest)
bptest(swiss_mod)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  swiss_mod
## BP = 5.9, df = 5, p-value = 0.3
```

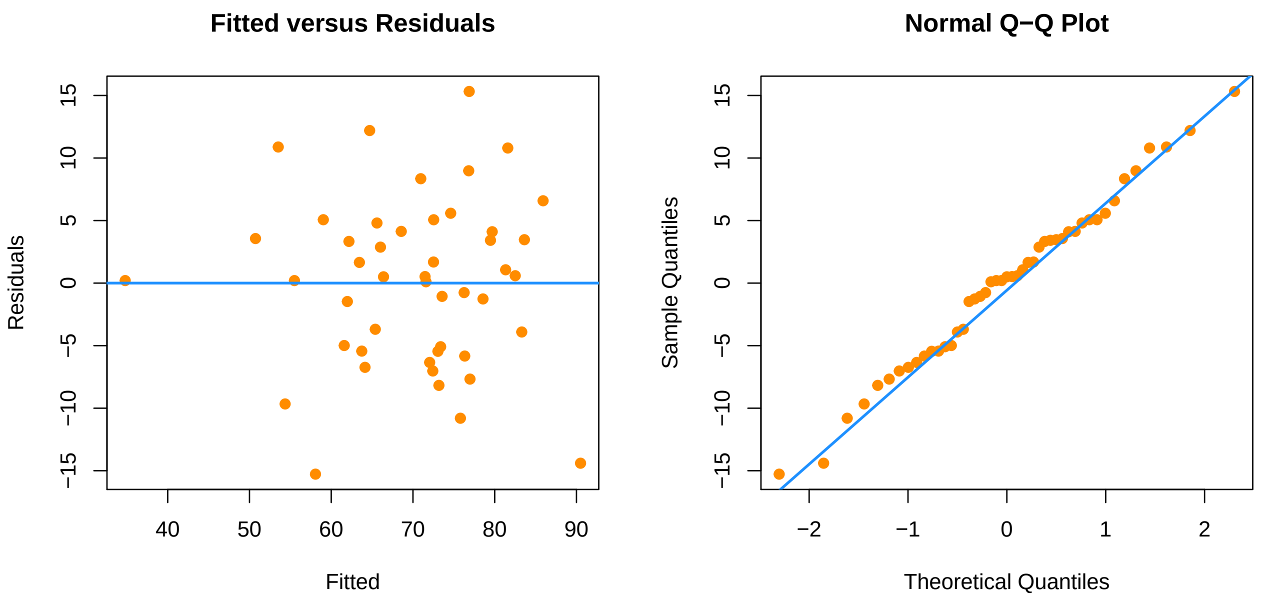
The fitted versus residuals plot looks pretty good. While we could argue that there is less variance for lower fitted values, there could also simply be less data in that range.

We see the Breusch–Pagan test does not reject, so we choose to believe that the constant variance assumption has not been violated.

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

Solution:

```r
diagnostics(swiss_mod, pcol = "darkorange", lcol = "dodgerblue")
```



```
## $p_val
## [1] 0.9318
##
## $decision
## [1] "Fail to Reject"
```

The points fall very close to the line in the Q-Q Plot output in **(b)**, and the Shapiro-Wilk test does not reject at any reasonable α, so we choose to believe that the normality assumption has not been violated.

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

Solution:

```r
swiss_mod_lev = hatvalues(swiss_mod)
swiss_mod_lev_mean = mean(swiss_mod_lev)
swiss_mod_lev[swiss_mod_lev > 2 * swiss_mod_lev_mean]
```

```
##    La Vallee V. De Geneve
##       0.3512       0.4558
```

Based on the heuristic given in the text, we say that the observations “La Vallee V.” and “De Geneve” are points of high leverage. They have greater potential for a large influence on the model fit.

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

Solution:

```r
swiss_mod_cook = cooks.distance(swiss_mod)
swiss_mod_cook[swiss_mod_cook > 4 / length(swiss_mod_cook)]
```

```
##  Porrentruy       Sierre   Neuchatel Rive Droite Rive Gauche
##      0.2077       0.1476      0.1249      0.1025      0.1124
```

The above observations (with their Cook’s distance reported) are considered influential according to the heuristic given in the text.

(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.

Solution:

```
swiss_mod_sub = lm(Fertility ~ ., data = swiss, subset = swiss_mod_cook <= 4 / length(swiss_mod_cook))
(coef(swiss_mod) - coef(swiss_mod_sub)) / coef(swiss_mod)
```

```
##      (Intercept)      Agriculture      Examination        Education
##         0.007033        -0.267754        -0.938558         0.207218
##         Catholic Infant.Mortality
##         0.054241        -0.260550
```

Here, we calculate the relative change in the coefficients. We see that for some coefficients, such as `Examination`, the change can be rather large in magnitude and can also change direction.

(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.

Solution:

```
swiss_removed = swiss[swiss_mod_cook > 4 / length(swiss_mod_cook), ]

predict(swiss_mod, swiss_removed)
```

```
##   Porrentruy       Sierre    Neuchatel Rive Droite Rive Gauche
##        90.50        76.88        53.52       54.36       58.07
```

```
predict(swiss_mod_sub, swiss_removed)
```

```
##   Porrentruy       Sierre    Neuchatel Rive Droite Rive Gauche
##        94.44        76.34        55.90       57.93       61.32
```

```
(predict(swiss_mod, swiss_removed) - predict(swiss_mod_sub, swiss_removed)) / predict(swiss_mod, swiss_
```

```
##   Porrentruy       Sierre    Neuchatel Rive Droite Rive Gauche
##    -0.043532     0.007048    -0.044412   -0.065556   -0.055892
```

Compared to the change in the estimated regression coefficients, the change in predicted `Fertility` is rather small. Why could this be? That's something we will touch on in an upcoming chapter!

Exercise 3 (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 100.

```
n = 100
set.seed(42)
x_1 = runif(n, -2, 2)
x_2 = runif(n, 0, 5)
```

Consider the model,

$$Y = 5 + 0x_1 + 1x_2 + \epsilon.$$

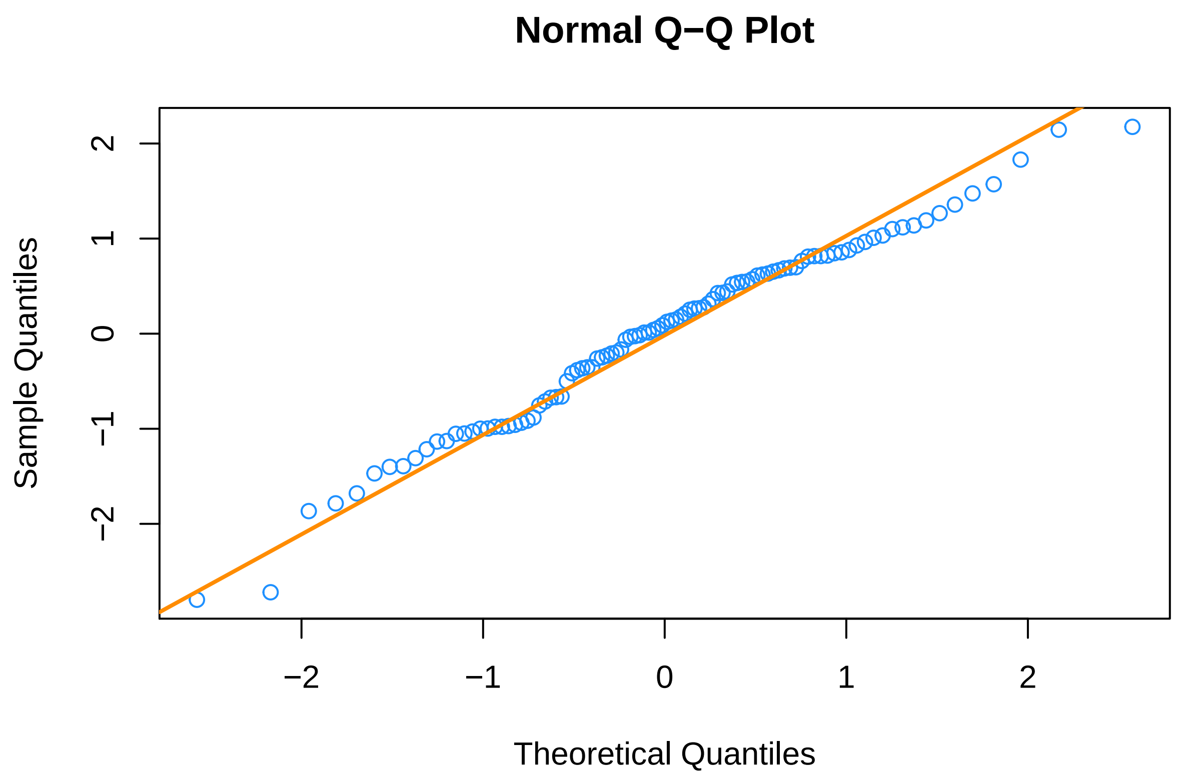
That is,

- $\beta_0 = 5$

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

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

```r
set.seed(420)
y_1 = 5 + 0 * x_1 + 1 * 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)
```

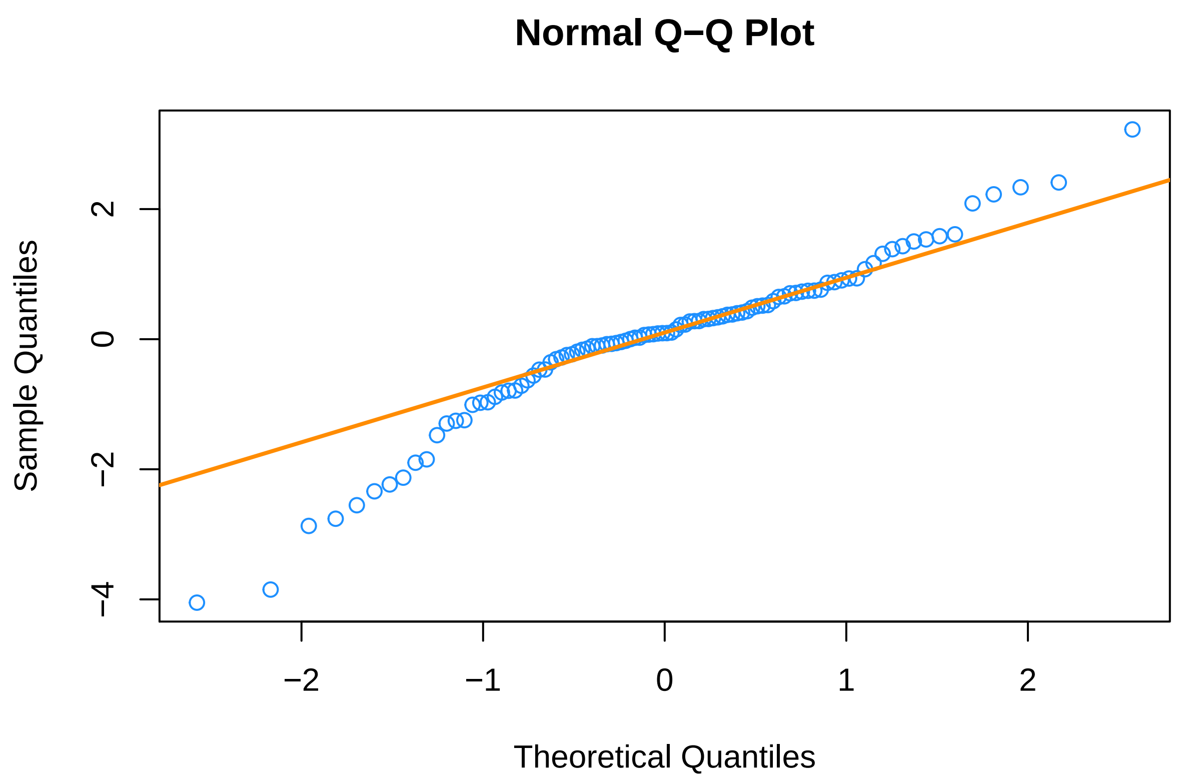


```r
shapiro.test(resid(fit_1))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(fit_1)
## W = 0.98, p-value = 0.2
```

Then, we simulate y_2 in a manner that **does** violate assumptions, which we again verify. In this case $\epsilon \sim N(0, \sigma = |x_1|)$.

```r
set.seed(42)
y_2 = 5 + 0 * x_1 + 1 * x_2  + rnorm(n = n, mean = 0, sd = abs(x_1))
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 = 0.95, p-value = 0.0005
```

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

```
num_sims = 2500
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 `2500` times. Each time, store the p-value for testing,

$$\beta_1 = 0,$$

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.)

Solution:

```
for(i in 1:num_sims) {
  y_1        = 5 + 0 * x_1 + 1 * x_2 + rnorm(n)
  fit_1      = lm(y_1 ~ x_1 + x_2)
  p_val_1[i] = summary(fit_1)$coef[2, 4]
```

```r
  y_2        = 5 + 0 * x_1 + 1 * x_2 + rnorm(n, 0, abs(x_1))
  fit_2      = lm(y_2 ~ x_1 + x_2)
  p_val_2[i] = summary(fit_2)$coef[2, 4]
}
```

(b) What proportion of the `p_val_1` values are less than 0.01? Less than 0.05? Less than 0.10? What proportion of the `p_val_2` values are less than 0.01? Less than 0.05? Less than 0.10? Arrange your results in a table. Briefly explain these results.

Solution:

```r
results = data.frame(
  alpha = c(0.01, 0.05, 0.10),
  good  = c(mean(p_val_1 < 0.01), mean(p_val_1 < 0.05), mean(p_val_1 < 0.10)),
  bad   = c(mean(p_val_2 < 0.01), mean(p_val_2 < 0.05), mean(p_val_2 < 0.10))
)
colnames(results) = c("alpha", "good assumptions", "assumptions violated")
knitr::kable(results)
```

alpha	good assumptions	assumptions violated
0.01	0.0060	0.0544
0.05	0.0484	0.1448
0.10	0.1028	0.2252

The results of `p_val_1` are roughly what we would expect. $\beta_1 = 0$ is true in this simulation, so we expect α of these simulations to reject by chance. We see these values are very far off for `p_val_2`, which is a result of the violation of assumptions. This is why we should only perform inference when assumptions are not violated.

Exercise 4 (TV Is Healthy?)

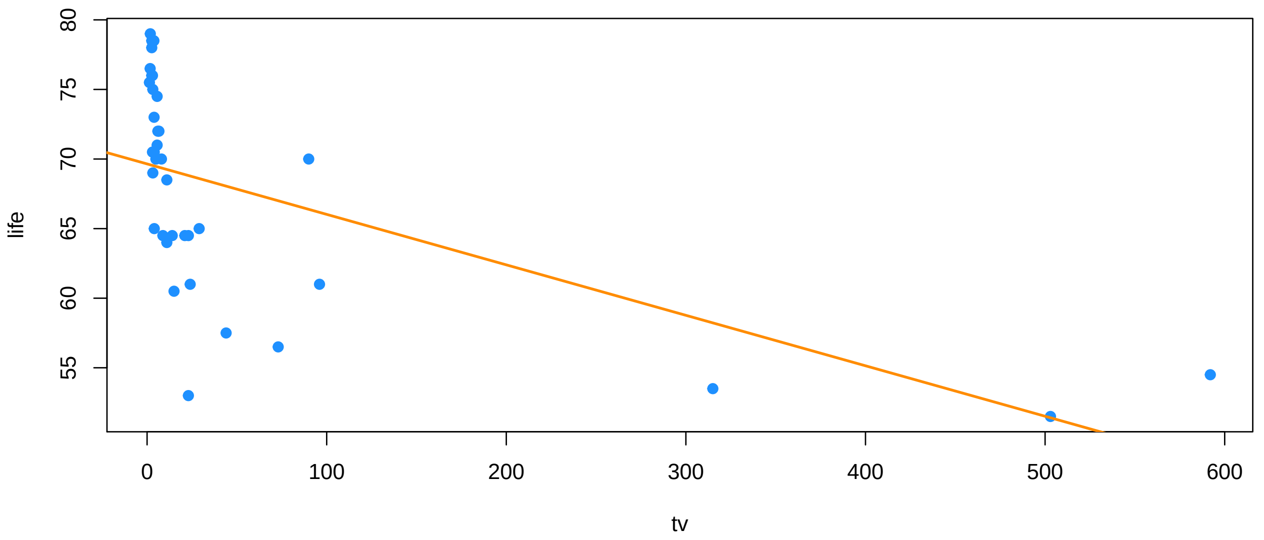
For this exercise, we will use the `tvdoctor` data, which can be found in the `faraway` package. After loading the `faraway` package, use `?tvdoctor` to learn about this dataset.

```r
library(faraway)
```

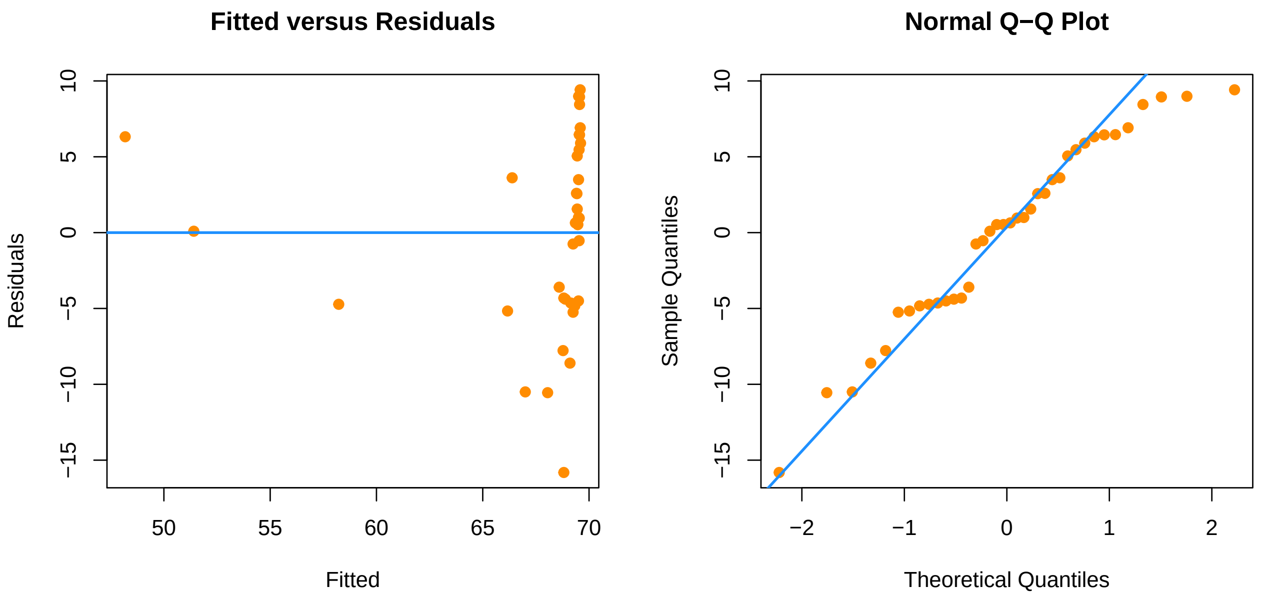
(a) Fit a simple linear regression with `life` as the response and `tv` as the predictor. Plot a scatterplot and add the fitted line. Check the assumptions of this model.

Solution:

```r
tv_mod_1 = lm(life ~ tv, data = tvdoctor)
plot(life ~ tv, data = tvdoctor, col = "dodgerblue", pch = 20, cex = 1.5)
abline(tv_mod_1, col = "darkorange", lwd = 2)
```



```r
diagnostics(tv_mod_1, testit = FALSE, pcol = "darkorange", lcol = "dodgerblue")
```

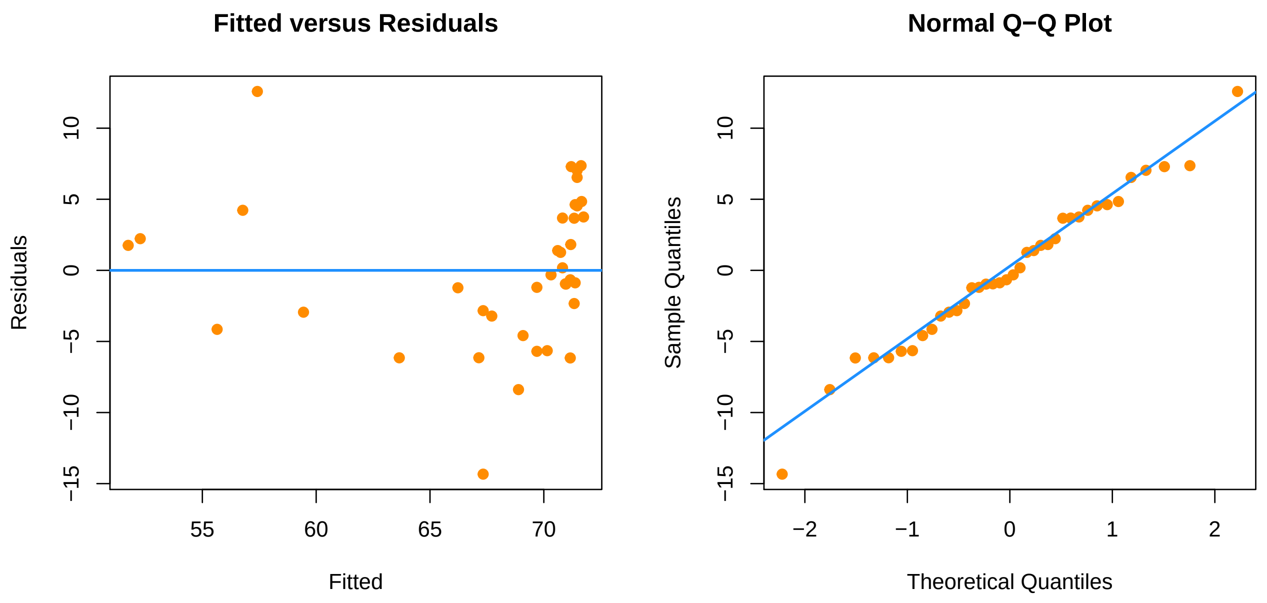


The Q-Q plot looks okay, not great. The fitted versus residuals plot, however, looks terrible. Something weird is happening there.

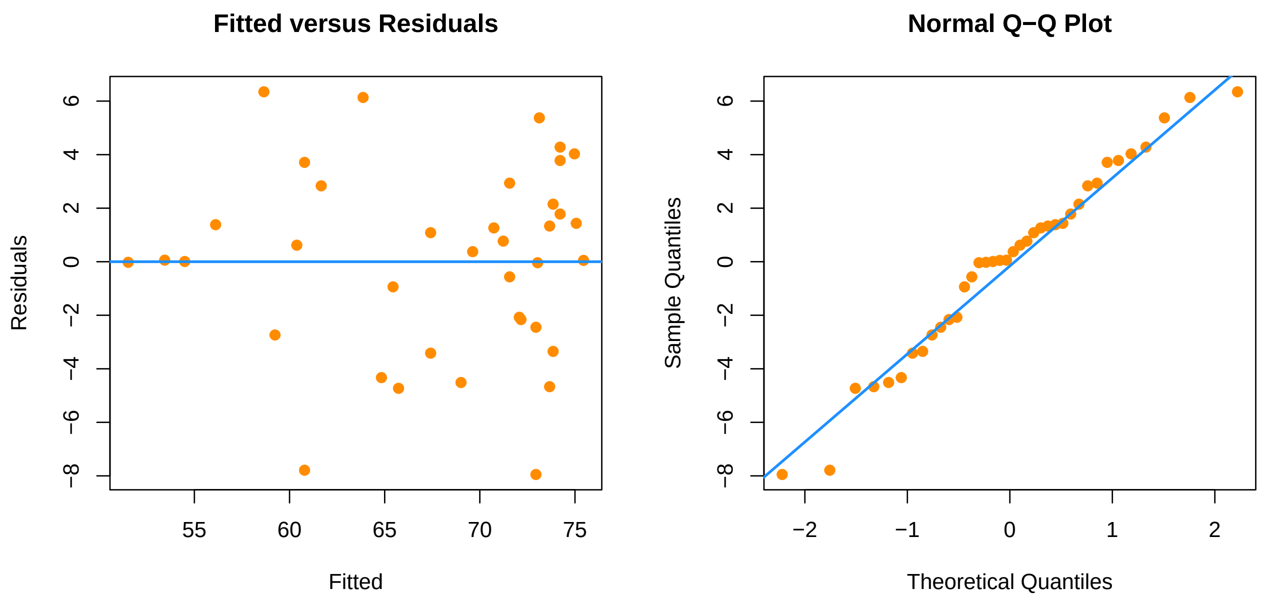
(b) Fit higher order polynomial models of degree 3, 5, and 7. For each, plot a fitted versus residuals plot and comment on the constant variance assumption. Based on those plots, which of these three models do you think are acceptable? Use a statistical test(s) to compare the models you just chose. Based on the test, which is preferred? Check the normality assumption of this model. Identify any influential observations of this model.

Solution:

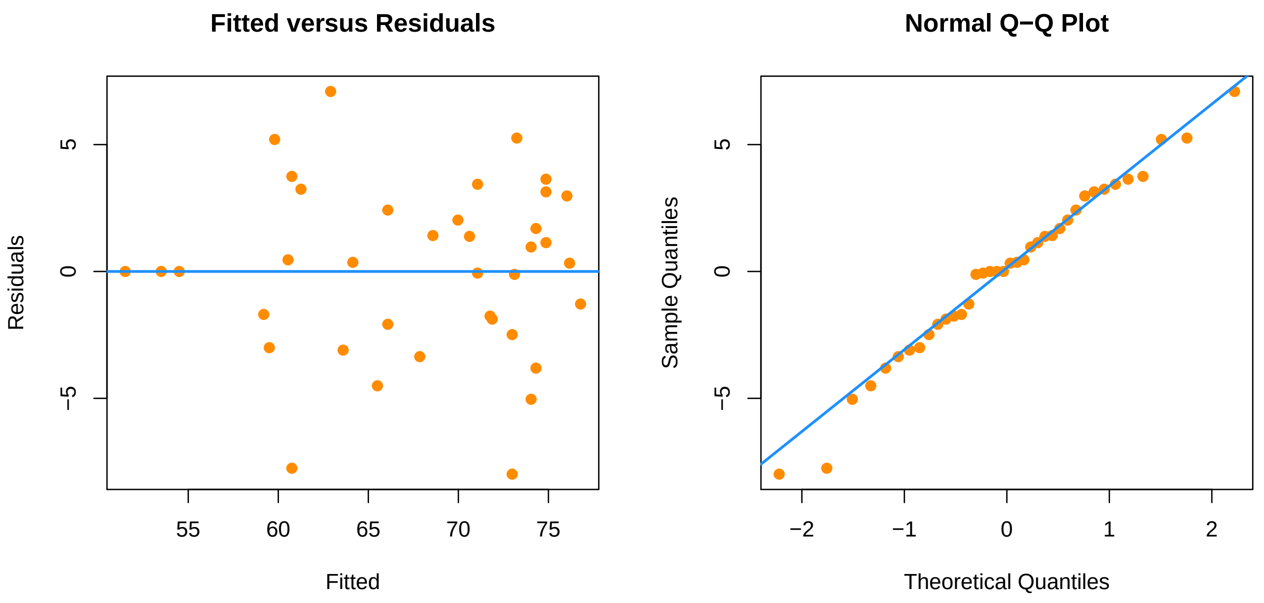
```r
tv_mod_3 = lm(life ~ poly(tv, 3), data = tvdoctor)
diagnostics(tv_mod_3, testit = FALSE, pcol = "darkorange", lcol = "dodgerblue")
```



```r
tv_mod_5 = lm(life ~ poly(tv, 5), data = tvdoctor)
diagnostics(tv_mod_5, testit = FALSE, pcol = "darkorange", lcol = "dodgerblue")
```



```r
tv_mod_7 = lm(life ~ poly(tv, 7), data = tvdoctor)
diagnostics(tv_mod_7, testit = FALSE, pcol = "darkorange", lcol = "dodgerblue")
```



```r
anova(tv_mod_5, tv_mod_7)
```

```
## Analysis of Variance Table
##
## Model 1: life ~ poly(tv, 5)
## Model 2: life ~ poly(tv, 7)
##   Res.Df RSS Df Sum of Sq    F Pr(>F)
## 1     32 457
## 2     30 428  2      28.9 1.01   0.38
```

```r
shapiro.test(resid(tv_mod_5))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(tv_mod_5)
## W = 0.97, p-value = 0.5
```

```r
cook_tv_mod_5 = cooks.distance(tv_mod_5)
cook_tv_mod_5[which(cook_tv_mod_5 > 4 / length(cook_tv_mod_5))]
```

```
## Bangladesh   Ethiopia      Kenya KoreaNorth    Myanmar
##   467.1758  3710.7211     0.4160     0.3983 34131.6047
```

Based on the plots, the models of degree 5 and 7 are acceptable. Based on the test, we prefer the model of degree 5. The Q-Q plot and the Shapiro-Wilk test suggest that there is no issue with the normality assumption. The above Cook's distances are for observations identified as influential.

Note this problem has suggested that as the number of people per TV goes up, life expectancy goes down. But does that mean that TV makes us live longer? Probably not! There is another variable in this dataset that we didn't see, `doctor`, which records the number of people per doctor. This variable has roughly the same relationship with `life`. Makes you think…

Exercise 5 (Brains)

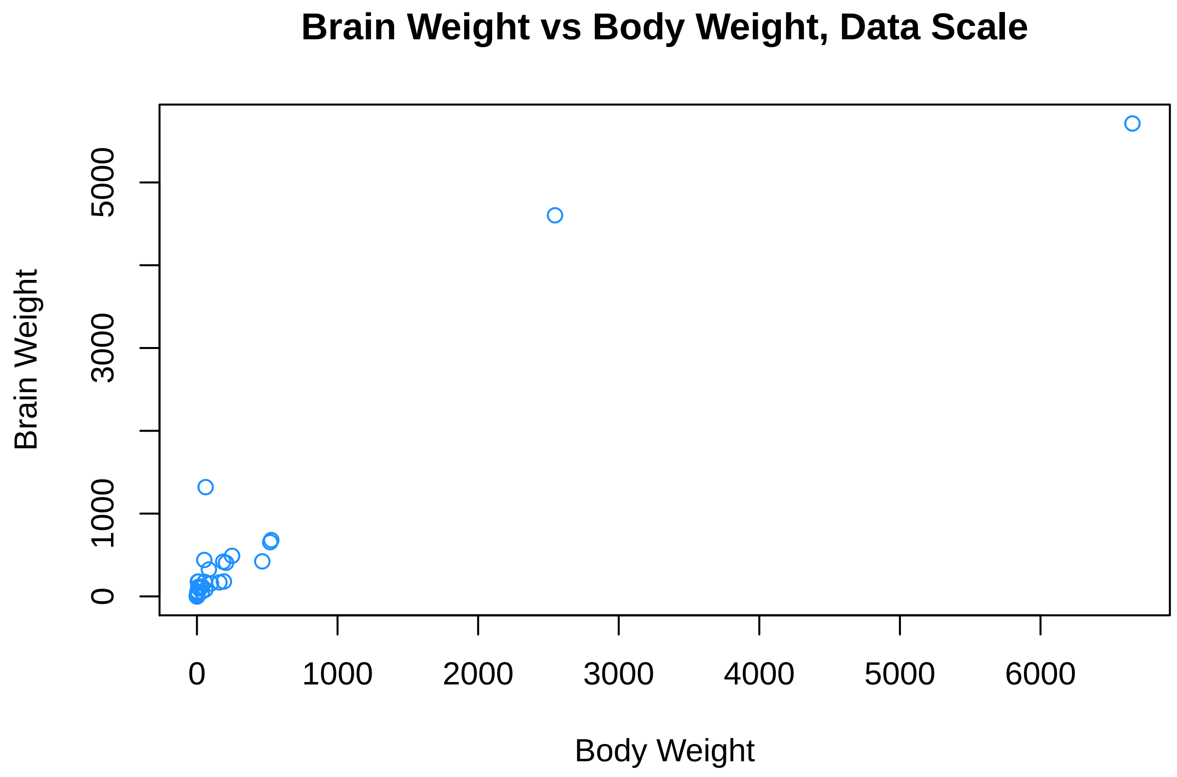
The data set `mammals` from the `MASS` package contains the average body weight in kilograms (x) and the average brain weight in grams (y) for 62 species of land mammals. Use `?mammals` to learn more.

```r
library(MASS)
```

(a) Plot average brain weight (y) versus average body weight (x).

Solution:

```r
plot(brain ~ body,
     data = mammals,
     main = "Brain Weight vs Body Weight, Data Scale",
     xlab = "Body Weight",
     ylab = "Brain Weight",
     col = "dodgerblue")
```



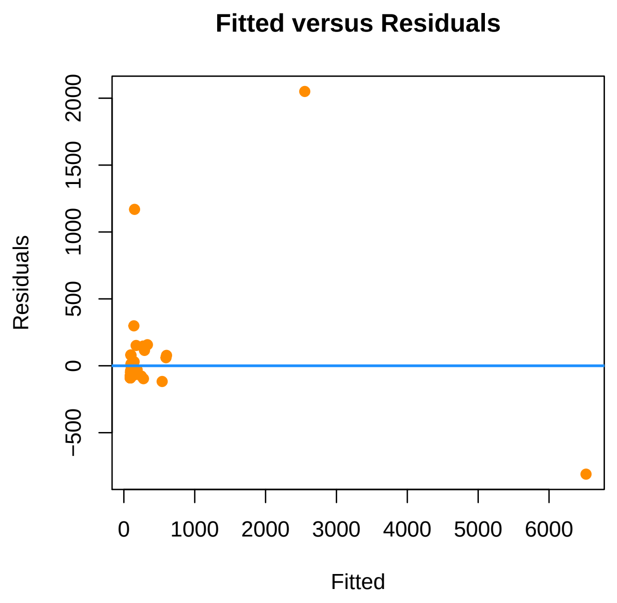
(b) Fit a linear model with `brain` as the response and `body` as the predictor. Test for significance of regression. Do you think this is an appropriate model?

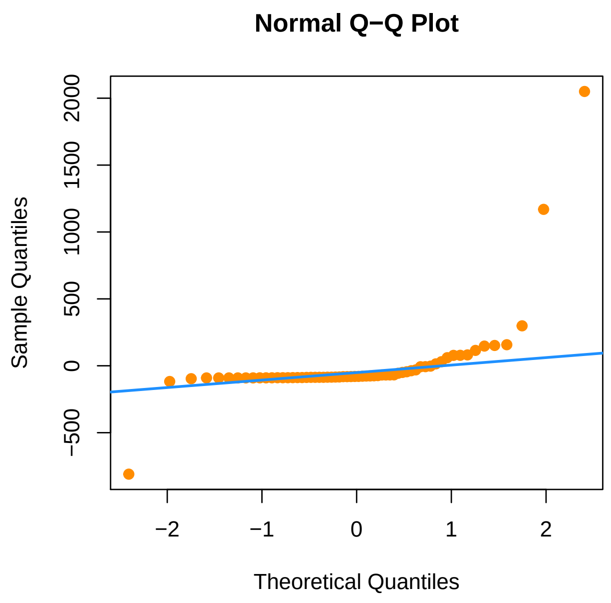
Solution:

```r
fit_bad = lm(brain ~ body, data = mammals)
anova(fit_bad)
```

```
## Analysis of Variance Table
##
## Response: brain
##           Df   Sum Sq  Mean Sq F value Pr(>F)
## body       1 46068314 46068314     411 <2e-16 ***
## Residuals 60  6722239   112037
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
diagnostics(fit_bad, testit = FALSE, pcol = "darkorange", lcol = "dodgerblue")
```





The model is incredibly significant, but is wildly inappropriate. Both the fitted vs residuals plot, and the Q-Q plot show large violation of assumptions.

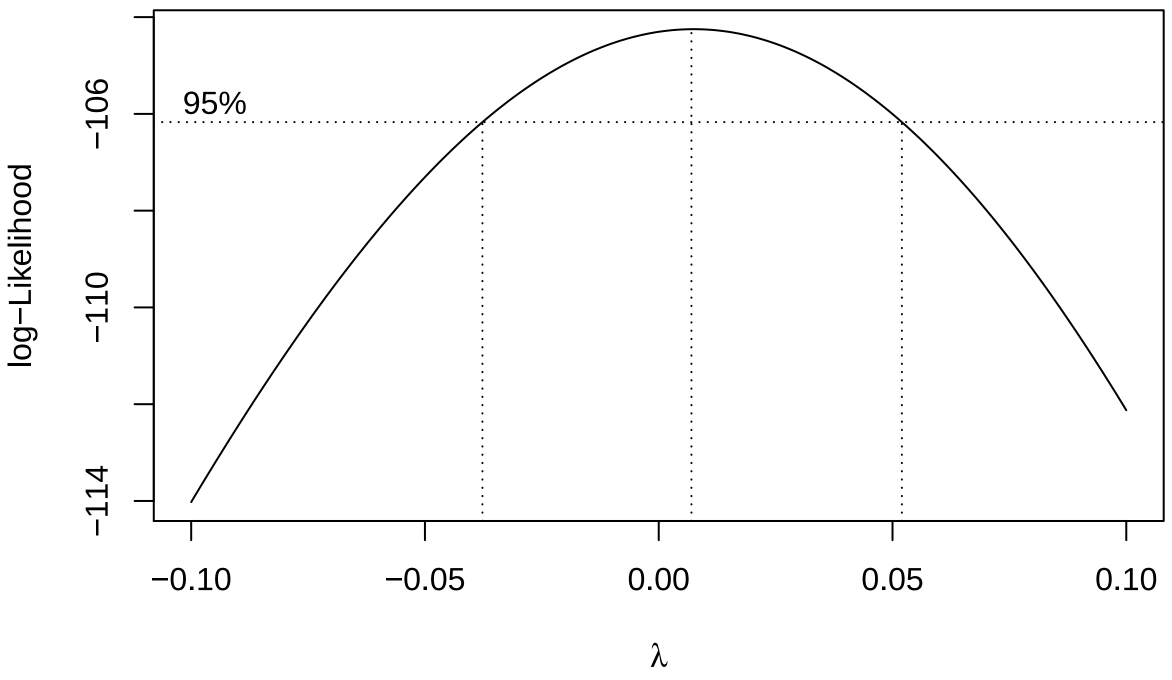
(c) Since the body weights do range over more than one order of magnitude and are strictly positive, we will use log(body weight) as our *predictor*, with no further justification. (Recall, *the log rule*: if the values of a variable range over more than one order of magnitude and the variable is strictly positive, then replacing the variable by its logarithm is likely to be helpful.) Use the Box-Cox method to verify that log(brain weight) is then a "recommended" transformation of the *response* variable. That is, verify that $\lambda = 0$ is among the "recommended" values of λ when considering,

$$g_\lambda(y) = \beta_0 + \beta_1 \log(\text{body weight}) + \epsilon$$

Include the relevant plot in your results, using an appropriate zoom onto the relevant values.

Solution:

```r
fit = lm(brain ~ log(body), data = mammals)
boxcox(fit, lambda = seq(-0.10, 0.10, by = 0.001), plotit = TRUE)
```

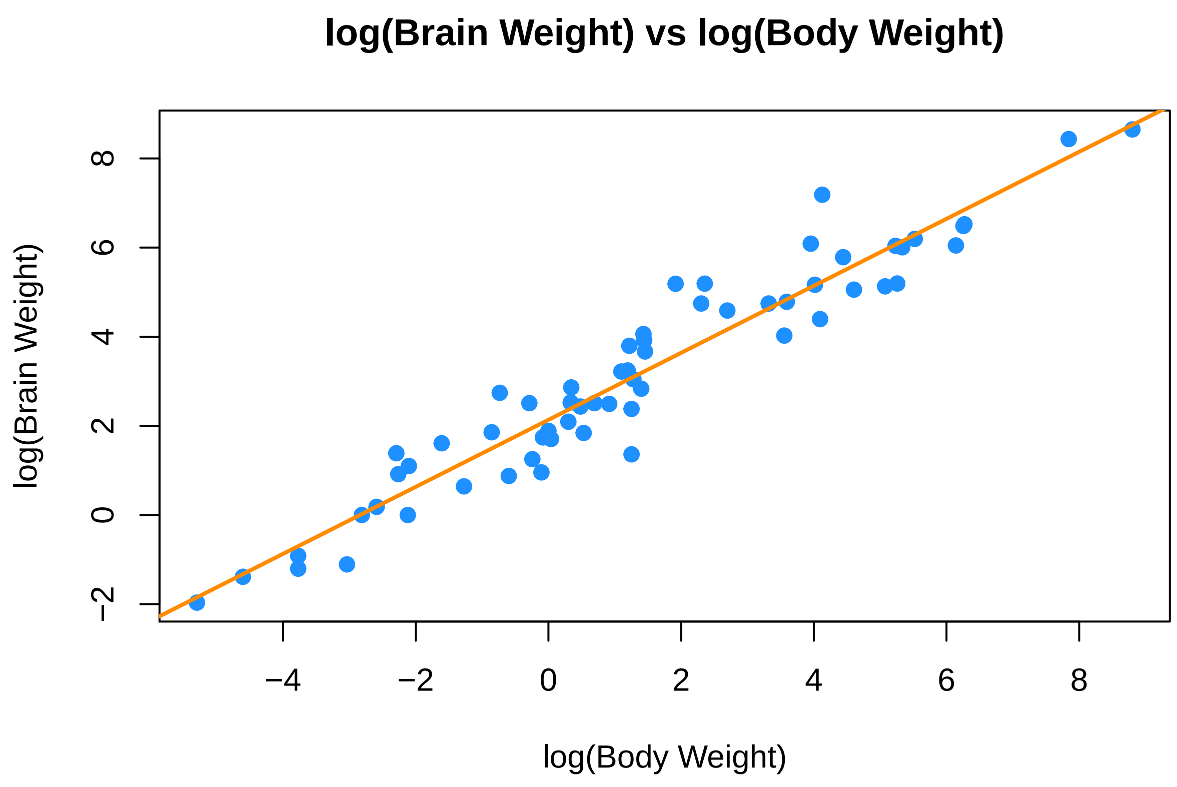


We see that the 95% confidence interval for λ estimated using the Box-Cox method does cover 0, so log transforming the response is appropriate.

(d) Fit the model justified in part **(c)**. That is, fit a model with log(brain weight) as the response and log(body weight) as a predictor. Plot log(brain weight) versus log(body weight) and add the regression line to the plot. Does a linear relationship seem to be appropriate here?

Solution:

```
fit_log = lm(log(brain) ~ log(body), data = mammals)
plot(log(brain) ~ log(body),
     data = mammals,
     main = "log(Brain Weight) vs log(Body Weight)",
     xlab = "log(Body Weight)",
     ylab = "log(Brain Weight)",
     col = "dodgerblue", pch = 20, cex = 1.5)
abline(fit_log, col = "darkorange", lwd = 2)
```

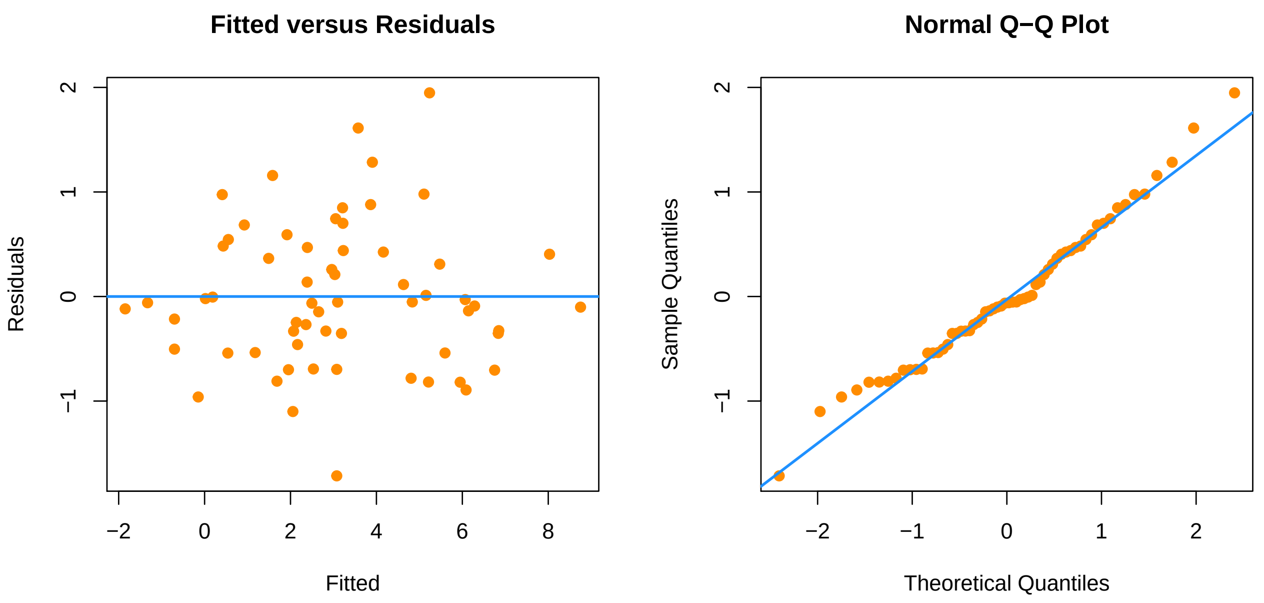


This plot suggests the linear relationship (between the log variables) is appropriate.

(e) Use a Q-Q plot to check the normality of the errors for the model fit in part **(d)**.

Solution:

```
diagnostics(fit_log, testit = FALSE, pcol = "darkorange", lcol = "dodgerblue")
```



The plot suggests that the normality assumption has not been violated.

(f) Use the model from part **(d)** to predict the brain weight of a male Snorlax which has a body weight of 1014.1 pounds. (A Snorlax would be a mammal, right?) Construct a 90% prediction interval.

Solution:

```r
snorlax = data.frame(body = 1014.1 / 2.2)
exp(predict.lm(fit_log, snorlax, interval = c("prediction"), level = 0.90))
```

```
##     fit   lwr  upr
## 1 849.9 258.2 2797
```

Note, we needed to convert from pounds to kilos, then change from a log scale back to the original scale.