HW3

9/21/2021

1

```
df = read.csv("auto-mpg.csv", head = TRUE)
names(df)
```

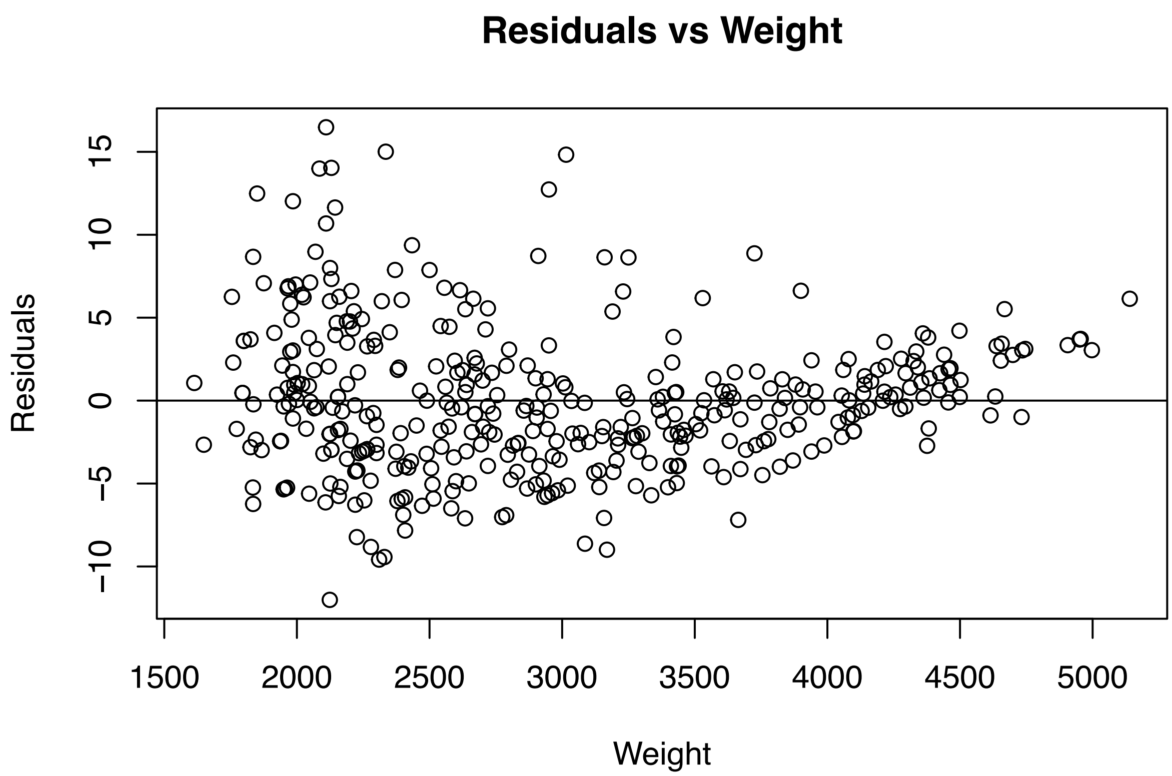
```
## [1] "mpg"          "cylinders"    "displacement" "horsepower"   "weight"
## [6] "acceleration" "model.year"   "origin"       "car.name"
```

```
str(df)
```

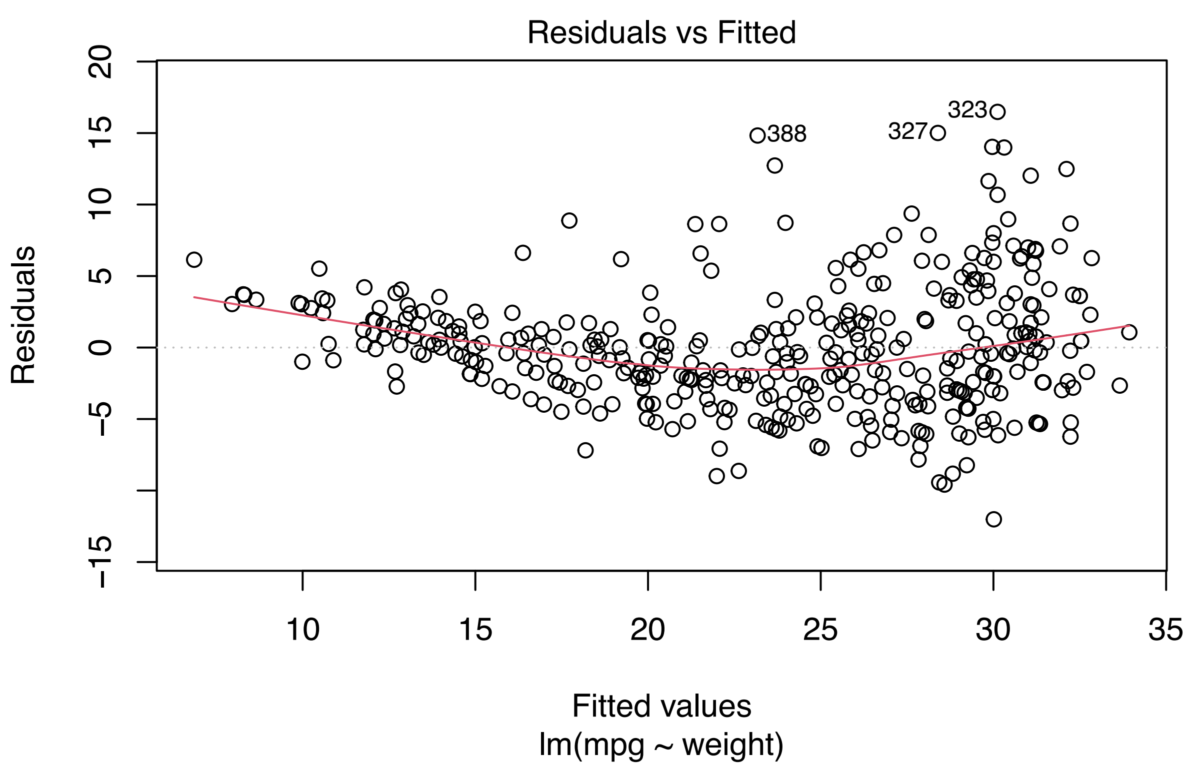
```
## 'data.frame':    398 obs. of  9 variables:
##  $ mpg         : num  18 15 18 16 17 15 14 14 14 15 ...
##  $ cylinders   : int  8 8 8 8 8 8 8 8 8 8 ...
##  $ displacement: num  307 350 318 304 302 429 454 440 455 390 ...
##  $ horsepower  : chr  "130" "165" "150" "150" ...
##  $ weight      : int  3504 3693 3436 3433 3449 4341 4354 4312 4425 3850 ...
##  $ acceleration: num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
##  $ model.year  : int  70 70 70 70 70 70 70 70 70 70 ...
##  $ origin      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ car.name    : chr  "chevrolet chevelle malibu" "buick skylark 320" "plymouth satellite" "amc rebe
```

1a

```
out_2a = lm(mpg ~ weight, df)
out_2a.res = resid(out_2a)
plot(df$weight, out_2a.res,
     xlab = "Weight", ylab = "Residuals",
     main="Residuals vs Weight")
abline(0, 0)
```



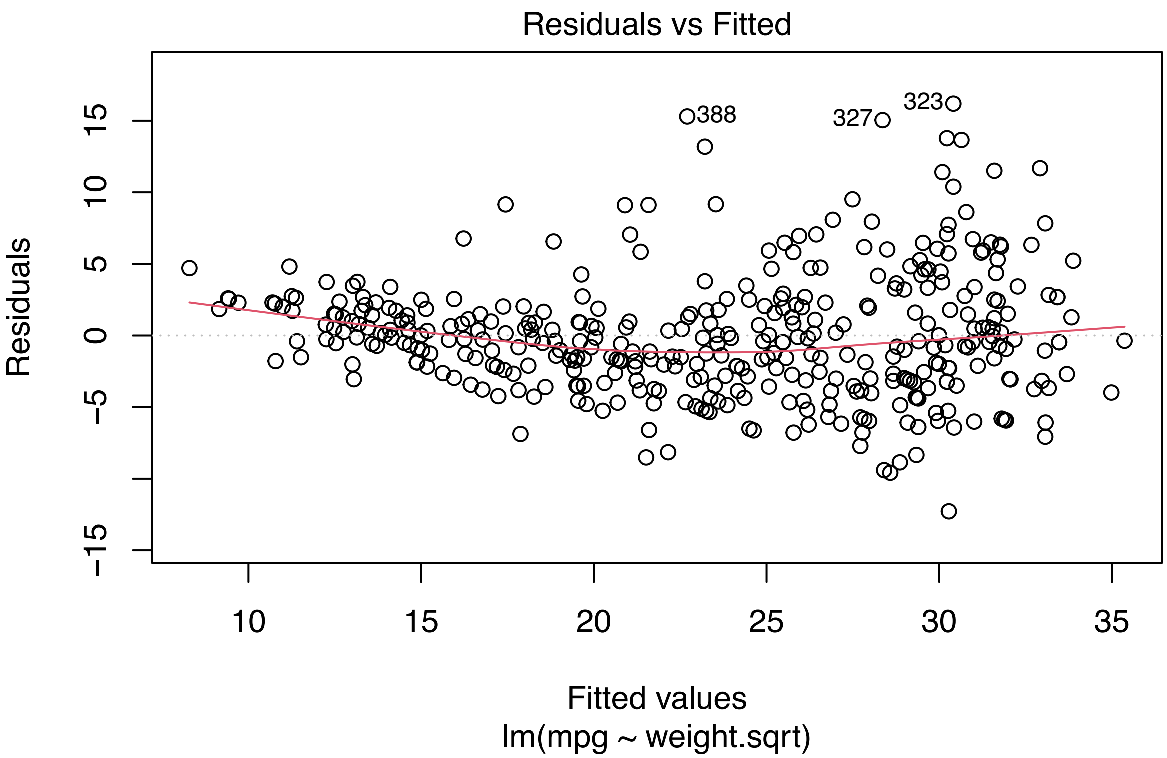

```r
plot(out_2a, which = 1)
```



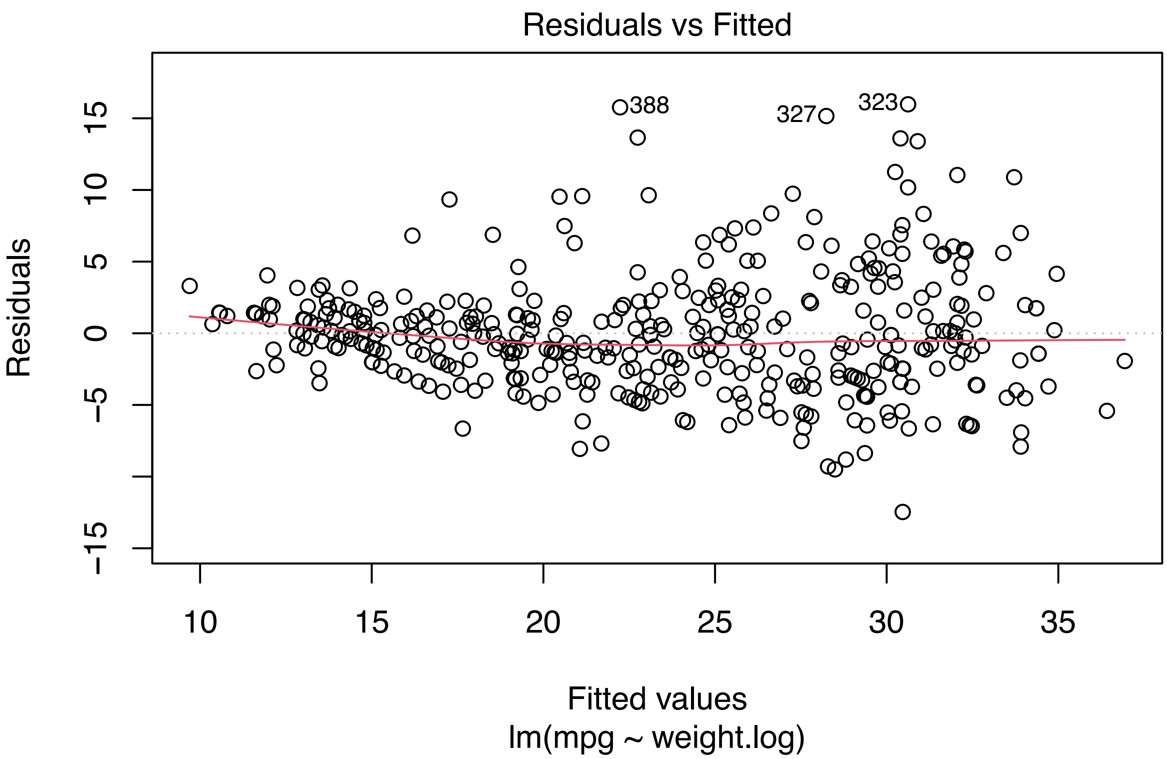
Answer: Since the residuals are randomly scattered, they are independent. However, they are not centered around 0 and do not have a constant spread across the vertical line so they don't have mean of 0 nor constant standard deviation.

1b

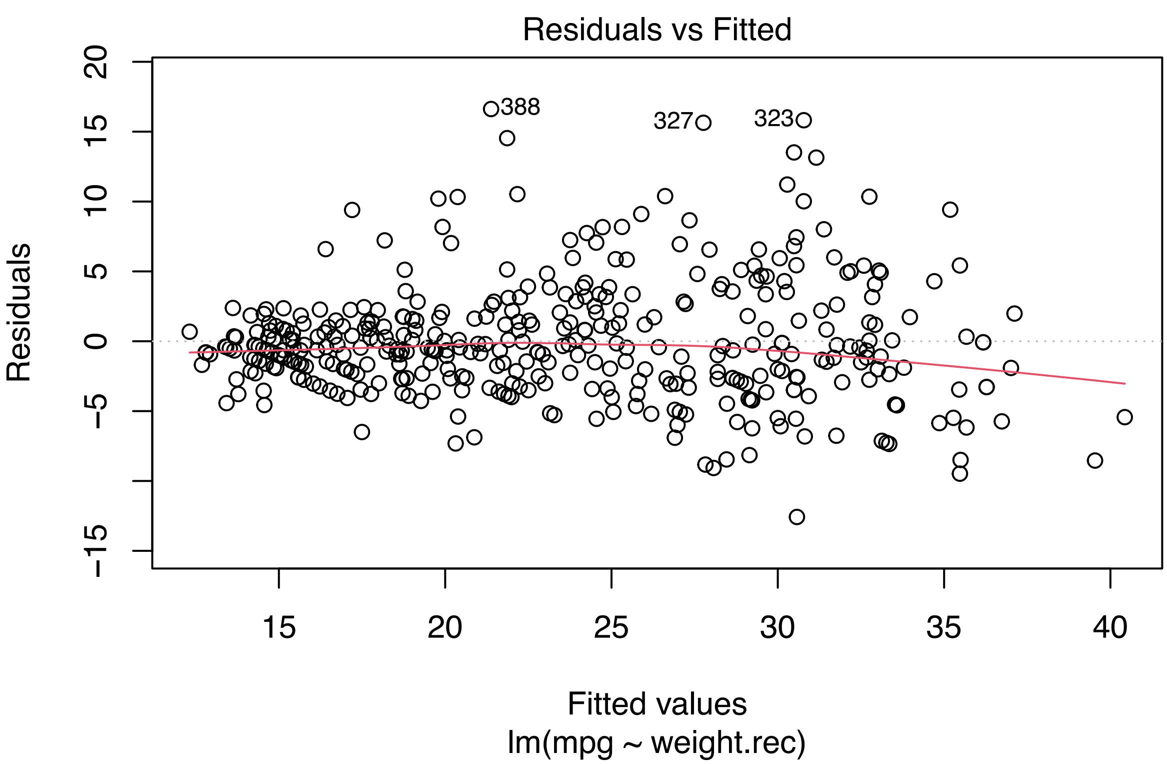
```r
df$weight.sqrt = sqrt(df$weight)
df$weight.log = log(df$weight)
df$weight.rec = 1/(df$weight)

out_sqrt = lm(mpg ~ weight.sqrt, data = df)
out_log = lm(mpg ~ weight.log, data = df)
out_rec = lm(mpg ~ weight.rec, data = df)

plot(out_sqrt, which = 1)
```




```r
plot(out_log, which = 1)
```



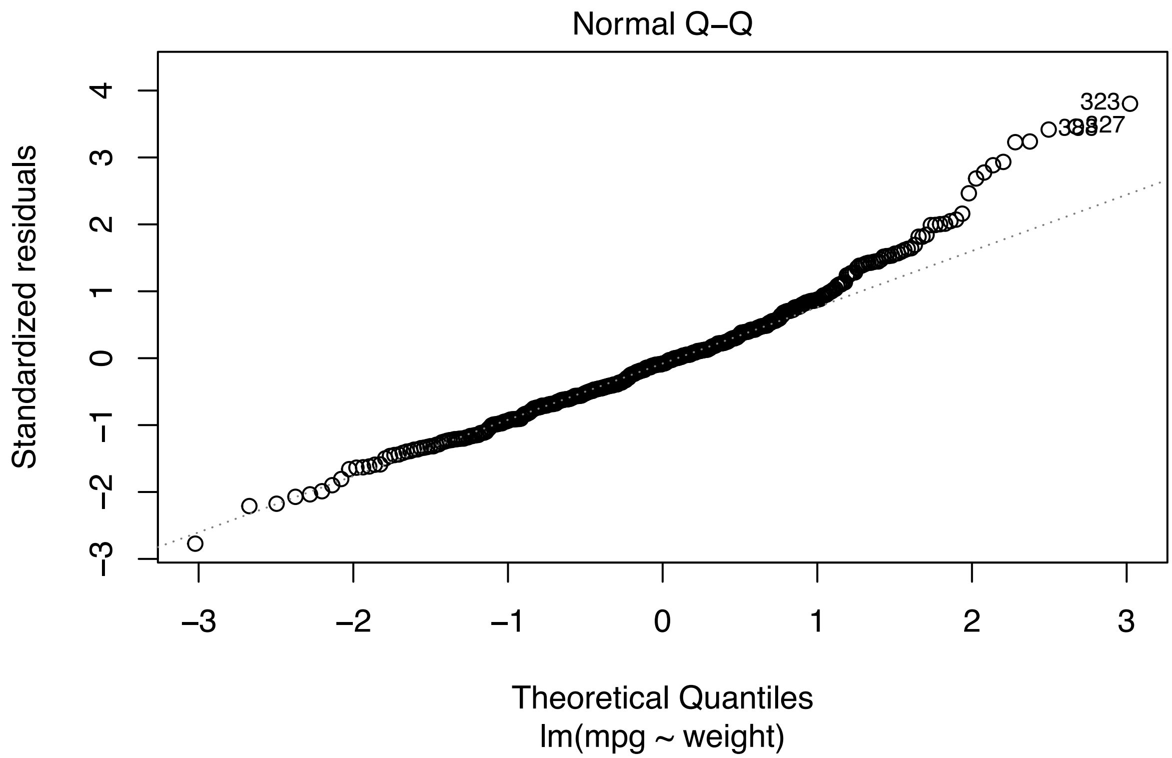

```
plot(out_rec, which = 1)
```



Answer: All three transformations violate linearity assumption. All the transformations seem to have randomly scattered residuals, which imply that they are independent. Only the log transformation seems to have residuals which have mean of 0. It is not obviously clear for the reciprocal transformation, but all three transformations seem to fail homogeneity assumption.

1c

```
plot(out_2a, which = 2)
```



Answer: Other than a few outliers at the end and some points on the right tip, most of the points lie on that diagonal line, so we can fairly assume that they seem normal.

1d

```
# From Lecture 3, pg 15
library(sandwich)
V = vcovHC(out_log)
se = sqrt(diag(V))
alpha = .1
z = -qnorm(alpha/2)
left  = out_log$coef - z*se
right = out_log$coef + z*se
print(cbind(left,right))
```

```
##                  left     right
## (Intercept) 201.33254 219.72152
## weight.log  -24.63461 -22.37181
```

Answer: The 90 percent confidence interval is (-24.63461, -22.37181)

2

```
df_2 = read.csv("auto-mpg.csv", head = TRUE)
df_2 = df_2[,-9]
```

```r
str(df_2)
```

```
## 'data.frame':    398 obs. of  8 variables:
##  $ mpg         : num  18 15 18 16 17 15 14 14 14 15 ...
##  $ cylinders   : int  8 8 8 8 8 8 8 8 8 8 ...
##  $ displacement: num  307 350 318 304 302 429 454 440 455 390 ...
##  $ horsepower  : chr  "130" "165" "150" "150" ...
##  $ weight      : int  3504 3693 3436 3433 3449 4341 4354 4312 4425 3850 ...
##  $ acceleration: num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
##  $ model.year  : int  70 70 70 70 70 70 70 70 70 70 ...
##  $ origin      : int  1 1 1 1 1 1 1 1 1 1 ...
```

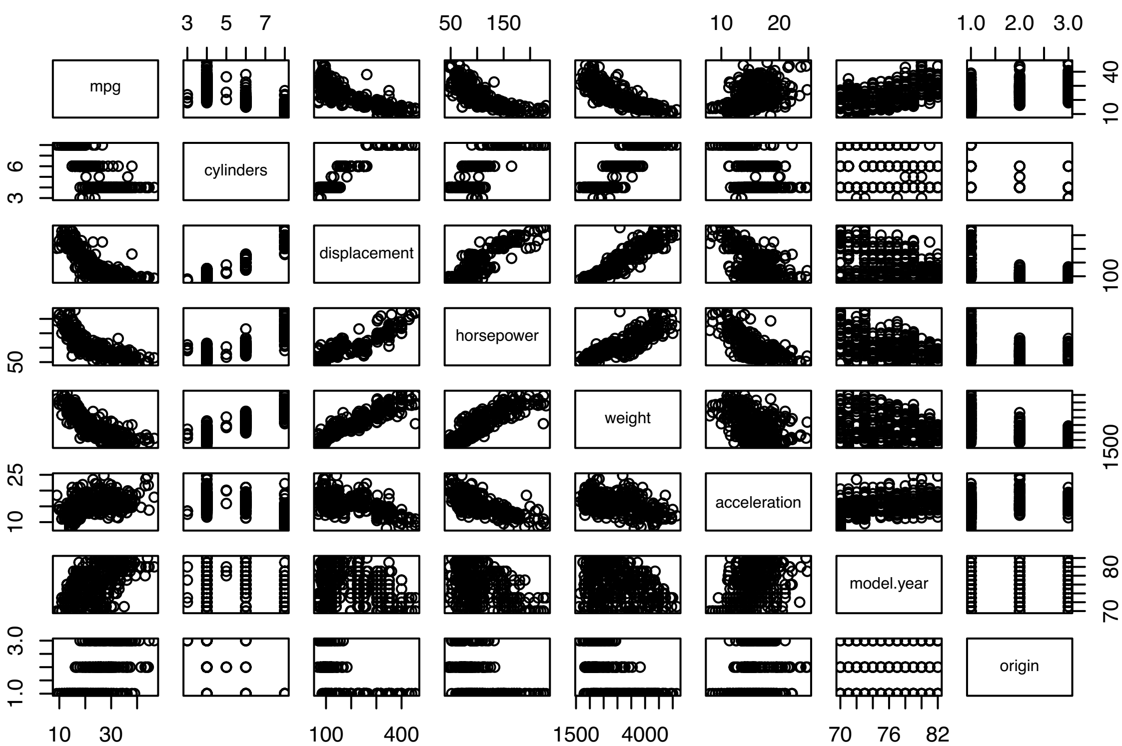
```r
df_2$horsepower = as.numeric(df_2$horsepower)
```

```
## Warning: NAs introduced by coercion
```

```r
df_2 = df_2[complete.cases(df_2), ]
```

2a

```r
pairs(df_2)
```



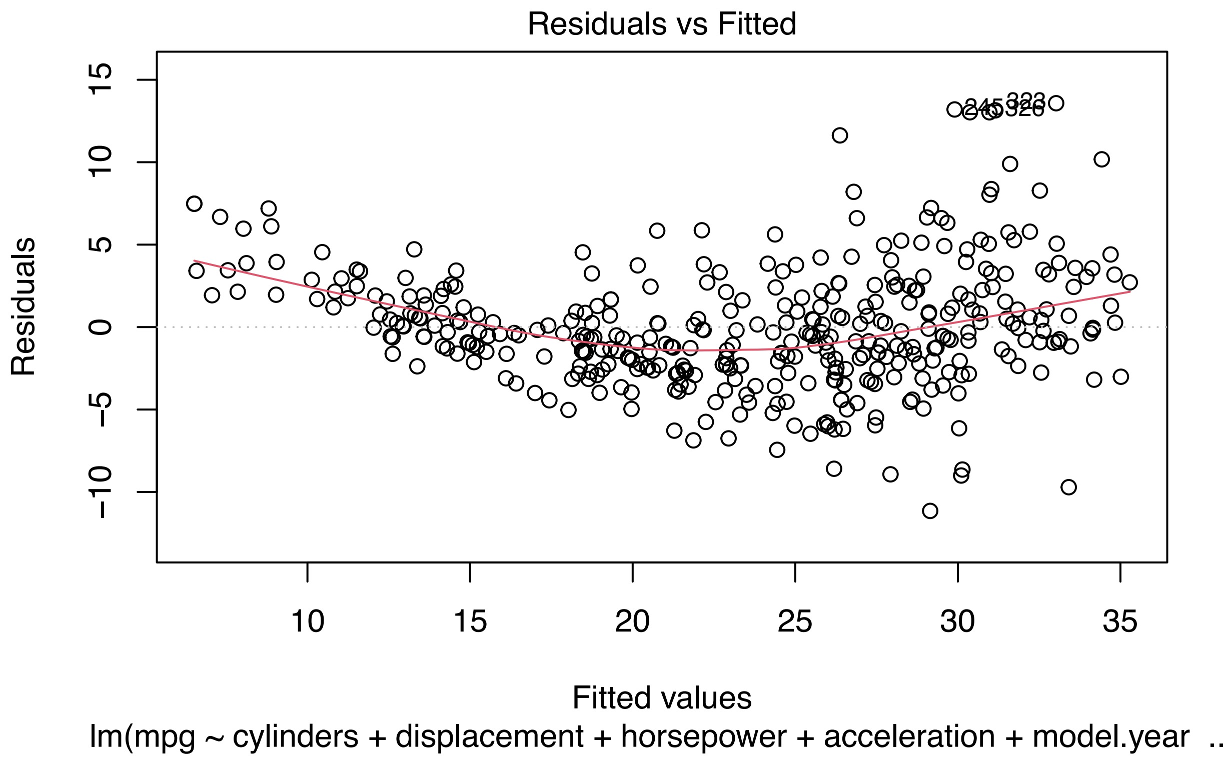
Answer: Based on the pairs plot, `mpg` seems to have negative relationship with `displacement`, `horsepower` and `weight` and positive relationship with `acceleration` and `model year`.

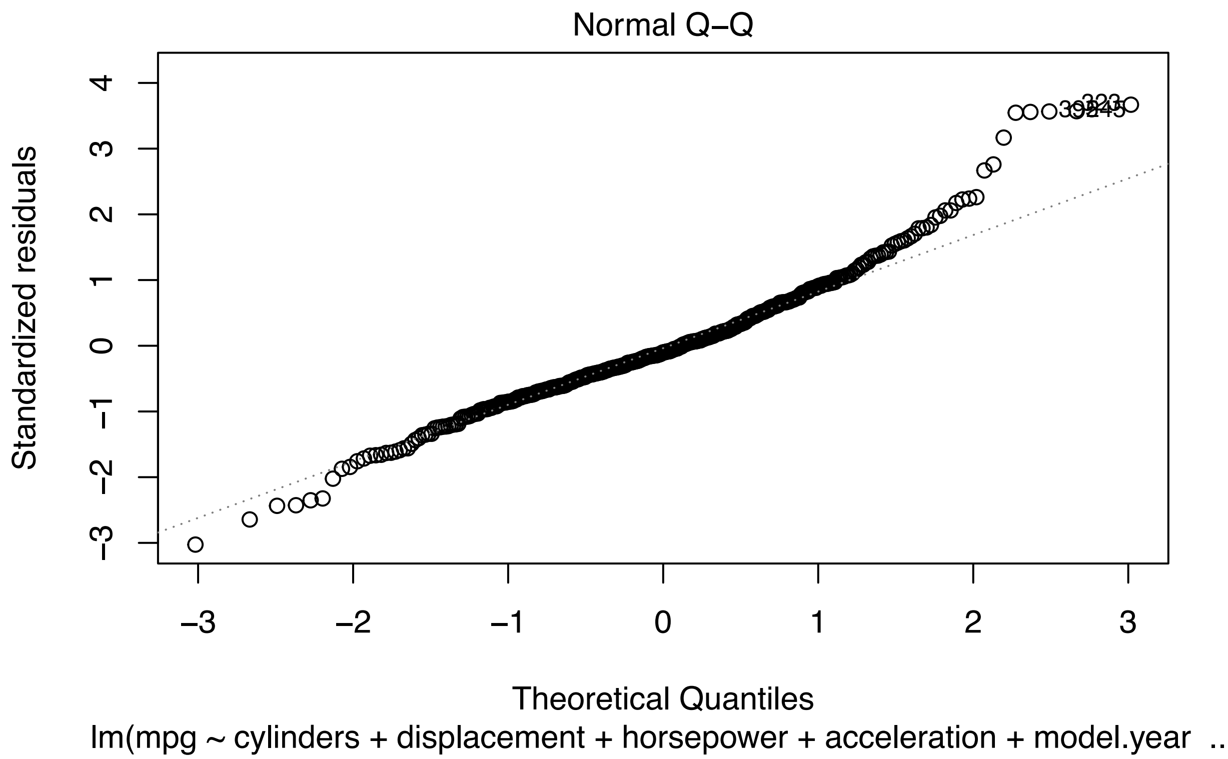
2b

```r
library(car)
```

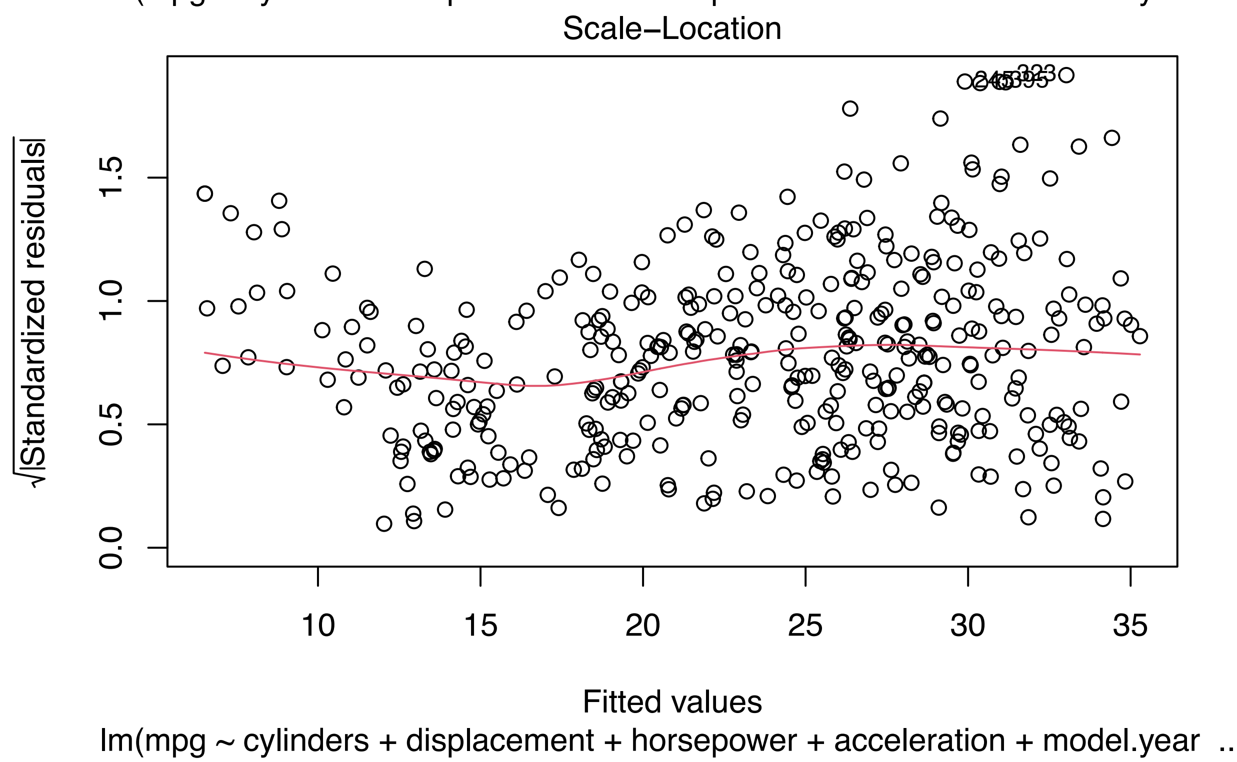
```
## Loading required package: carData
```

```r
library(leaps)

out_2b = lm(mpg ~ cylinders + displacement + horsepower
            + acceleration + model.year + origin, data = df_2)

plot(out_2b)
```

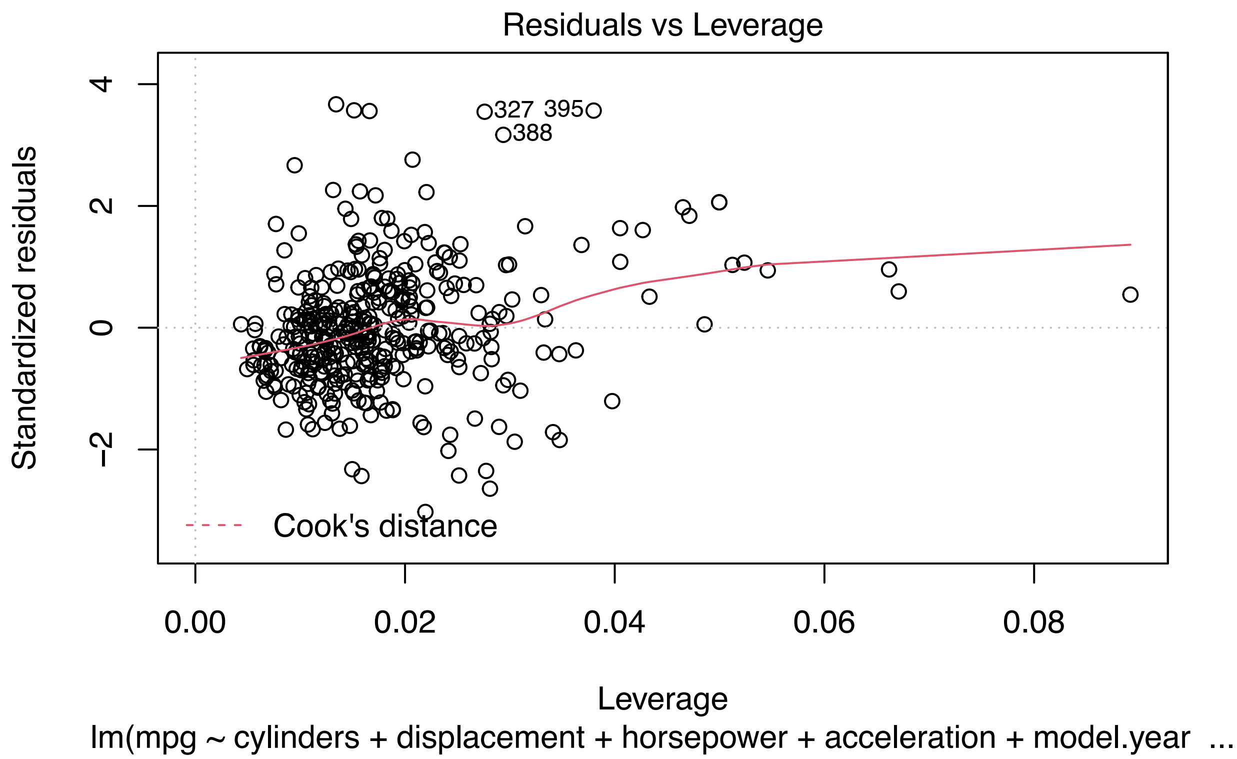




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(mpg ~ cylinders + displacement + horsepower + acceleration + model.year ...



Scale−Location
√|Standardized residuals|
Fitted values
lm(mpg ~ cylinders + displacement + horsepower + acceleration + model.year ...



Answer: The residuals of the regression model fitted on all the other variables seem normally distributed but do not seem to be independent. They seem to be centered around 0 so have mean of 0. They also approximately have a constant spread.

Next, I used `regsubsets()` from `leaps` package to get the best subsets and checked vif of each sub-model for any multicollinearity problem.

```
car::vif(out_2b)
```

```
##    cylinders displacement   horsepower acceleration   model.year       origin
##    10.563572    18.587690     7.888801     2.017986     1.209915     1.756092
```

```
best.subs.df_2 = leaps::regsubsets(mpg ~., data = df_2, nvmax = 5)
summary(best.subs.df_2)
```

```
## Subset selection object
## Call: regsubsets.formula(mpg ~ ., data = df_2, nvmax = 5)
## 7 Variables  (and intercept)
##              Forced in Forced out
## cylinders        FALSE      FALSE
## displacement     FALSE      FALSE
## horsepower       FALSE      FALSE
## weight           FALSE      FALSE
## acceleration     FALSE      FALSE
## model.year       FALSE      FALSE
## origin           FALSE      FALSE
## 1 subsets of each size up to 5
## Selection Algorithm: exhaustive
##          cylinders displacement horsepower weight acceleration model.year
## 1  ( 1 ) " "       " "          " "        "*"    " "          " "
## 2  ( 1 ) " "       " "          " "        "*"    " "          "*"
## 3  ( 1 ) " "       " "          " "        "*"    " "          "*"
```

```
## 4  ( 1 ) " "       "*"          " "         "*"    " "           "*"
## 5  ( 1 ) " "       "*"          "*"         "*"    " "           "*"
##           origin
## 1  ( 1 ) " "
## 2  ( 1 ) " "
## 3  ( 1 ) "*"
## 4  ( 1 ) "*"
## 5  ( 1 ) "*"
```

```r
best.from.5 = lm(mpg ~ displacement + horsepower+ weight + model.year + origin, data = df_2)
best.from.4 = lm(mpg ~ displacement + weight + model.year + origin, data = df_2)
best.from.3 = lm(mpg ~ weight + model.year + origin, data = df_2)
best.from.2 = lm(mpg ~ weight + model.year, data = df_2)
best.from.1 = lm(mpg ~ weight, data = df_2)

car::vif(best.from.5)
```

```
## displacement   horsepower       weight   model.year       origin
##    11.815410     6.064714     8.208873     1.237146     1.755107
```

```r
car::vif(best.from.4)
```

```
## displacement       weight   model.year       origin
##     8.694250     7.820387     1.176033     1.615041
```

```r
car::vif(best.from.3)
```

```
##     weight model.year     origin
##   1.625522   1.105651   1.520292
```

```r
car::vif(best.from.2)
```

```
##     weight model.year
##   1.105651   1.105651
```

Since for models with number of predictors more than 3 have predictors whose vif was over 2.5, I decided to keep models with number of predictors less than or equal to 3. In order to choose the model, I looked at each model's adjusted R^2 value.

```r
summary(best.from.3)
```

```
##
## Call:
## lm(formula = mpg ~ weight + model.year + origin, data = df_2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.9440 -2.0948 -0.0389  1.7255 13.2722
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.805e+01  4.001e+00  -4.510 8.60e-06 ***
## weight      -5.994e-03  2.541e-04 -23.588  < 2e-16 ***
## model.year   7.571e-01  4.832e-02  15.668  < 2e-16 ***
## origin       1.150e+00  2.591e-01   4.439 1.18e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.348 on 388 degrees of freedom
```

```
## Multiple R-squared:  0.8175, Adjusted R-squared:  0.816
## F-statistic: 579.2 on 3 and 388 DF,  p-value: < 2.2e-16
```

```
summary(best.from.2)
```

```
##
## Call:
## lm(formula = mpg ~ weight + model.year, data = df_2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.8505 -2.3014 -0.1167  2.0367 14.3555
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.435e+01  4.007e+00  -3.581 0.000386 ***
## weight      -6.632e-03  2.146e-04 -30.911  < 2e-16 ***
## model.year   7.573e-01  4.947e-02  15.308  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.427 on 389 degrees of freedom
## Multiple R-squared:  0.8082, Adjusted R-squared:  0.8072
## F-statistic: 819.5 on 2 and 389 DF,  p-value: < 2.2e-16
```

```
summary(best.from.1)
```

```
##
## Call:
## lm(formula = mpg ~ weight, data = df_2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.9736  -2.7556  -0.3358   2.1379  16.5194
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 46.216524   0.798673   57.87   <2e-16 ***
## weight      -0.007647   0.000258  -29.64   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.333 on 390 degrees of freedom
## Multiple R-squared:  0.6926, Adjusted R-squared:  0.6918
## F-statistic: 878.8 on 1 and 390 DF,  p-value: < 2.2e-16
```

I chose the model with three predictors, since ithas the highest adjusted R-squared value.

```
summary(best.from.3)
```

```
##
## Call:
## lm(formula = mpg ~ weight + model.year + origin, data = df_2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.9440 -2.0948 -0.0389  1.7255 13.2722
```

```
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.805e+01  4.001e+00  -4.510 8.60e-06 ***
## weight      -5.994e-03  2.541e-04 -23.588  < 2e-16 ***
## model.year   7.571e-01  4.832e-02  15.668  < 2e-16 ***
## origin       1.150e+00  2.591e-01   4.439 1.18e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.348 on 388 degrees of freedom
## Multiple R-squared:  0.8175, Adjusted R-squared:  0.816
## F-statistic: 579.2 on 3 and 388 DF,  p-value: < 2.2e-16
```

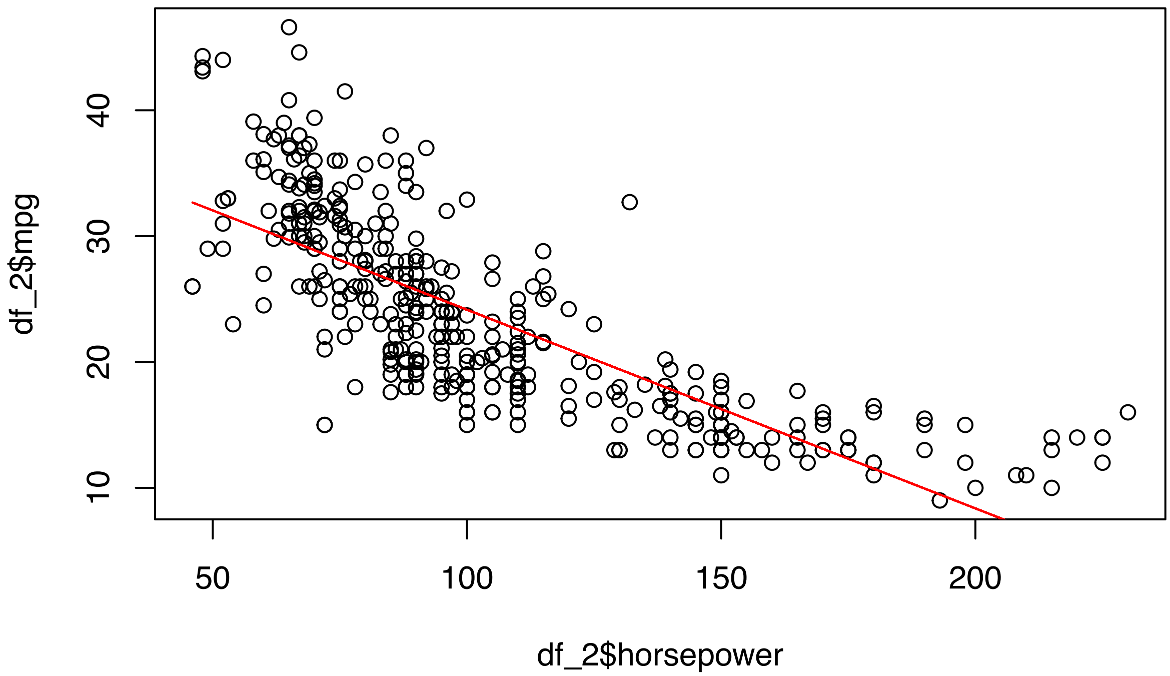
```r
# From Lecture 3, pg 15
V = vcovHC(best.from.3)
se = sqrt(diag(V))
alpha = .05
z = -qnorm(alpha/2)
left  = best.from.3$coef - z*se
right = best.from.3$coef + z*se
print(cbind(left,right))
```

```
##                     left         right
## (Intercept) -25.479213396 -10.612486903
## weight       -0.006451767  -0.005536468
## model.year    0.662799634   0.851452588
## origin        0.600971918   1.699809660
```

Answer: B0 = -1.788e+01, B1 = -6.023e-03, B2 = 7.559e-01, B3 = 1.166e+00 where B0 is the intercept, B1 is the slope of weight, B2 is the slope of model.year and B3 is the slope of origin. The 95% confidence interval for the intercept is (-25.479213396, -10.612486903), for the weight is (-0.006451767, -0.005536468), for the model.year is (0.662799634, 0.851452588) and for the origin is (0.600971918, 1.699809660)

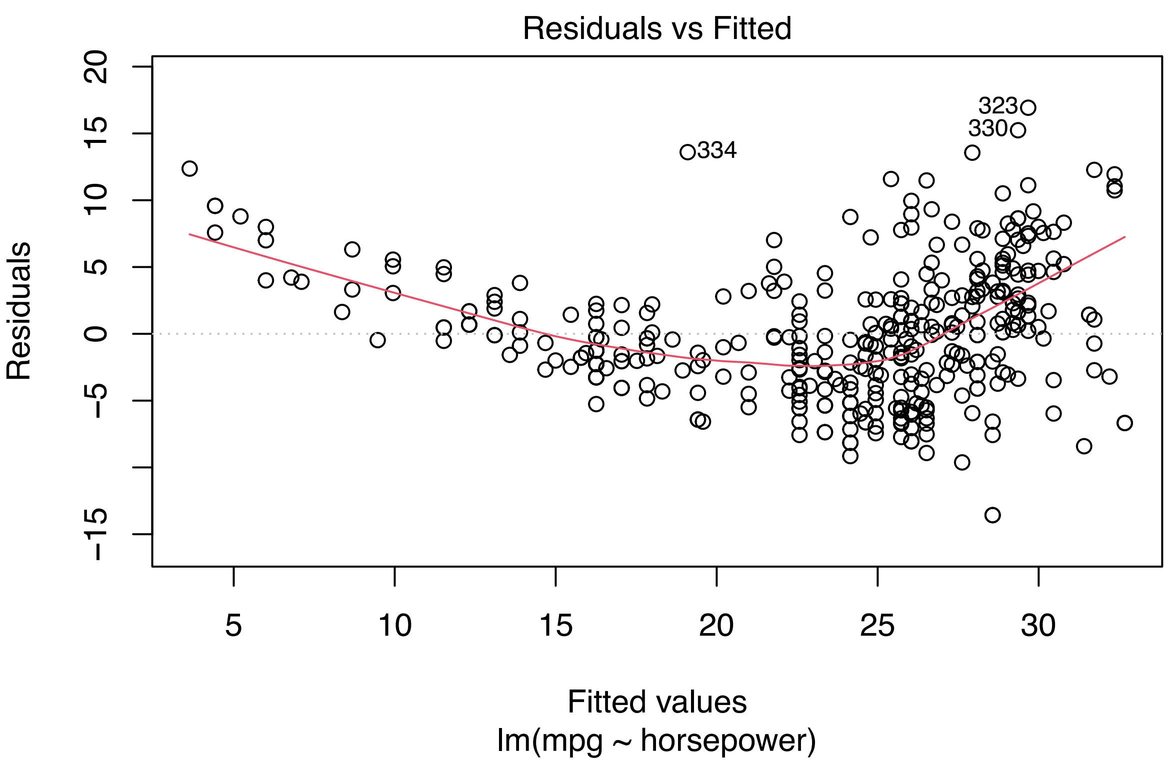
2c

```r
out_2c_1 = lm(mpg ~ horsepower, data = df_2)
plot(df_2$horsepower, df_2$mpg)
lines(df_2$horsepower, fitted(out_2c_1), col = "red")
```

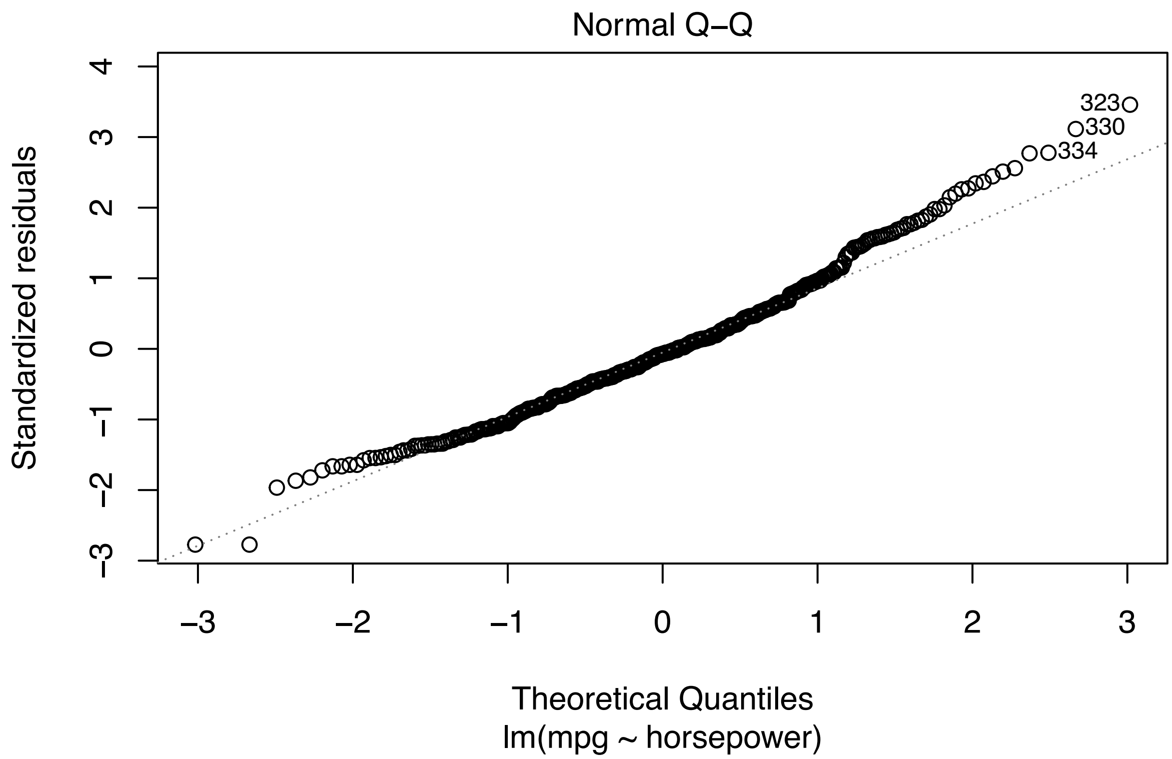


df_2$mpg
10
20
30
40
50
100
150
200
df_2$horsepower

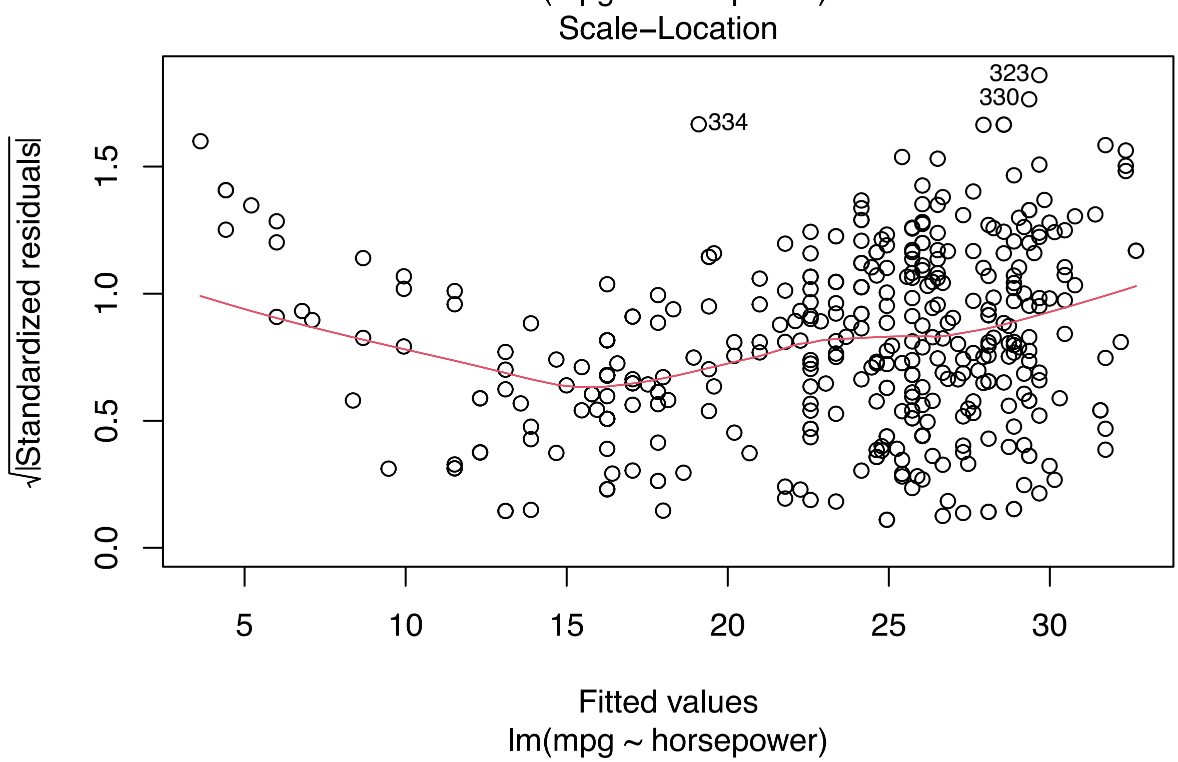

```
plot(out_2c_1)
```



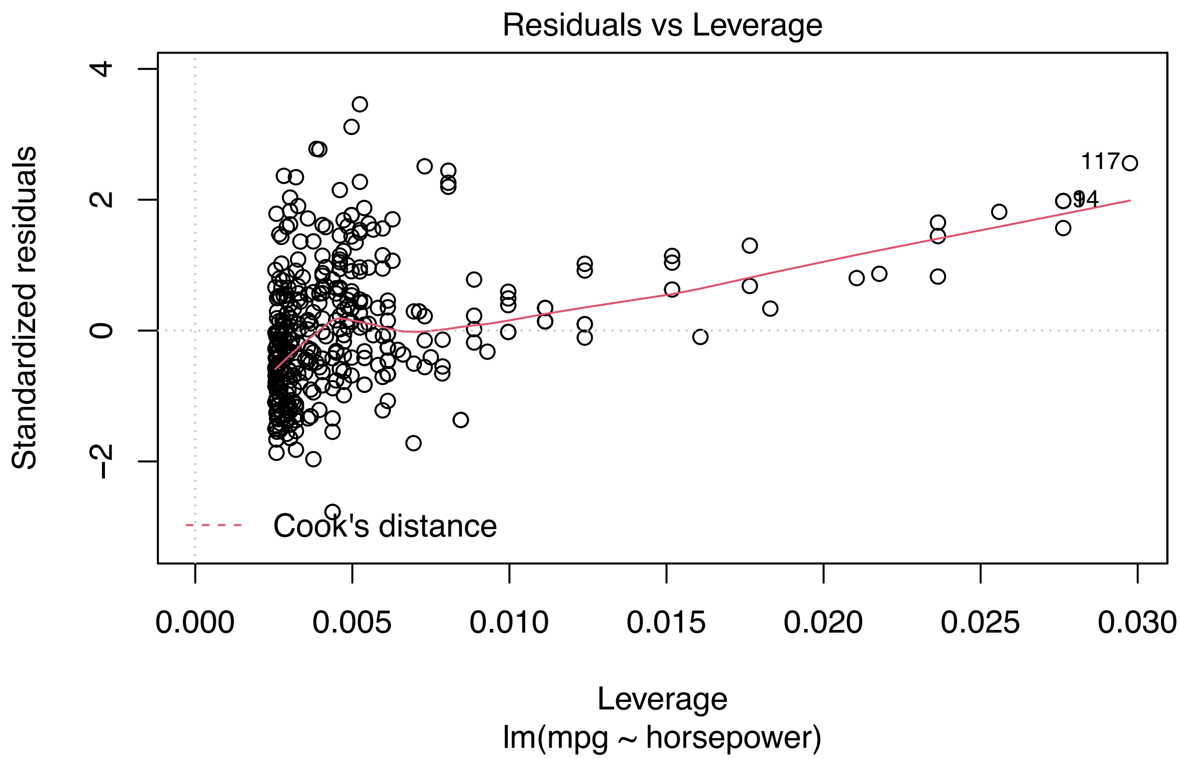
Residuals vs Fitted
Residuals
−15
−5
0
5
10
15
20
334
323
330
5
10
15
20
25
30
Fitted values
lm(mpg ~ horsepower)



Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(mpg ~ horsepower)
323
330
334

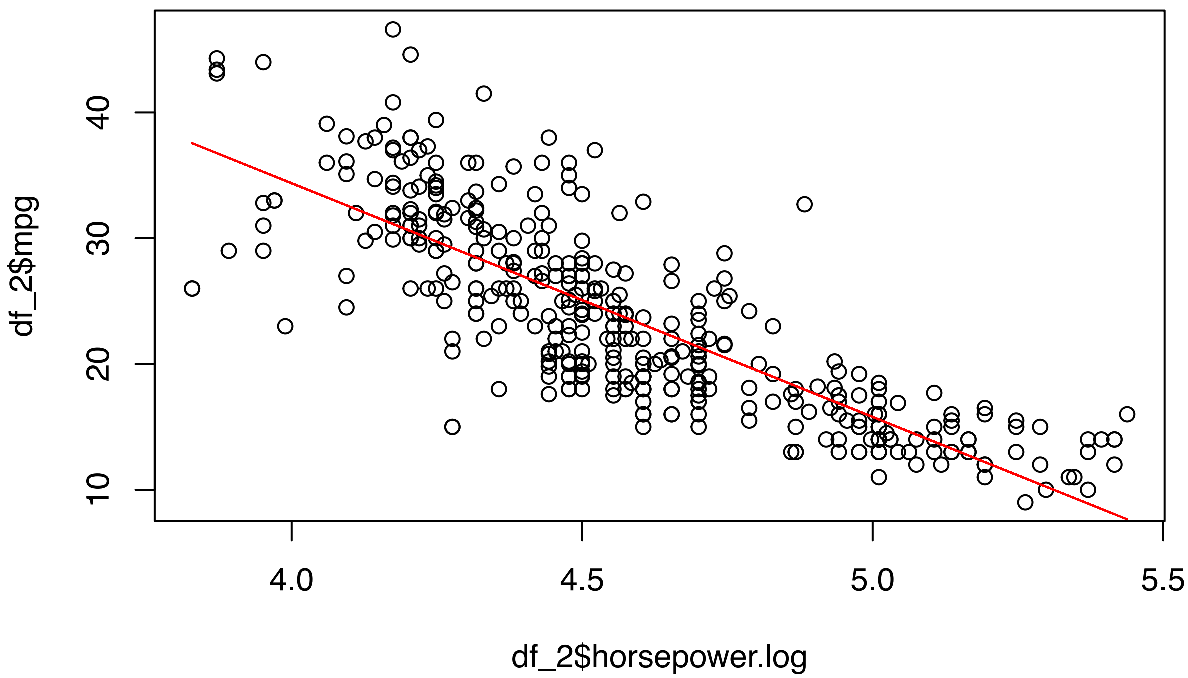


Scale−Location
√|Standardized residuals|
Fitted values
lm(mpg ~ horsepower)
323
330
334

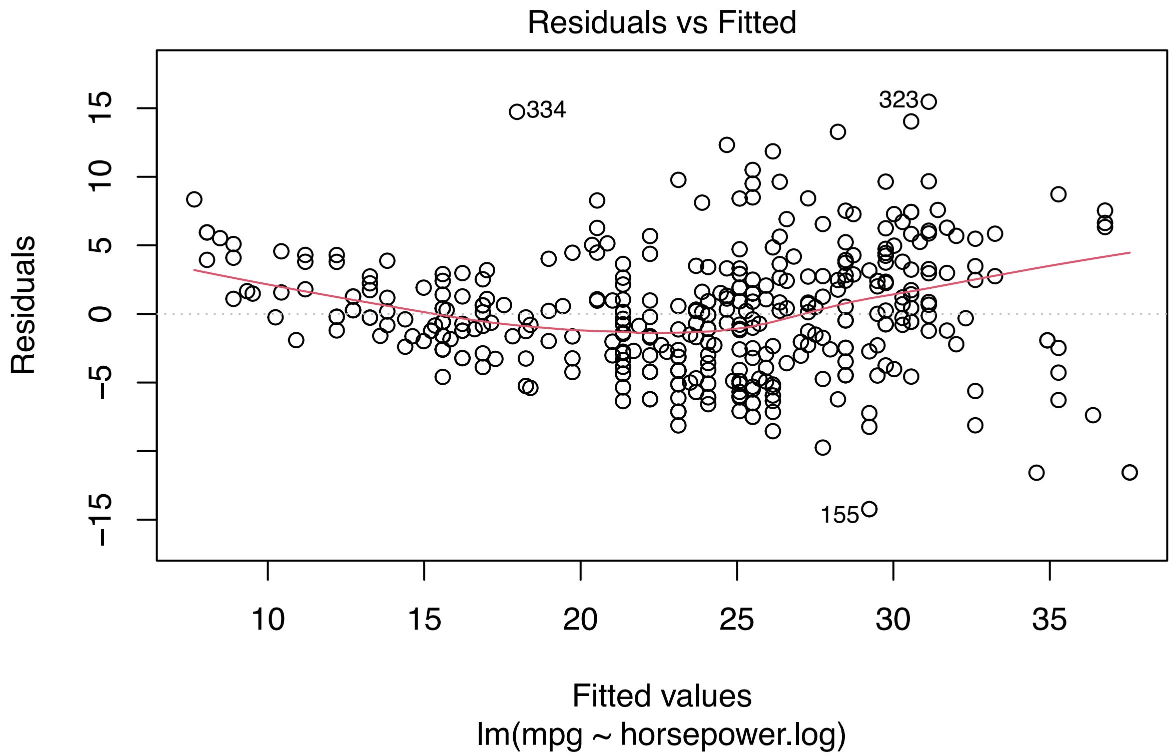


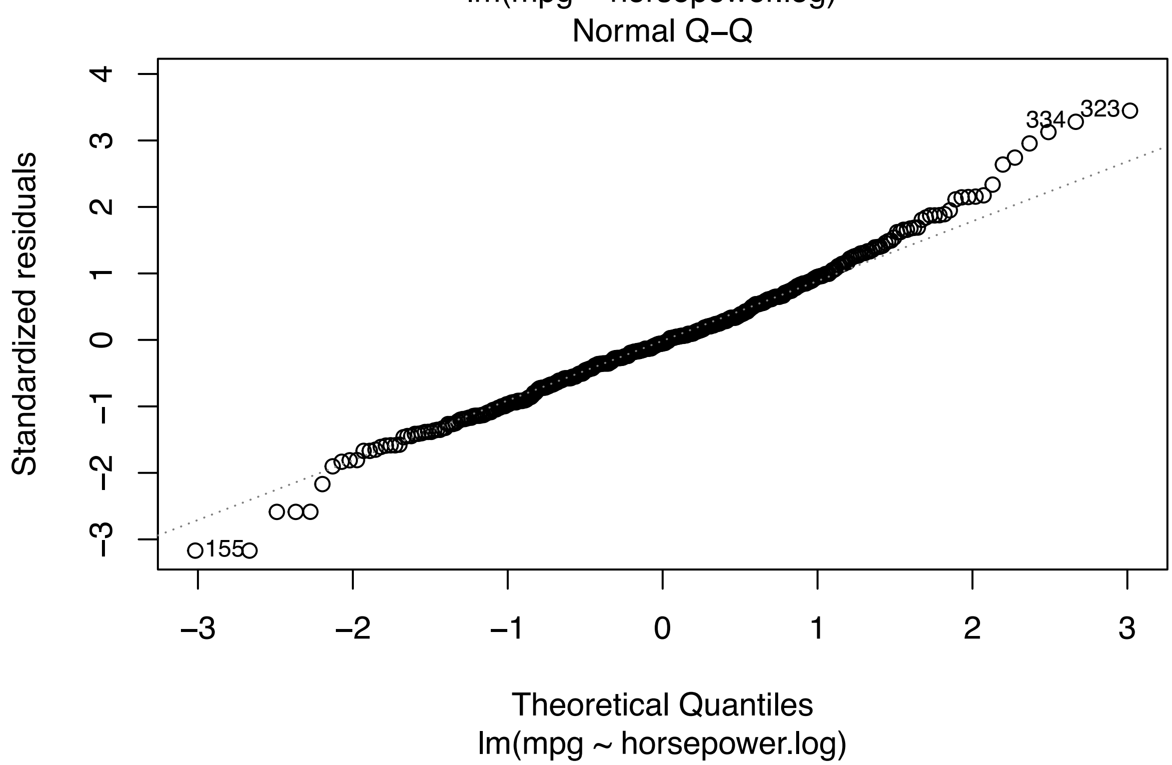
From the residuals vs fitted values plot, the residuals show a curved pattern and therefore are not independent, violating linearity assumption for the model. I tried log transformation and below is the result:

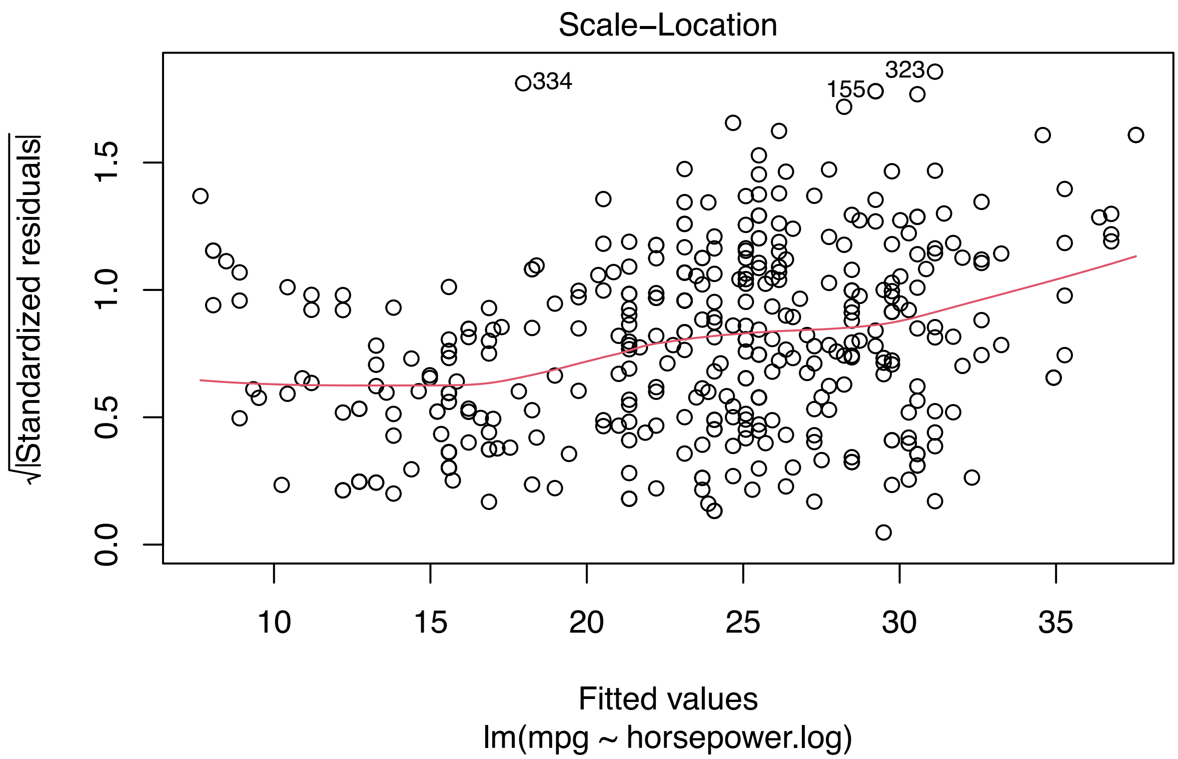
```
df_2$horsepower.log = log(df_2$horsepower)
out_2c_2 = lm(mpg ~ horsepower.log, data = df_2)
plot(df_2$horsepower.log, df_2$mpg)
lines(df_2$horsepower.log, fitted(out_2c_2), col = "red")
```

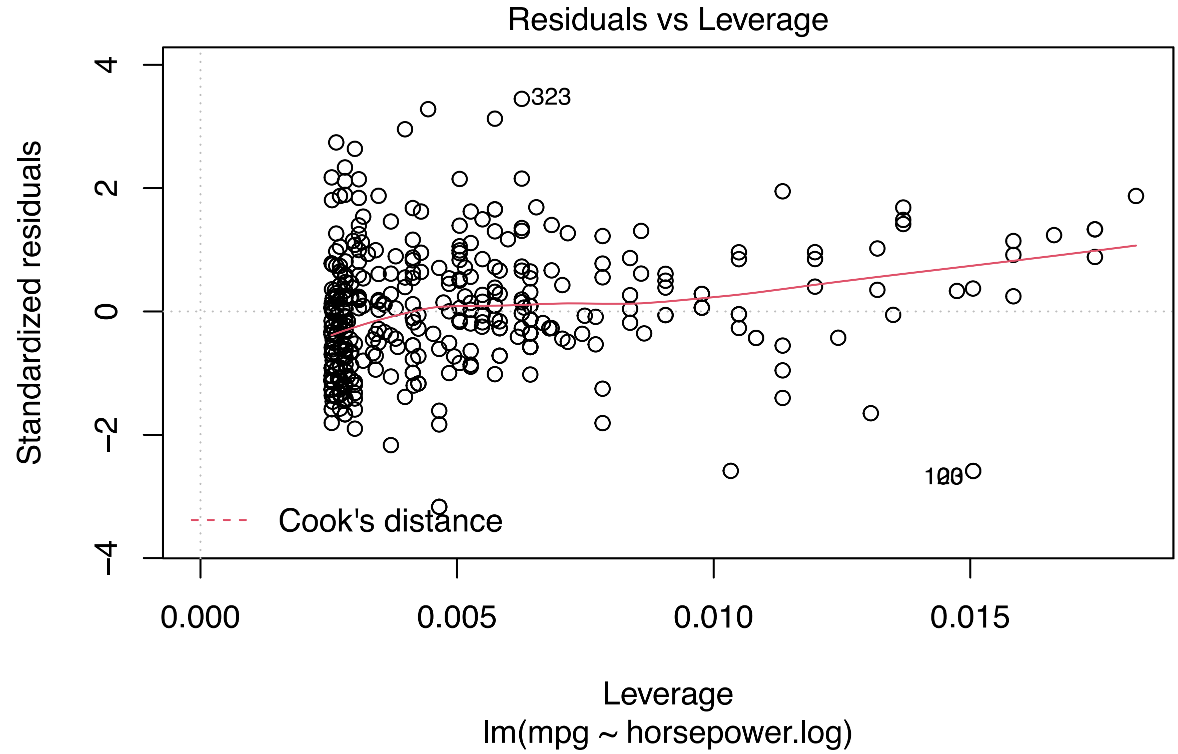


```
plot(out_2c_2)
```


Residuals vs Fitted
Residuals
Fitted values
lm(mpg ~ horsepower.log)
334
323
155
−15
−5
0
5
10
15
10
15
20
25
30
35


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(mpg ~ horsepower.log)
334
323
155
−3
−2
−1
0
1
2
3
4





After the transformation, the first plot seems more linear than the one before the transformation. The residuals still show a curved pattern, indicating violation of independence assumption.

3

3a

```
par(mfrow = c(2,2))
df_3 = read.table("gpa.txt")
names(df_3) = c("GPA", "ACT")
summary(lm(GPA ~ poly(ACT,1), data = df_3))
```

```
##
## Call:
## lm(formula = GPA ~ poly(ACT, 1), data = df_3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.74004 -0.33827  0.04062  0.44064  1.22737
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.07405    0.05688   54.04  < 2e-16 ***
## poly(ACT, 1)  1.89416    0.62313    3.04  0.00292 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6231 on 118 degrees of freedom
## Multiple R-squared:  0.07262,    Adjusted R-squared:  0.06476
## F-statistic:  9.24 on 1 and 118 DF,  p-value: 0.002917
```

```
summary(lm(GPA ~ poly(ACT,2), data = df_3))
```

```
##
## Call:
## lm(formula = GPA ~ poly(ACT, 2), data = df_3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.73899 -0.30901  0.02954  0.43309  1.31931
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    3.07405    0.05708  53.853  < 2e-16 ***
## poly(ACT, 2)1  1.89416    0.62530   3.029  0.00302 **
## poly(ACT, 2)2 -0.26584    0.62530  -0.425  0.67151
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6253 on 117 degrees of freedom
## Multiple R-squared:  0.07405,    Adjusted R-squared:  0.05822
## F-statistic: 4.678 on 2 and 117 DF,  p-value: 0.0111
```

```
summary(lm(GPA ~ poly(ACT,3), data = df_3))
```

```
##
## Call:
## lm(formula = GPA ~ poly(ACT, 3), data = df_3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -2.74940 -0.31673  0.02534  0.44473  1.26005
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     3.07405    0.05732  53.634  < 2e-16 ***
## poly(ACT, 3)1  1.89416    0.62786   3.017  0.00314 **
## poly(ACT, 3)2 -0.26584    0.62786  -0.423  0.67278
## poly(ACT, 3)3 -0.13725    0.62786  -0.219  0.82734
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6279 on 116 degrees of freedom
## Multiple R-squared:  0.07443,    Adjusted R-squared:  0.0505
## F-statistic: 3.109 on 3 and 116 DF,  p-value: 0.02915
```

```
summary(lm(GPA ~ poly(ACT,4), data = df_3))
```

```
##
## Call:
## lm(formula = GPA ~ poly(ACT, 4), data = df_3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.6813 -0.3422  0.0222  0.4421  1.1400
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     3.07405    0.05689  54.031  < 2e-16 ***
## poly(ACT, 4)1  1.89416    0.62325   3.039  0.00294 **
## poly(ACT, 4)2 -0.26584    0.62325  -0.427  0.67051
## poly(ACT, 4)3 -0.13725    0.62325  -0.220  0.82609
## poly(ACT, 4)4  1.02866    0.62325   1.650  0.10157
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6232 on 115 degrees of freedom
## Multiple R-squared:  0.09585,    Adjusted R-squared:  0.0644
## F-statistic: 3.048 on 4 and 115 DF,  p-value: 0.01984
```

I would choose a simple regression model for this data set, since 1) there is not much of difference between the adjusted R^2 values 2) the p-values for polynomial variables are greater than 0.05, and 3) the issue of over-fitting rises as the order of polynomial increases.

```
out_3a = lm(GPA ~ poly(ACT, 1), data = df_3)
summary(out_3a)
```

```
##
## Call:
## lm(formula = GPA ~ poly(ACT, 1), data = df_3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.74004 -0.33827  0.04062  0.44064  1.22737
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)   3.07405    0.05688   54.04  < 2e-16 ***
## poly(ACT, 1)  1.89416    0.62313    3.04  0.00292 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6231 on 118 degrees of freedom
## Multiple R-squared:  0.07262,    Adjusted R-squared:  0.06476
## F-statistic:  9.24 on 1 and 118 DF,  p-value: 0.002917
```

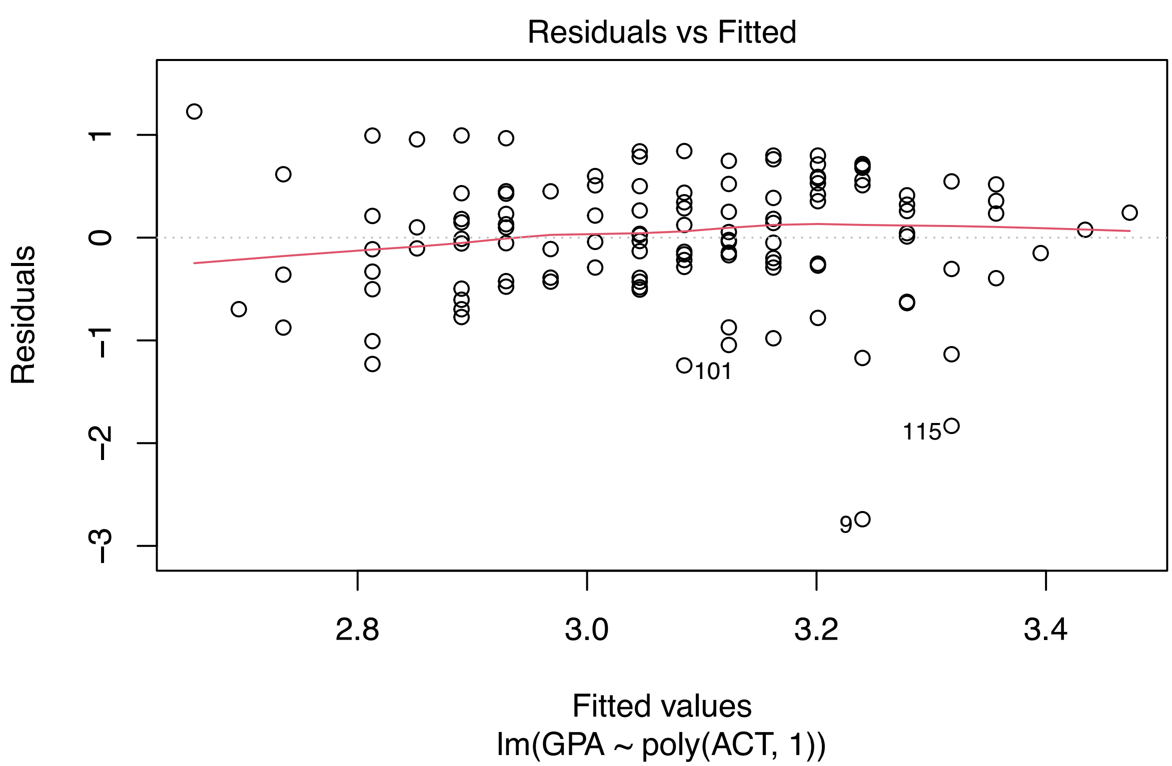
```
# From Lecture 3, pg 15
V = vcovHC(out_3a)
se = sqrt(diag(V))
alpha = .05
z = -qnorm(alpha/2)
left  = out_3a$coef - z*se
right = out_3a$coef + z*se
print(cbind(left,right))
```

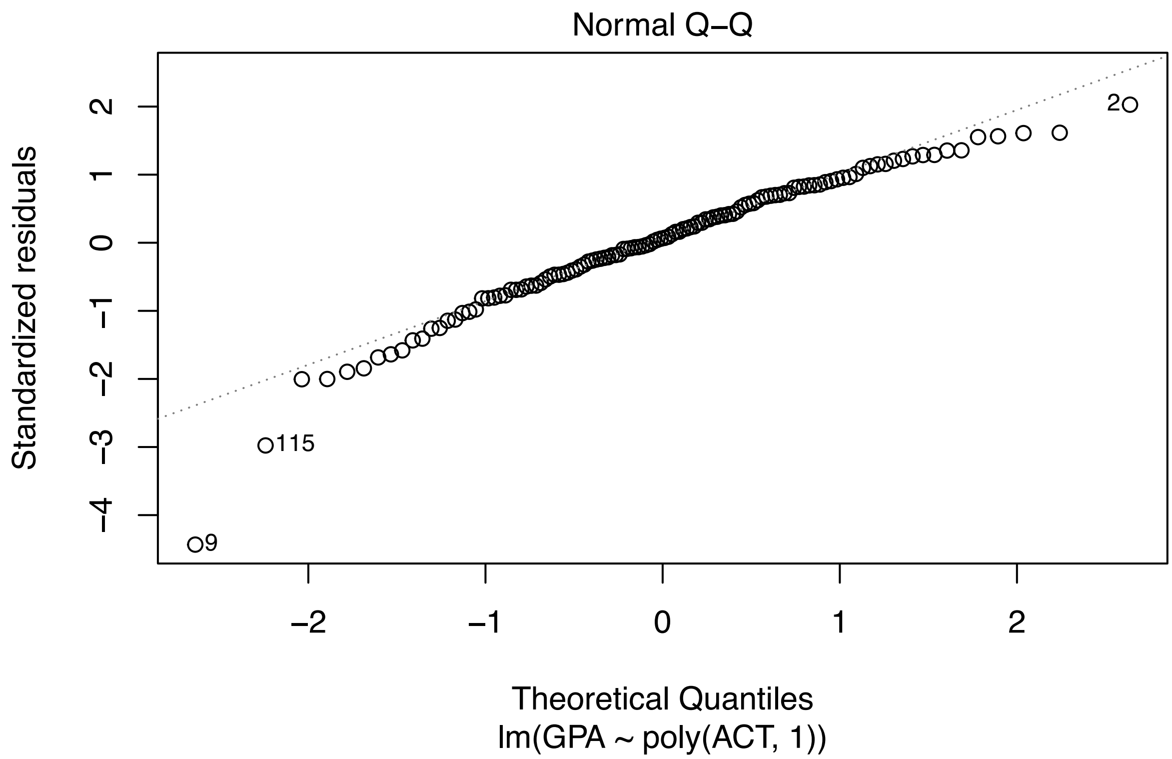
```
##                   left    right
## (Intercept)  2.9613517 3.186748
## poly(ACT, 1) 0.4930033 3.295319
```

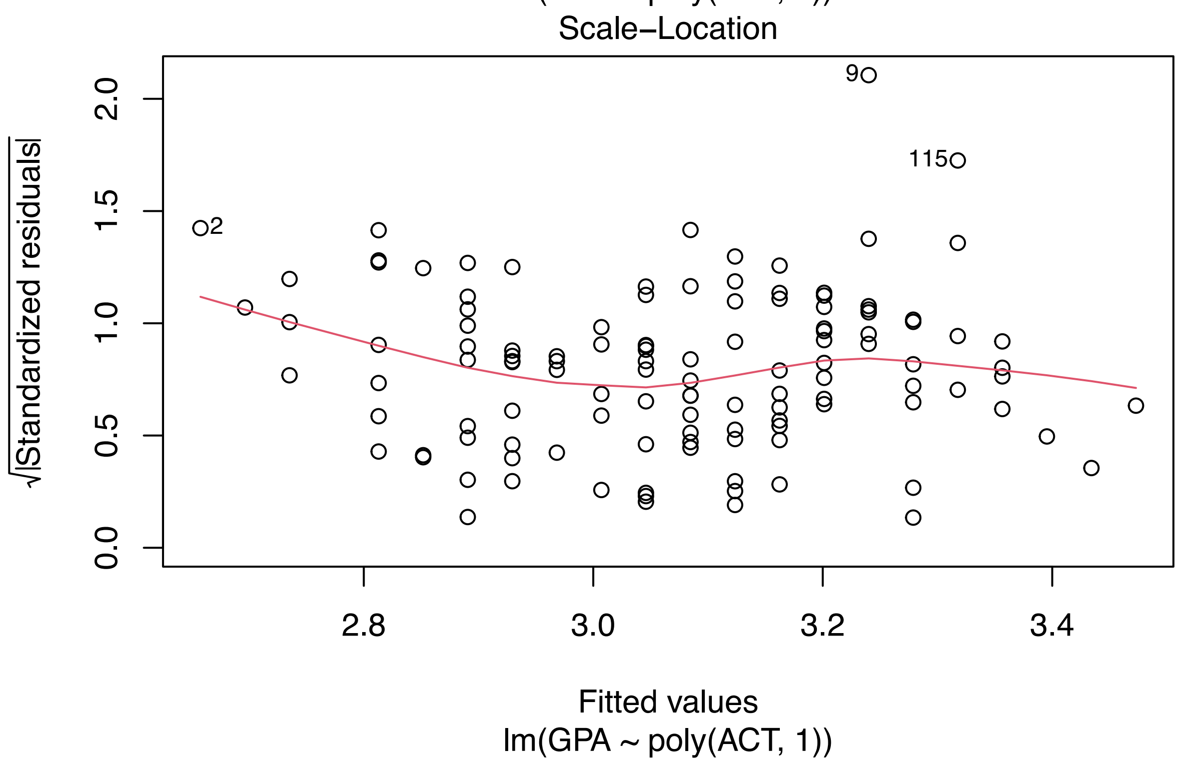
Answer: B0 = 3.07405, B1 = 1.89416 where B0 is the intercept and B1 is the slope of ACT. The 95% confidence interval for B0 = (2.9613517, 3.1867483), B1 = (0.4930033, 3.295319)

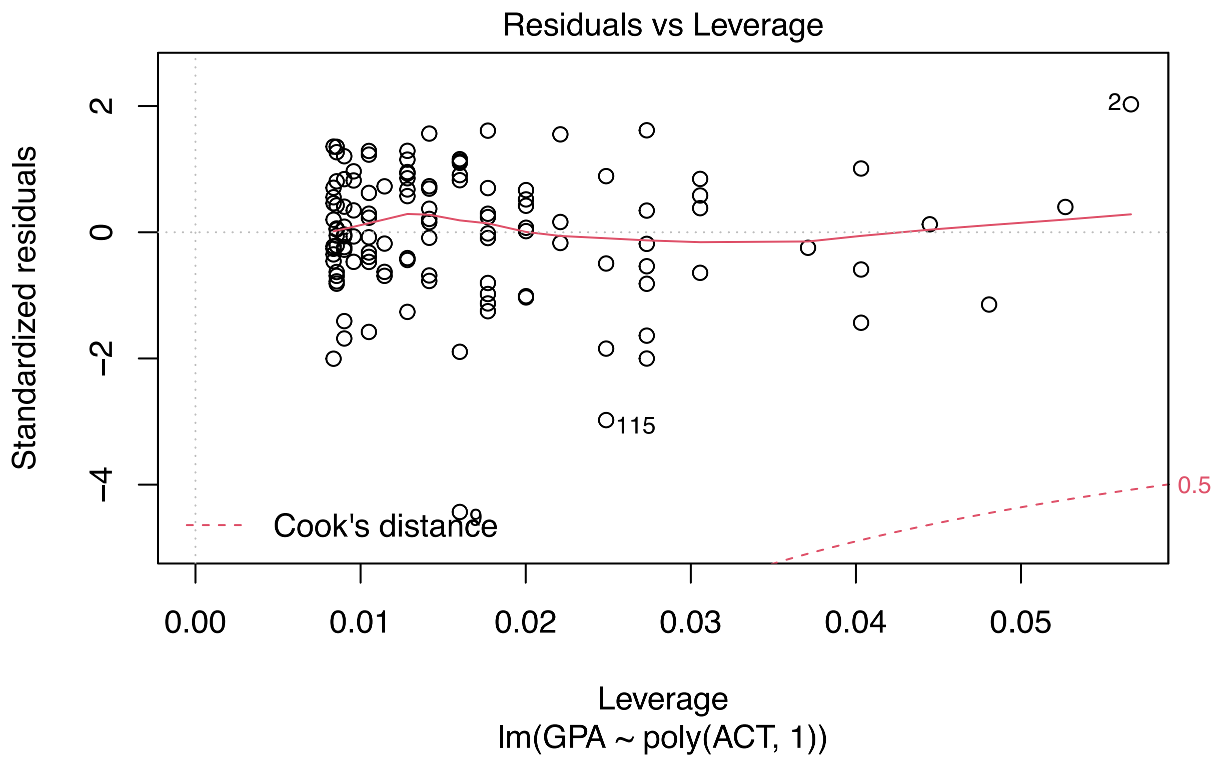
3b

```
plot(out_3a)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(GPA ~ poly(ACT, 1))
2
115
9
−4
−3
−2
−1
0
1
2


Scale−Location
√|Standardized residuals|
Fitted values
lm(GPA ~ poly(ACT, 1))
9
115
2
0.0
0.5
1.0
1.5
2.0
2.8
3.0
3.2
3.4



Answer: The residuals seem to be normally distributed, independent, have mean of 0 and constant spread. I would not adjust the model.

3c

```
predict(out_3a, interval = "prediction", level = 0.6, newdata = data.frame(ACT = mean(df_3$ACT)))
```

```
##       fit      lwr      upr
## 1 3.07405 2.545521 3.602579
```

Answer: The 60% prediction interval for the GPA of students with the mean ACT score is (2.545521, 3.602579).

Homework 3

Problem 4

$$\mathbb{E}[\hat{\beta}|X_1,\ldots,X_n] = \mathbb{E}\left[\frac{\sum_i Y_i U_i}{\sum_i U_i^2}|X_1,\ldots,X_n\right] \approx \frac{\mathbb{E}[\sum_i Y_i U_i|X_1,\ldots,X_n]}{\mathbb{E}[\sum_i U_i^2|X_1,\ldots,X_n]}$$

Now plug-in the values for Y_i and U_i:

$$\begin{aligned}
\frac{\mathbb{E}[\sum_i(\beta X_i+\epsilon_i)(X_i+\delta_i)|X_1,\ldots,X_n]}{\mathbb{E}[\sum_i(X_i+\delta_i)^2|X_1,\ldots,X_n]} &= \frac{\mathbb{E}[\sum_i \beta X_i^2+\beta\delta X_i+\epsilon_i X_i+\epsilon_i\delta_i|X_1,\ldots,X_n]}{\mathbb{E}[\sum_i X_i^2+\delta_i^2+2\delta_i X_i|X_1,\ldots,X_n]} \\
&= \frac{\sum_i \beta X_i^2+0+0+0}{\sum_i X_i^2+\sum_i \mathbb{E}[\delta_i^2|X_1,\ldots,X_n]} \\
&= \beta\frac{\sum_i X_i^2}{\sum_i X_i^2+n\tau^2} = \beta\frac{n\bar{X}^2}{n\bar{X}^2+n\tau^2} \\
&= \beta\frac{\bar{X}^2}{\bar{X}^2+\tau^2},
\end{aligned}$$

where we have used linearity of expectation, pairwise independence of ϵ, δ, X_i and the fact that $\epsilon_i \sim \mathcal{N}(0,\sigma^2)$ and $\delta_i \sim \mathcal{N}(0,\tau^2)$.

As n gets larger the bias is not affected. As $\tau \to \infty$ we have $\mathbb{E}[\hat{\beta}|X_1,\ldots,X_n] \to 0$.