HW11

107070008

```
auto <- read.table("auto-data.txt", header=FALSE, na.strings = "?")
names(auto) <- c("mpg", "cylinders", "displacement", "horsepower",
                 "weight", "acceleration", "model_year", "origin", "car_name")
cars <- auto
cars_log <- with(cars, data.frame(log(mpg), log(cylinders), log(displacement),
                                  log(horsepower), log(weight), log(acceleration),
                                  model_year, origin))
```

Question 1) Let's deal with nonlinearity first.Create a new dataset that log-transforms several variables from our original dataset (called cars in this case):

a. Run a new regression on the cars_log dataset, with mpg.log. dependent on all other variables

i. Which log-transformed factors have a significant effect on log.mpg. at 10% significance?

```
regr <- lm(log.mpg. ~ log.cylinders.+ log.displacement.+
             log.horsepower.+ log.weight.+
             log.acceleration.+ model_year+
             factor(origin), data = cars_log)
summary(regr)
```

```
##
## Call:
## lm(formula = log.mpg. ~ log.cylinders. + log.displacement. +
##     log.horsepower. + log.weight. + log.acceleration. + model_year +
##     factor(origin), data = cars_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.39727 -0.06880  0.00450  0.06356  0.38542
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        7.301938   0.361777  20.184  < 2e-16 ***
## log.cylinders.    -0.081915   0.061116  -1.340  0.18094
## log.displacement.  0.020387   0.058369   0.349  0.72707
## log.horsepower.   -0.284751   0.057945  -4.914 1.32e-06 ***
## log.weight.       -0.592955   0.085165  -6.962 1.46e-11 ***
```

```
## log.acceleration. -0.169673   0.059649  -2.845  0.00469 **
## model_year         0.030239   0.001771  17.078  < 2e-16 ***
## factor(origin)2    0.050717   0.020920   2.424  0.01580 *
## factor(origin)3    0.047215   0.020622   2.290  0.02259 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.113 on 383 degrees of freedom
##   (6 observations deleted due to missingness)
## Multiple R-squared:  0.8919, Adjusted R-squared:  0.8897
## F-statistic:   395 on 8 and 383 DF,  p-value: < 2.2e-16
```

none of them.

ii. Do some new factors now have effects on mpg, and why might this be?

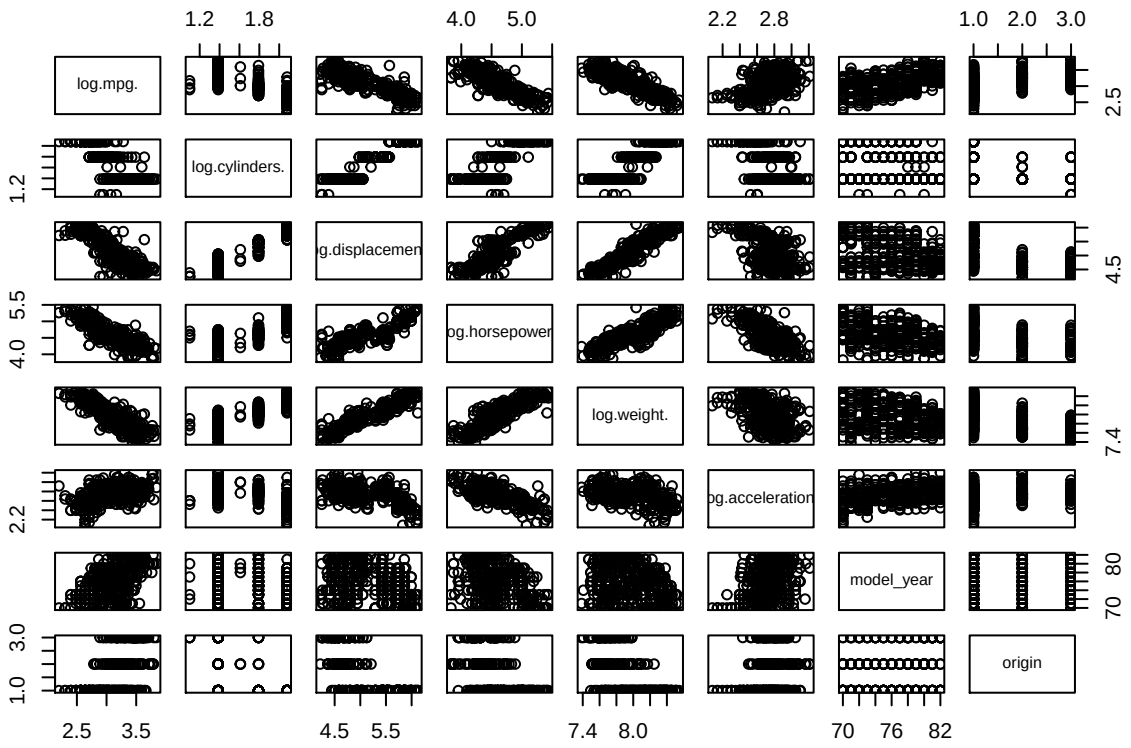
```
regr_o <- lm(mpg ~ cylinders+ displacement+
             horsepower+ weight+ acceleration+ model_year+
              factor(origin), data = cars)
summary(regr_o)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders + displacement + horsepower + weight +
##     acceleration + model_year + factor(origin), data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.0095 -2.0785 -0.0982  1.9856 13.3608
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -1.795e+01  4.677e+00  -3.839 0.000145 ***
## cylinders       -4.897e-01  3.212e-01  -1.524 0.128215
## displacement     2.398e-02  7.653e-03   3.133 0.001863 **
## horsepower      -1.818e-02  1.371e-02  -1.326 0.185488
## weight          -6.710e-03  6.551e-04 -10.243  < 2e-16 ***
## acceleration     7.910e-02  9.822e-02   0.805 0.421101
## model_year       7.770e-01  5.178e-02  15.005  < 2e-16 ***
## factor(origin)2  2.630e+00  5.664e-01   4.643 4.72e-06 ***
## factor(origin)3  2.853e+00  5.527e-01   5.162 3.93e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.307 on 383 degrees of freedom
##   (6 observations deleted due to missingness)
## Multiple R-squared:  0.8242, Adjusted R-squared:  0.8205
## F-statistic: 224.5 on 8 and 383 DF,  p-value: < 2.2e-16
```

horsepower, accerlation have significant effect on mpg, because people can see the hidden trend differences among datas with logistic regression.

iii. Which factors still have insignificant or opposite (from correlation) effects on mpg? Why might this be?

```
plot(cars_log)
```



log.cylinders. has insignificant effect on mpg, and log.displacement has an opposite effect on mpg, because its trend is opposite of the regression coeefficient.

b. Let's take a closer look at weight, because it seems to be a major explanation of mpg

i. Create a regression (call it regr_wt) of mpg over weight from the original cars dataset

```
regr_wt <- lm(mpg ~ weight, data = cars, na.action=na.exclude)
```

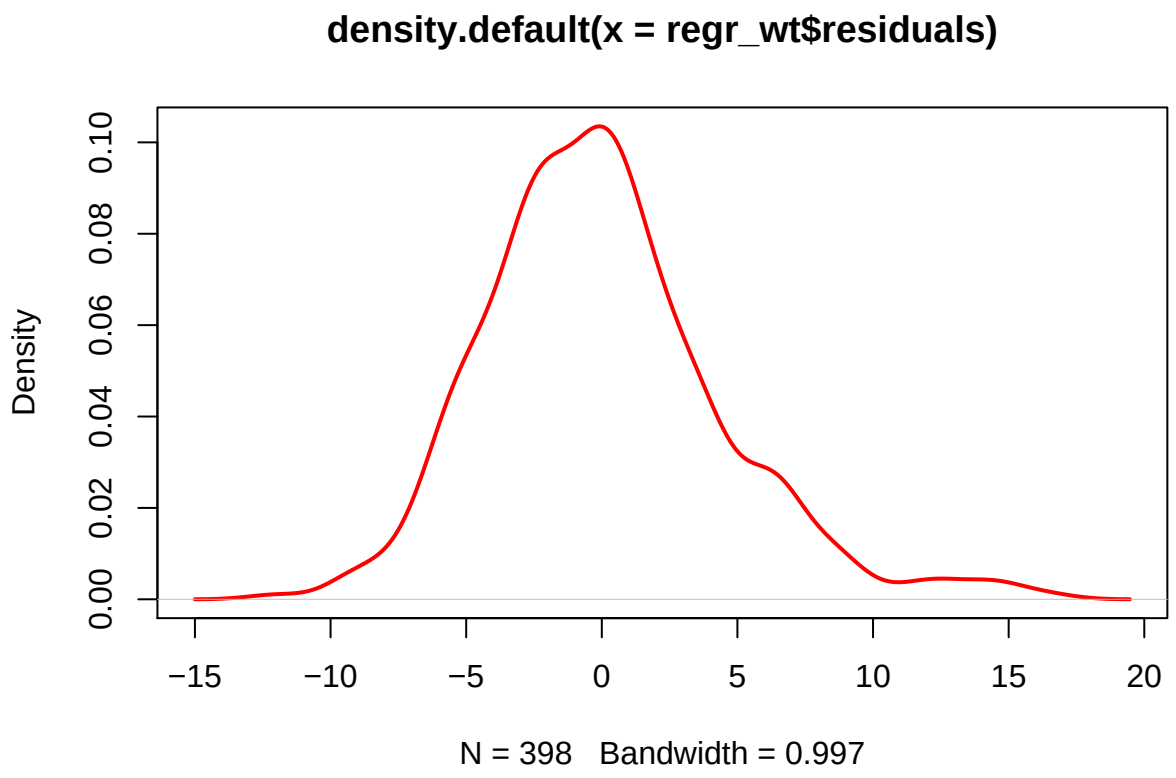
ii. Create a regression (call it regr_wt_log) of log.mpg. on log.weight. from cars_log

```
regr_wt_log <- lm(log.mpg. ~ log.weight., data = cars_log, na.action=na.exclude)
```

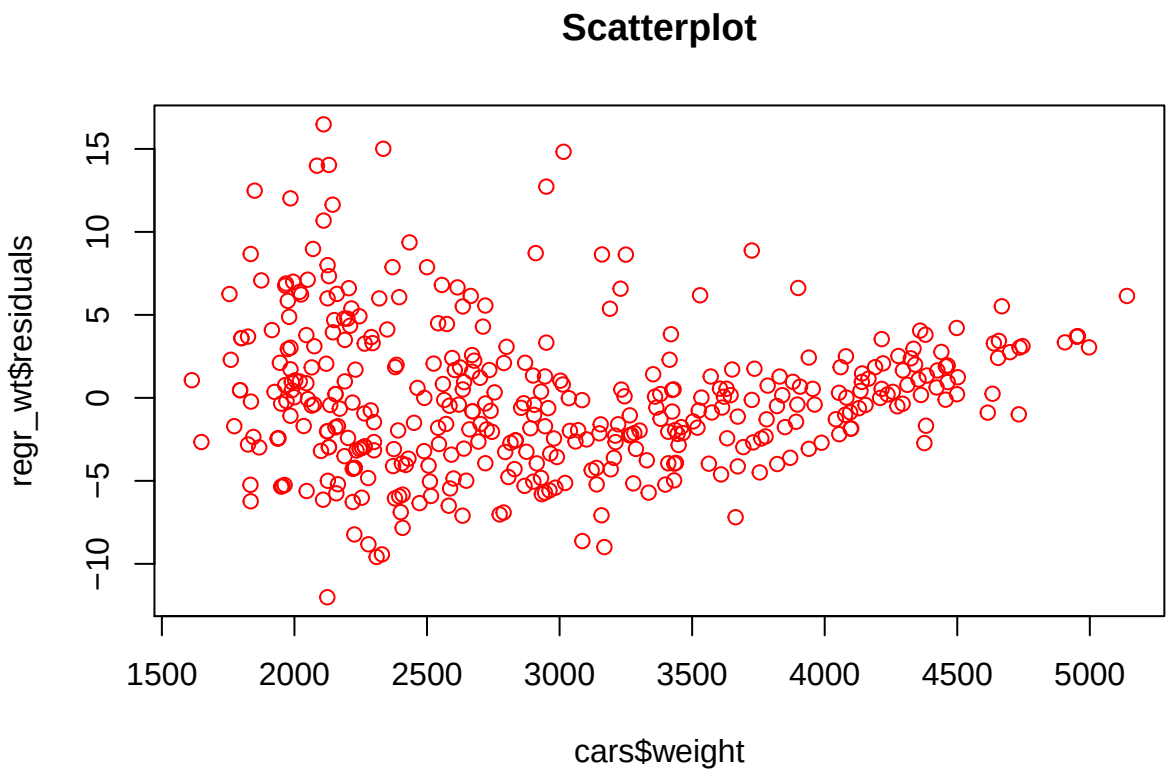
iii. Visualize the residuals of both regression models (raw and log-transformed):

1. density plots of residuals
2. scatterplot of log.weight. vs. residuals

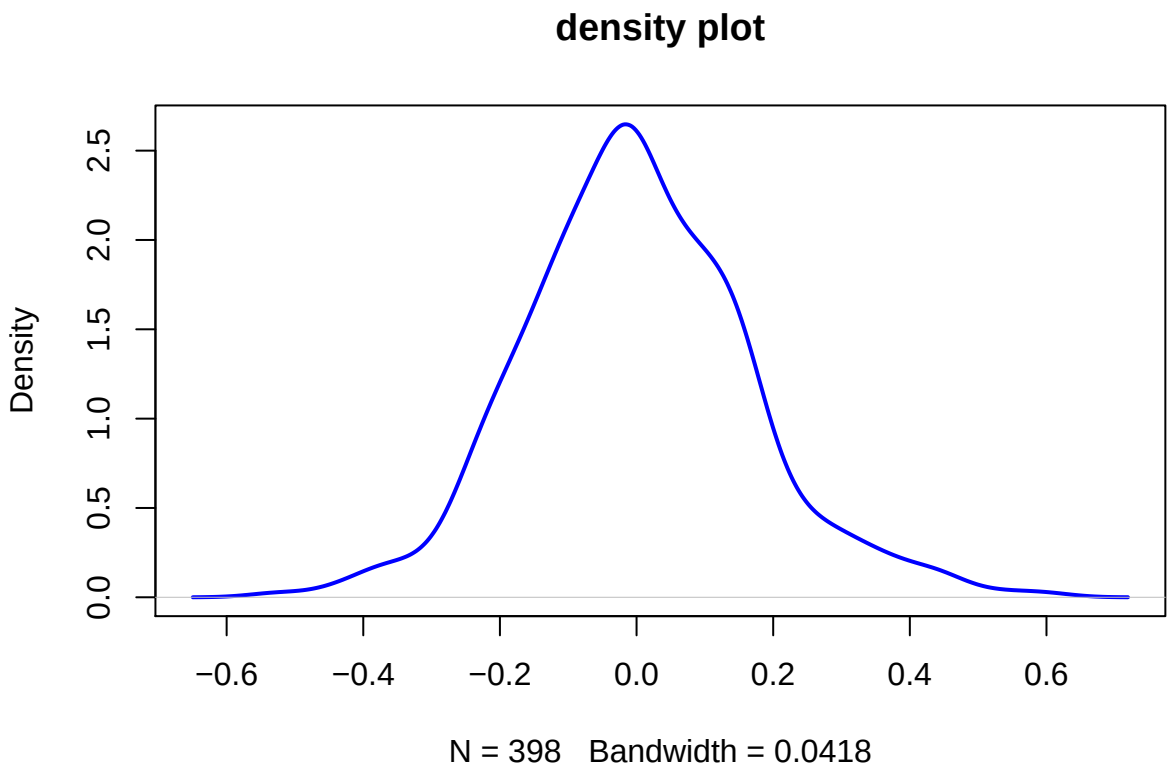
```
plot(density(regr_wt$residuals), lwd = 2, col = "red")
```



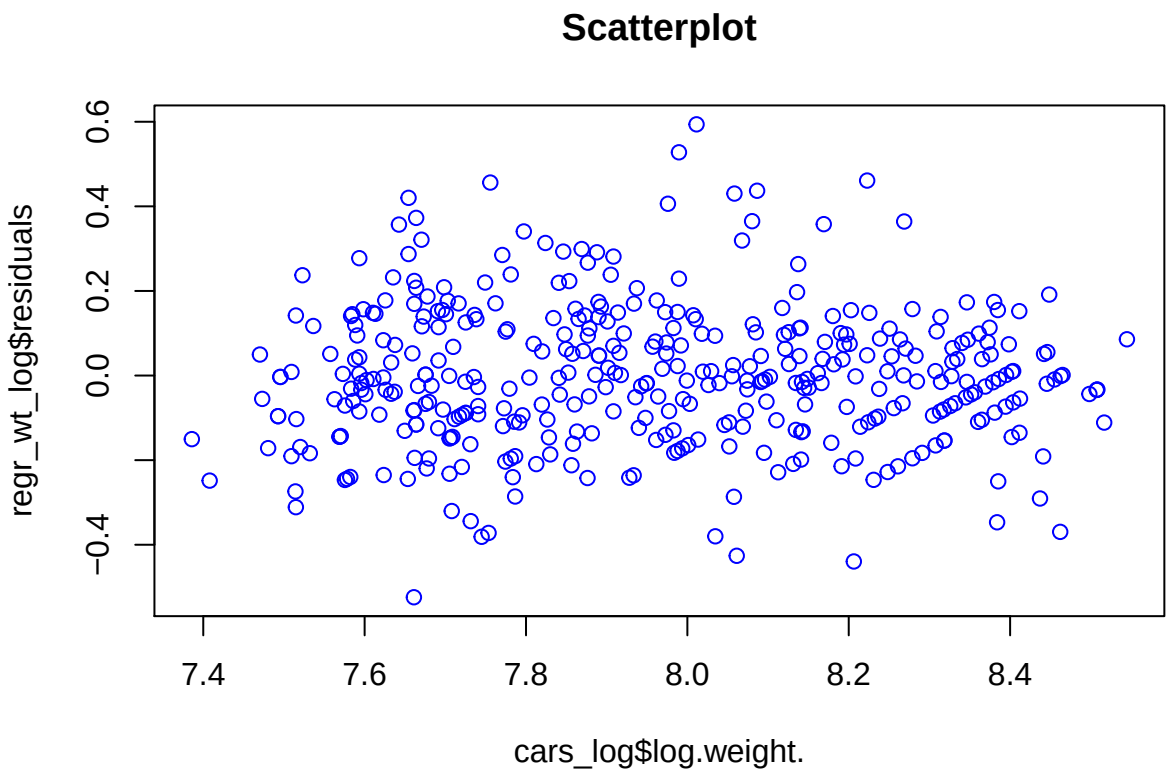

```
plot(cars$weight, regr_wt$residuals, main="Scatterplot", col = "red")
```




```
plot(density(regr_wt_log$residuals), lwd = 2, col = "blue", main = "density plot")
```



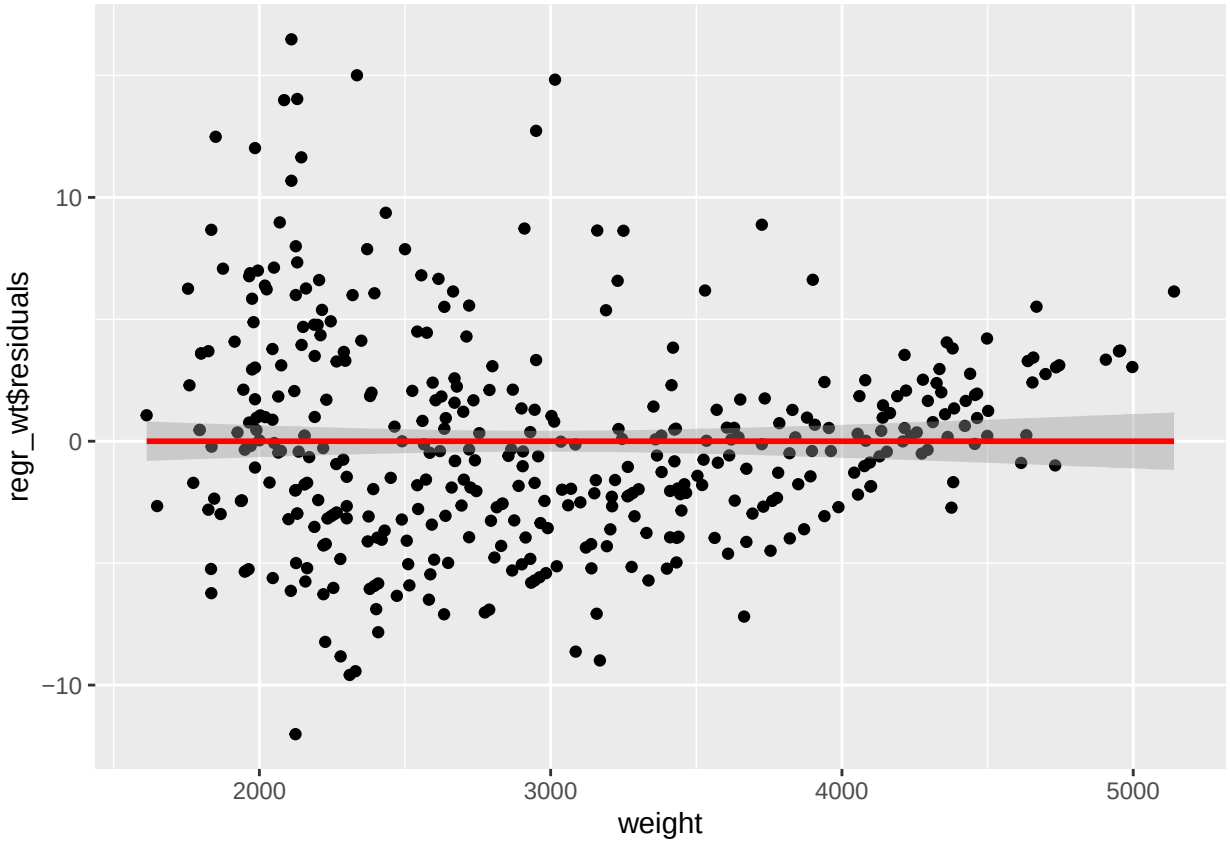

```
plot(cars_log$log.weight., regr_wt_log$residuals, main="Scatterplot", col = "blue")
```



iv. Which regression produces better distributed residuals for the assumptions of regression?

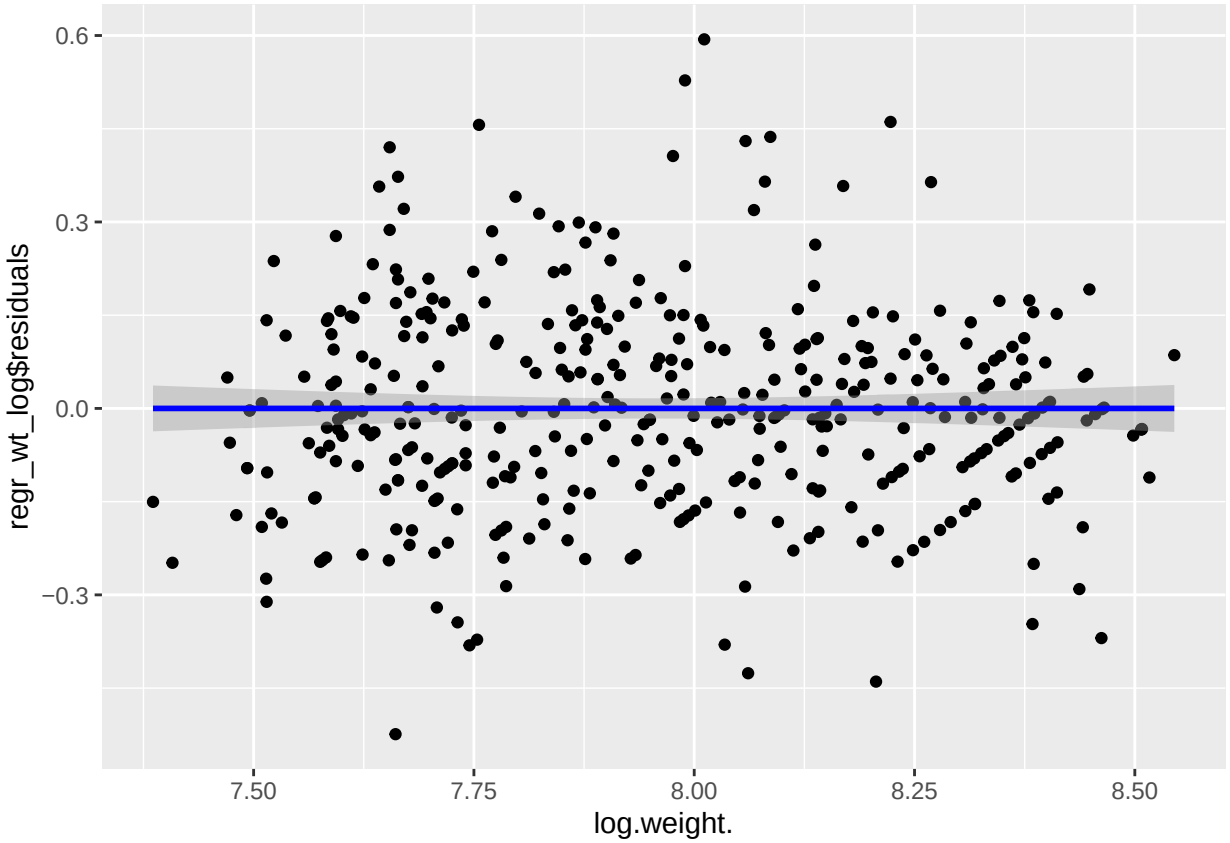
```
library(ggplot2)
ggplot(cars, aes(x = weight, y = regr_wt$residuals)) +
  geom_point() +
  stat_smooth(method = "lm", col = "red")
```

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




```
ggplot(cars_log, aes(x = log.weight., y = regr_wt_log$residuals)) +
  geom_point() +
  stat_smooth(method = "lm", col = "blue")
```

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



logistic regression has a better residuals distribution, because it is linear, constant vaiance, and independent.On the other hand, regression of raw data doesn't have constant variance properties.

v. How would you interpret the slope of log.weight. vs log.mpg. in simple words?

```
summary(regr_wt_log)
```

```
##
## Call:
## lm(formula = log.mpg. ~ log.weight., data = cars_log, na.action = na.exclude)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.52408 -0.10441 -0.00805  0.10165  0.59384
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  11.5219     0.2349   49.06   <2e-16 ***
## log.weight.  -1.0583     0.0295  -35.87   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.165 on 396 degrees of freedom
## Multiple R-squared:  0.7647, Adjusted R-squared:  0.7641
## F-statistic:  1287 on 1 and 396 DF,  p-value: < 2.2e-16
```

The number of slope in regression is -1.0583. It means that if log.weight. increase a unit, then log.mpg will increase 11.529+(-1.0583).

c. Let's examine the 95% confidence interval of the slope of log.weight. vs. log.mpg.

i. Create a bootstrapped confidence interval

```
boot_regr <- function(model, dataset) {
  boot_index <- sample(1: nrow(dataset), replace = TRUE)
  data_boot <- dataset[boot_index,]
  regr_boot <- lm(model, data = data_boot)
  regr_boot$coefficients
}
coeffs <- replicate(2500, boot_regr(log.mpg.~log.weight., cars_log))
quantile(coeffs["log.weight.",], c(0.025, 0.975))
```

```
##      2.5%     97.5%
## -1.112324 -1.006791
```

ii. Verify your results with a confidence interval using traditional statistics (i.e., estimate of coefficient and its standard error from lm() results)

```
wt_regr_log <- lm(log.mpg.~log.weight., cars_log)
confint(wt_regr_log)
```

```
##                 2.5 %    97.5 %
## (Intercept) 11.060154 11.983659
## log.weight. -1.116264 -1.000272
```

Question 2)Let's tackle multicollinearity next. Consider the regression model:

a. Using regression and R2, compute the VIF of log.weight. using the approach shown in class

```
regr_log <- lm(log.mpg. ~ log.cylinders. + log.displacement. + log.horsepower. +
                log.weight. + log.acceleration. + model_year +
                factor(origin), data=cars_log)

weight_regr <- lm(log.weight. ~ log.displacement. + log.horsepower. +
                    log.acceleration. + model_year + factor(origin),
                  data=cars_log, na.action = na.exclude)
r2_weight <- summary(weight_regr)$r.squared
r2_weight
```

```
## [1] 0.942433
```

```
vif_weight <- 1/(1- r2_weight)
vif_weight
```

```
## [1] 17.37105
```

b. Let's try a procedure called Stepwise VIF Selection to remove highly collinear predictors. Start by Installing the 'car' package in RStudio – it has a function called vif()

i. Use vif(regr_log) to compute VIF of the all the independent variables

```
library(car)
```

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

```
vif(regr_log)
```

```
##                       GVIF Df GVIF^(1/(2*Df))
## log.cylinders.    10.456738  1        3.233688
## log.displacement. 29.625732  1        5.442952
## log.horsepower.   12.132057  1        3.483110
## log.weight.       17.575117  1        4.192269
## log.acceleration.  3.570357  1        1.889539
## model_year         1.303738  1        1.141814
## factor(origin)     2.656795  2        1.276702
```

ii. Eliminate from your model the single independent variable with the largest VIF score that is also greater than 5

```
regr_log1 <- lm(log.mpg. ~ log.cylinders. + log.horsepower. +
                log.weight. + log.acceleration. + model_year +
                factor(origin), data=cars_log)
vif(regr_log1)
```

```
##                       GVIF Df GVIF^(1/(2*Df))
## log.cylinders.     5.433107  1        2.330903
## log.horsepower.   12.114475  1        3.480585
## log.weight.       11.239741  1        3.352572
## log.acceleration.  3.327967  1        1.824272
## model_year         1.291741  1        1.136548
## factor(origin)     1.897608  2        1.173685
```

iii. Repeat steps (i) and (ii) until no more independent variables have VIF scores above 5

```
regr_log2 <- lm(log.mpg. ~ log.cylinders. + log.weight. +
                 log.acceleration. + model_year +
                 factor(origin), data=cars_log)
vif(regr_log2)
```

```
##                       GVIF Df GVIF^(1/(2*Df))
## log.cylinders.    5.321090  1        2.306749
## log.weight.       4.788498  1        2.188264
## log.acceleration. 1.400111  1        1.183263
## model_year        1.201815  1        1.096273
## factor(origin)    1.792784  2        1.157130
```

```
regr_log3 <- lm(log.mpg. ~ log.weight. +
                  log.acceleration. + model_year +
                  factor(origin), data=cars_log)
vif(regr_log3)
```

```
##                       GVIF Df GVIF^(1/(2*Df))
## log.weight.       1.926377  1        1.387940
## log.acceleration. 1.303005  1        1.141493
## model_year        1.167241  1        1.080389
## factor(origin)    1.692320  2        1.140567
```

iv. Report the final regression model and its summary statistics

```
regr_log3
```

```
##
## Call:
## lm(formula = log.mpg. ~ log.weight. + log.acceleration. + model_year +
##     factor(origin), data = cars_log)
##
## Coefficients:
##       (Intercept)        log.weight.  log.acceleration.         model_year
##           7.43116           -0.87661            0.05151            0.03273
##   factor(origin)2    factor(origin)3
##           0.05799            0.03233
```

```
summary(regr_log3)
```

```
##
## Call:
## lm(formula = log.mpg. ~ log.weight. + log.acceleration. + model_year +
##     factor(origin), data = cars_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.38275 -0.07032  0.00491  0.06470  0.39913
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        7.431155   0.312248  23.799  < 2e-16 ***
## log.weight.       -0.876608   0.028697 -30.547  < 2e-16 ***
## log.acceleration.  0.051508   0.036652   1.405  0.16072
## model_year         0.032734   0.001696  19.306  < 2e-16 ***
## factor(origin)2    0.057991   0.017885   3.242  0.00129 **
```

```
## factor(origin)3    0.032333   0.018279   1.769  0.07770 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1156 on 392 degrees of freedom
## Multiple R-squared:  0.8856, Adjusted R-squared:  0.8841
## F-statistic: 606.8 on 5 and 392 DF,  p-value: < 2.2e-16
```

c.Using stepwise VIF selection, have we lost any variables that were previously significant? If so, how much did we hurt our explanation by dropping those variables? (hint: look at model fit)

Yes, We sloghtly hurt the salience of the regression model, because the previous model's R squaerd is 0.8919, and the new model's R squaerd is 0.8856.

d. From only the formula for VIF, try deducing/deriving the following:

i. If an independent variable has no correlation with other independent variables, what would its VIF score be? For only formula, if an independent variable has no correlation with other independent variables, the value of R-squared will close to 0.Therefore, the VIF will be 1/(1-0) = 1.(closing to 1)

ii. Given a regression with only two independent variables (X1 and X2), how correlated would X1 and X2 have to be, to get VIF scores of 5 or higher? To get VIF scores of 10 or higher? 5 < 1/(1-R^2), R = +- 0.8944272, and they are highly correlated. They will have a serious corlinearity problem.

Question 3) Might the relationship of weight on mpg be different for cars from different origins? Let's try visualizing this. First, plot all the weights, using different colors and symbols for the three origins:

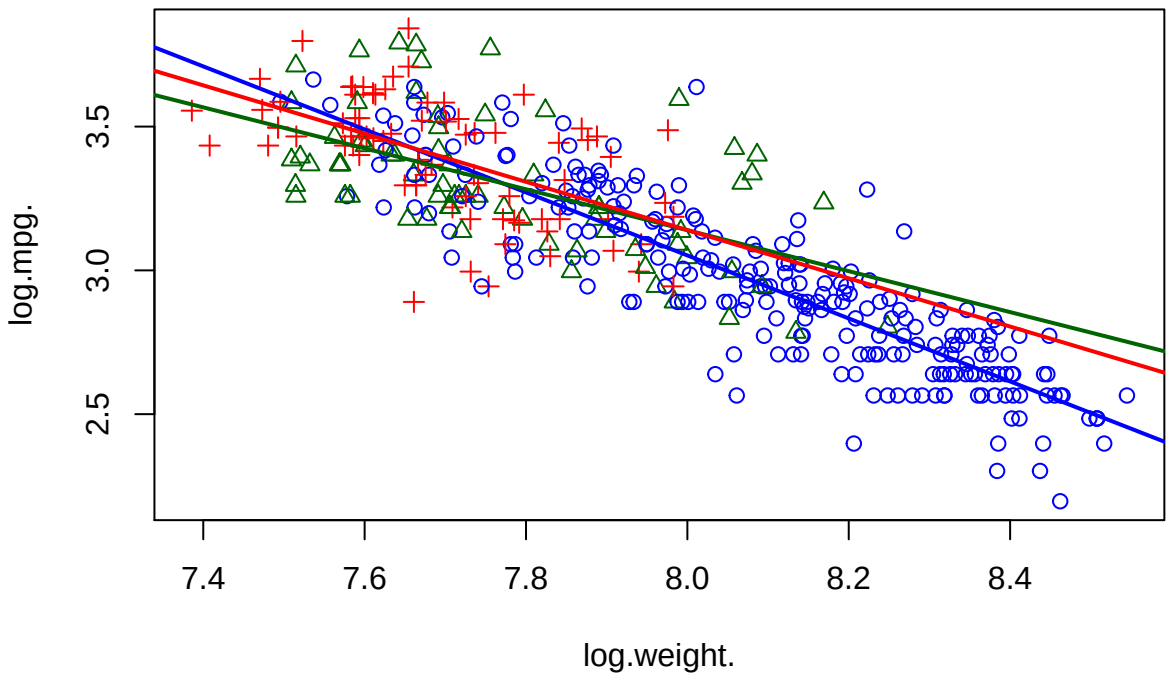
a. Let's add three separate regression lines on the scatterplot, one for each of the origins: Here's one for the US to get you started:

```
origin_colors = c("blue", "darkgreen", "red")
with(cars_log, plot(log.weight., log.mpg., pch=origin, col=origin_colors[origin]))

cars_us <- subset(cars_log, origin==1)
wt_regr_us <- lm(log.mpg. ~ log.weight., data=cars_us)
abline(wt_regr_us, col=origin_colors[1], lwd=2)

cars_us <- subset(cars_log, origin==2)
wt_regr_us <- lm(log.mpg. ~ log.weight., data=cars_us)
abline(wt_regr_us, col=origin_colors[2], lwd=2)

cars_us <- subset(cars_log, origin==3)
wt_regr_us <- lm(log.mpg. ~ log.weight., data=cars_us)
abline(wt_regr_us, col=origin_colors[3], lwd=2)
```



b.[not graded] Do cars from different origins appear to have different weight vs. mpg relationships?

```
no, they are the same.
```