HW11

Black and Red texts are answers !

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

```
auto <- read.table("auto-data.txt", header=FALSE, na.strings = "?", str
ingsAsFactors = F)
names(auto) <- c("mpg", "cylinders", "displacement", "horsepower", "wei
ght",
"acceleration", "model_year", "origin", "car_name")
cars_log <- with(auto, data.frame(log(mpg), log(cylinders), log(displac
ement), log(horsepower), log(weight), log(acceleration), model_year, or
igin))
```

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

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

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

log.horsepower., log.weight., log.acceleration., model_year, origin

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

Horsepower, accerleration become significant factor after log-transforming. Before that, they are non-linear correlated to mpg, which would not meet the assumption of regression.

iii. Which factors still have insignificant or opposite effect on mpg? Why might this be?

Cylinders and displacement still have no significant effect on mpg. I think they shared high multicollinearity with other variable like **weight** and **horsepower** thus become insignificant.

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 on weight from the original cars dataset

```
regr_wt <- lm(mpg ~ weight, data = auto)
summary(regr_wt)
```

```
##
## Call:
## lm(formula = mpg ~ weight, data = auto)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -12.012  -2.801  -0.351   2.114  16.480
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 46.3173644  0.7952452   58.24   <2e-16 ***
## weight      -0.0076766  0.0002575  -29.81   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.345 on 396 degrees of freedom
## Multiple R-squared:  0.6918, Adjusted R-squared:  0.691
## F-statistic: 888.9 on 1 and 396 DF,  p-value: < 2.2e-16
```

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

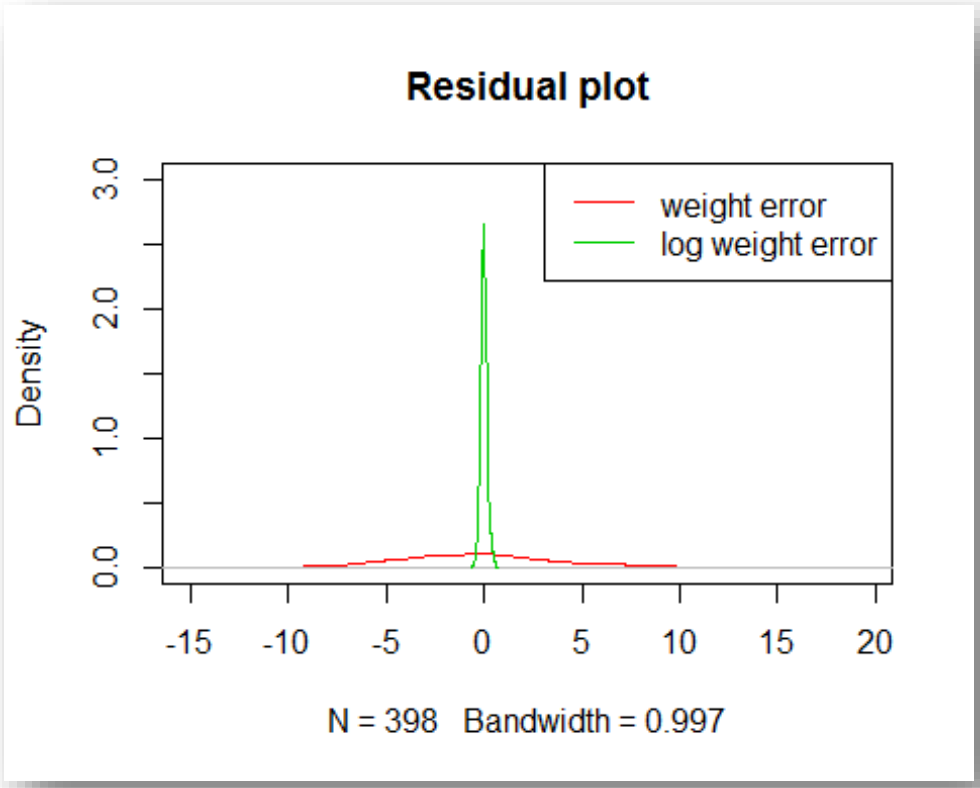
```
regr_wt_log <- lm(log.mpg. ~log.weight., data = cars_log)
summary(regr_wt_log)
```

```
##
## Call:
## lm(formula = log.mpg. ~ log.weight., data = cars_log)
##
## 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
```

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

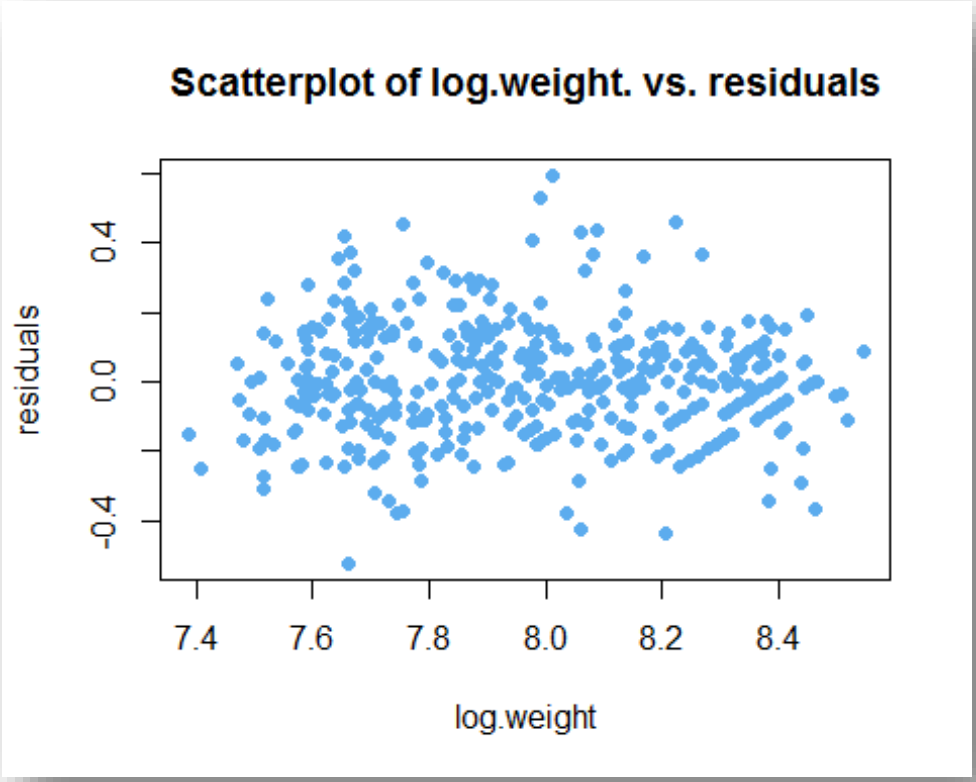
1. density plots of residuals

```
plot(density(regr_wt$residuals), col = 2,ylim = c(0,3),main = "Residual
 plot")
lines(density(regr_wt_log$residuals),col = 3)
legend("topright",c("weight error","log weight error"),lty=c(1,1),col =
 c(2,3))
```



2. scatterplot of log.weight. vs. residuals

```
plot(cars_log$log.weight.,regr_wt_log$residuals,col="steelblue2",pch =
16,
     xlab="log.weight", ylab = "residuals", main = "Scatterplot of log.
weight. vs. residuals")
```



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

Assumptions of regression requires error terms to be random and normally distributed with zero mean. From the above two figures, we can find out that the regression using **log.weight.** as independent variable perform well. The errors lie extremely centralized with randomness.

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

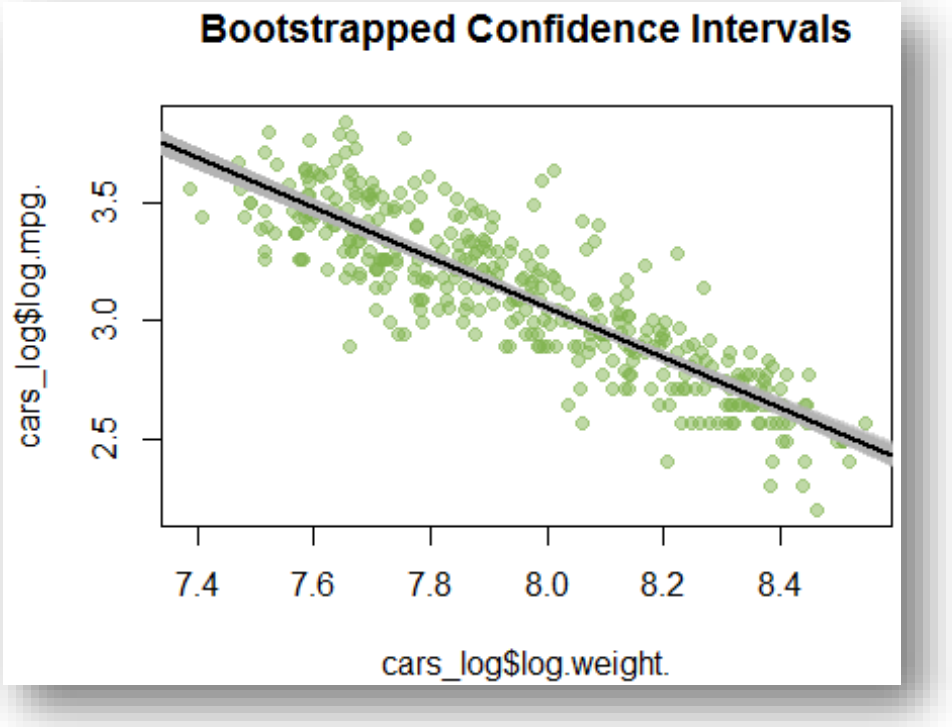
1% change in weight leads to -1.0583% change in mpg.

c). What is the 95% confidence interval of the slope of log.weight. vs. log.mpg.?

i. Create a bootstrapped confidence interval

```
plot(cars_log$log.weight.,cars_log$log.mpg., col=rgb(0.5, 0.7, 0.3, 0.
5),
     pch=19, main = "Bootstrapped Confidence Intervals")
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)
  abline(regr_boot, lwd=1, col=rgb(0.7, 0.7, 0.7, 0.5))
  return(regr_boot$coefficients)
```

```
}
coeffs<-replicate(300, boot_regr(log.mpg. ~ log.weight., cars_log))
abline(a=mean(coeffs["(Intercept)",]), b=mean(coeffs["log.weight.",]),
lwd=2)
```

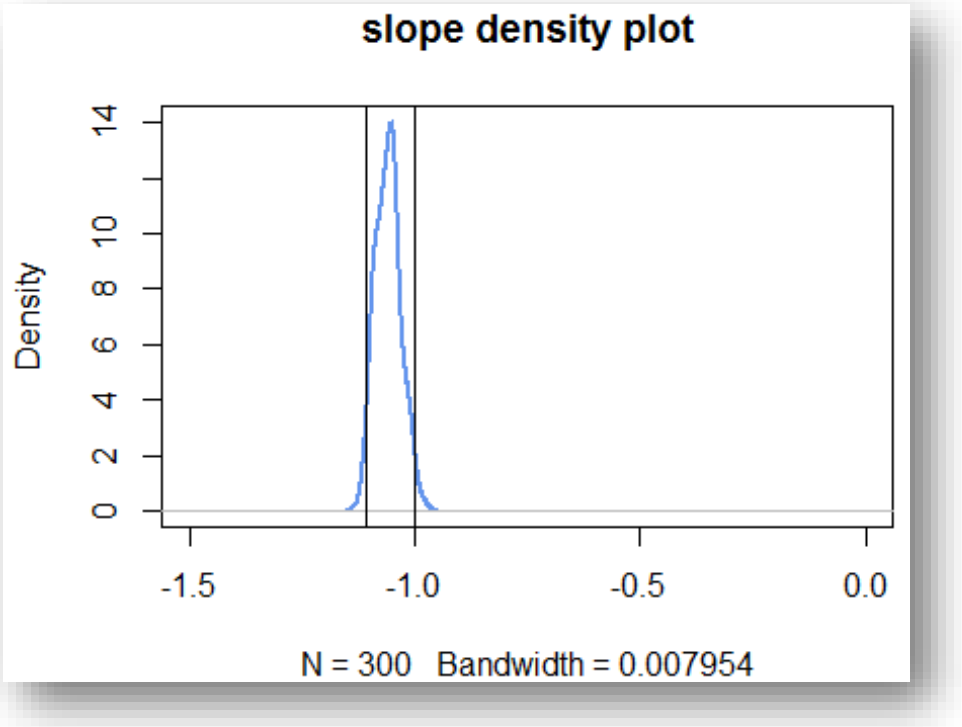


ii. Verify your results with a confidence interval using traditional statistics

```
quantile(coeffs["log.weight.",], c(0.025,0.975))

##      2.5%     97.5%
## -1.107093 -1.001534

plot(density(coeffs["log.weight.",]),xlim = c(-1.5,0),col="cornflowerbl
ue",lwd=2,
     main = "slope density plot")
abline(v=quantile(coeffs["log.weight.",], c(0.025,0.975)))
```




```
print(confint(regr_wt_log)) # another function to show the coefficient.

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

After finding out the 95% confidence interval of the slope, we can claim that the coefficient is significant since our result is -1.0583, which lies between 95% confidence interval.

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

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

a). Using regression and R2, calculate the VIF of log.weight. as demonstrated in class

```
log_weight <- lm(log.weight. ~ log.cylinders. + log.displacement.
                 + log.horsepower. +log.acceleration. + model_year +
                   factor(origin),  data=cars_log)
# R_square
r2_weight <- summary(log_weight)$r.squared
r2_weight

## [1] 0.9431014
```

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

## [1] 17.57512
```

log.weight. shares more than half its variance with other independent variables.

b). Let's try a procedure called Stepwise VIF Selection to remove highly collinear variables. 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)
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. Remove the independent variable with the largest VIF score greater than 5 from the model

```
source("vif_func.R")
vif_func(cars_log[,-1],trace = T, thresh = 5)

## Loading required package: fmsb

## Warning: package 'fmsb' was built under R version 3.3.3

##  var               vif
##  log.cylinders.    10.3992892027075
##  log.displacement. 26.7183497033435
##  log.horsepower.   11.9223498240788
##  log.weight.       16.1211180925851
##  log.acceleration. 3.50651971018354
##  model_year        1.23495195952232
##  origin            2.10227629368548
##
## removed:  log.displacement. 26.71835
##
##  var               vif
##  log.cylinders.    5.19596792899155
##  log.horsepower.   11.9222190706628
##  log.weight.       11.0466363915695
##  log.acceleration. 3.31722431686995
##  model_year        1.23412127724178
```

```
##  origin            1.66493628842562
##
## removed:  log.horsepower. 11.92222
##
##  var               vif
##  log.cylinders.    5.11438728283876
##  log.weight.       4.77886733786272
##  log.acceleration. 1.39592029539436
##  model_year        1.16606140374477
##  origin            1.6106880471501
##
## removed:  log.cylinders. 5.114387

## [1] "log.weight."       "log.acceleration." "model_year"
## [4] "origin"
```

I removed **displacement, horsepower and cylinders**.

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

From the above, after removing the variables that have large VIF scores, **log.weight., log.accerleration., model_year** and **origin** were remained.

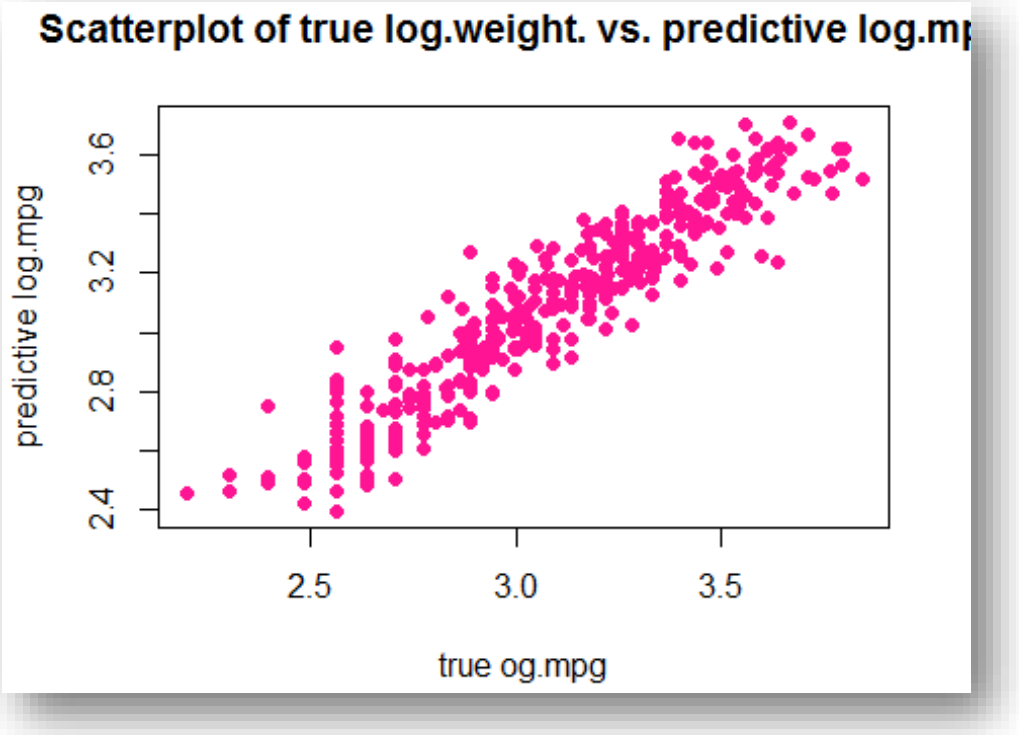
iv. Report the final regression model and its summary statistics

```
regr_final <- lm(log.mpg. ~ log.weight. + log.acceleration. + model_yea
r + factor(origin),
                 data  = cars_log)
summary(regr_final)

##
## 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

plot(cars_log$log.mpg.,regr_final$fitted.values,col="deeppink1",pch = 16,
     xlab="true og.mpg", ylab = "predictive log.mpg", main = "Scatterplot of true log.weight. vs. predictive log.mpg")
```



c). Using stepwise VIF selection, have we lost any variables that were previously significant? If so, was it reasonable to drop those variables? (hint: look at model fit)

Stopwise VIF selection drops variable **horesepower**, which was previosly significant, since it has high VIF value. Comparing to the adjust R-square of full and selected version regression, I think it's reasonable because they are almost the same. In selected version, we used even much fewer variables!

d). General questions on VIF:

i. If an independent variable has no correlation with other independent variables, what would its VIF score be?

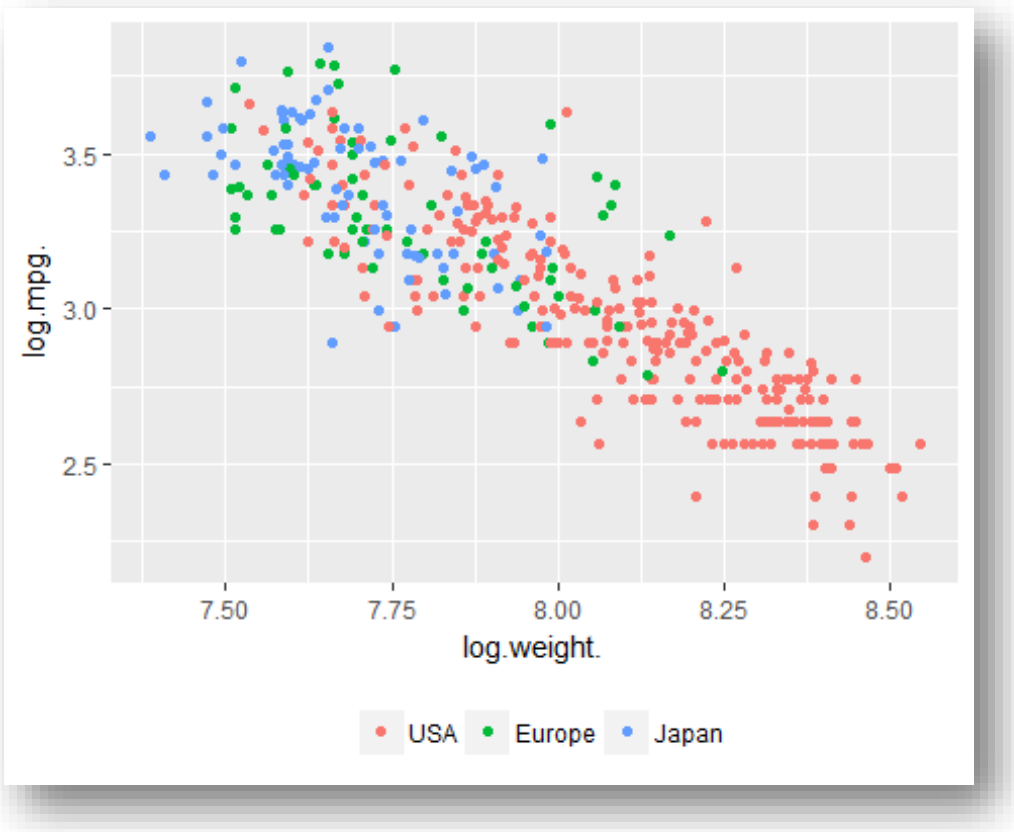
If no correlation, R-squared would be zero. Due to the formula of VIF, it turns out that VIF become 1 (VIF=1) .

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?

correlation between them above **0.894** could make VIF scores of 5 or higher.
correlation between them above **0.949** could make VIF scores of 10 or higher.

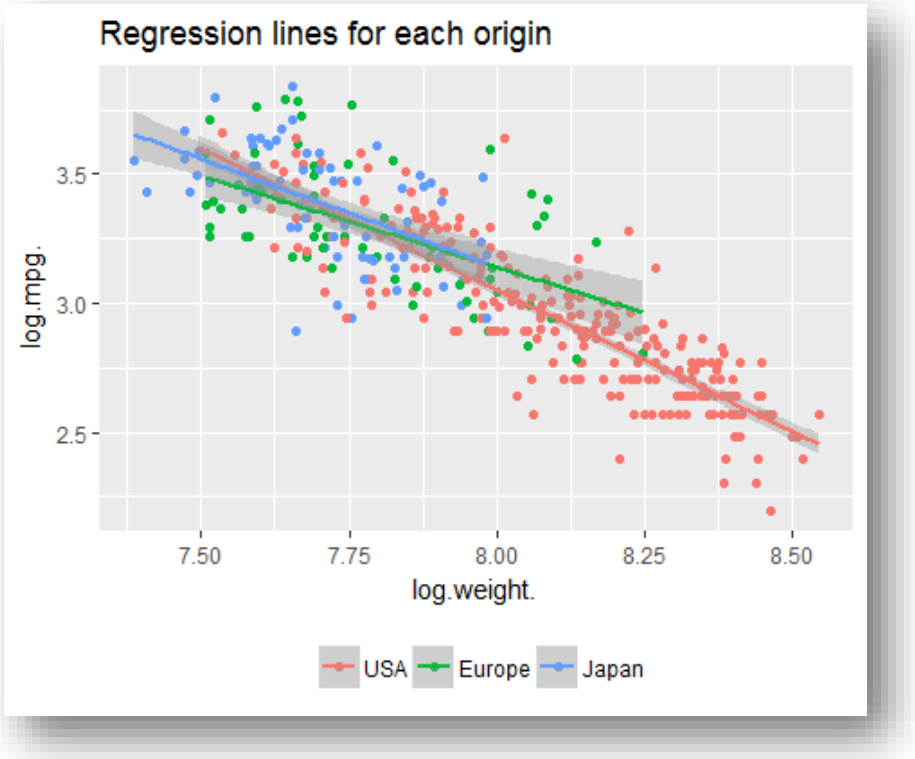
Question 3) Might the relationship of weight on mpg be different for cars from different origins? Let's try visualizing this. First, plot all the weights, using different colors and symbols for the three origins:

```
library(ggplot2)
cars_log$country <- factor(cars_log$origin,labels=c("USA","Europe","Japan"))
ggplot(cars_log,aes(x= log.weight., y = log.mpg.,col = factor(country)))
+
  geom_point()+
  theme(legend.position="bottom", legend.direction="horizontal") +
  theme(legend.title=element_blank())
```



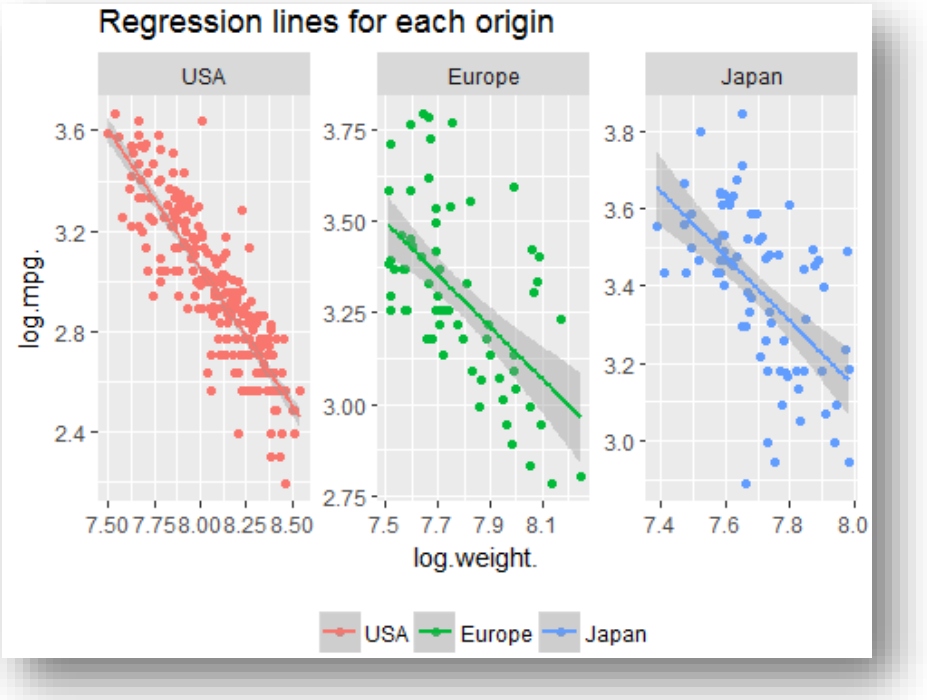
a). Let's add three separate regression lines on the scatterplot, one for each of the origins:

```
three_regr <- ggplot(cars_log,aes(x= log.weight., y = log.mpg.,col = fa
ctor(country)))+
  ggtitle("Regression lines for each origin")+
  geom_point()+
  geom_smooth(aes(log.weight.,log.mpg.,color=factor(country)),method=lm,
se=TRUE)+
  theme(legend.position="bottom", legend.direction="horizontal") +
  theme(legend.title=element_blank())
three_regr
```



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

```
three_regr+facet_wrap(~factor(country),scales = "free")
```



Here I separated three countries, and it's obvious that they have different amounts of data points. Both Europe and Japan show higher standard deviation of regression lines because they have fewer data. In terms of relationships between log.weight. and log.mpg., I don't think there's significant difference between each origin. On another perspective, number of data points might affect the comparison.