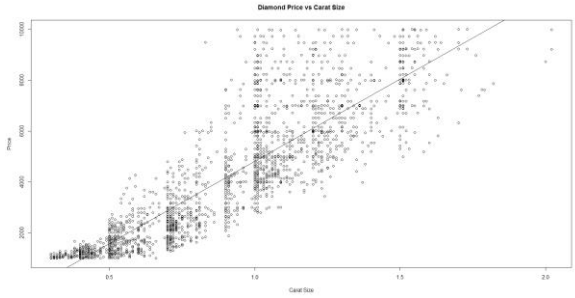
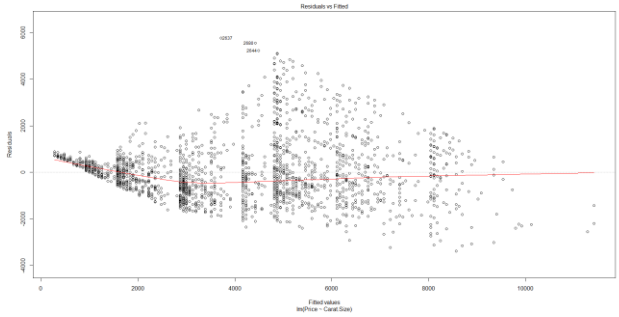
HW 5 Key

Name: Will Schievelbein

Q1: One can observe evidence of heteroscedasticity in the model from the ‘fanning’ out of the data mainly in the middle of predicted price. Additionally, there is nonlinearity at the low predicted price diamonds as the observed price of diamonds curves up from the prediction line for low value diamonds. Finally, there is an issue that this data is likely censored as there is a sharp cut-off at $10000 and exactly $1000 which is likely causing the nonlinearity problem.

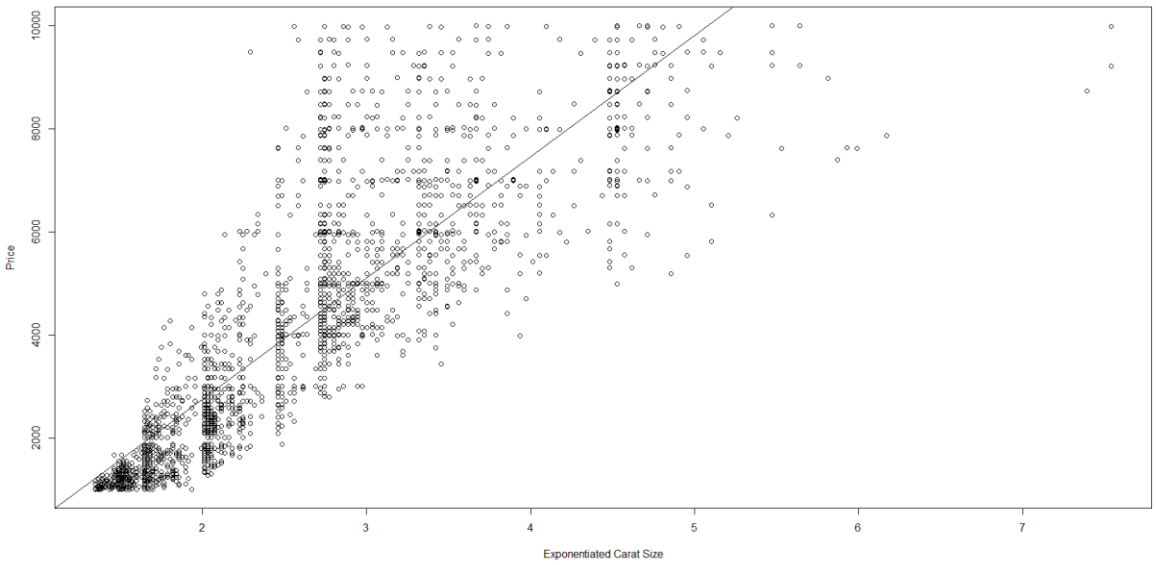





```
model_carat <- lm(Price ~ Carat.Size, data=diamonds)


plot(diamonds$Price ~ diamonds$Carat.Size, xlab='Carat Size', ylab='Price',main='Diamond Price vs
Carat Size')

abline(model_carat)

plot(model_carat) #First plot is the residuals vs fitted value plot
```

Q2: We could try a reciprocal transformation (1/x), exponential (e^x) or a log.

Q3: Exponential transformation would not be appropriate as that transformation would make small diamonds seem smaller and large diamonds seem larger to the regression. Below is a plot of the transformation for observation:



Q4: With a fan shape, we are trying to reduce the very high value diamonds. We could accomplish this with a log or square root transformation.

Q5:

```
diamonds$price_ln <- log(Price)
diamonds$size_ln <- log(Carat.Size)
diamonds$size_sqrt <- (Carat.Size)**.5
```

Q6:

Price_ln

$$\log(price) = 6.495 + 1.826 Carat.Size + \varepsilon$$

Size_ln

$$Price = 5057.7 + 5080 * \log(Carat.Size) + \varepsilon$$

Q7:

Model price_ln predicts a half carat diamond would cost $1649.

Model Size_ln predicts a half carat diamond would cost $1551

```
exp(6.495 + 1.82614*.5)
5057.657 + 5080*log(.5)
```

Q8:

Model_lnsqrt

$$\log(price) = 4.968 + 3.401 * sqrt(Carat.Size) + \varepsilon$$

Model_lnln

$$\log(price) = 8.404879 + 1.501 * \log(\text{Carat. Size}) + \varepsilon$$

```
model_lnsqrt <- lm(price_ln ~ size_sqrt, data=diamonds)
model_lnln <- lm(price_ln ~ size_ln, data=diamonds)
coef(model_lnsqrt)
coef(model_lnln)
```

9: Model log sqrt predicts a price of $1592.20 for a half carat diamond.

```
exp(4.968 + 3.401*sqrt(.5))
```

Model log log predicts a price of $1579.06 for a half carat diamond.

```
exp(8.405 + 1.501*log(.5))
```

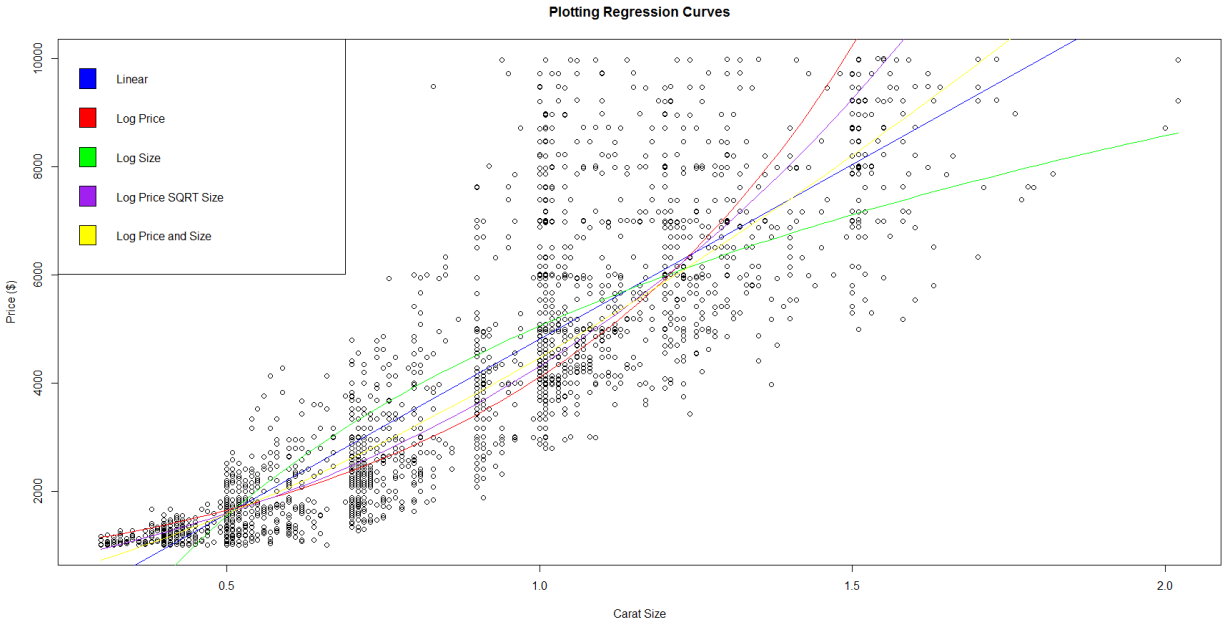
10: Model log log would predict the price to increase by $\left(\frac{p_1}{p_2}\right)^{\beta_1}$, 5.2 times increase.

$$\log\left(\frac{price_{1.5}}{price_{.5}}\right) = \log(price_{1.5}) - \log(price_{.5}) = \beta_0 + \beta_1 \log(1.5) - (\beta_0 + \beta_1 \log(.5))$$

$$= \beta_1(\log(1.5) - \log(.5)) = \beta_1 \log\left(\frac{1.5}{.5}\right) \rightarrow \frac{price_{1.5}}{price_{.5}} = 3^{1.5008} = 5.2$$

11: The best model is negotiable, but in my opinion the best fit is achieved with the log-log model. The log-log model has the best (largest) R^2 with .82, and the best (lowest) AIC of 789. A plot of the untransformed fits on the same graph is displayed below.

In statistics, it is important to balance 'best statistical model' with interpretability. For that reason, another model can be argued.




```
#Plotting for better visualization
graphics.off()
```

```
plot(diamonds$Price ~ diamonds$Carat.Size, xlab='Carat Size', ylab='Price ($)', main='Plotting Re
gression Curves')
abline(model_carat, col='blue')
curve(exp(6.495086 + 1.82614*x), add=T, col='red')
curve(5057.657 + 5080.563*log(x), add=T, col='green')
curve(exp(4.968119 + 3.401203*sqrt(x)),add=T, col='purple')
curve(exp(8.4049 + 1.5008*log(x)), add=T, col='yellow')
```

12: The interval found by the linear model is $1505-$1647 while the interval for the log-log model is $1553-$1605. It is expected that the log-log model would be more exact in the low price region as that area has been 'enlarged' in the transformation.

```
#Prediction confidence interval

predict.lm(model_carat, newdata = data.frame(Carat.Size = .5), interval='confidence')

exp(predict.lm(model_lnln, newdata = data.frame(size_ln = log(.5)), interval='confidence'))
```

13:

```
diamonds$Color <- factor(diamonds$Color, ordered=FALSE)
model_transform_interact <- lm(price_ln ~ Color*size_ln, data=diamonds)
lm(formula = price_ln ~ Color * size_ln, data = diamonds)

Residuals:
     Min       1Q   Median       3Q      Max
-0.69502 -0.15312 -0.01442  0.14805  0.82085

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    8.64089    0.02221 389.007  < 2e-16 ***
ColorE        -0.08904    0.02730  -3.261 0.001123 **
ColorF        -0.09594    0.02676  -3.585 0.000343 ***
ColorG        -0.12047    0.02555  -4.716 2.53e-06 ***
ColorH        -0.24939    0.02515  -9.916  < 2e-16 ***
ColorI        -0.35913    0.02564 -14.004  < 2e-16 ***
ColorJ        -0.51220    0.02653 -19.304  < 2e-16 ***
ColorK        -0.62826    0.03162 -19.868  < 2e-16 ***
size_ln        1.63357    0.03724  43.870  < 2e-16 ***
ColorE:size_ln -0.03357    0.04590  -0.731 0.464719
ColorF:size_ln  0.06166    0.04719   1.307 0.191477
ColorG:size_ln  0.12794    0.04867   2.629 0.008619 **
ColorH:size_ln  0.18899    0.05362   3.525 0.000431 ***
ColorI:size_ln  0.16548    0.05712   2.897 0.003797 **
ColorJ:size_ln  0.29776    0.06122   4.864 1.22e-06 ***
ColorK:size_ln  0.17961    0.08651   2.076 0.037971 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2274 on 2674 degrees of freedom
Multiple R-squared:  0.8822,   Adjusted R-squared:  0.8815
F-statistic:  1335 on 15 and 2674 DF,  p-value: < 2.2e-16
```

14: The prediction interval is from $3288-$8036 and the confidence interval is $4992-$5293. A linear model prediction interval is $3601.545-$7539.181which is tighter than the interval for the transformed model.

```
exp(predict.lm(model_transform_interact, newdata=data.frame(size_ln=log(1), Color='F'),
interval='confidence'))
exp(predict.lm(model_transform_interact, newdata=data.frame(size_ln=log(1), Color='F'),
interval='prediction'))
```

```
LinearFull <- lm(Price ~ Carat.Size * Color, data=diamonds)
predict(LinearFull, list(Carat.Size=1,Color='F'), interval='prediction')
```

15: This question is open ended, please provide what you consider the best model. One should be measuring with adjusted R^2, or AIC. Additionally, any interpretability concerns can be made to limit the model. Below is some code to generate the best model according to AIC criteria with the three predictors. There is a triple interaction:

```
diamonds$Clarity <- factor(diamonds$Clarity, ordered=FALSE)
diamonds$Cut <- factor(diamonds$Cut, ordered=FALSE)

ModelFull <- lm(price_ln ~ size_ln*Color*Clarity*Cut, data=diamonds)
model_best <- step(ModelFull)
summary(model_best)
```

Adjusted R-squared: 0.9728
AIC: -4174.937

REMAINDER OF MY R CODE:

```
diamonds <-
read.csv("https://raw.githubusercontent.com/brianlukoff/sta371g/master/data/diamonds.csv")

diamonds$Color <- ordered(diamonds$Color,c('D','E','F','G','H','I','J','K'))
diamonds$Clarity <- ordered(diamonds$Clarity,c('IF','VVS1','VVS2','VS1','VS2','SI1','SI2'))
diamonds$Cut <- ordered(diamonds$Cut,c('Ideal','Excellent','Very Good','Good'))

model_carat <- lm(Price ~ Carat.Size, data=diamonds)

plot(diamonds$Price ~ diamonds$Carat.Size, xlab='Carat Size', ylab='Price',main='Diamond Price vs
Carat Size')
abline(model_carat)
plot(model_carat) #First plot is the residuals vs fitted value plot

diamonds$price_ln <- log(Price)
diamonds$size_ln <- log(Carat.Size)
diamonds$size_sqrt <- (Carat.Size)**.5

attach(diamonds)

model_priceln <- lm(price_ln ~ Carat.Size)
model_sizeln <- lm(Price ~ size_ln)
coef(model_priceln)
coef(model_sizeln)

exp(6.495 + 1.82614*.5)
5057.657 + 5080*log(.5)

model_lnsqrt <- lm(price_ln ~ size_sqrt, data=diamonds)
model_lnln <- lm(price_ln ~ size_ln, data=diamonds)
coef(model_lnsqrt)
coef(model_lnln)

exp(4.968 + 3.401*sqrt(.5))
exp(8.405 + 1.501*log(.5))

#11
predicted1.5 <- exp(8.4049 + 1.5008*log(1.5))
predicted.5 <- exp(8.4049 + 1.5008*log(.5))

predicted1.5/predicted.5
exp(1.5008*log(1.5/.5))

#Examining which is the best fit: model_carat, model_priceln, model_sizeln, model_lnsqrt, and
model_lnln)
AIC(model_carat,model_priceln, model_sizeln, model_lnsqrt, model_lnln)
summary(model_carat)$r.squared
summary(model_priceln)$r.squared
summary(model_sizeln)$r.squared
summary(model_lnsqrt)$r.squared
summary(model_lnln)$r.squared

#Plotting for better visualization
graphics.off()
plot(diamonds$Price ~ diamonds$Carat.Size, xlab='Carat Size', ylab='Price ($)', main='Plotting
Regression Curves')
abline(model_carat, col='blue')
curve(exp(6.495086 + 1.82614*x), add=T, col='red')
curve(5057.657 + 5080.563*log(x), add=T, col='green')
curve(exp(4.968119 + 3.401203*sqrt(x)),add=T, col='purple')
curve(exp(8.4049 + 1.5008*log(x)), add=T, col='yellow')


legend("topleft", c("Linear", "Log Price", 'Log Size','Log Price SQRT Size','Log Price and
Size'), fill=c("blue", "red", 'green','purple','yellow'))
```

```
#Prediction confidence interval
predict.lm(model_carat, newdata = data.frame(Carat.Size = .5), interval='confidence')
exp(predict.lm(model_lnln, newdata = data.frame(size_ln = log(.5)), interval='confidence'))

diamonds$Color <- factor(diamonds$Color, ordered=FALSE)
model_transform_interact <- lm(price_ln ~ Color*size_ln, data=diamonds)

exp(predict.lm(model_transform_interact, newdata=data.frame(size_ln=log(1), Color='F'),
interval='confidence'))
exp(predict.lm(model_transform_interact, newdata=data.frame(size_ln=log(1), Color='F'),
interval='prediction'))

LinearFull <- lm(Price ~ Carat.Size * Color, data=diamonds)
predict(LinearFull, list(Carat.Size=1,Color='F'), interval='prediction')

diamonds$Clarity <- factor(diamonds$Clarity, ordered=FALSE)
diamonds$Cut <- factor(diamonds$Cut, ordered=FALSE)

ModelFull <- lm(price_ln ~ size_ln*Color*Clarity*Cut, data=diamonds)
model_best <- step(ModelFull)
summary(model_best)
```