Student: Yinan Kang

CSCI E-106: Data Modeling

Assignment 4

Due: March, 4 2019 at 7:19 pm EST

Instructions: Students should submit their reports on Canvas. The report needs to clearly state what question is being solved, step-by-step walk-through solutions, and final answers clearly indicated. Please solve by hand where appropriate.

Please submit two files: (1) a R Markdown file (.Rmd extension) and (2) a PDF document generated using knitr for the .Rmd file submitted in (1) where appropriate. Please, use RStudio Cloud for your solutions.

All questions are coming from Kutner, M. *et al*: Applied Linear Statistical Models, Fifth Edition.

1. (3.26)
2. (3.28)
3. (3.29)
4. (3.31)
5. (3.32)

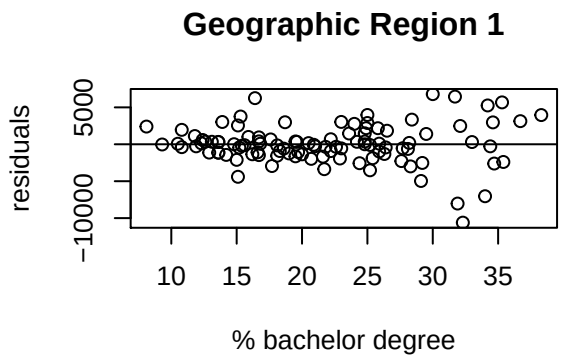
YK_Assignment4

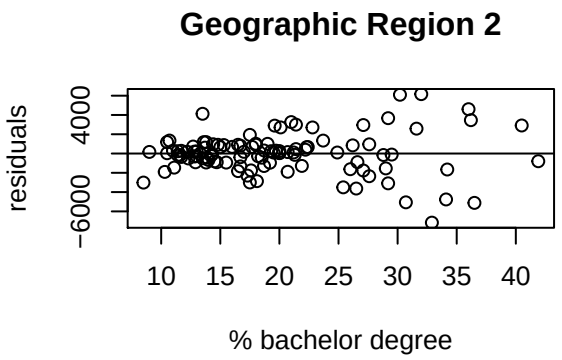
Yinan Kang

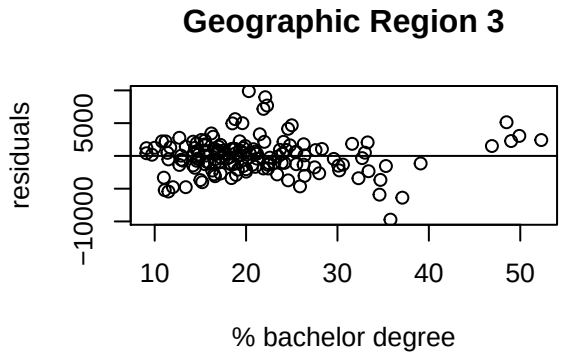
3/1/2019

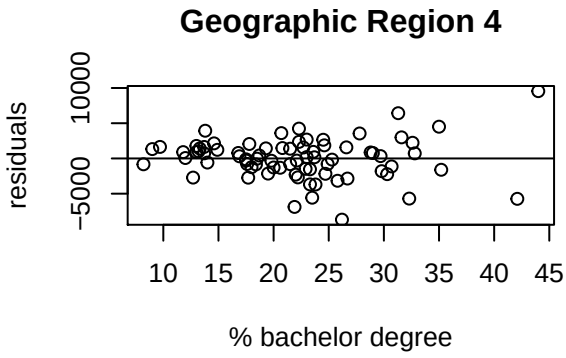
Problem 3.26 - CDI Data

```
#Importing data
cdi.df <- read.csv("CDI.csv")

#Split data by 'Geographic Region'
cdi.1 <- filter(cdi.df,cdi.df$Geographic.Region == 1)
cdi.2 <- filter(cdi.df,cdi.df$Geographic.Region == 2)
cdi.3 <- filter(cdi.df,cdi.df$Geographic.Region == 3)
cdi.4 <- filter(cdi.df,cdi.df$Geographic.Region == 4)

#Regress 'per capita income' (Y) against '% with bachelor's degree' (X)
cdi.1.model <- lm(cdi.1$Per.Capita.Income ~ cdi.1$X.Bachelor.s.degrees)
cdi.2.model <- lm(cdi.2$Per.Capita.Income ~ cdi.2$X.Bachelor.s.degrees)
cdi.3.model <- lm(cdi.3$Per.Capita.Income ~ cdi.3$X.Bachelor.s.degrees)
cdi.4.model <- lm(cdi.4$Per.Capita.Income ~ cdi.4$X.Bachelor.s.degrees)

#Residual Plots
par(mfrow=c(2,2))
cdi.1.resplot <- plot(cdi.1$X.Bachelor.s.degrees, cdi.1.model$residuals, ylab="residuals",
                      xlab="% bachelor degree", main="Geographic Region 1")
abline(0,0)
cdi.2.resplot <- plot(cdi.2$X.Bachelor.s.degrees, cdi.2.model$residuals, ylab="residuals",
                      xlab="% bachelor degree", main="Geographic Region 2")
abline(0,0)
cdi.3.resplot <- plot(cdi.3$X.Bachelor.s.degrees, cdi.3.model$residuals, ylab="residuals",
                      xlab="% bachelor degree", main="Geographic Region 3")
abline(0,0)
cdi.4.resplot <- plot(cdi.4$X.Bachelor.s.degrees, cdi.4.model$residuals, ylab="residuals",
                      xlab="% bachelor degree", main="Geographic Region 4")
abline(0,0)
```

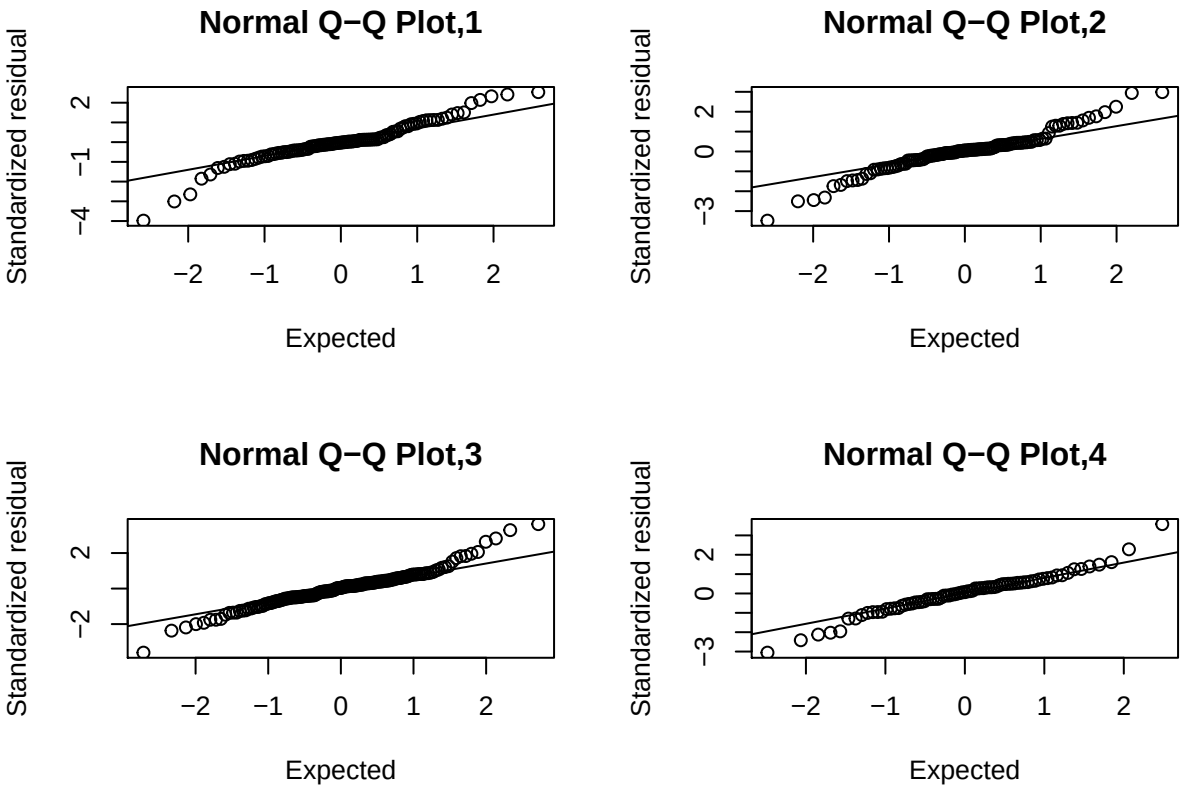









```r
# Normal Probability Plots

## Standardizing Residuals
cdi.1.stan <- rstandard(cdi.1.model)
cdi.2.stan <- rstandard(cdi.2.model)
cdi.3.stan <- rstandard(cdi.3.model)
cdi.4.stan <- rstandard(cdi.4.model)

par(mfrow=c(2,2))
qqnorm(cdi.1.stan, main="Normal Q-Q Plot,1", xlab="Expected", ylab="Standardized residual")
qqline(cdi.1.stan)
qqnorm(cdi.2.stan, main="Normal Q-Q Plot,2", xlab="Expected", ylab="Standardized residual")
qqline(cdi.2.stan)
qqnorm(cdi.3.stan, main="Normal Q-Q Plot,3", xlab="Expected", ylab="Standardized residual")
qqline(cdi.3.stan)
qqnorm(cdi.4.stan, main="Normal Q-Q Plot,4", xlab="Expected", ylab="Standardized residual")
qqline(cdi.4.stan)
```



Analysis: Observing the four geographic regions via both the Residual Plots and Normal Probability Plots, I expect the models for Regions 1 and 2 to have slightly higher error variances. They have significant deviances from expected residual in both the upper and lower tails, and are less dense in the middle regions relative to, say, Region 3 (which also has noticeable deviances at tails, but the larger number of near-expected errors in the middle may lower overall variance). The model for Region 4 may have a variance similar to Region 1 and 2, but observationally, there are less extreme deviances from expected residuals, particularly at the upper tail.

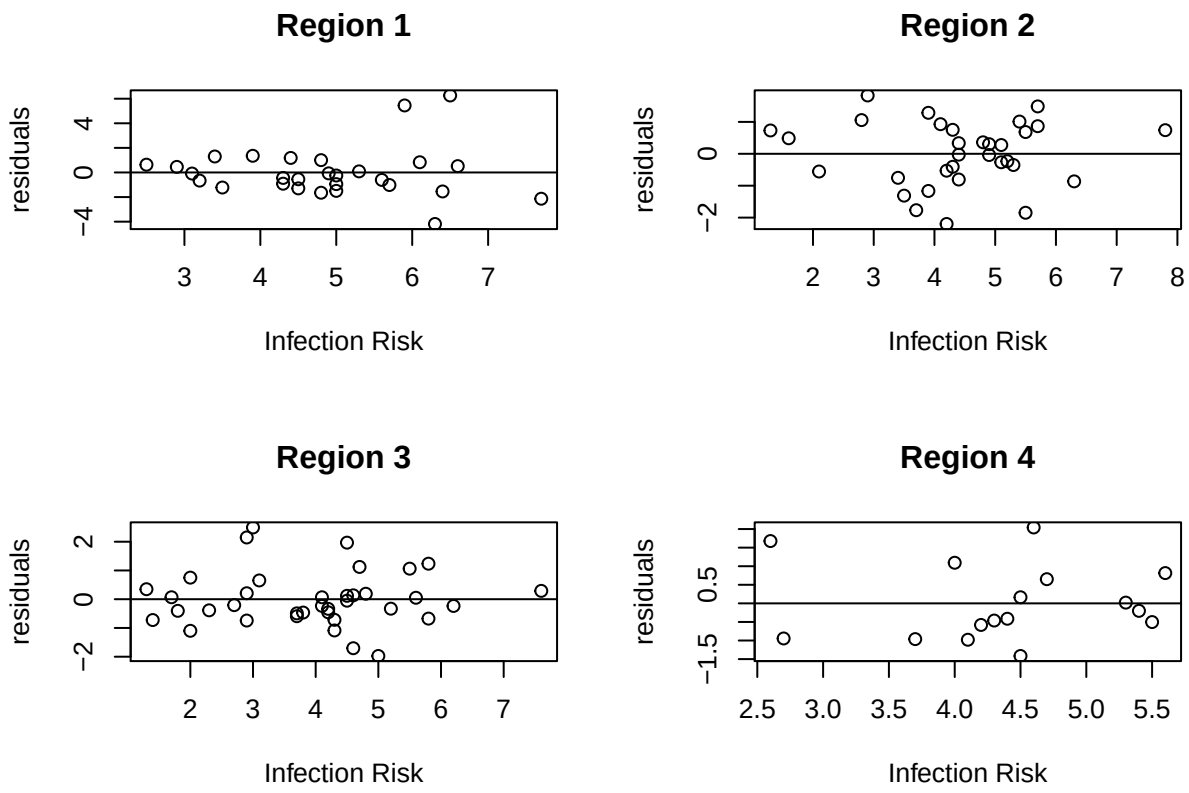
Problem 3.28 - SENIC

```
rm(list=ls())
# Import Data
senic <- read.table("/cloud/project/APC1.DAT", quote="\"", comment.char="")
names(senic) <- c("ID", "length.of.stay", "age" , "infection.risk", "routine.culturing.ratio", "routine
head(senic,3)
```

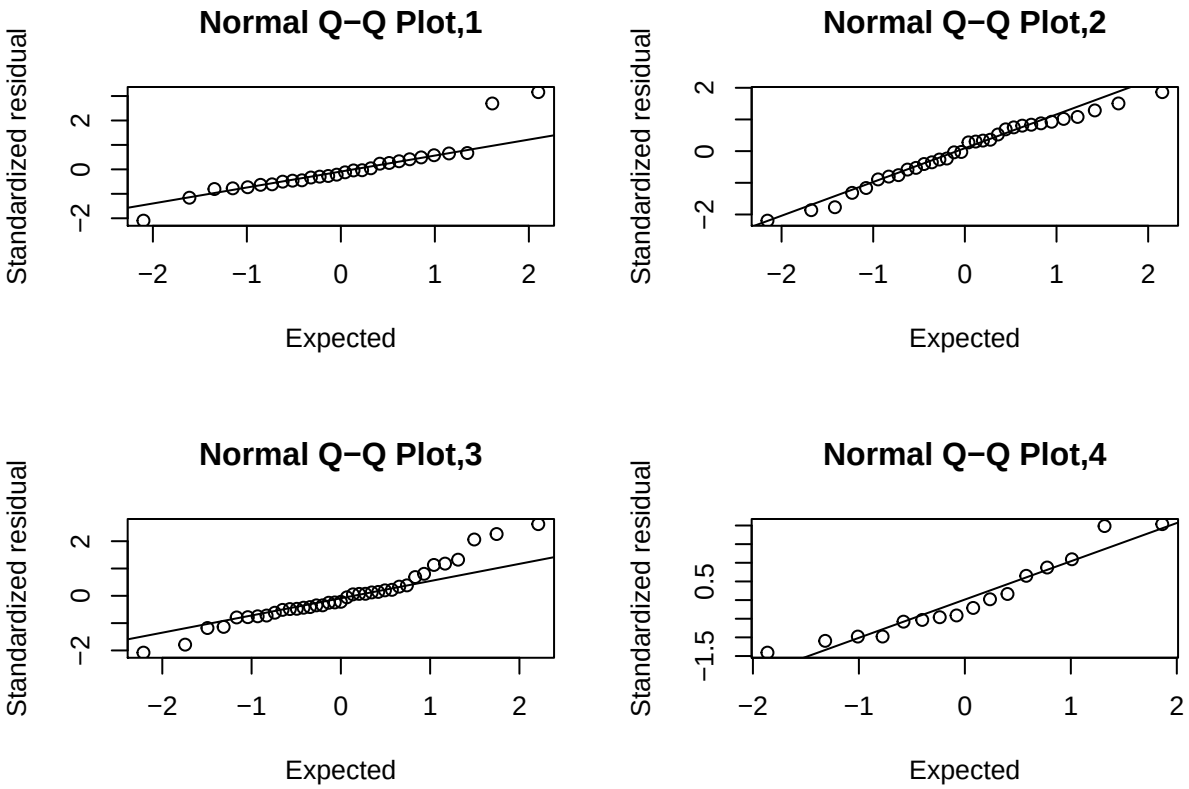
```
##   ID length.of.stay  age infection.risk routine.culturing.ratio
## 1  1           7.13 55.7            4.1                     9.0
## 2  2           8.82 58.2            1.6                     3.8
## 3  3           8.34 56.9            2.7                     8.1
##   routine.chest.xray.ratio number.of.beds medical.school.affiliation
## 1                     39.6           279                           2
## 2                     51.7            80                           2
## 3                     74.0           107                           2
##   region average.daily.census number.of.nurses
## 1      4                  207              241
## 2      2                   51               52
## 3      3                   82               54
```

```
##   available.facilities.services
## 1                            60
## 2                            40
## 3                            20
```

```r
# Breaking up by Region
senic.1 <- filter(senic, senic$region == 1)
senic.2 <- filter(senic, senic$region == 2)
senic.3 <- filter(senic, senic$region == 3)
senic.4 <- filter(senic, senic$region == 4)

# 1.46 dictates regress 'avg length of stay in hospital' (Y) vs. 'infection risk' (X)
senic.1.model <- lm(senic.1$length.of.stay ~ senic.1$infection.risk)
senic.2.model <- lm(senic.2$length.of.stay ~ senic.2$infection.risk)
senic.3.model <- lm(senic.3$length.of.stay ~ senic.3$infection.risk)
senic.4.model <- lm(senic.4$length.of.stay ~ senic.4$infection.risk)

#Residual Plots
par(mfrow=c(2,2))
senic.1.resplot <- plot(senic.1$infection.risk, senic.1.model$residuals, ylab="residuals",
                        xlab="Infection Risk", main="Region 1")
abline(0,0)
senic.2.resplot <- plot(senic.2$infection.risk, senic.2.model$residuals, ylab="residuals",
                        xlab="Infection Risk", main="Region 2")
abline(0,0)
senic.3.resplot <- plot(senic.3$infection.risk, senic.3.model$residuals, ylab="residuals",
                        xlab="Infection Risk", main="Region 3")
abline(0,0)
senic.4.resplot <- plot(senic.4$infection.risk, senic.4.model$residuals, ylab="residuals",
                        xlab="Infection Risk", main="Region 4")
abline(0,0)
```




```
# Normal Probability Plots

## Standardizing Residuals
senic.1.stan <- rstandard(senic.1.model)
senic.2.stan <- rstandard(senic.2.model)
senic.3.stan <- rstandard(senic.3.model)
senic.4.stan <- rstandard(senic.4.model)

par(mfrow=c(2,2))
qqnorm(senic.1.stan, main="Normal Q-Q Plot,1", xlab="Expected", ylab="Standardized residual")
qqline(senic.1.stan)
qqnorm(senic.2.stan, main="Normal Q-Q Plot,2", xlab="Expected", ylab="Standardized residual")
qqline(senic.2.stan)
qqnorm(senic.3.stan, main="Normal Q-Q Plot,3", xlab="Expected", ylab="Standardized residual")
qqline(senic.3.stan)
qqnorm(senic.4.stan, main="Normal Q-Q Plot,4", xlab="Expected", ylab="Standardized residual")
qqline(senic.4.stan)
```



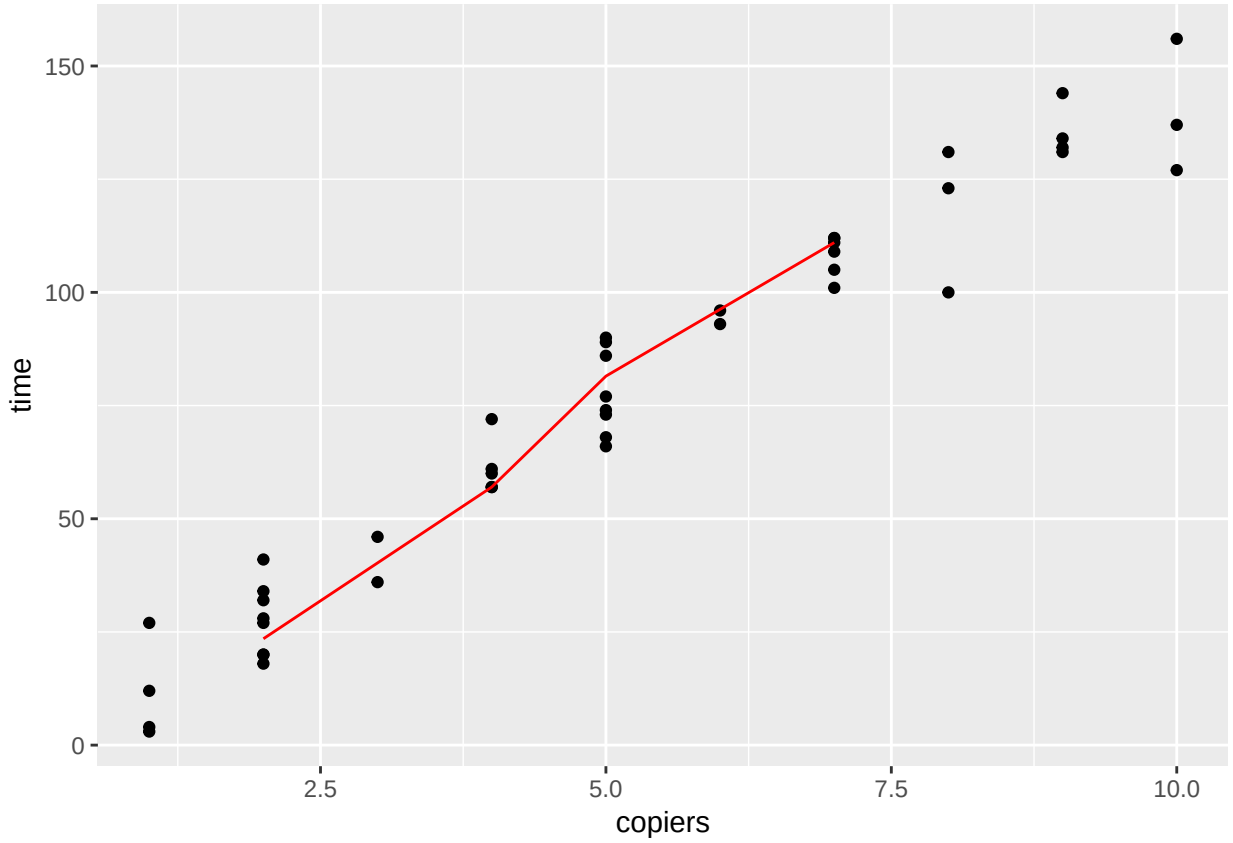
Analysis: Observing the Normal Probability plots, it looks like the model for Region 3 has the highest variance amongst the four regions, as it has the most deviances from expected residuals in both the lower and upper tails. The model for Region 1 has very few deviances, but the few deviances are high in magnitude. Models for Regions 2 and 4 look similar in variance, though Region 4 has noticeably less observations than the others.

Problem 3.29 - Copier Data

(a)

```
require(ggplot2)
rm(list=ls())
# Importing Data
copier.colnames <- c("time", "copiers")
copier.df <- read.table(url("http://users.stat.ufl.edu/~rrandles/sta4210/Rclassnotes/data/textdatasets/
colnames(copier.df) <- copier.colnames

# Splitting bands
band.1 <- filter(copier.df, between(copier.df$copiers, 0.5, 2.5))
band.2 <- filter(copier.df, between(copier.df$copiers, 2.5, 4.5))
band.3 <- filter(copier.df, between(copier.df$copiers, 4.5, 6.5))
band.4 <- filter(copier.df, between(copier.df$copiers, 6.5, 8.5))

# Calculating medians
band1medians <- apply(band.1, 2, median)
band2medians <- apply(band.2, 2, median)
band3medians <- apply(band.3, 2, median)
```

```
band4medians <- apply(band.4, 2, median)
medians.df <- as.data.frame(rbind(band1medians, band2medians, band3medians, band4medians))

# Plotting band smooths
ggplot() + geom_point(data = copier.df, aes(x = copiers, y = time)) +
  geom_line(data = medians.df, aes(x = copiers, y = time), col = "red")
```



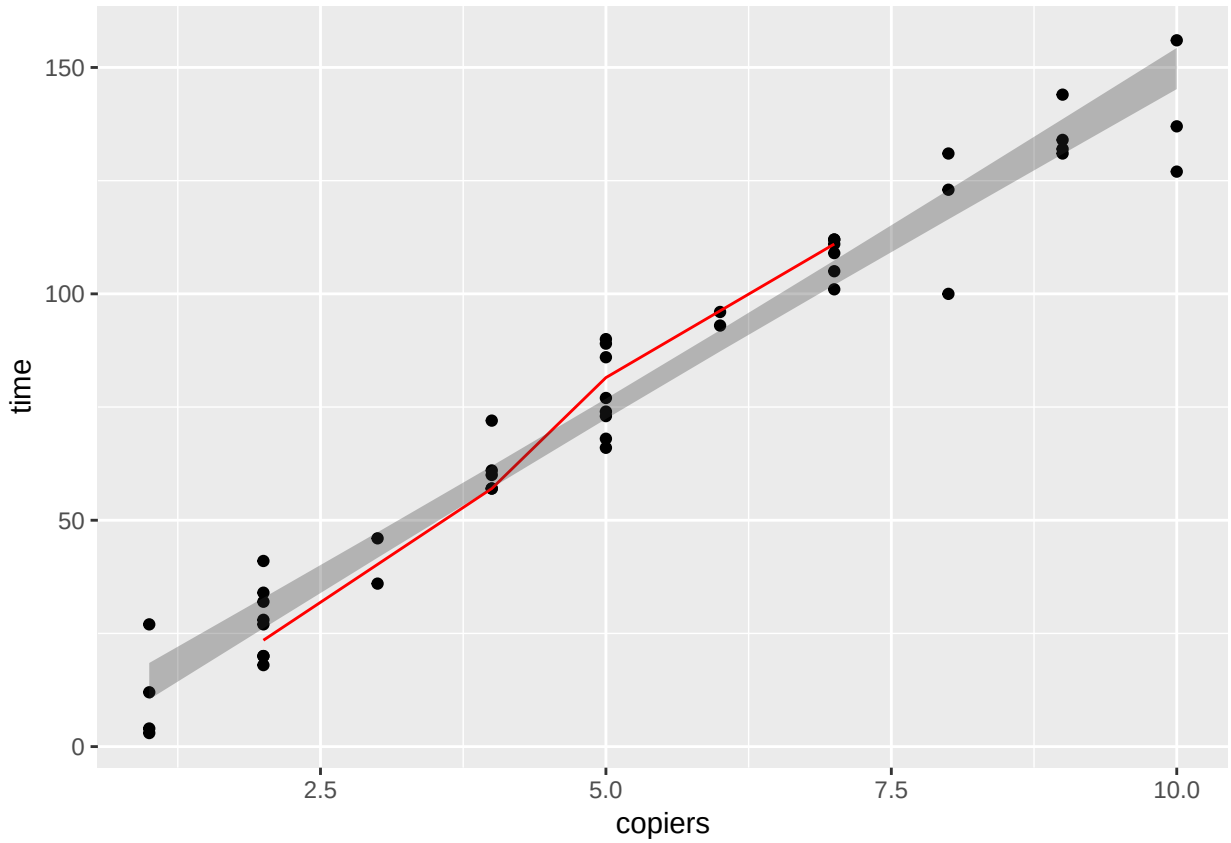
Does the band smooth suggest regression is linear?

Response: The line connecting the median values is not precisely linear, but trends up in relative constant proportion as X increases. This would've been observable just looking at the scatterplot of the data by itself.

(b)

```
require(ggplot2)
# Generating 90% confidence bands
copier.model <- lm(copier.df$time ~ copier.df$copiers)
copier.conf <- predict(copier.model, newdata=data.frame(copier.df), interval="confidence",
                       level = 0.9)

# Setting up data frames to be used in 'ggplot()'
copier.conf <- as.data.frame(copier.conf)
copier.conf<- cbind(copier.conf,copier.df)

# Plotting confidence band over band smooth and scatterplot
ggplot(data = copier.df, aes(x = copiers, y = time)) + geom_point() +
```

```
  geom_line(data = medians.df, aes(x = copiers, y = time), col = "red") +
  geom_ribbon(data=copier.conf,aes(ymin=lwr,ymax=upr), alpha=0.3)
```



Does band smooth fall entirely inside confidence band? What does this tell you about appropriateness of linear regression function?

Response: No, the band smooth does not fall entirely inside 90% confidence band. This tells us that a linear regression function may not yield the truest regression predictions.

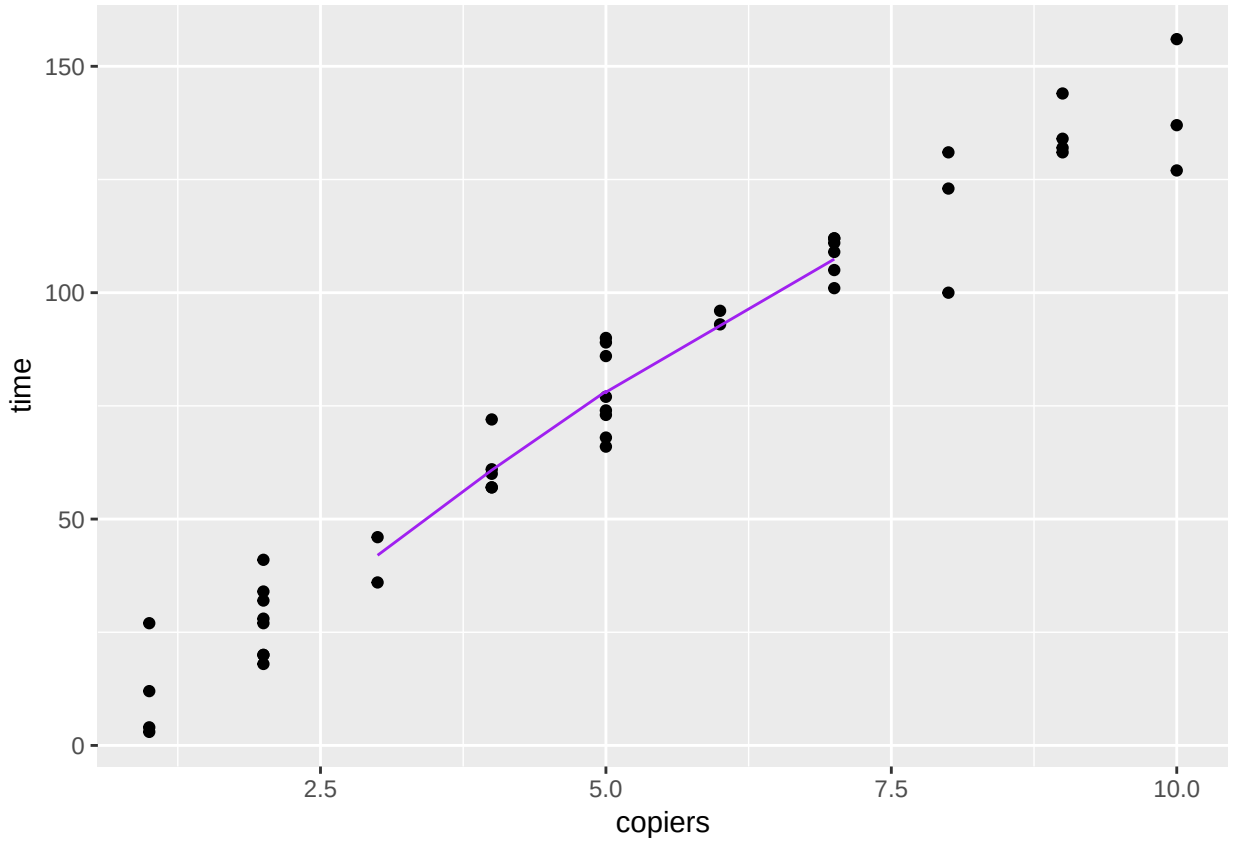
(c)

```
rm(list = setdiff(ls(), c("copier.df")))
# Dividing into 6 neighborhoods
neigh.1 <- filter(copier.df, between(copier.df$copiers, 0.5, 3.5))
neigh.2 <- filter(copier.df, between(copier.df$copiers, 1.5, 4.5))
neigh.3 <- filter(copier.df, between(copier.df$copiers, 2.5, 5.5))
neigh.4 <- filter(copier.df, between(copier.df$copiers, 3.5, 6.5))
neigh.5 <- filter(copier.df, between(copier.df$copiers, 4.5, 7.5))
neigh.6 <- filter(copier.df, between(copier.df$copiers, 5.5, 8.5))

# Making list of neighborhoods for loop
neigh.list <- list(neigh.1, neigh.2, neigh.3, neigh.4, neigh.5, neigh.6)

# Blank data frame to store prediction and results
neigh.df <- data.frame()
```

```r
# Loop for making a regression on each neighborhood, and storing results
for (i in 1:6) {
  df <- neigh.list[[i]]
  model <- lm(df$time ~ df$copiers)
  prediction <- predict(model)
  prediction <- t(t(prediction))
  assign(paste0("model.",i),model)
  df <- cbind(df,prediction)
  assign(paste0("full.",i), df)
  med <- ceiling(nrow(df)/2)
  neigh.df[i,1] <- i
  neigh.df[i,2] <- df[med,2]
  neigh.df[i,3] <- df[med,3]
}
colnames(neigh.df) <- c("neigh","Xc.middle","Yc.pred")
head(neigh.df,2)
```

```
##   neigh Xc.middle  Yc.pred
## 1     1         3 42.00000
## 2     2         4 60.80645
```

```r
# Creating plot with simplified Lowess curve
ggplot() + geom_point(data = copier.df, aes(x = copiers, y = time)) +
  geom_line(data = neigh.df, aes(x = Xc.middle, y = Yc.pred), col = "purple")
```



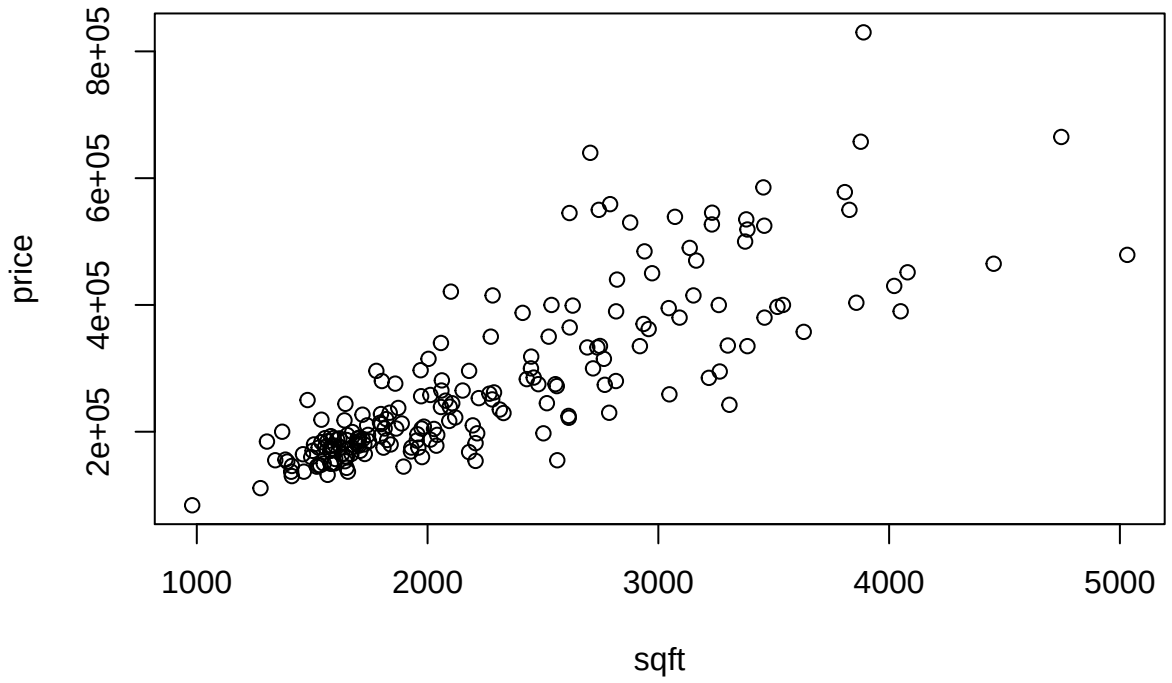
In what ways is simplified Lowess curve different than band smooth in (a)?

Response: The simplified Lowess curve here is smoother compared to the band smooth curve made in (a),

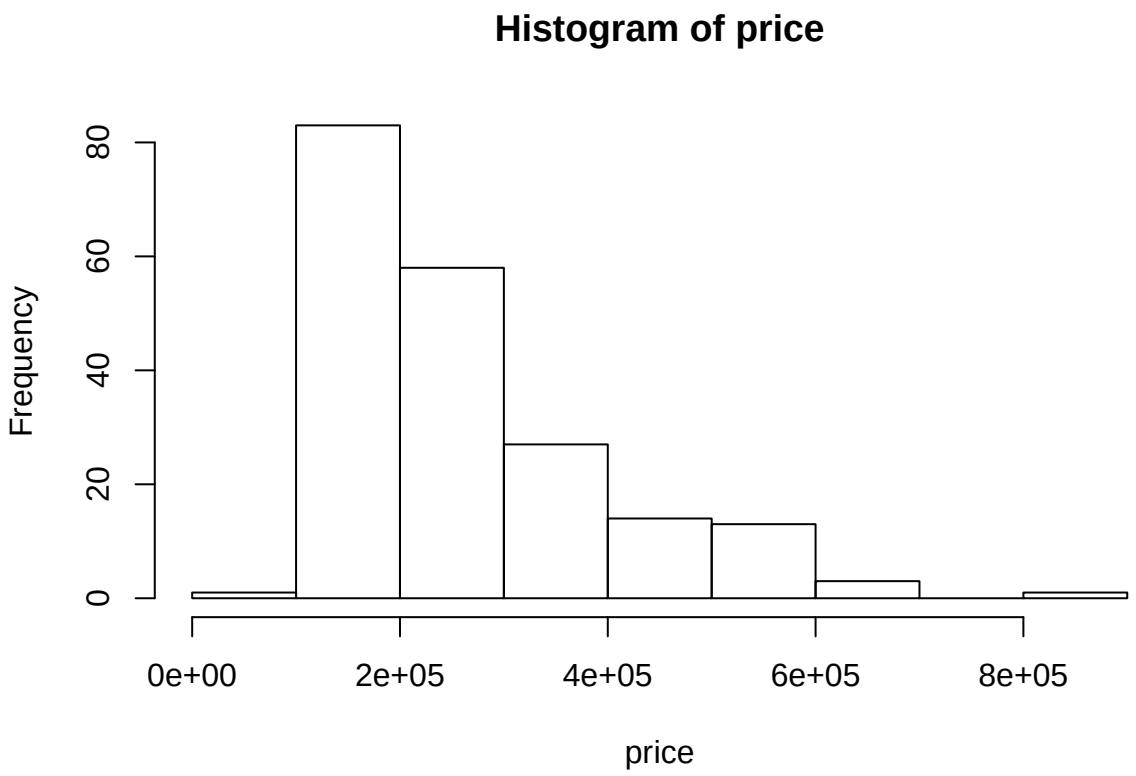
thus appearing more linear. Its range is also more centralized in the middle section of the data.

Problem 3.31 - Real Estate

```
# Importing data
rm(list=ls())
real.estate <- read.table(url("http://users.stat.ufl.edu/~rrandles/sta4210/Rclassnotes/data/textdatasets

# Setting seed and taking random sample of 200
set.seed(3319)

training.rows <- sample(c(1:nrow(real.estate)), size = 200, replace = FALSE)
training.data <- real.estate[training.rows,]

attach(training.data)

# Doing exploratory quick visuals
plot(price~sqft)
```



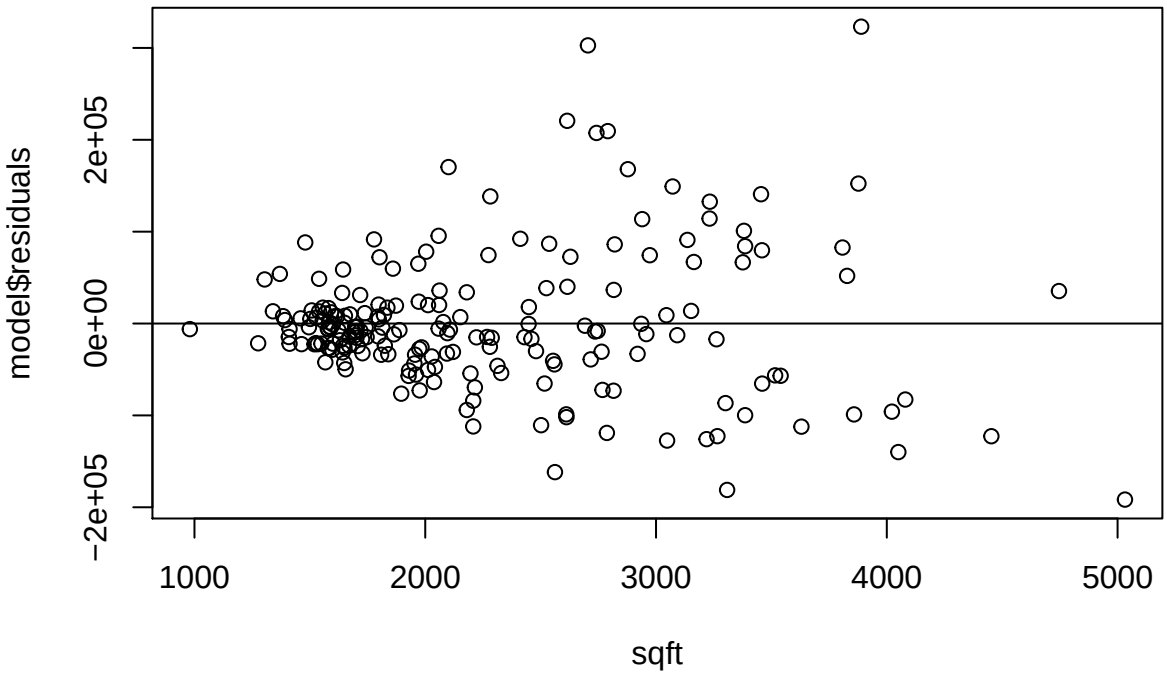

```
hist(price)
```



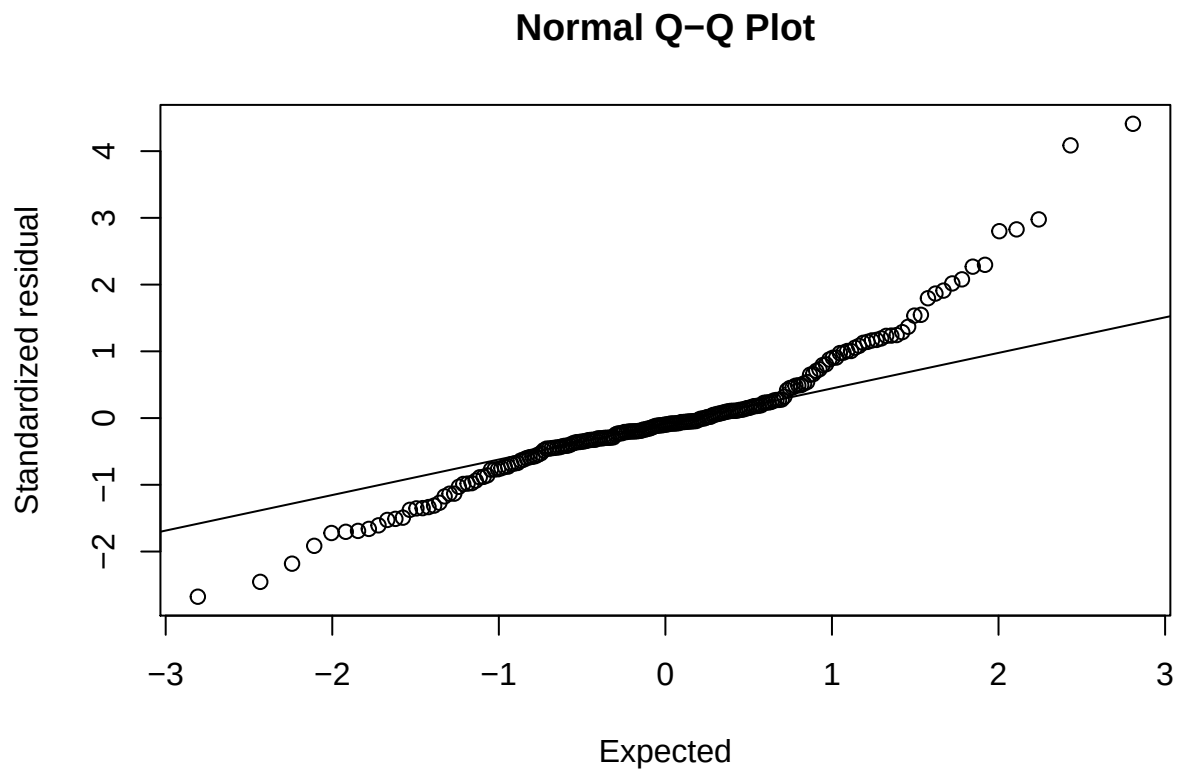

```
## The Scatterplot and Histogram show the data is skewed left

# Generating Model
model <- lm(price ~ sqft, data = training.data)
summary(model)
```

```
##
## Call:
## lm(formula = price ~ sqft, data = training.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -191633  -33133   -7272   20081  323128
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -50314.491  16593.650  -3.032  0.00275 **
## sqft           143.273      6.975  20.542  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 74350 on 198 degrees of freedom
## Multiple R-squared:  0.6806, Adjusted R-squared:  0.679
## F-statistic:   422 on 1 and 198 DF,  p-value: < 2.2e-16
```

```
## Residual Plot
plot(sqft, model$residuals)
abline(0,0)
```




```
## Normal Probability Plot
model.stan <- rstandard(model)
qqnorm(model.stan, main="Normal Q-Q Plot", xlab="Expected", ylab="Standardized residual")
qqline(model.stan)
```




```
anova(model)
```

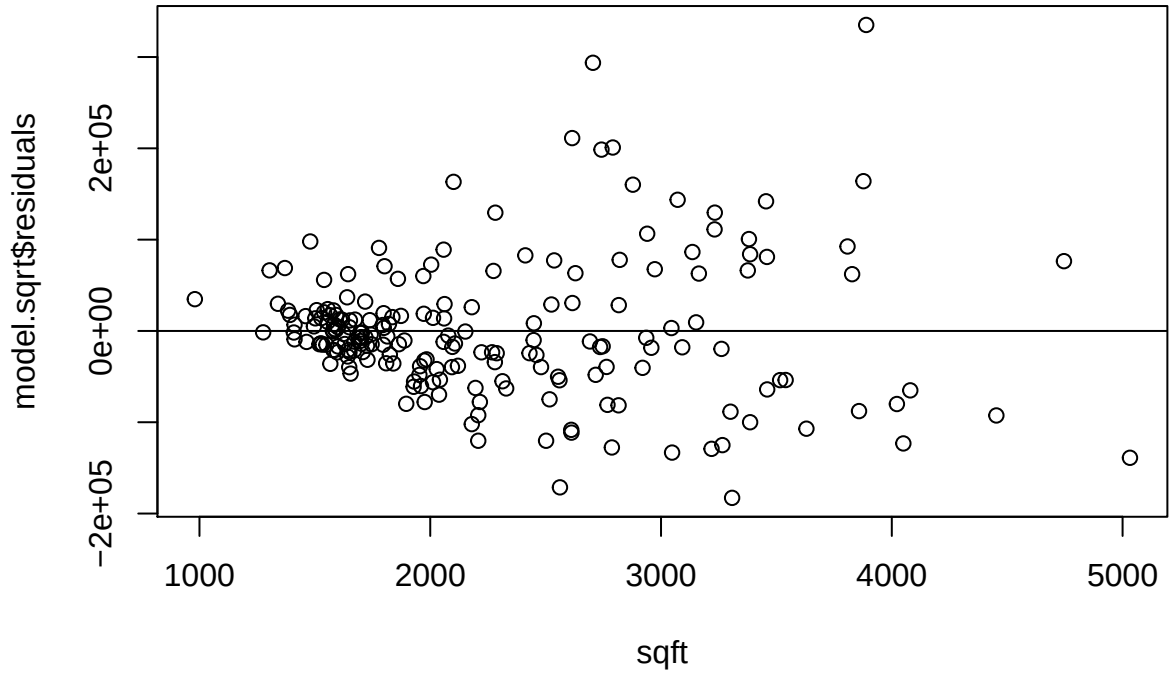
```
## Analysis of Variance Table
##
```

```
## Response: price
##             Df     Sum Sq    Mean Sq F value    Pr(>F)
## sqft         1 2.3323e+12 2.3323e+12  421.96 < 2.2e-16 ***
## Residuals 198 1.0944e+12 5.5274e+09
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

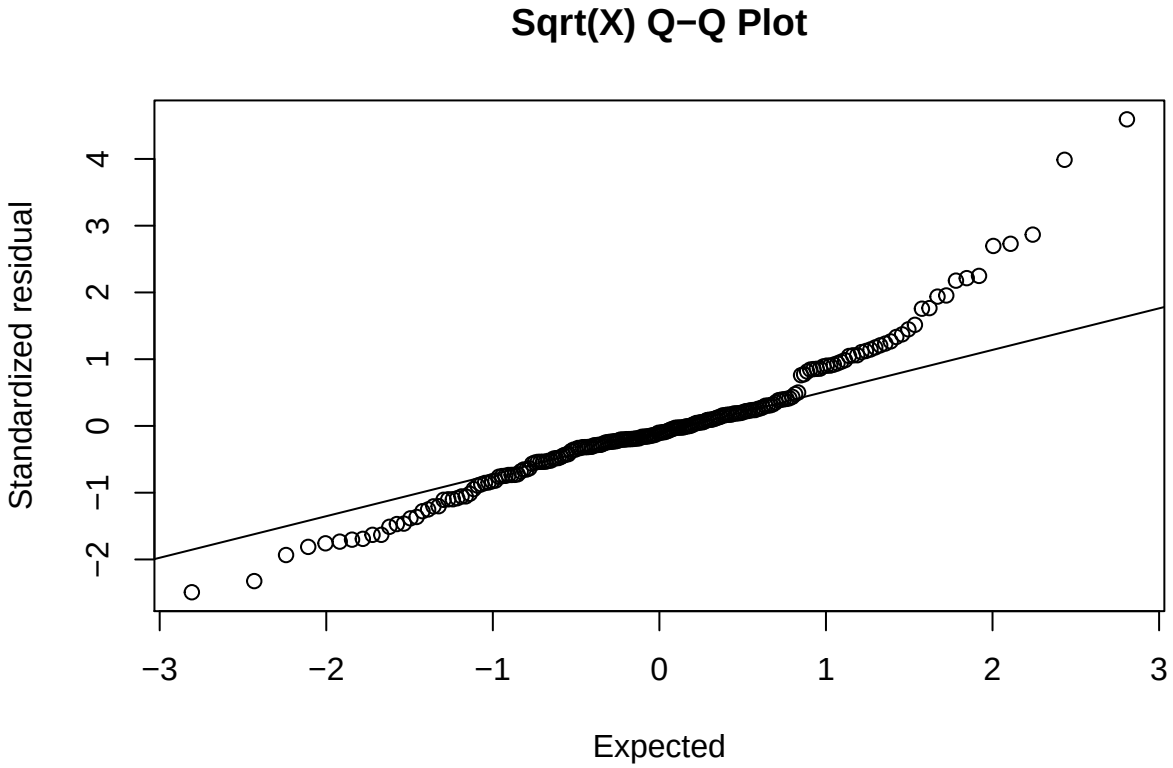
From viewing the residual plot, it looks like the linear regression model does has high NON-CONSTANCY OF VARIANCE, particularly for houses past ~2200 sqft. The highly irregular standardized residuals on the tails of the Q-Q plot also indicate NON-NORMALCY OF ERROR distribution, which would not be ideal for a linear regression model. There is no reason to believe errors are non-independent.

Looking at the Q-Q plot, we see a curvature at the upper tail that, according to textbook Figure 3.13, may be a prototype for a square root transformation of X.

```
sqft.sq <- sqrt(sqft)
model.sqrt<- lm(price~sqft.sq)
## Residual Plot
plot(sqft, model.sqrt$residuals)
abline(0,0)
```



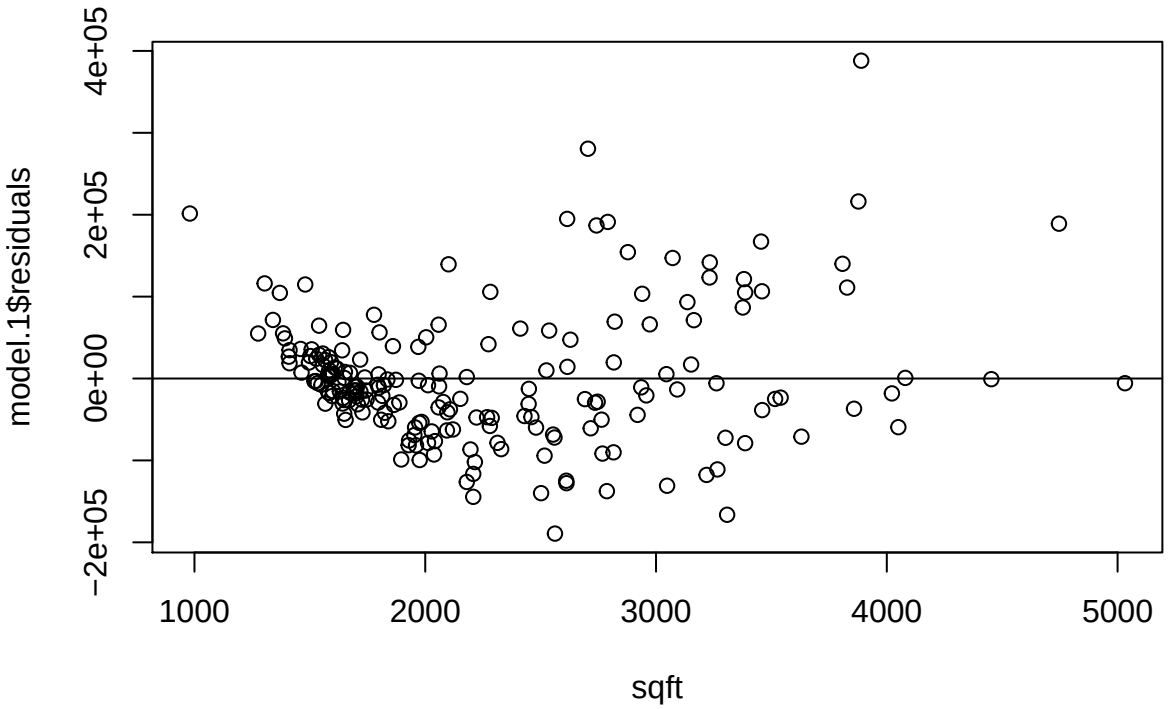

```
## Normal Probability Plot
model.sqrt.stan <- rstandard(model.sqrt)
qqnorm(model.sqrt.stan, main="Sqrt(X) Q-Q Plot", xlab="Expected", ylab="Standardized residual")
qqline(model.sqrt.stan)
```



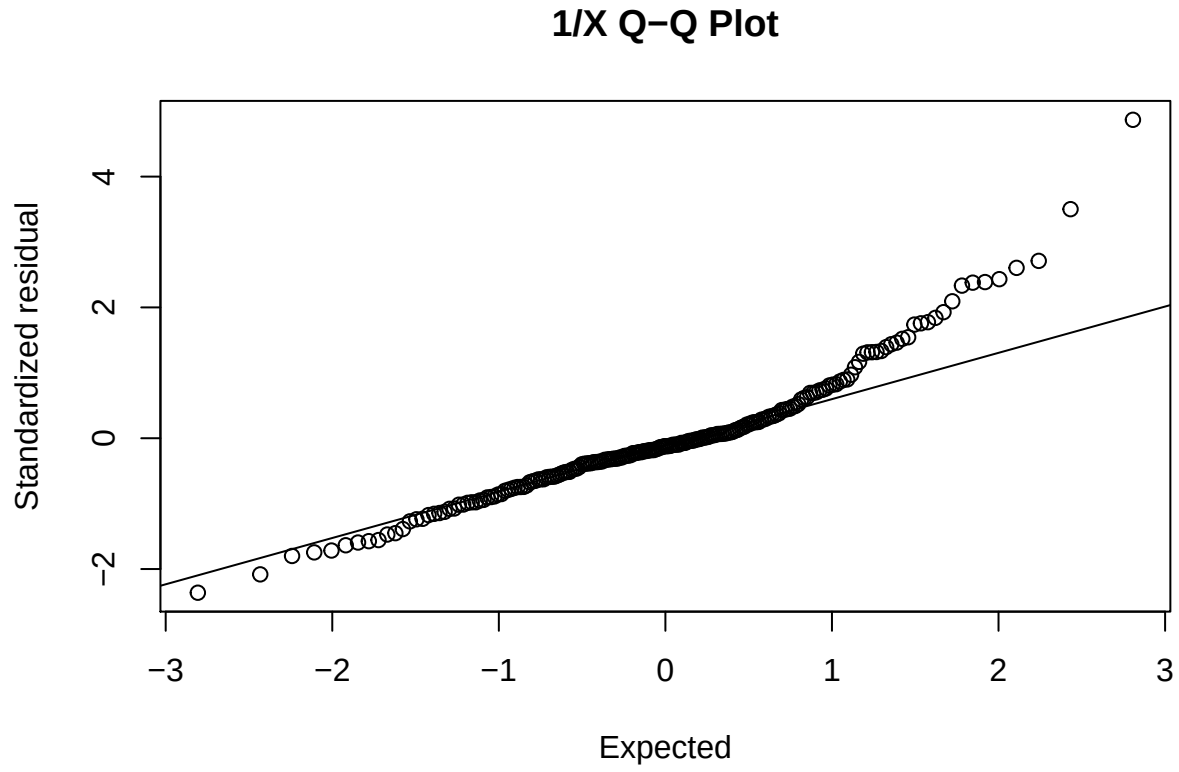
Looking at the plots for the square root transformation, we see that it actually made the residual variance worse from a modeling perspective. Offline, I also attempted a log(X) transformation, which yielded similarly poor results.

Next, will try a 1/X transformation:

```
sqft.1 <- 1/sqft
model.1 <- lm(price~sqft.1)
## Residual Plot
plot(sqft, model.1$residuals)
abline(0,0)
```



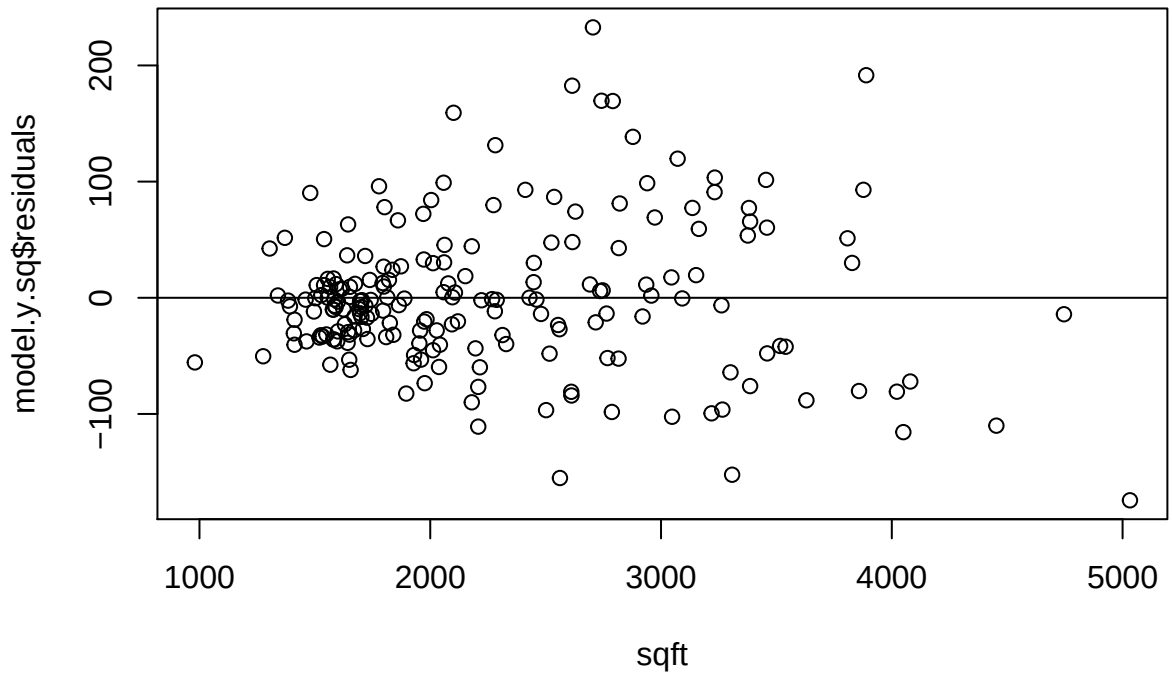

```
## Normal Probability Plot
model.1.stan <- rstandard(model.1)
qqnorm(model.1.stan, main="1/X Q-Q Plot", xlab="Expected", ylab="Standardized residual")
qqline(model.1.stan)
```



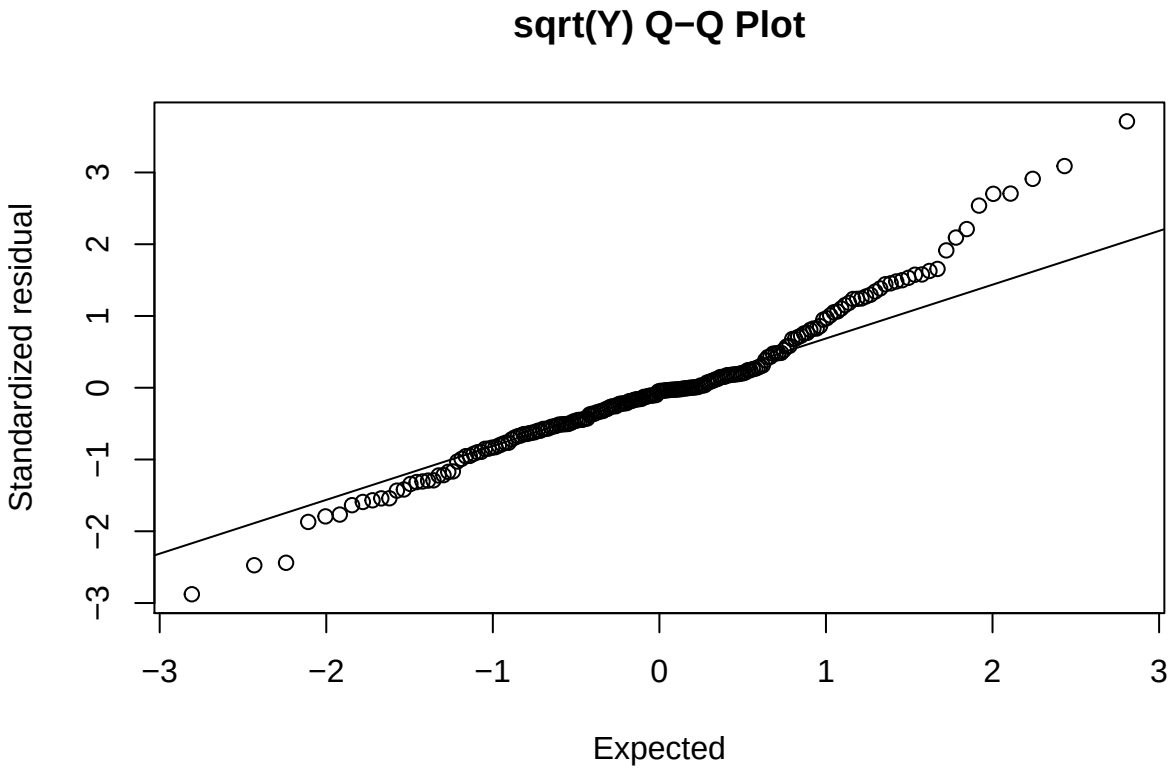
This transformation of X (sqft) also did not help.

Next, I will try a transformation of sqrt(y) (sqrt(price)) using original X (sqft)

```r
price.sq <- sqrt(price)
model.y.sq <- lm(price.sq~sqft)

## Residual Plot
plot(sqft, model.y.sq$residuals)
abline(0,0)
```



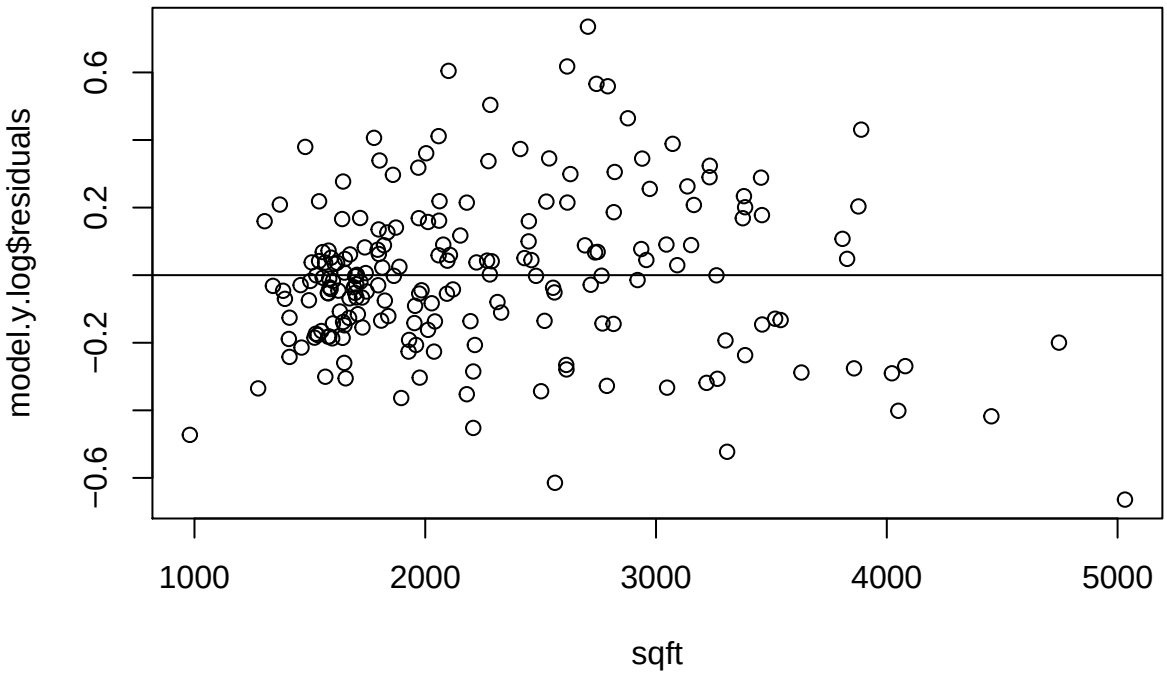

```r
## Normal Probability Plot
model.ysq.stan <- rstandard(model.y.sq)
qqnorm(model.ysq.stan, main="sqrt(Y) Q-Q Plot", xlab="Expected", ylab="Standardized residual")
qqline(model.ysq.stan)
```



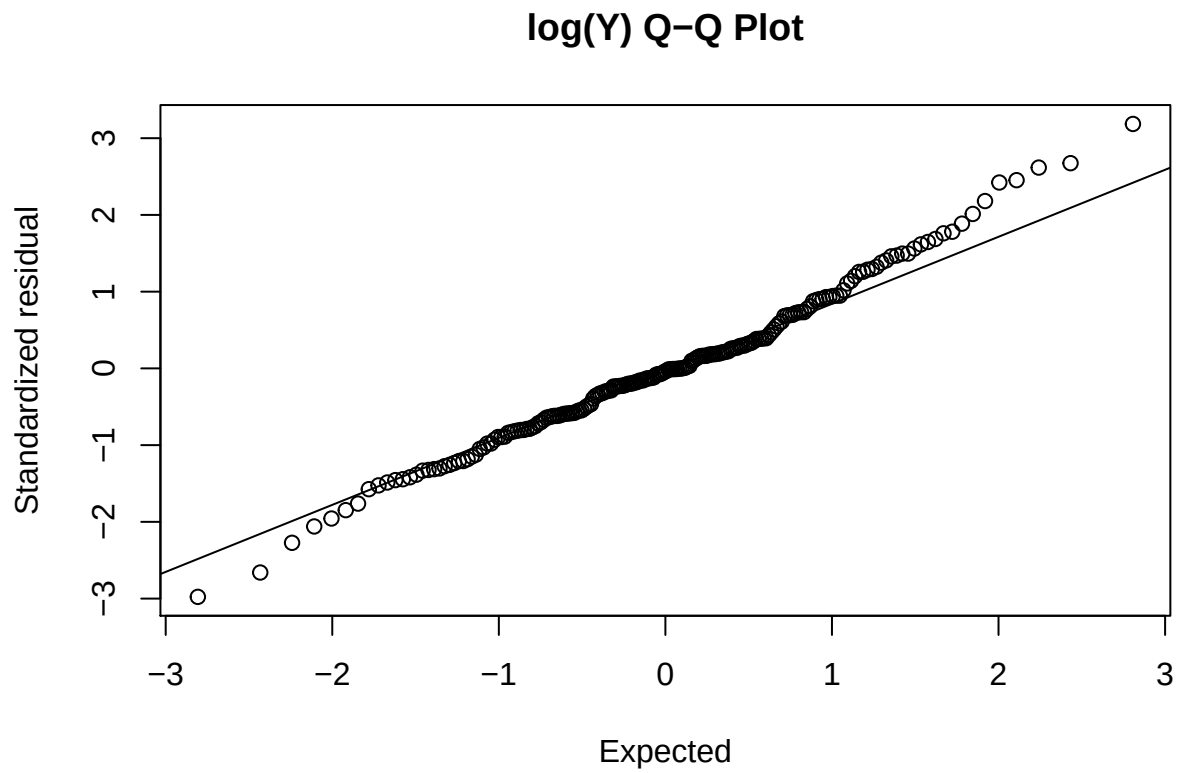
This looks to be about the same quality model as the others so far, perhaps marginally better simply from observation. I also tried sqrt(X) (sqrt(sqft)) and it produced similar results.

Lastly, I will try a log(Y) transformation:

```
price.log <- log(price)
model.y.log <- lm(price.log~sqft)

## Residual Plot
plot(sqft, model.y.log$residuals)
abline(0,0)
```




```
## Normal Probability Plot
model.y.log.stan <- rstandard(model.y.log)
qqnorm(model.y.log.stan, main="log(Y) Q-Q Plot", xlab="Expected", ylab="Standardized residual")
qqline(model.y.log.stan)
```




```
summary(model.y.log.stan)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -2.977409 -0.620903 -0.038726 -0.001141  0.557333  3.185410
```

Judging by the diagnostics above, this model seems like a (relative) winner. The QQ-Plot and the Residual Plot seem better than the original and the other attempts thus far, and the R-Sq value of this model is equatable to the original of ~ 0.7. Will use this for prediction.

Predicting prices of houses with X = 1000 sqft, X = 4900 sqft

```
# Creating 'newdata' df
new.data <- data.frame(sqft = c(1100,4900))

# Generating predictions

final.predictions <- predict(model.y.log, newdata=new.data)

# Transforming from 'log' back to '$'
final.predictions <- exp(final.predictions)
print(final.predictions)
```

```
##        1        2
## 142684.5 873829.0
```

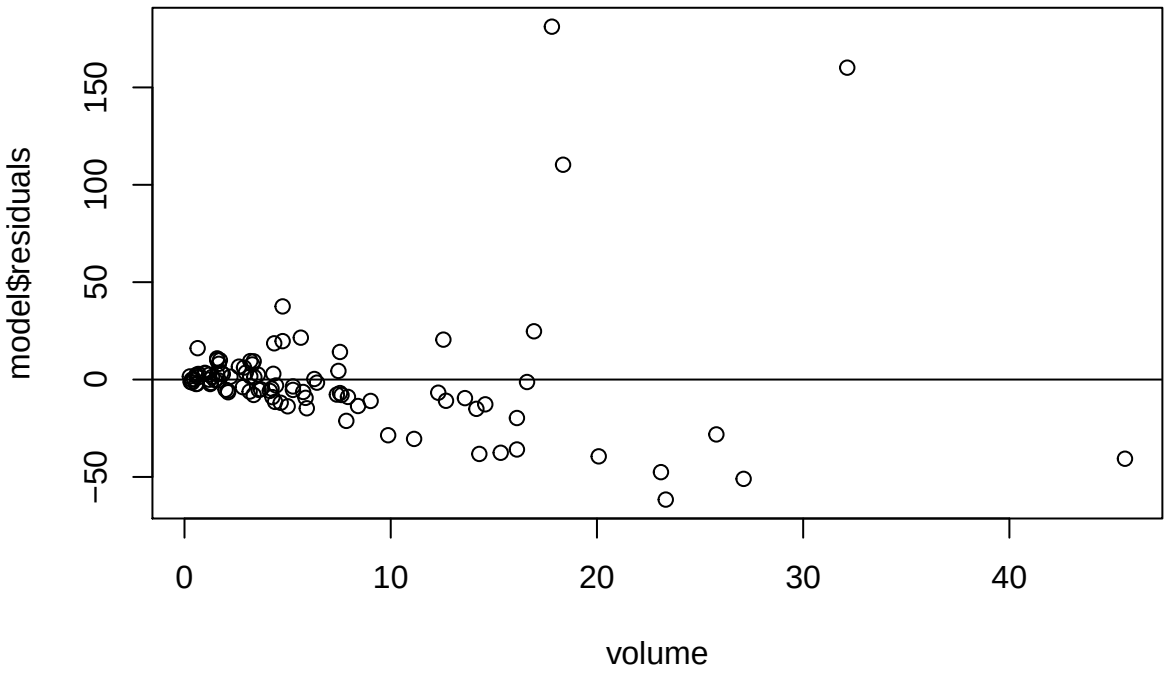
Using the final model based on the log(Y) transformation, the house with X=1100 sqft has predicted price of 142684.5 dollars, and the house with X=4900 sqft has predicted price of 873829.0 dollars.

Problem 3.32 - Prostate Cancer

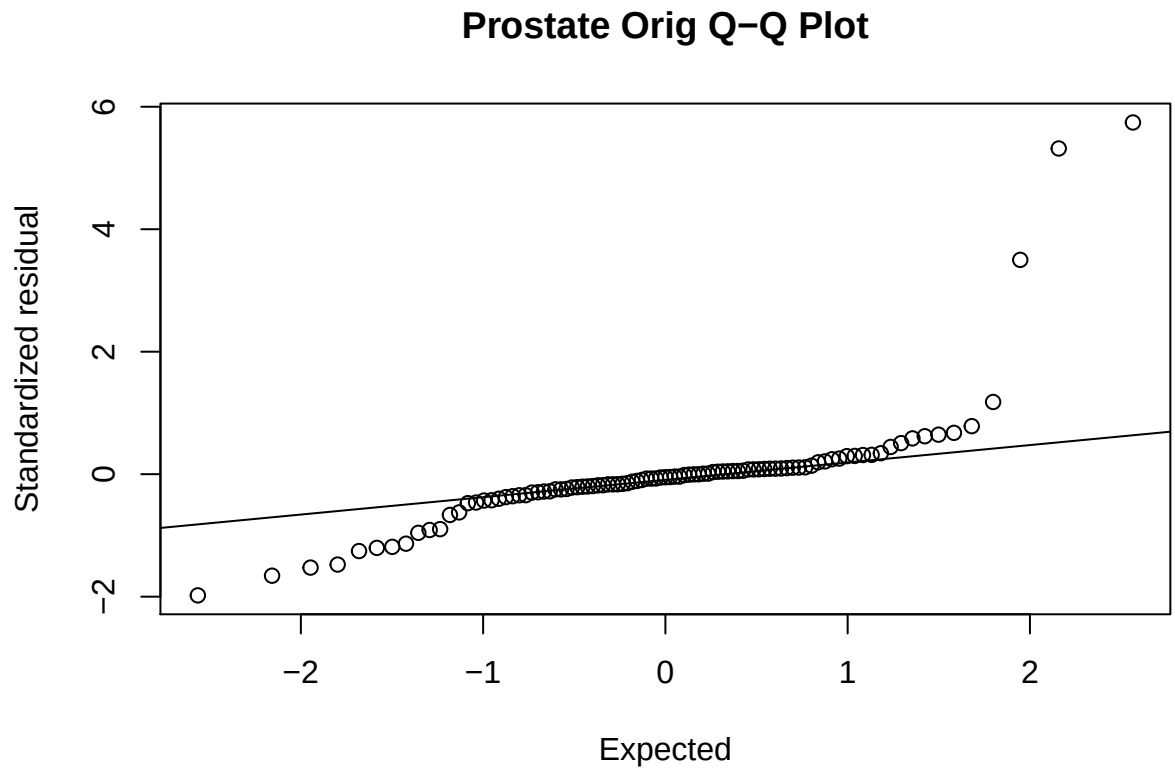
```
# Importing Data
rm(list=ls())
pro.df <- read.table(url("http://users.stat.ufl.edu/~rrandles/sta4210/Rclassnotes/data/textdatasets/Kut
head(pro.df,2)
```

```
##   id PSAlevel volume weight age hyperplasia svInvasion capsularPenetration
## 1  1    0.651 0.5599 15.959  50           0          0                    0
## 2  2    0.852 0.3716 27.660  58           0          0                    0
##   gleasonScore
## 1            6
## 2            7
```

```
attach(pro.df)

# Creating original regression model
model <- lm(PSAlevel ~ volume)

# Diagnostic Visuals

## Residual Plot
plot(volume, model$residuals)
abline(0,0)
```




```
## Normal Probability Plot
model.stan <- rstandard(model)
qqnorm(model.stan, main="Prostate Orig Q-Q Plot", xlab="Expected", ylab="Standardized residual")
qqline(model.stan)
```

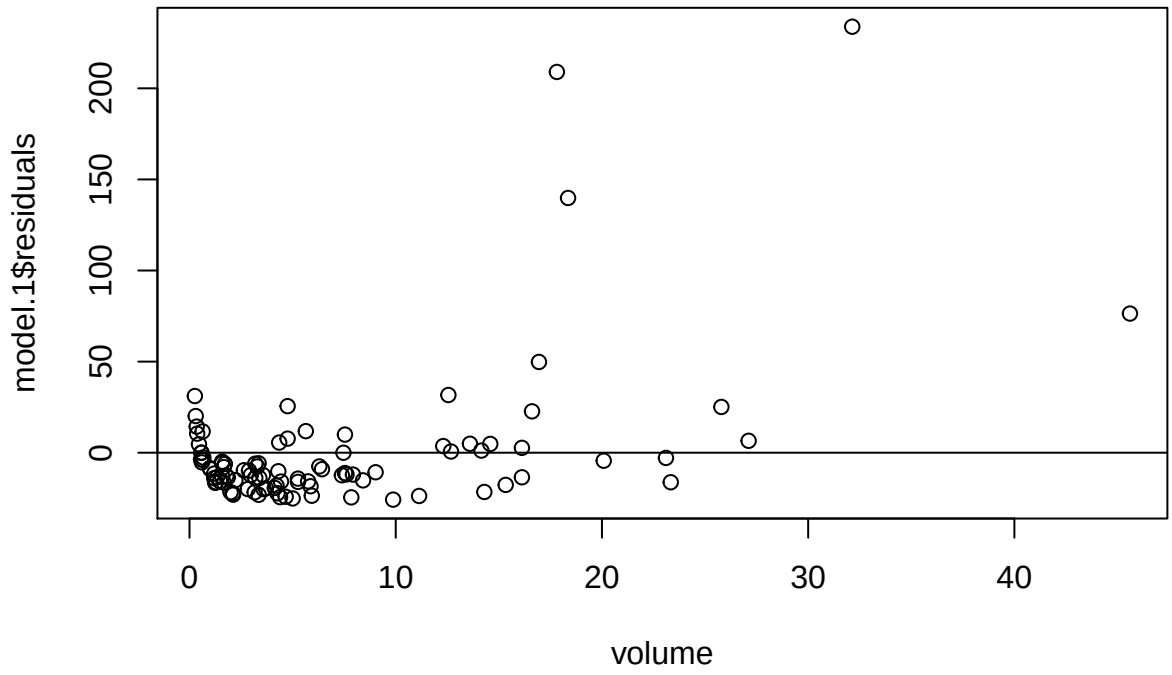


First impressions from the diagnostic visuals, it looks like a linear regression is appropriate for much of the model. However, variance is non-constant, as residuals increase dramatically (with a negative tendency) as cancer volume increases. Furthermore, the tails on the Q-Q plot show non-normality or errors. There is no

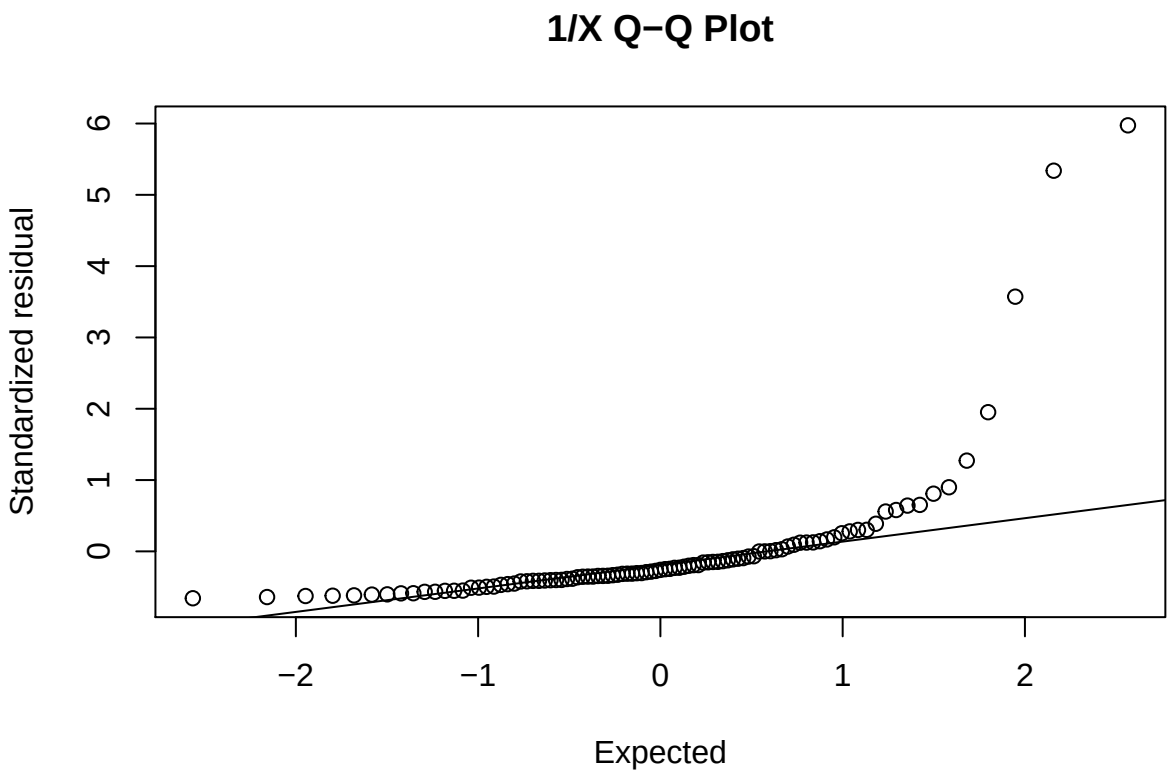
reason to believe errors would be non-independent, so that is a positive.

The lowering slope of the residuals plot looks like the situation on Figure 3.13(c), so I will first attempt a (1/X) transformation:

```
volume.1 <- 1/volume
model.1 <- lm(PSAlevel ~ volume.1)

## Residual Plot
plot(volume, model.1$residuals)
abline(0,0)
```



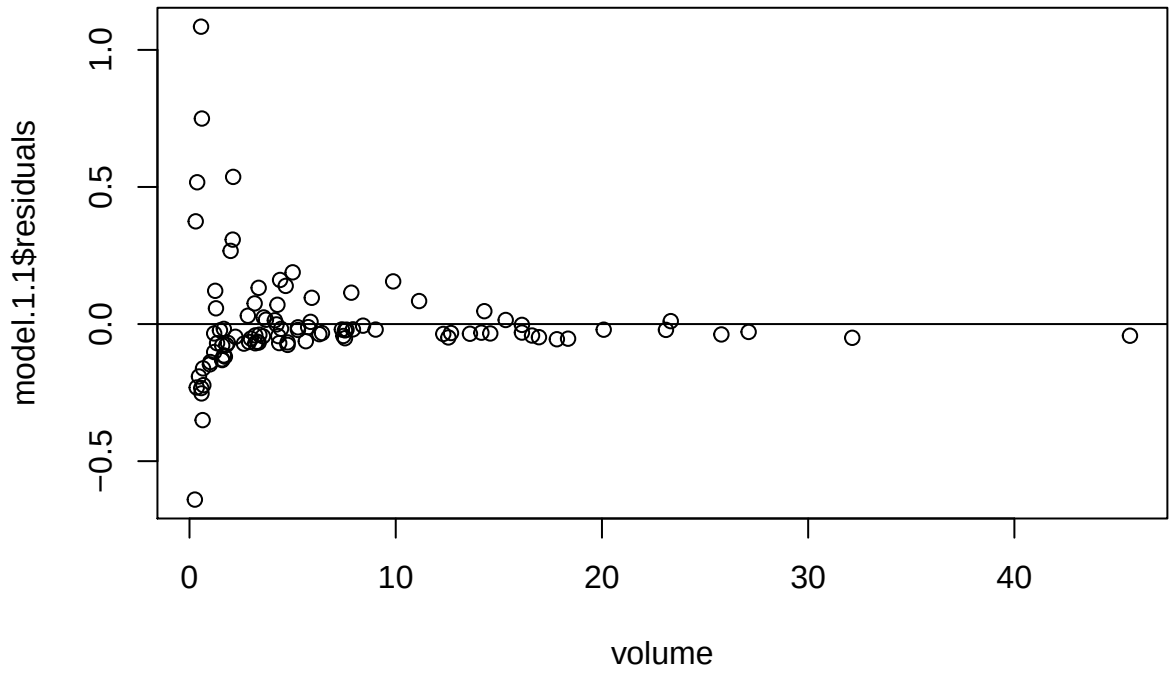

```
## Normal Probability Plot
model.1.stan <- rstandard(model.1)
qqnorm(model.1.stan, main="1/X Q-Q Plot", xlab="Expected", ylab="Standardized residual")
qqline(model.1.stan)
```



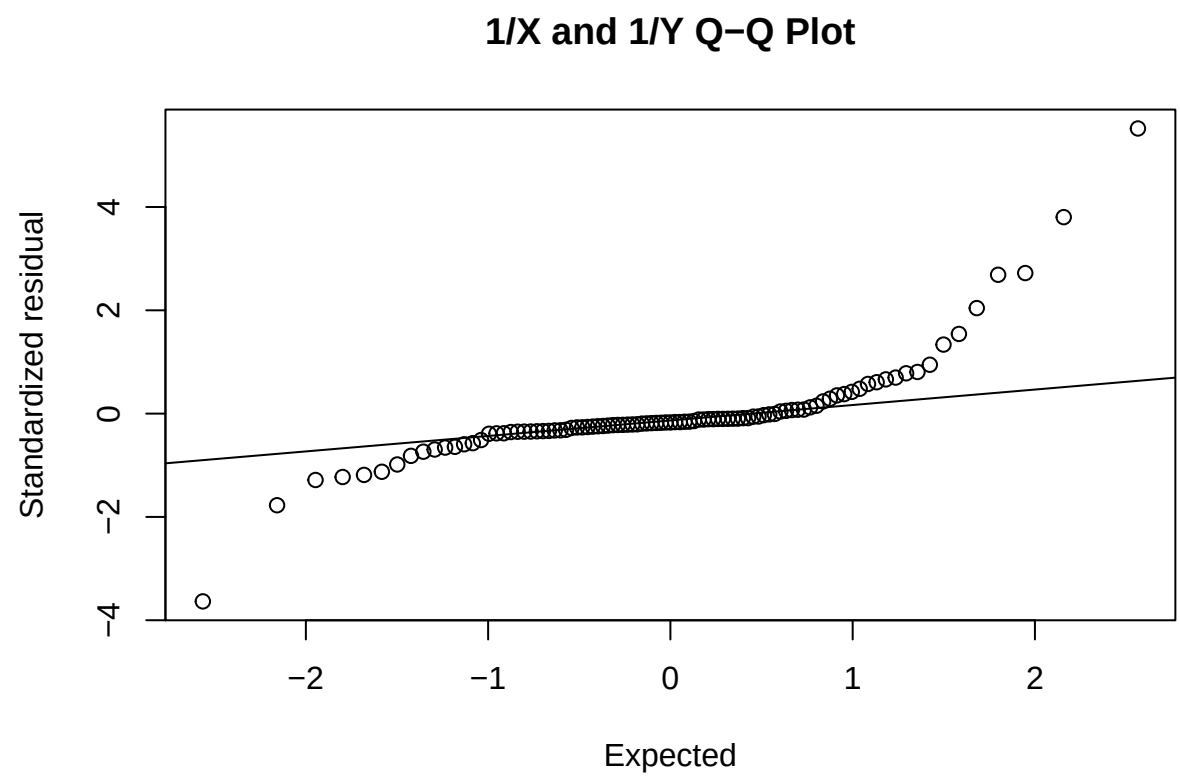
This transformation worked well for 'correcting' the lower residuals. The lower tail on the Q-Q plot became more manageable, while the negative residuals tightened up towards the middle on the residual graph.

Will next try a simultaneous 1/Y transformation:

```
PSAlevel.1 <- 1/PSAlevel
model.1.1 <- lm(PSAlevel.1 ~ volume.1)

## Residual Plot
plot(volume, model.1.1$residuals)
abline(0,0)
```



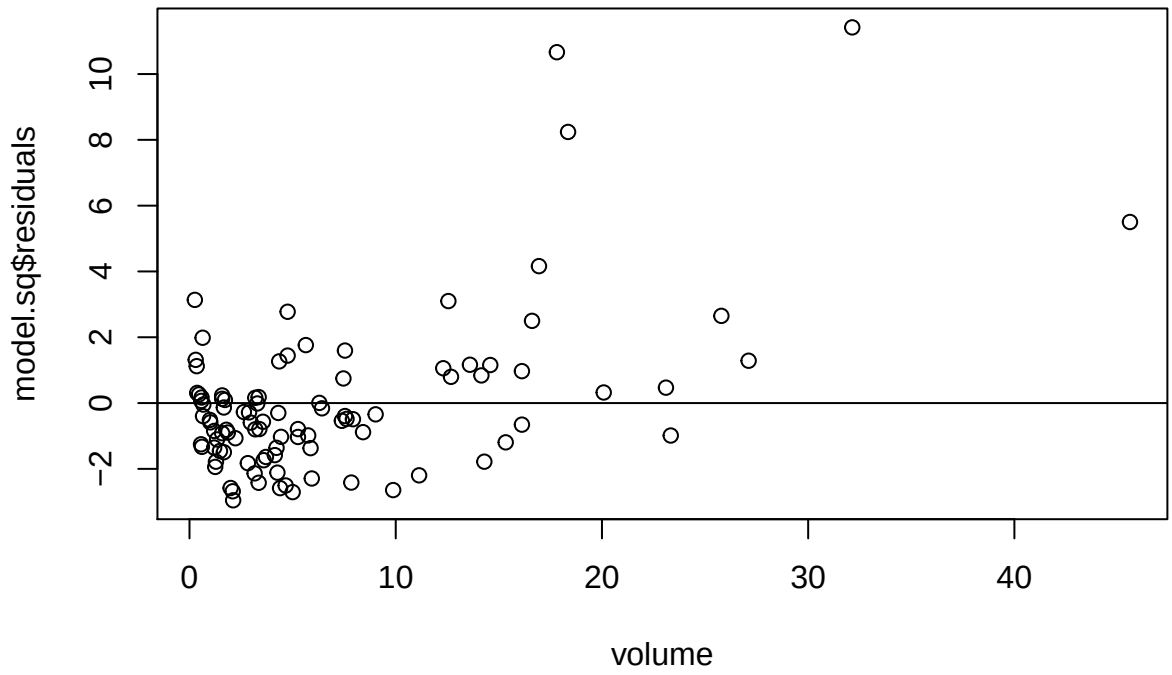

```
## Normal Probability Plot
model.1.1.stan <- rstandard(model.1.1)
qqnorm(model.1.1.stan, main="1/X and 1/Y Q-Q Plot", xlab="Expected", ylab="Standardized residual")
qqline(model.1.1.stan)
```



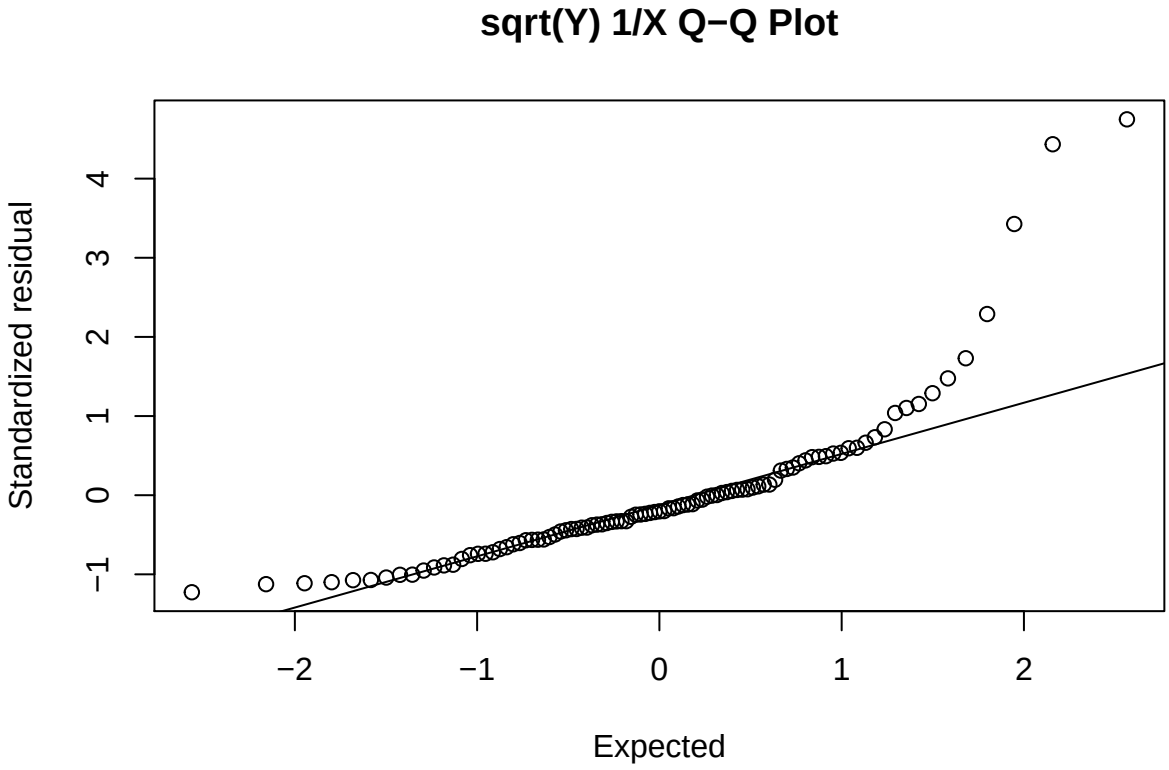
This transformation made the upper tails worse, which was not the intent.

Will next stick with 1/X but try sqrt(Y)

```r
PSA.sq <- sqrt(PSAlevel)
model.sq <- lm(PSA.sq~volume.1)

## Residual Plot
plot(volume, model.sq$residuals)
abline(0,0)
```




```r
## Normal Probability Plot
model.sq.stan <- rstandard(model.sq)
qqnorm(model.sq.stan, main="sqrt(Y) 1/X Q-Q Plot", xlab="Expected", ylab="Standardized residual")
qqline(model.sq.stan)
```



This transformation also did not do the trick on the upper tails. Being that the 1/X transformation was successful, but the transformations on Y were not, the final model will only incorporate the 1/X transformation and leave Y as normal.

Estimating mean PSA level for cancer volume = 20cc using 'model.1':

```
# Setting up new data df... including the 1/X transformation
new.data <- data.frame(volume.1=1/20)

# Running prediction:
final.prediction <- predict(model.1,newdata = new.data)
final.prediction
```

```
##      1
## 30.967
```

Using 'model.1' which was the best model (using the 1/X) transformation, the prediction of a patient with cancer volume = 20cc is PSAlevel = 30.967003