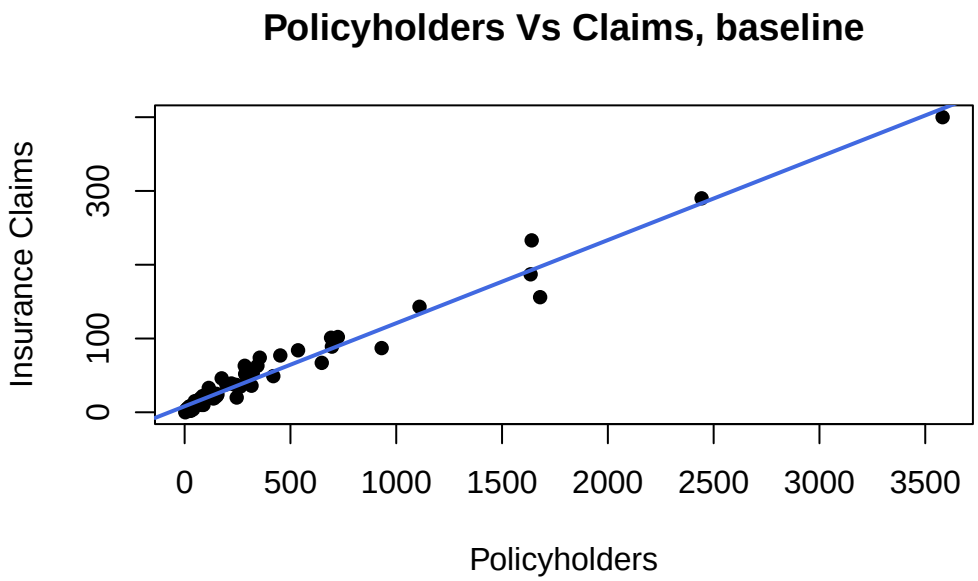
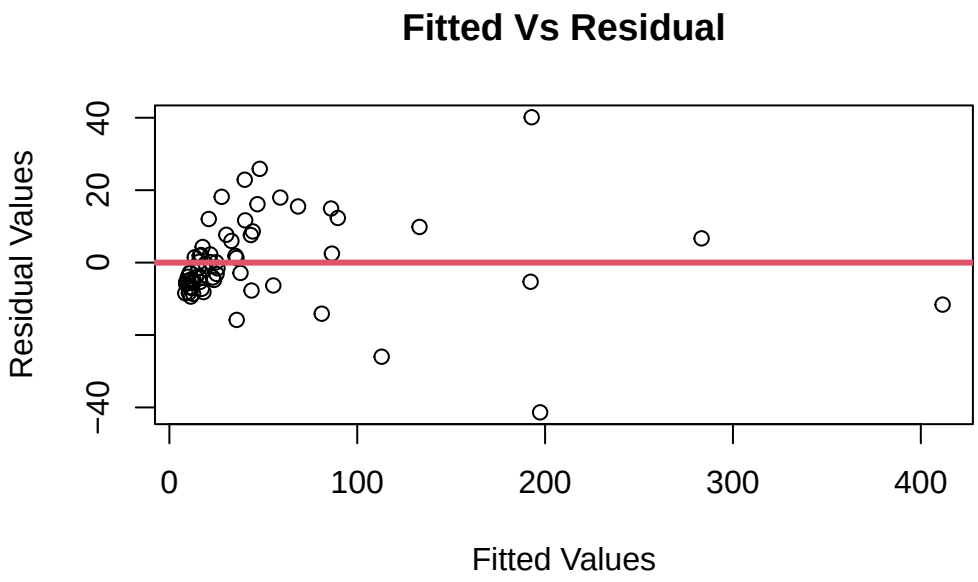
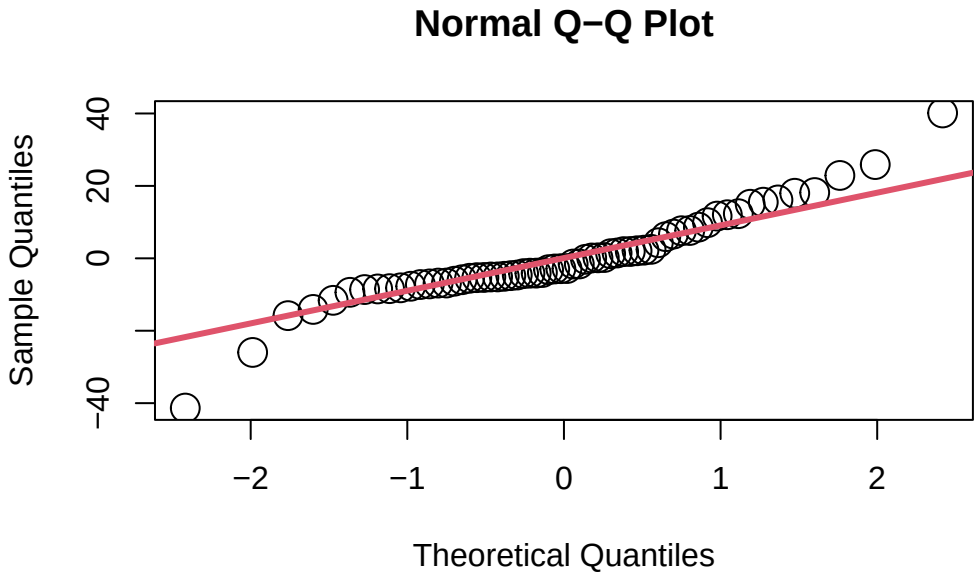
Lab 4

Brad Staples

Question 1




Fitted Vs Residual
Residual Values
40
20
0
−40
0
100
200
300
400
Fitted Values

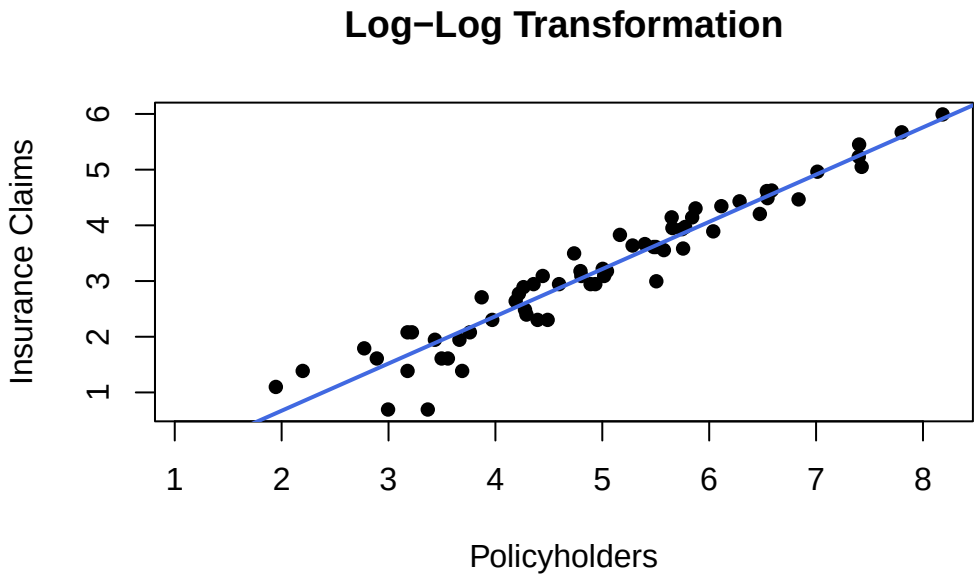

Normal Q−Q Plot
Sample Quantiles
40
20
0
−40
−2
−1
0
1
2
Theoretical Quantiles

1a:

The linearity in this model is pretty good, as most of the points fall reasonably close to the fitted line, so no attention is needed there. The residuals show an immediate and strong fanning pattern, indicating clear heteroscedasticity and the need for a transformation. The QQ-plot is a little suspect and the residuals are right-skewed, suggesting that a transformation is also needed to improve normality. Independence appears to be acceptable as well, with no obvious patterns indicating any grouping or dependence structure.

1b

```
ins2<-Insurance[Insurance$Claims>0,]
plot(log(Holders), log(Claims),
     xlab="Policyholders",
     ylab="Insurance Claims",
     main = "Log-Log Transformation",
     pch=16)
mod2 <- lm(log(Claims) ~ log(Holders), data = ins2)
abline(mod2, col="royalblue", lwd=2)
```



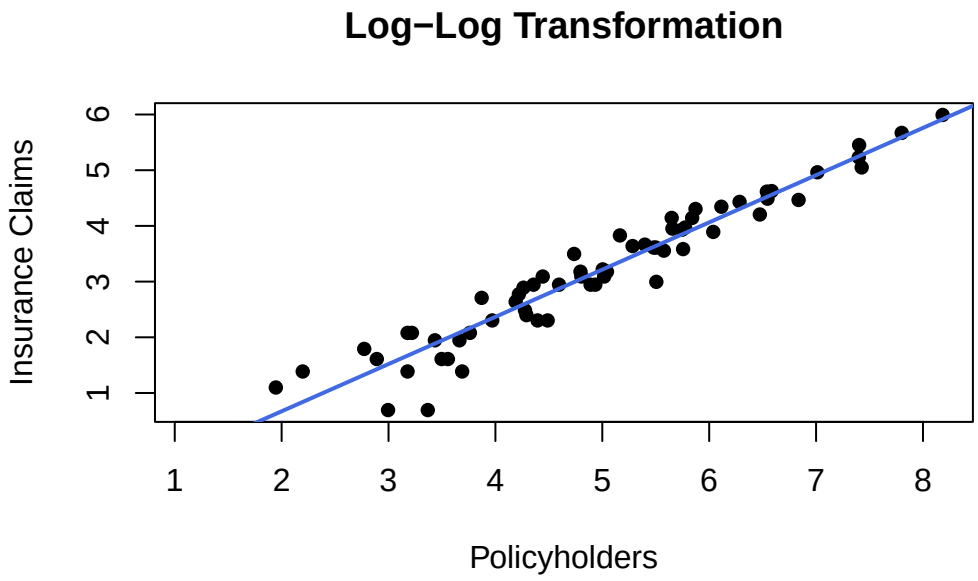
```
summary(mod2)$coef
```

```
                Estimate Std. Error   t value     Pr(>|t|)
(Intercept)  -1.0242040 0.15545373 -6.588481 1.174985e-08
log(Holders)  0.8478836 0.03013842 28.132981 1.230321e-36
```

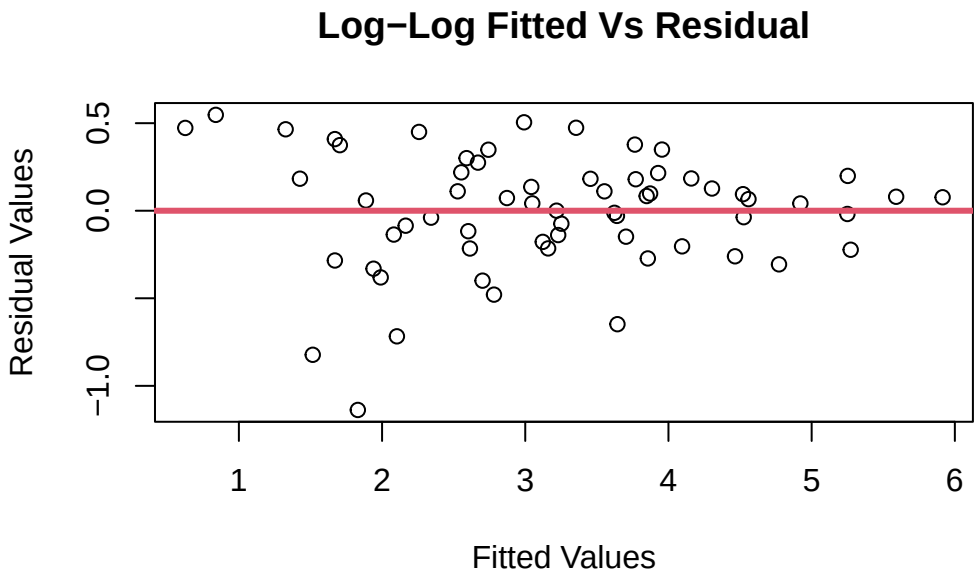
The slope of this model shows that for every 1% increase in policyholders there is a 0.85% expected percent increase in the number of claims filed. This means that claims grow slightly less proportionally than policyholders which is helpful for estimating premiums and planning for the future.

1c

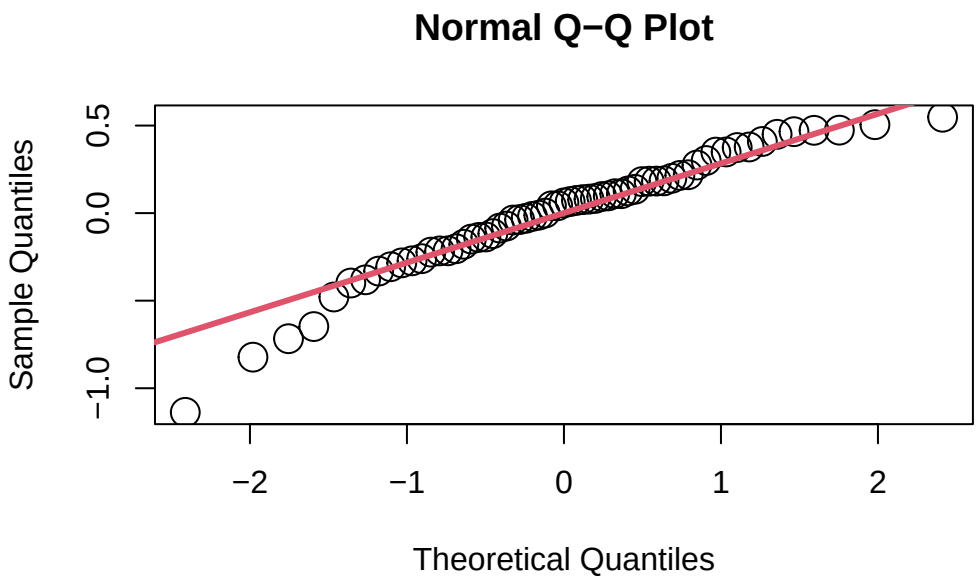
```
plot(log(Holders), log(Claims),
     xlab="Policyholders",
     ylab="Insurance Claims",
     main = "Log-Log Transformation",
     pch=16)
mod2 <- lm(log(Claims) ~ log(Holders), data = ins2)
abline(mod2, col="royalblue", lwd=2)
```



```r
plot(mod2$fitted.values, mod2$residuals,
     main="Log-Log Fitted Vs Residual",
     xlab="Fitted Values",
     ylab="Residual Values")
abline(h=0, col=2, lwd=3)
```




```r
qqnorm(mod2$residuals, cex=2)
qqline(mod2$residuals, col=2, lwd=3)
```




```
summary(mod2)$coef
```

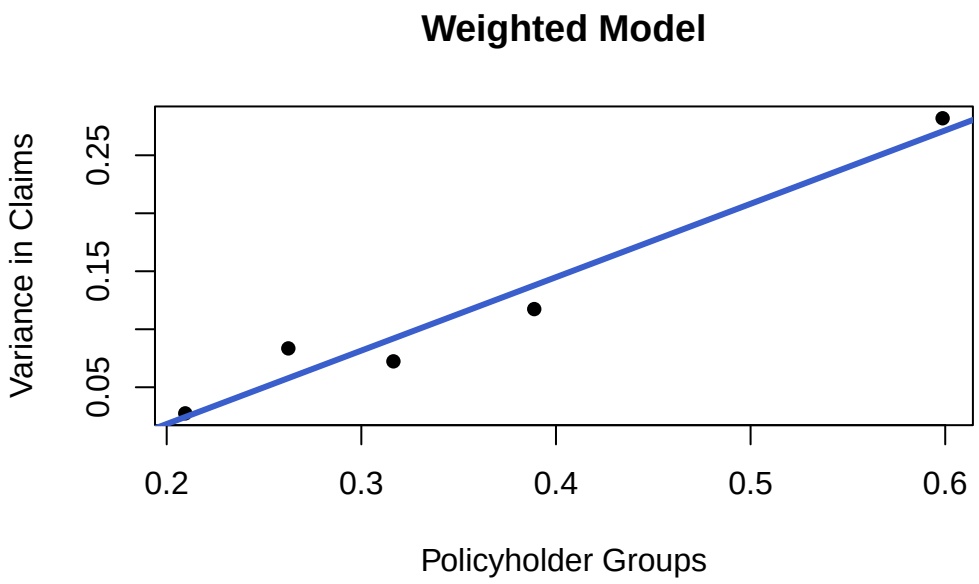
```
               Estimate Std. Error   t value     Pr(>|t|)
(Intercept)  -1.0242040 0.15545373 -6.588481 1.174985e-08
log(Holders)  0.8478836 0.03013842 28.132981 1.230321e-36
```

Linearity in this model looks great, with more of the points spread along the regression line. Homoscedasticity has improved substantially compared to the original and despite the points not being super close to the center, there is no noticeable fanning or patterns. Normality looks a lot better than before, with the Q–Q plot more closely aligned along the diagonal and without the wiggles seen in the pre-transformed model. Independence was not a concern before and is not one now.

1d

```
fv<-mod2$fitted.values
grps<-(fv>4.133775)+(fv>3.566796)+(fv>2.773839)+(fv>2.090657)

grp.var<-by(mod2$residuals, grps, var)
grp.med<-by(mod2$fitted, grps, median)
```

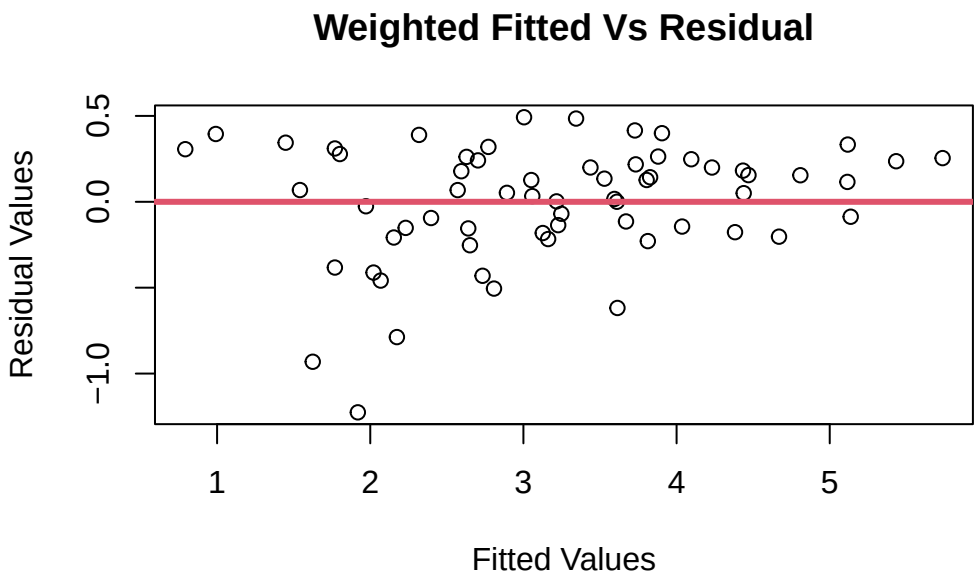
```r
plot(1/grp.med, grp.var, pch=16, xlab="Policyholder Groups", ylab="Variance in Claims",
     main="Weighted Model")
var.mod<-lm(grp.var~I(1/grp.med))
abline(var.mod, col="royalblue3", lwd=3)
```



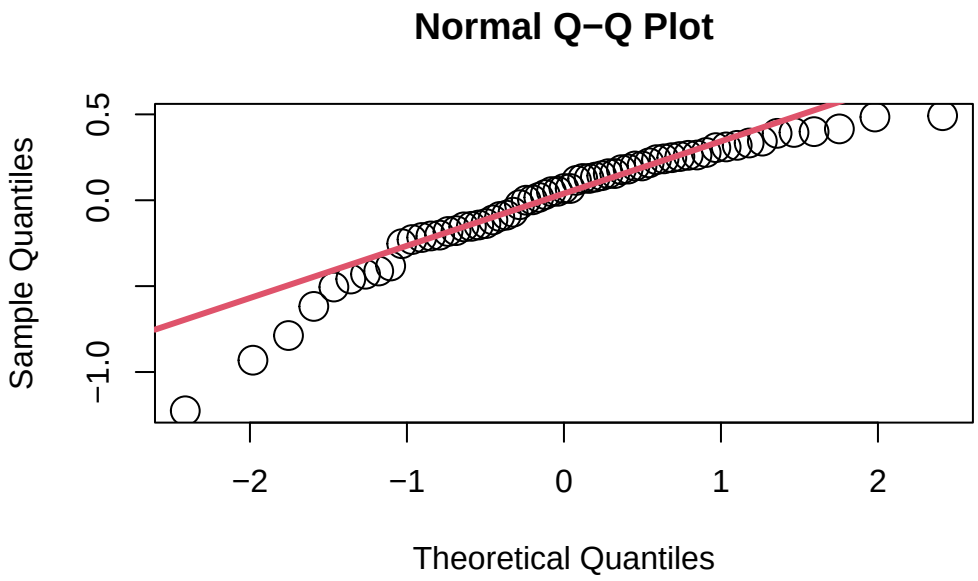

```r
var.est<-var.mod$coef[1]+(var.mod$coef[2]*mod2$fitted.values)

mod3<-lm(I(log(ins2$Claims))~I(log(ins2$Holders)),weights=1/var.est)
summary(mod3)$coef
```

```
                      Estimate Std. Error   t value     Pr(>|t|)
(Intercept)          -0.7502853 0.14877015 -5.043252 4.380413e-06
I(log(ins2$Holders))  0.7927096 0.03343807 23.706797 1.776399e-32
```

```r
plot(mod3$fitted.values, mod3$residuals,
     main="Weighted Fitted Vs Residual",
     xlab="Fitted Values",
     ylab="Residual Values")
abline(h=0, col=2, lwd=3)
```




```
qqnorm(mod3$residuals, cex=2)
qqline(mod3$residuals, col=2, lwd=3)
```



In the weighted model the new slope is 0.7927096 and the new intercept is -0.7502853 which is not vastly different from our log log slope of 0.8478836 and intercept of -1.024204. The residuals and normality given by their respetive plots look very similar to the log-log model, since the weighted model was based on it. Linearity remains well balanced, and independence is even less of a concern with the defined groups. Overall, all assumptions appear to be valid for this weighted model

1e

Confidence Interval

```
predict(mod2, newdata=data.frame(Holders=700), interval="confidence", level=0.98)
```

```
      fit      lwr      upr
1 4.53035 4.378048 4.682651
```

Prediction Inverval

```
predict(mod2, newdata=data.frame(Holders=80), interval="prediction", level=0.98)
```

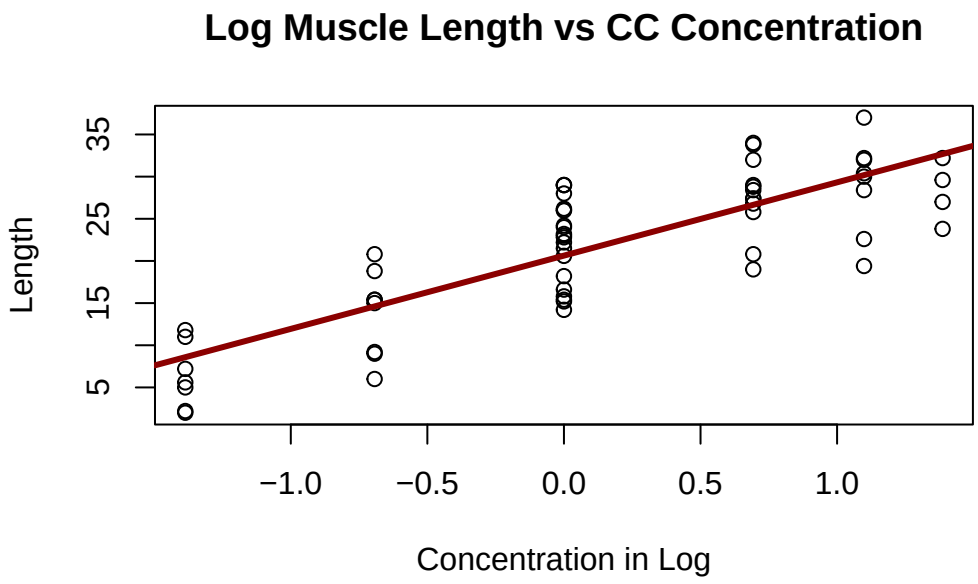
```
       fit     lwr     upr
1 2.691245 1.88427 3.49822
```

Question 2

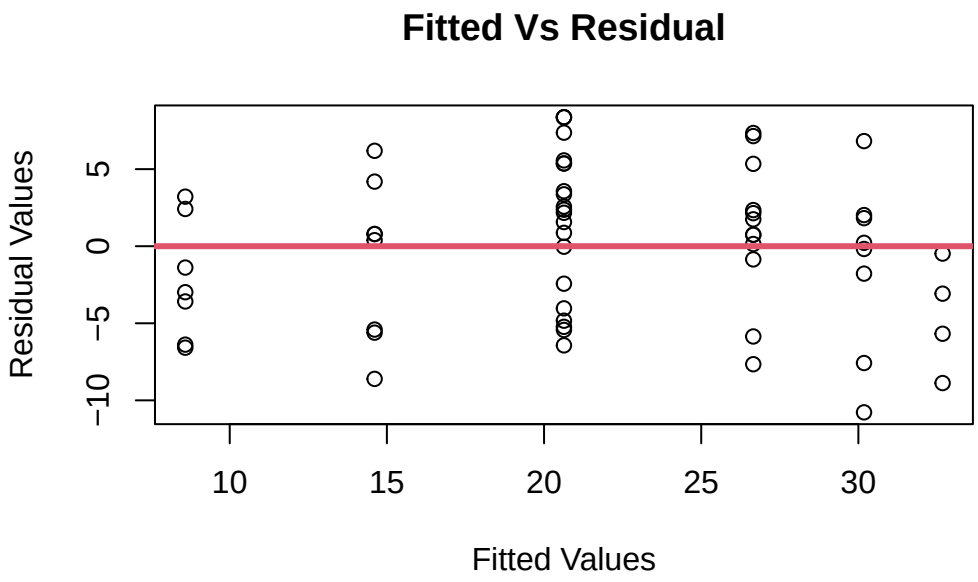
```
library(MASS)
attach(muscle)
```

2a

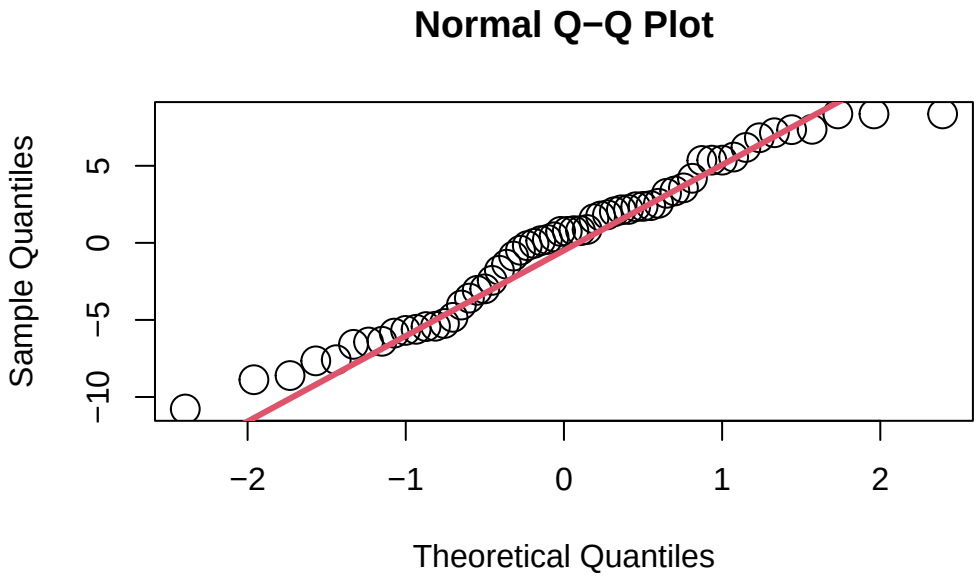
```r
plot(log(Conc), Length,
     xlab="Concentration in Log",
     ylab="Length",
     main="Log Muscle Length vs CC Concentration")
reg5<-lm(Length~log(Conc))
abline(reg5, col="darkred", lwd=3)
```




```r
plot(reg5$fitted.values, reg5$residuals,
     main="Fitted Vs Residual",
     xlab="Fitted Values",
     ylab="Residual Values")
abline(h=0, col=2, lwd=3)
```




```
qqnorm(reg5$residuals, cex=2)
qqline(reg5$residuals, col=2, lwd=3)
```



I considered both the log and square root transformations for calcium concentration, as the regression models for both are very similar. The log model fits slightly closer to the regression line overall than the square root model but the difference is very minimal. I favored the log transformation mainly because the fitted vs. residual plot shows a slightly better spread of points around zero compared to the square root's fitted vs. residual plot, but the difference is still only slight. While log leads to an overall better transformation, square root has some validity as well.

2b

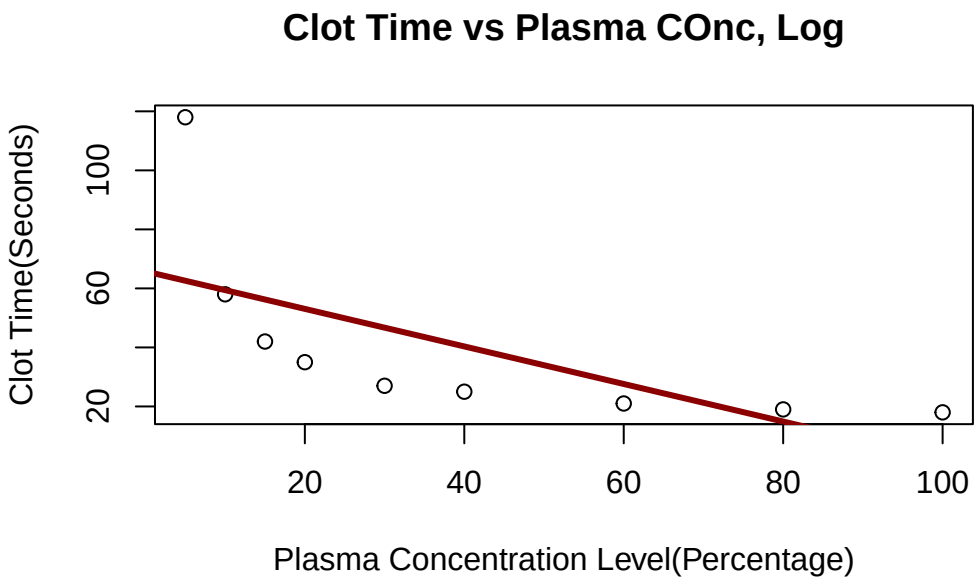
Using a log model, we expect a relatively large change in muscle length when the concentration is between 1.0 and 2.0 and a noticeable decrease in muscle length when calcium chloride when concentrations between 2.0 and 3.0. This is very visible on the graph with our extrapolated points being far below the regression line on the tail ends of the plot. Because of the way log functions, the expected change in muscle length is not constant across the concentration range. Increases at lower concentrations have a much larger impact than higher concentrations.

2c

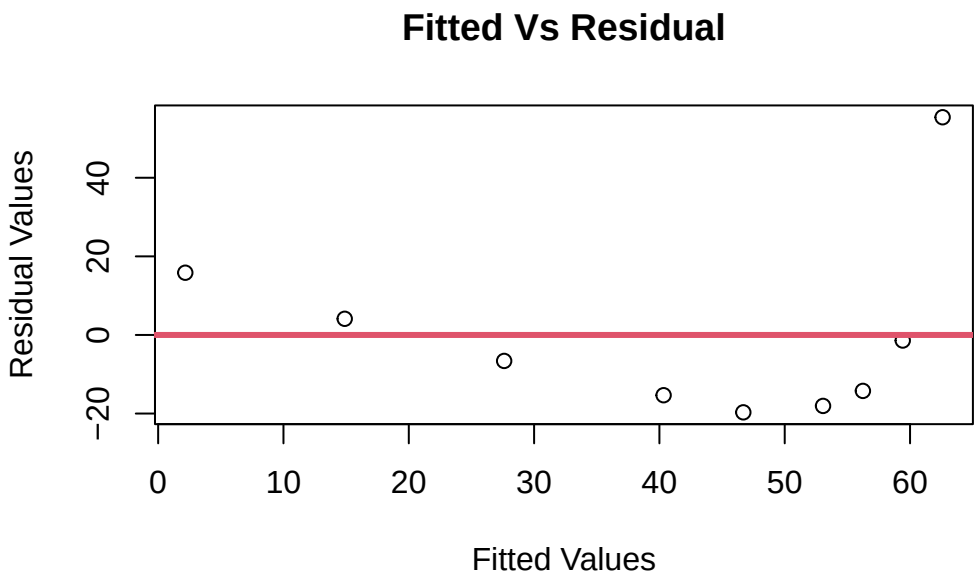
Linearity in the model is still somewhat rough, as many points lie far from the regression line. Homoscedasticity is moderately acceptable; while there is a slight fanning pattern in the fitted vs. residual plot, it is less pronounced than in the original model. If we generate a QQ-plot we see a large deviation from the regression line throughout, meaning normality an area of major concern.the grouping of the data points suggests a potential lack of independence, which could be contributing to the other issues observed in the regression. Overall there are improvements to linearity and homoscedasticity but normality and indepence remain pain points.

Question 3

```
library(faraway)
attach(clot)
plot(conc[lot=="one"], time[lot=="one"],
     xlab = "Plasma Concentration Level(Percentage)",
     ylab = "Clot Time(Seconds)",
     main = "Clot Time vs Plasma COnc, Log")
blood1<-lm(time[lot=="one"]~conc[lot=="one"])
abline(blood1, col="darkred", lwd=3)
```

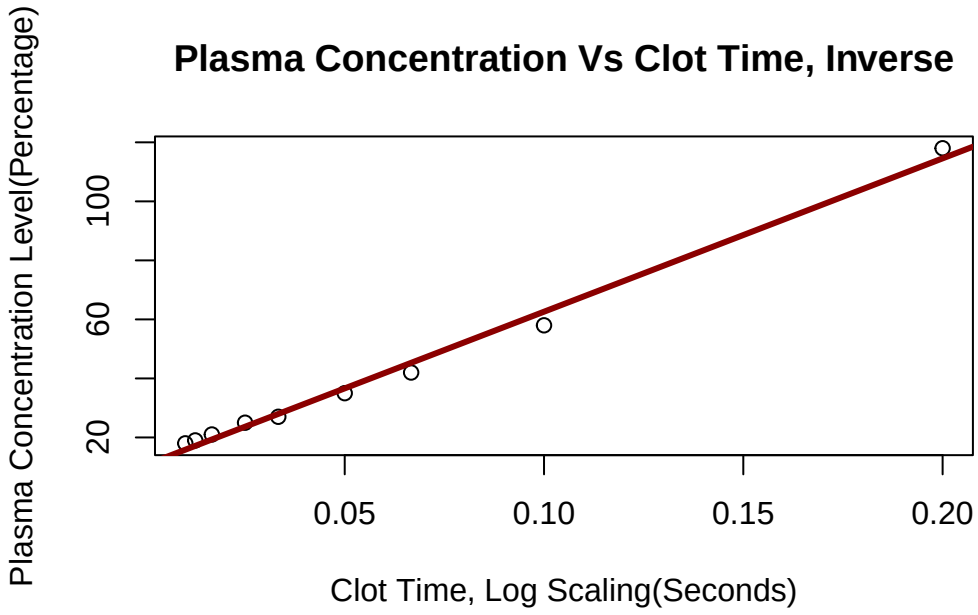



```
plot(blood1$fitted.values, blood1$residuals,
     main="Fitted Vs Residual",
     xlab="Fitted Values",
     ylab="Residual Values")
abline(h=0, col=2, lwd=3)
```

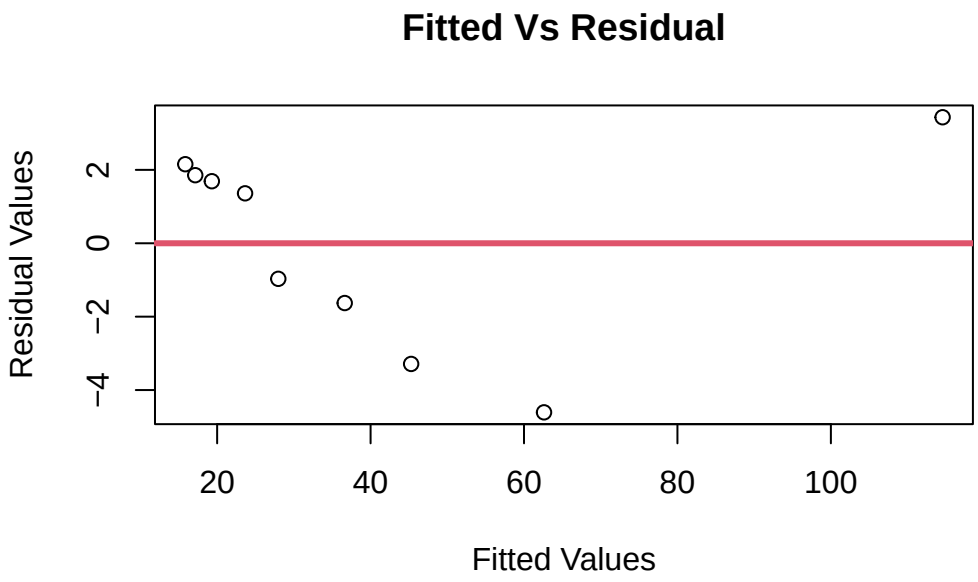


3a

```
plot(I(1/conc[lot=="one"]), time[lot=="one"],
     xlab = "Clot Time, Log Scaling(Seconds)",
     ylab = "Plasma Concentration Level(Percentage)",
     main="Plasma Concentration Vs Clot Time, Inverse")
blood4<-lm(time[lot=="one"]~I(1/conc[lot=="one"]))
abline(blood4, col="darkred", lwd=3)
```




```
plot(blood4$fitted.values, blood4$residuals,
     main="Fitted Vs Residual",
     xlab="Fitted Values",
     ylab="Residual Values")
abline(h=0, col=2, lwd=3)
```



Inverse is by far the best transformation for this plot, linearity is very good in the inverse transformation with points aligning much more closely along the regression line compared to the other transformations. While Homoscedaticty is not great in the fitted vs residual plot, none of these did much better than inverse. Normality is going to be rough for all the models but no other transformation improves it more substantially. Overall inverse does the best job with at least satisfying linearity and outperforming the alternatives.

3b

```
summary(blood4)$coef
```

```
                         Estimate Std. Error   t value     Pr(>|t|)
(Intercept)              10.65001   1.377308  7.732484 1.131840e-04
I(1/conc[lot == "one"]) 519.57842  16.980011 30.599416 1.027351e-08
```

In the inverse model the new slope is 519.5784161 and the new intercept is 10.6500109

3c

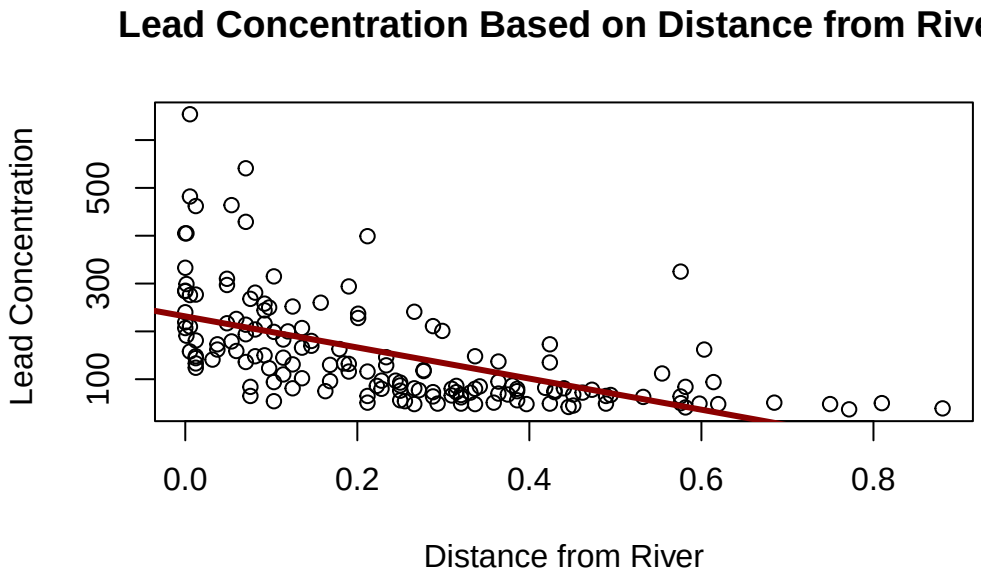
```r
lot1 <- clot[lot=="one", ]
lot1$invConc <- 1/lot1$conc
blood4<-lm(time ~ invConc, data = lot1)
predict(blood4, newdata = data.frame(invConc = 1/50), interval="confidence", level=0.95)
```

```
       fit      lwr      upr
1 21.04158 18.29064 23.79252
```

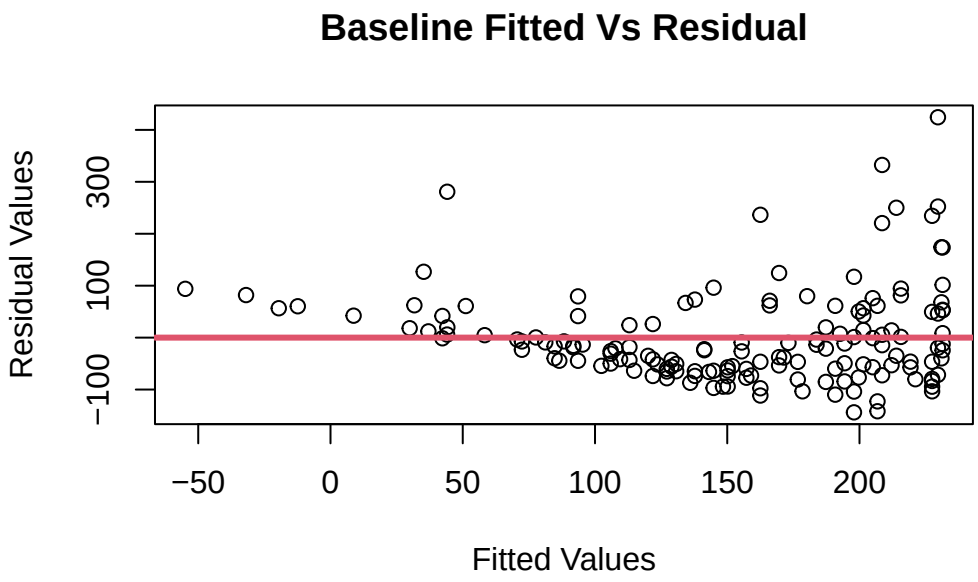
Using the inverse transformed model, the estimated average clotting time at a plasma concentration of 50% is 21.04518 seconds. The 90% confidence interval for this mean ranges from 18.29064 to 23.79252 seconds, meaning we are 95% confidence the mean clotting time at 50% concentration will be within this interval

Question 4

```r
library(sp)
data(meuse)
attach(meuse)
plot(dist, lead,
     main="Lead Concentration Based on Distance from River",
     xlab="Distance from River",
     ylab="Lead Concentration ")
model1<-lm(lead~dist, data=meuse)
abline(model1, col="darkred", lwd=3)
```

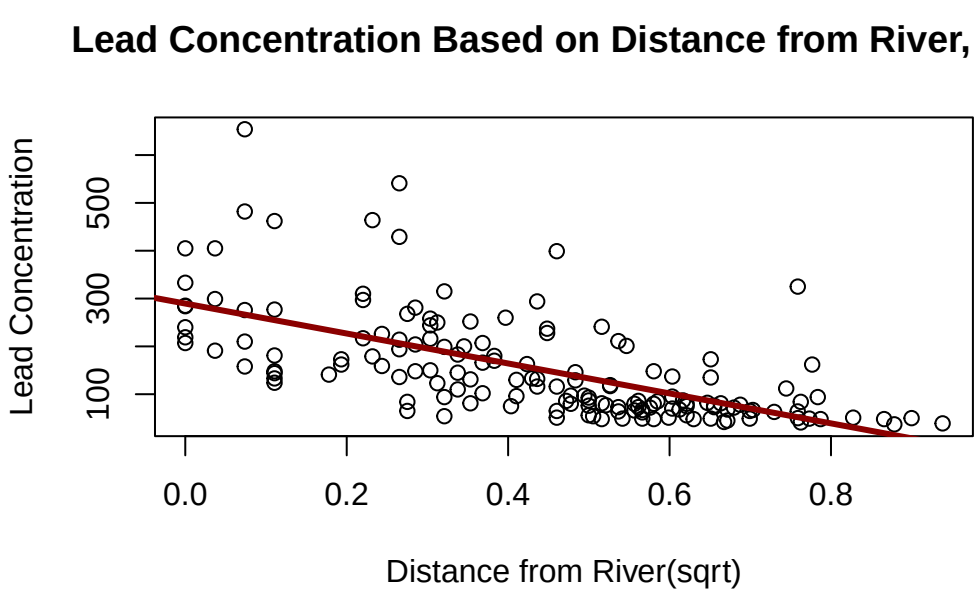



```
plot(model1$fitted.values, model1$residuals,
     main="Baseline Fitted Vs Residual",
     xlab="Fitted Values",
     ylab="Residual Values")
abline(h=0, col=2, lwd=3)
```

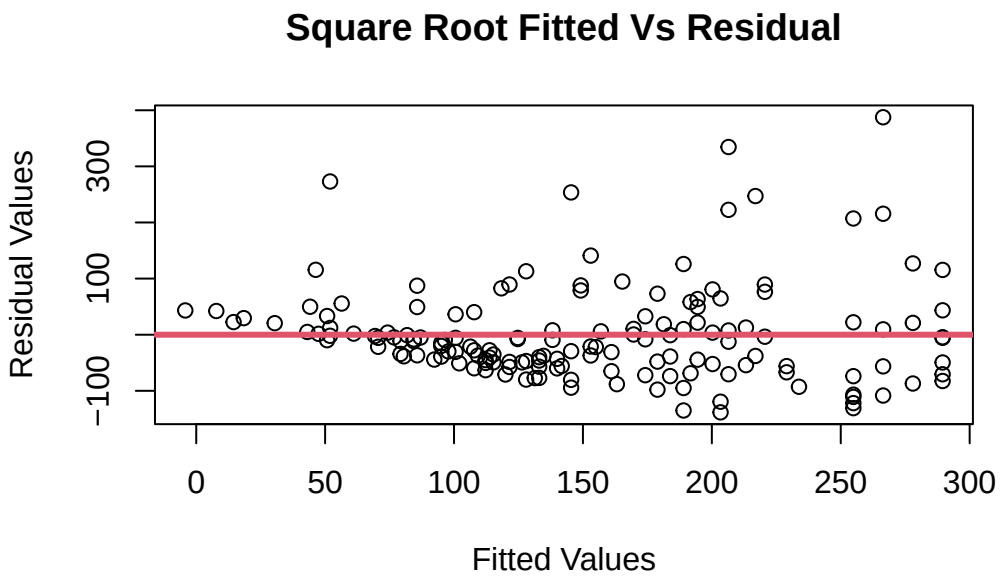


4a

```
plot(sqrt(dist), lead,
     main="Lead Concentration Based on Distance from River, Sqrt",
     xlab="Distance from River(sqrt)",
     ylab="Lead Concentration")
model3<-lm(lead~sqrt(dist))
abline(model3, col="darkred", lwd=3)
```



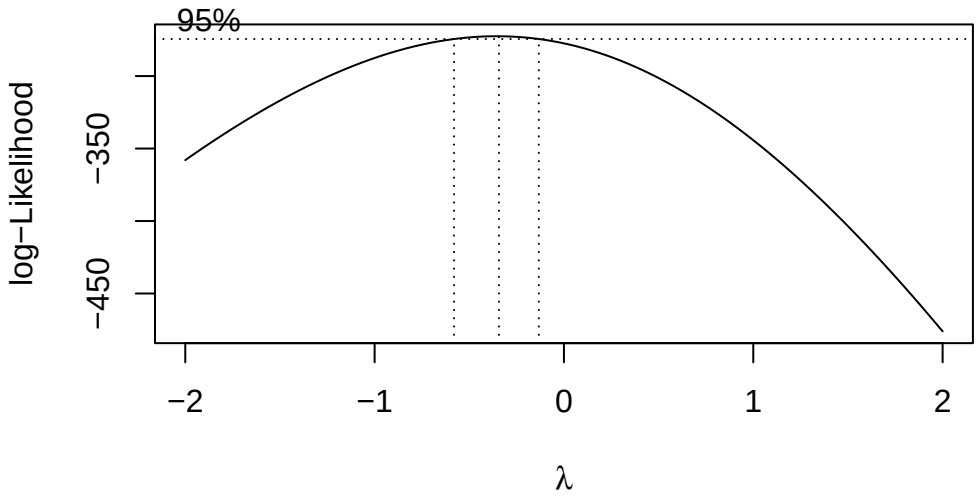

```
plot(model3$fitted.values, model3$residuals,
     main="Square Root Fitted Vs Residual",
     xlab="Fitted Values",
     ylab="Residual Values")
abline(h=0, col=2, lwd=3)
```



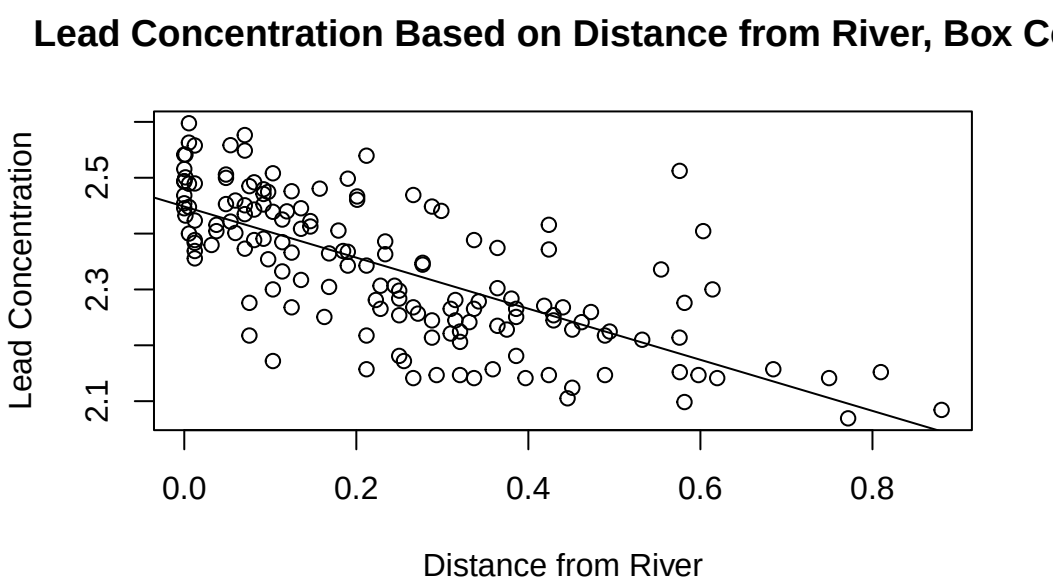
I prefer the square root transformation for modeling lead concentrations as a function of distance from the river. This transformation improves linearity compared to the baseline model, with the points more evenly distributed along the regression line and a longer fitted line that better captures the trend. The residual plot shows a spread that is more centered across the x-axis with overall less heteroscedasticity than the untransformed model.

4b

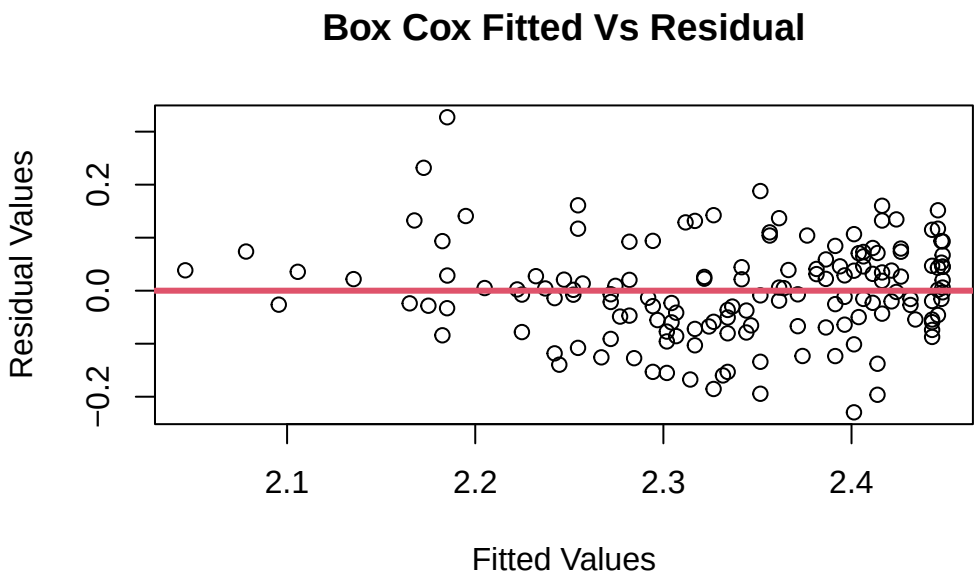
```
bc<-boxcox(model1)
```




```
lambda <- bc$x[which.max(bc$y)]
lead_trans<-(lead^lambda-1)/lambda
model6<-lm(lead_trans~dist)
plot(dist, lead_trans,
     main="Lead Concentration Based on Distance from River, Box Cox",
     xlab="Distance from River",
     ylab="Lead Concentration")
abline(model6)
```



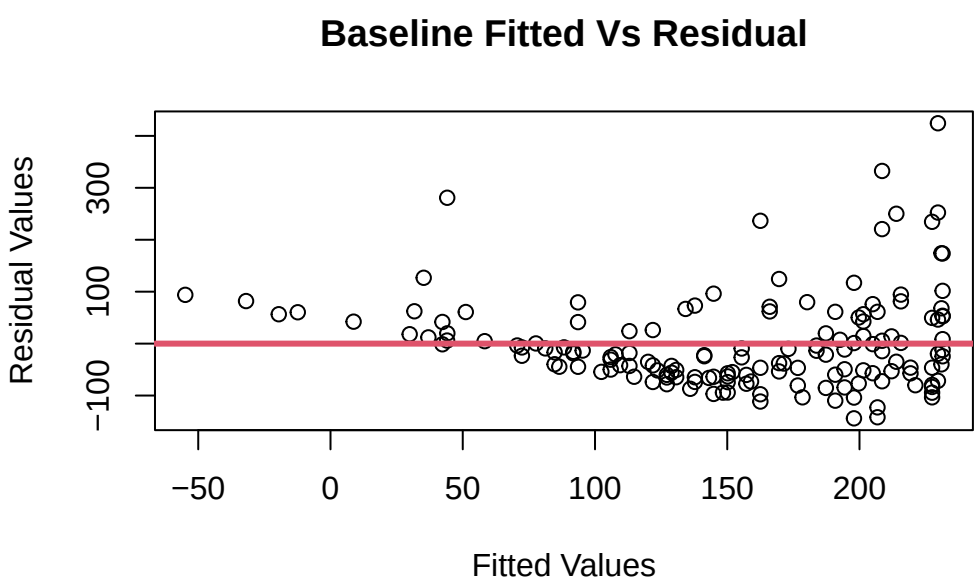

```
plot(model6$fitted.values, model6$residuals,
     main="Box Cox Fitted Vs Residual",
     xlab="Fitted Values",
     ylab="Residual Values")
abline(h=0, col=2, lwd=3)
```



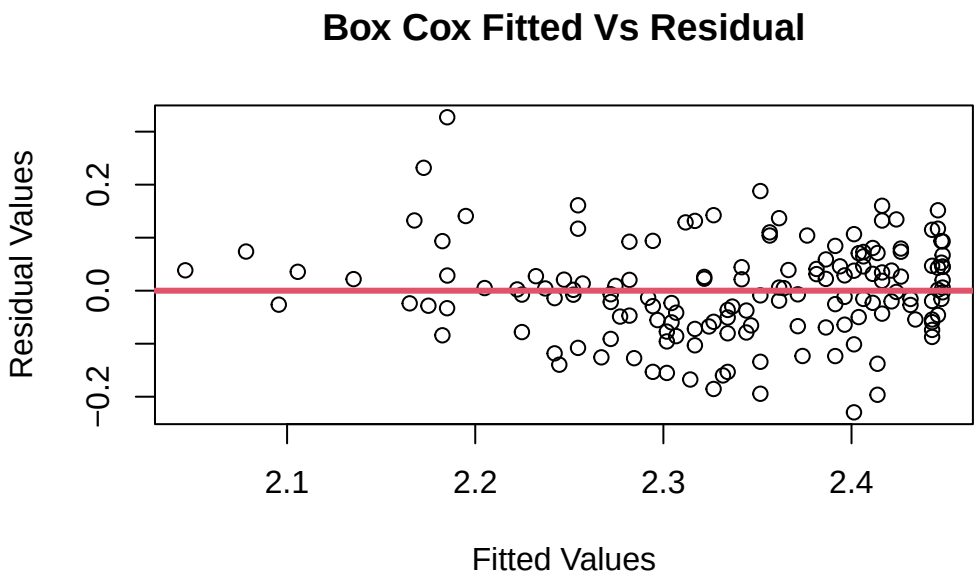
Box Cox easily outperforms other transformation, providing a much better linear fit than any of the previous models. The data is more evenly spread along the fitted line and the spread of values is more uniforms. The improvements are also reflected in the fitted vs residual plot, and while there is still heteroscedasticity, there is still a better spread around zero than compared to previous models.

4c

```
plot(model1$fitted.values, model1$residuals,
     main="Baseline Fitted Vs Residual",
     xlab="Fitted Values",
     ylab="Residual Values")
abline(h=0, col=2, lwd=3)
```




```
plot(model6$fitted.values, model6$residuals,
     main="Box Cox Fitted Vs Residual",
     xlab="Fitted Values",
     ylab="Residual Values")
abline(h=0, col=2, lwd=3)
```



The box cox model does a good job in spreading out the concentration of data points and reducing patterns in the overall spread. The baseline plot shows a large concentration of datapoints below the middle line, with a slight curve shape to the data, but once we apply box cox we get a fitted vs residual plot without a curving pattern and a better spread of datapoints above and below the median line.

4d 95% Condfidence Interval

```
conft<-predict(model6,list(dist=c(.25, .7, 1)),interval="confidence", level=0.95)
conft_norm<-(lambda * conft + 1)^(1/lambda)
conft_norm
```

```
        fit       lwr      upr
1 111.04028 103.40846 119.4484
2  45.72974  40.08394  52.4977
3  28.58685  24.00618  34.4231
```