excercises_week_5

Lindsey Greenhill

2/24/2021

excercises 4.4 and 4.8

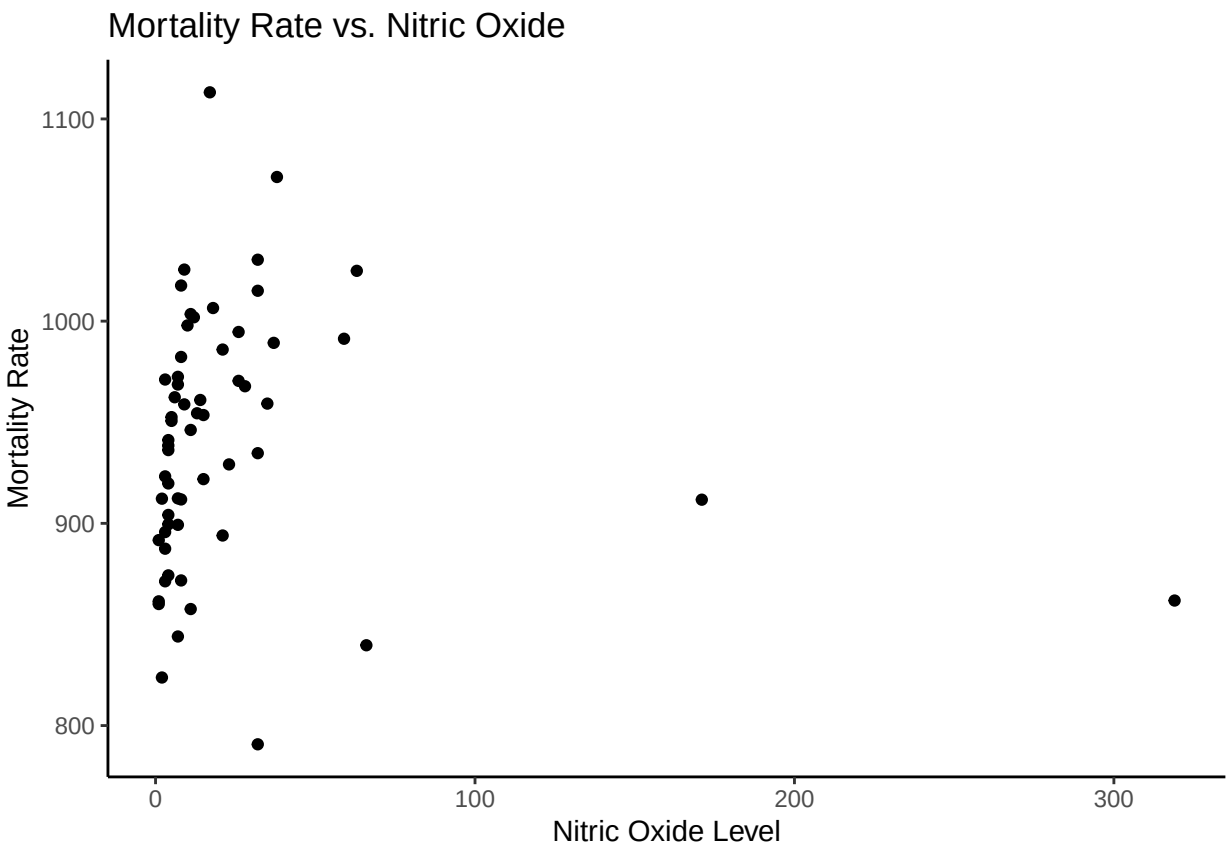
Question 3

Part a

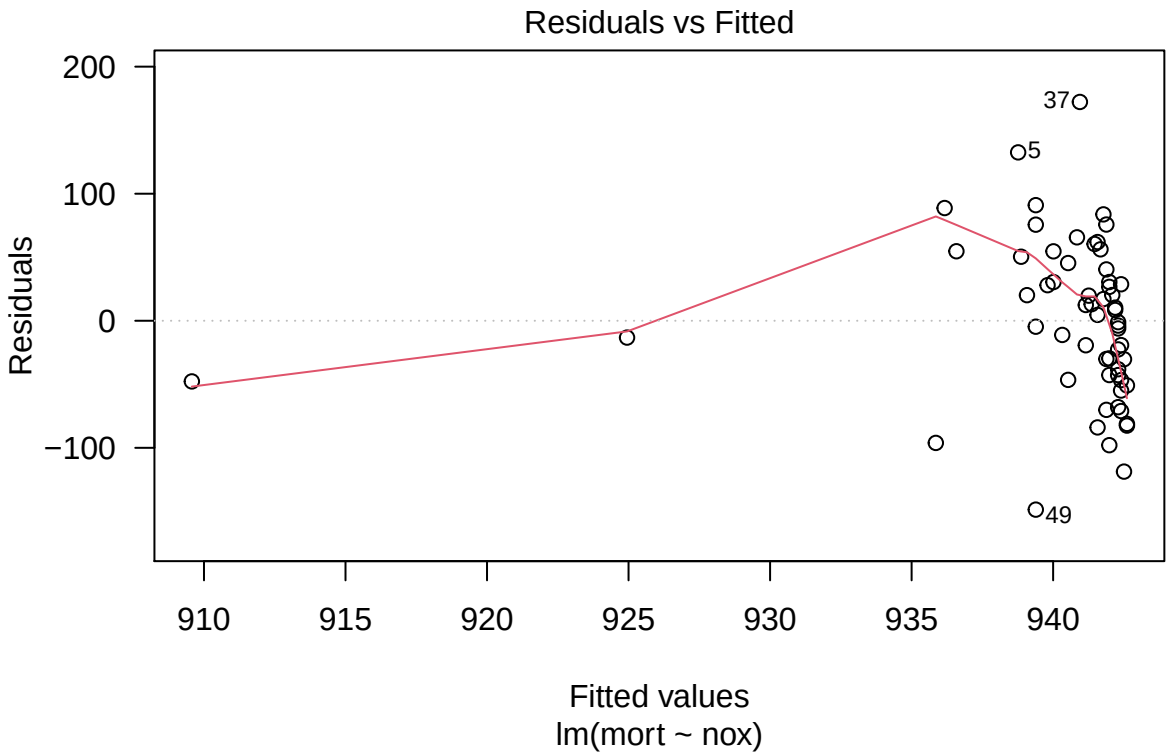
The scatterplot we created of Mortality rate vs. Nitric Oxide suggests that a linear regression without any transformations will not fit the data very well, as many of the data points are clustered around the start of the range and only a few data points appear around the end of the range.

Fitting a linear regression (fit_1) and looking at its residuals confirms that a linear regression may not be the best fit for this data. The residuals are not randomly distributed and are clustered at the end of the range, telling us that the linear regression is not a good fit.

```
# creating scatterplot. Data looks like a lot of the data is very clustered
# around 0 with some values much further away.

pollution %>%
  ggplot(aes(x = nox, y = mort)) +
  geom_point() +
  labs(title = "Mortality Rate vs. Nitric Oxide",
       x =  "Nitric Oxide Level",
       y =  "Mortality Rate") +
  theme_classic()
```



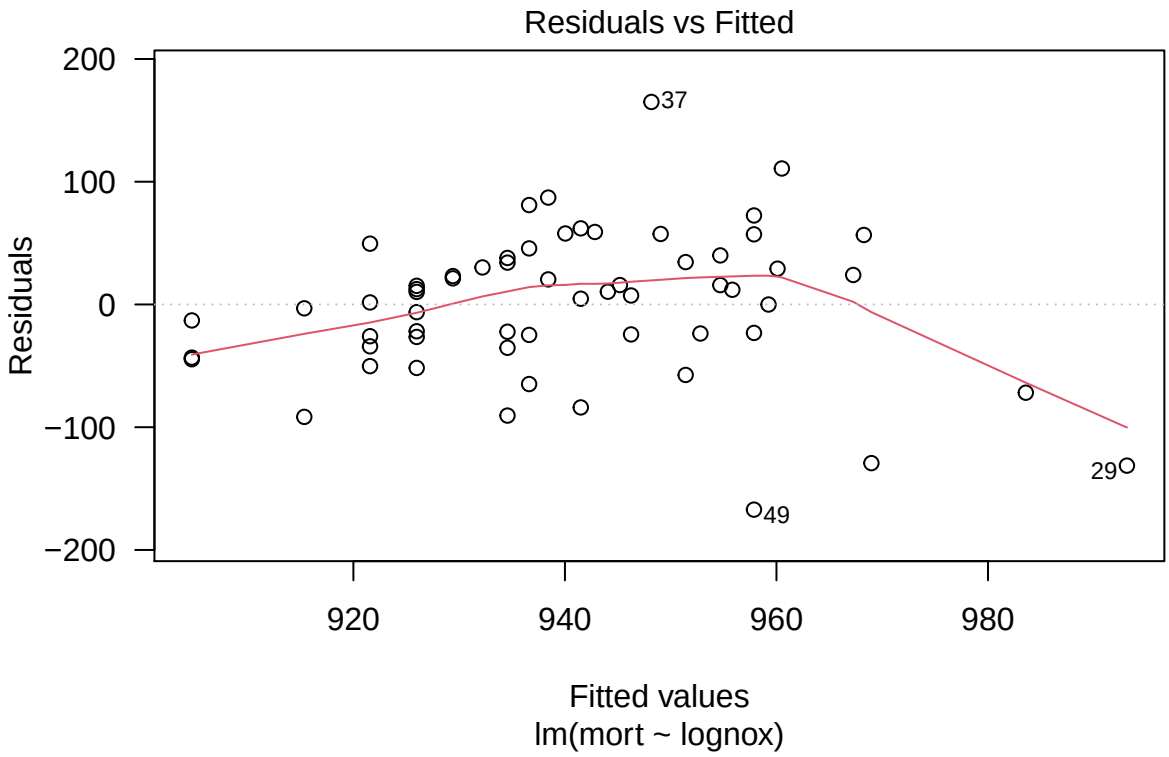

```
# fitting the linear regression

fit_1 <- lm(mort ~ nox, data = pollution)

# creating the residuals

plot(fit_1, which = 1, las = 1, )
```



Part b

I chose to do a logarithmic transformation of the data, meaning that I ran a regression on mortality rate vs. log(nox), where nox is our nitric oxide variable from part a. The residuals for this regression look much better than the linear regression from part a. They are relatively randomly distributed and aren't clustered around any one point, telling us that the model is not a bad fit.

```
# creating a log version of nox

pollution <- pollution %>%
  mutate(lognox = log(nox))


fit_2 <- lm(mort ~ lognox, data = pollution)

# creating the residuals. This residual plots look a lot better than the other
# residuals. They are more randomly distributed and not clustered.

plot(fit_2, which = 1, las = 1, )
```



Part c

The slope coefficient from the model in b is 15,34. The interpretation of this coefficient is a difference of 1% in nitrix oxide levels is associated with an (approximate) positive difference of .1534% in mortality rate.

Part d

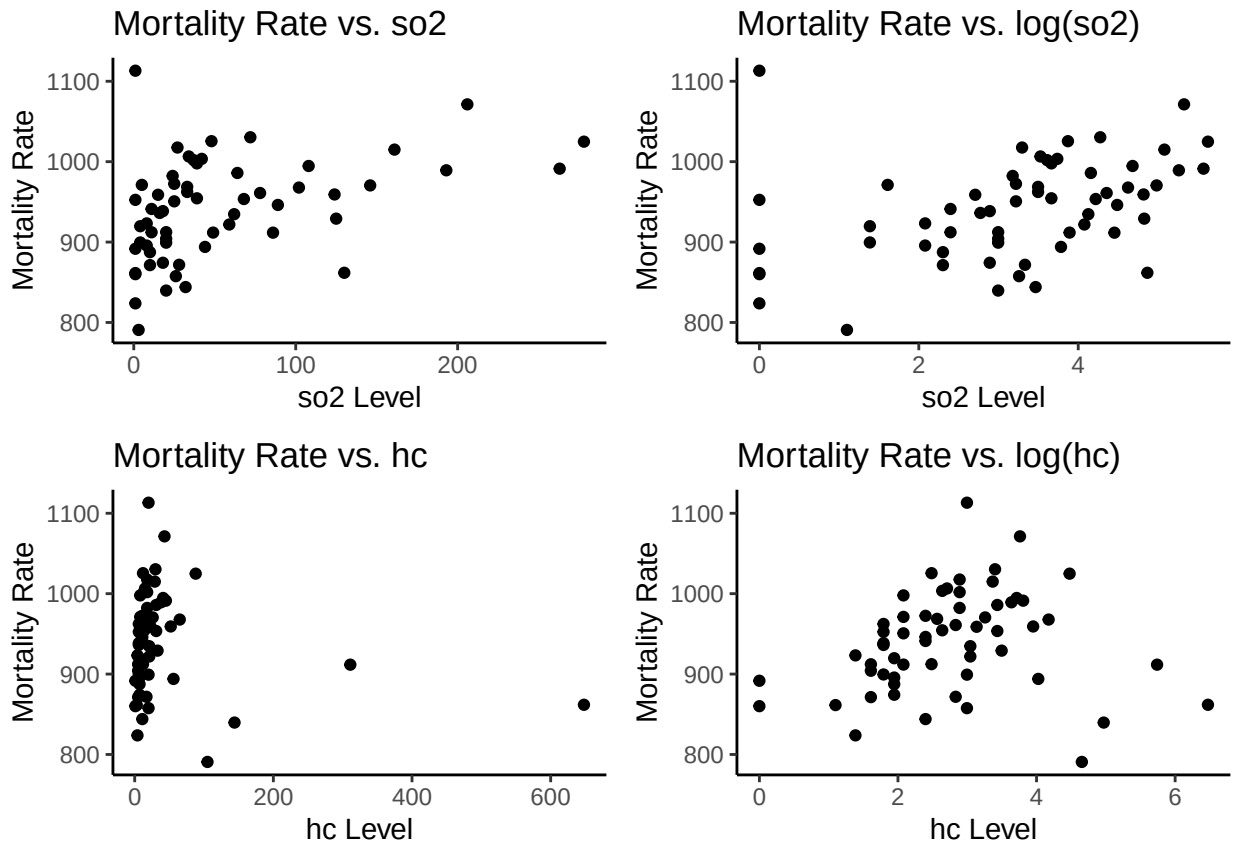
To determine whether or not I should use logarithmic transformations for sulfur dioxide and hydrocarbons, I ran a linear regression on mortality rate vs. each variable and mortality rate vs. each log variable and compared the scatter plots and residual plots.

The graphs below show the scatter plots for both the non log so2 and hc variables as well as the log verison of each variable. It looks like a logarithmic transformation might be a better for for both of these variables based on their scatterplots. I will confirm this by looking at the residual plots for each variable.

The residual plot for the variable follows the scatterplot. For both hc and so2, the log version of the regression produces residuals that appear more randomly distributed, especially for hc. So, I decided to use the log version of both so2 and hc in my regression.

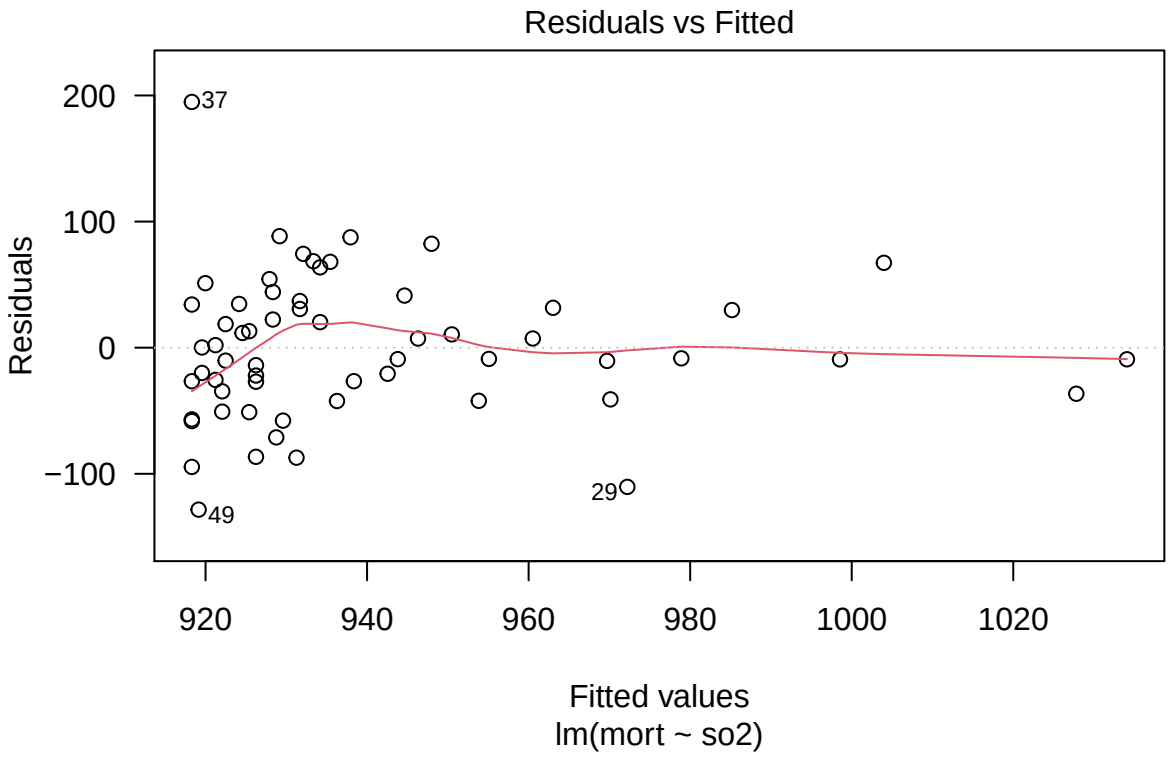
```
# creating logged versions of the variables

pollution <- pollution %>%
  mutate(logso2 = log(so2),
         loghc = log(hc),
         logmort = log(mort))
```

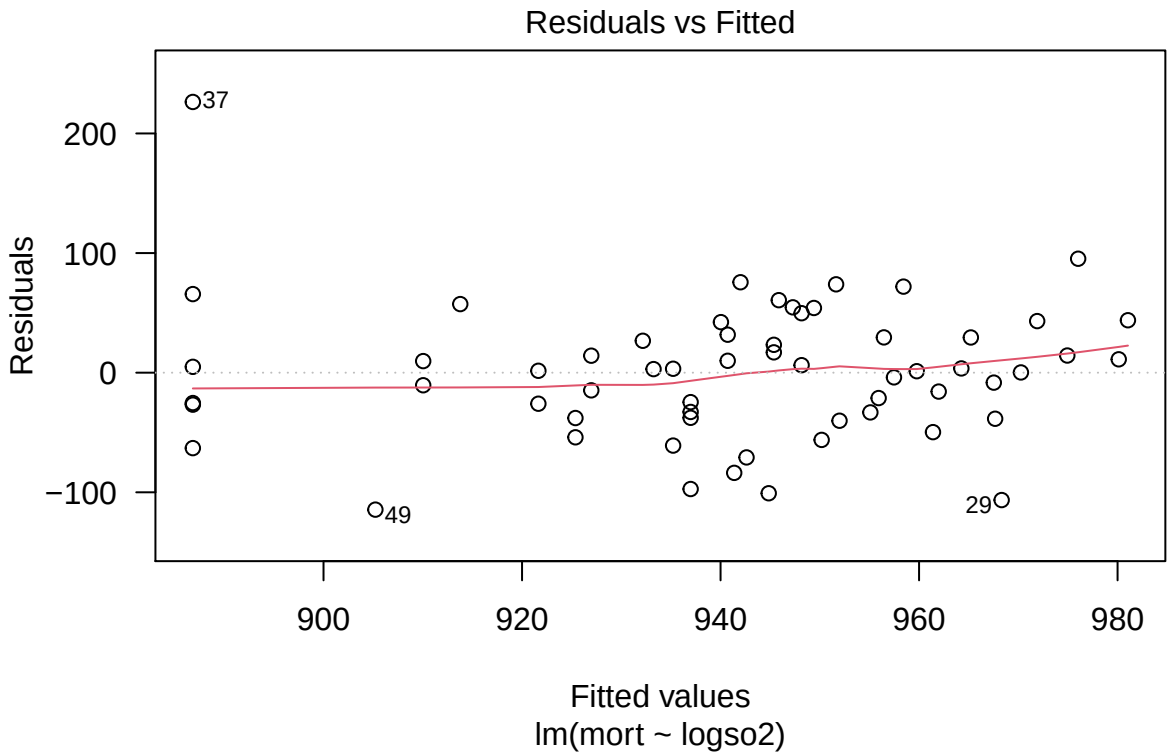
```r
# maybe want to do the log of so2

so2_plot <- pollution %>%
  ggplot(aes(x = so2, y = mort)) +
  geom_point() +
  labs(title = "Mortality Rate vs. so2",
       x =  "so2 Level",
       y =  "Mortality Rate") +
  theme_classic()

# definitely want to do the log of hc

hc_plot <- pollution %>%
  ggplot(aes(x = hc, y = mort)) +
  geom_point() +
  labs(title = "Mortality Rate vs. hc",
       x =  "hc Level",
       y =  "Mortality Rate") +
  theme_classic()

logso2_plot <- pollution %>%
  ggplot(aes(x = logso2, y = mort)) +
  geom_point() +
  labs(title = "Mortality Rate vs. log(so2)",
       x =  "so2 Level",
       y =  "Mortality Rate") +
  theme_classic()

# definitely want to do the log of hc

loghc_plot <- pollution %>%
  ggplot(aes(x = loghc, y = mort)) +
  geom_point() +
  labs(title = "Mortality Rate vs. log(hc)",
       x =  "hc Level",
       y =  "Mortality Rate") +
  theme_classic()

ggarrange(so2_plot, logso2_plot,
          hc_plot, loghc_plot)
```



The plots below show the residuals for the so2 model and the log(so2) model. The residuals for the log model look more randomly distributed than the residuals for the non log model, so I am inclined to use the logso2 variable in my final model.

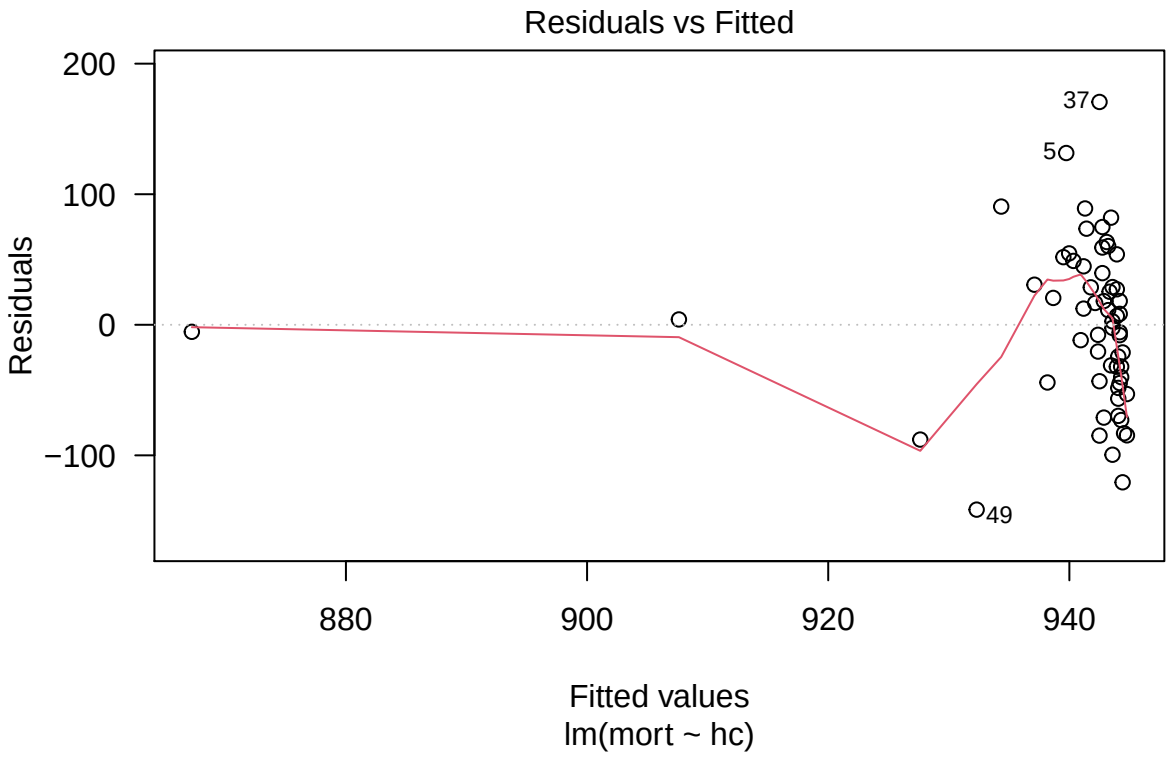
```
# creating models and residual plots for so2 and hc for log and non log

fit_so2 <- lm(mort ~ so2, data = pollution)
fit_logso2 <- lm(mort ~ logso2, data = pollution)

fit_hc <- lm(mort ~ hc, data = pollution)
fit_loghc <- lm(mort ~ loghc, data = pollution)

# using par to display all of them together
{
plot(fit_so2, which = 1, las = 1, )
plot(fit_logso2, which = 1, las = 1, )
}
```

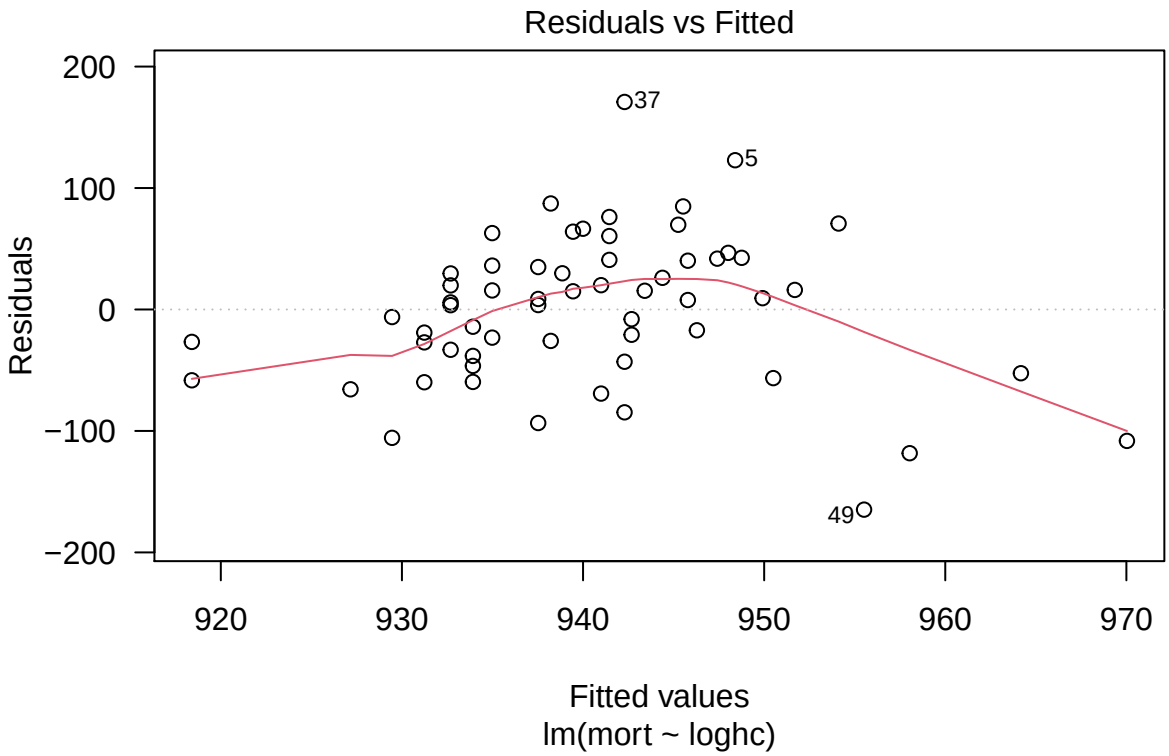

Residuals vs Fitted
Residuals
200
100
0
−100
37
49
29
920
940
960
980
1000
1020
Fitted values
lm(mort ~ so2)



The plots below show the residuals for the hc model and the log(hc) model. The residuals for the log model look dramatically more randomly distributed than the residuals for the non log model, so I am inclined to use the loghc variable in my final model.

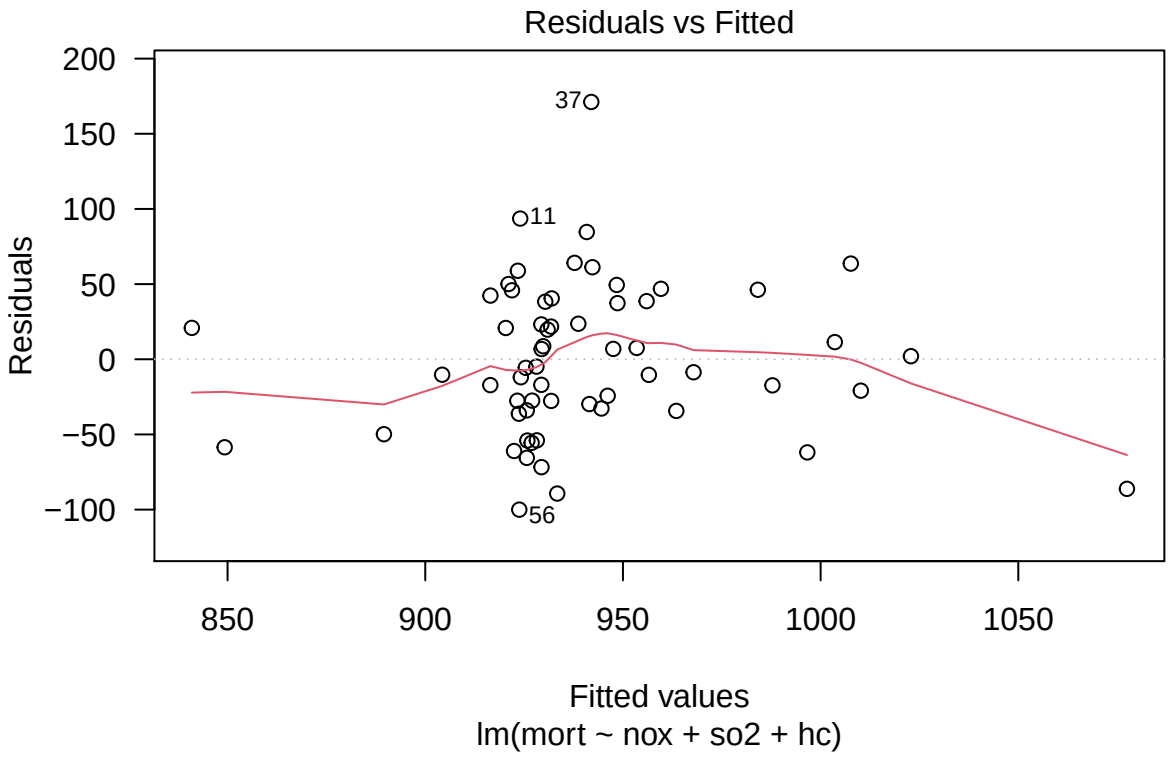
```
{
plot(fit_hc, which = 1, las = 1, )
plot(fit_loghc, which = 1, las = 1, )
}
```


Residuals vs Fitted
200
100
0
−100
Residuals
37
5
49
880
900
920
940
Fitted values
lm(mort ~ hc)

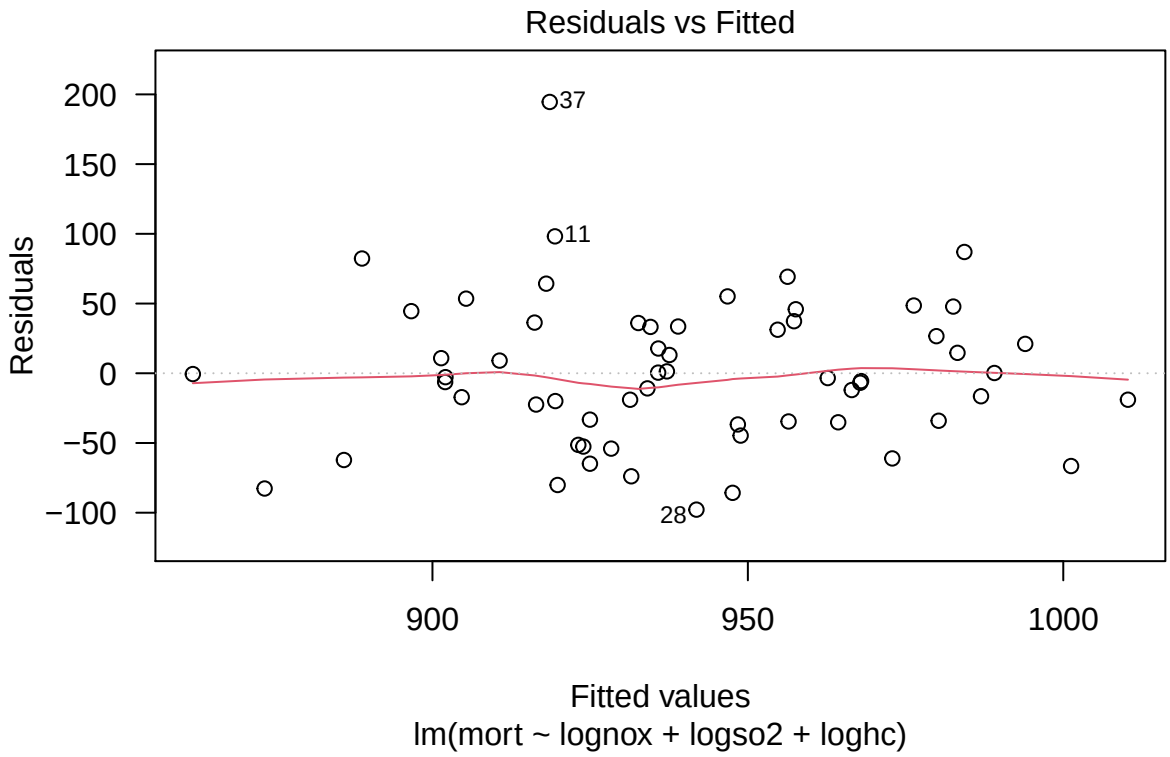


The plots below compare the residuals for the non log and log regressions including nox, so2, and hc (at this point, y is still not in log form). The residuals for the log regression are more evenly distributed and random, confirming my decision above to use the log version of both so2 and hc.

```
# creating two regression to compare

fit_all <- lm(mort ~ nox + so2 + hc, data = pollution)

fit_all_log <- lm(mort ~ lognox + logso2 + loghc, data = pollution)

# the residuals look better for the logged model

plot(fit_all, which = 1, las = 1, )
```



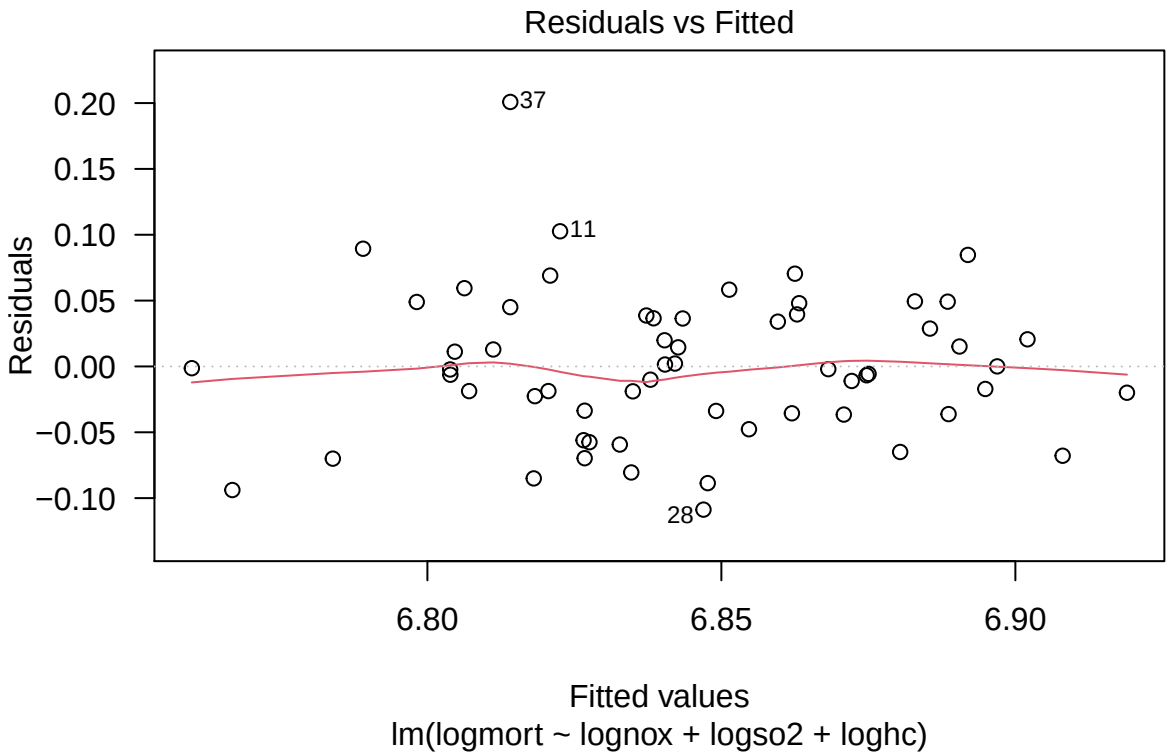

```
plot(fit_all_log, which = 1, las = 1, )
```




```
# log level.
# level log: 1% change in x is associated with a beta/100 change in y, on average
# log log: 1% change in x is associated with a beta % change in y
```

I also decided to see how running a regression on log mortality rate vs the log other variables affected the model. I would prefer to have both sides of the regression be logged, so the interpretation of the coefficients is simpler. Looking at the residual plot below, using log y in the regression doesn't appear to make the residuals any less random, so I will use log mortality in my model.

```
# benefit of also logging y: the coefficients become easier to interpret

fit_all_log_y <- lm(logmort ~ lognox + logso2 + loghc, data = pollution)

# logging the y doesn't seem to make the residuals less random

plot(fit_all_log_y, which = 1, las = 1, )
```

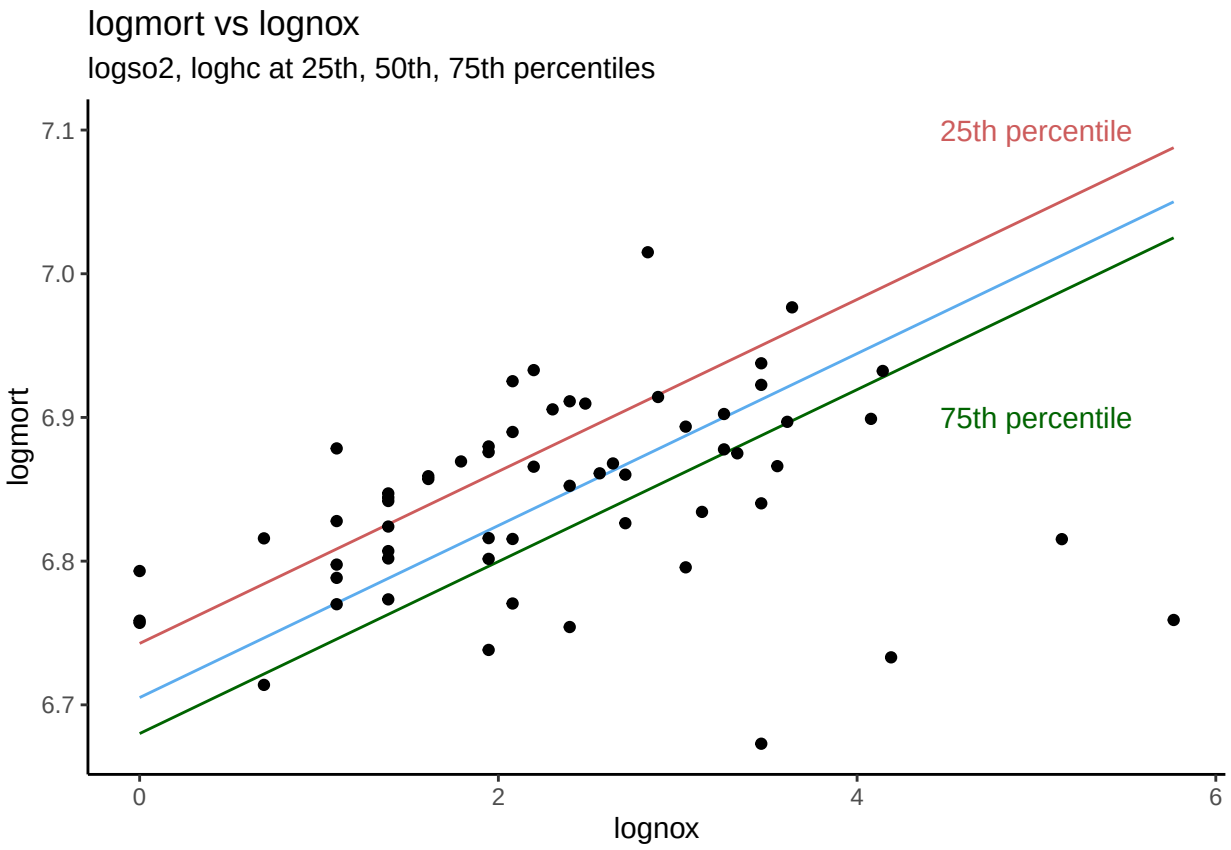


The plot below shows the fitted regression model. Because I couldn't include all 3 IV's, I displayed logmort vs lognox with loghc and logso2 held constant at their 25th, 50th, and 75th quantiles.

The table below shows the regression results for the model.

```
# showing plot of fitted model

pollution %>%
  ggplot(aes(y = logmort)) +
  stat_function(fun = function(.x) 6.82675
                + .05984*.x +
                  (.01431*mean(pollution$logso2)) +
                  (-.06081*mean(pollution$loghc)),
                col  = "steelblue2") +
    stat_function(fun = function(.x) 6.82675
                + .05984*.x +
                  (.01431*quantile(pollution$logso2, probs = .25)) +
                  (-.06081*quantile(pollution$loghc, probs = .25)),
                col  = "indianred") +
      stat_function(fun = function(.x) 6.82675
                + .05984*.x +
                  (.01431*quantile(pollution$logso2, probs = .75)) +
                  (-.06081*quantile(pollution$loghc, probs = .75)),
                col  = "darkgreen") +
  annotate("text", x = 5, y = 6.9, label = "75th percentile",
           col = "darkgreen") +
    annotate("text", x = 5, y = 7.1, label = "25th percentile",
```

```
          col = "indianred") +
  geom_point(aes(x = pollution$lognox)) +
  labs(title = "logmort vs lognox",
       subtitle = "logso2, loghc at 25th, 50th, 75th percentiles",
       x = "lognox") +
  theme_classic()
```




```
# showing regression table

stargazer(fit_all_log_y, type = "latex",
          title = "Regression Results Log Model")
```

% Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harvard.edu
% Date and time: Fri, Feb 26, 2021 - 15:03:57

The coefficient for lognox is .06. In context, this means that a 1% increase in nitric oxide levels is associated with a .06% increase in mortality rate, on average.

The coefficient for logso2 is .014. In context, this means that a 1% increase in so2 levels is associated with a .014% increase in mortality rate, on average.

the coefficient for loghc is -.061. In context, this means that a 1% increase in hc levels is associated with a .061% decrease in mortality rate, on average.

Table 1: Regression Results Log Model

	Dependent variable:
	logmort
lognox	0.060^{**}
	(0.023)
logso2	0.014^{*}
	(0.008)
loghc	-0.061^{***}
	(0.021)
Constant	6.827^{***}
	(0.023)
Observations	60
R^2	0.285
Adjusted R^2	0.247
Residual Std. Error	0.058 (df = 56)
F Statistic	7.449^{***} (df = 3; 56)
Note:	*p<0.1; **p<0.05; ***p<0.01

Part e

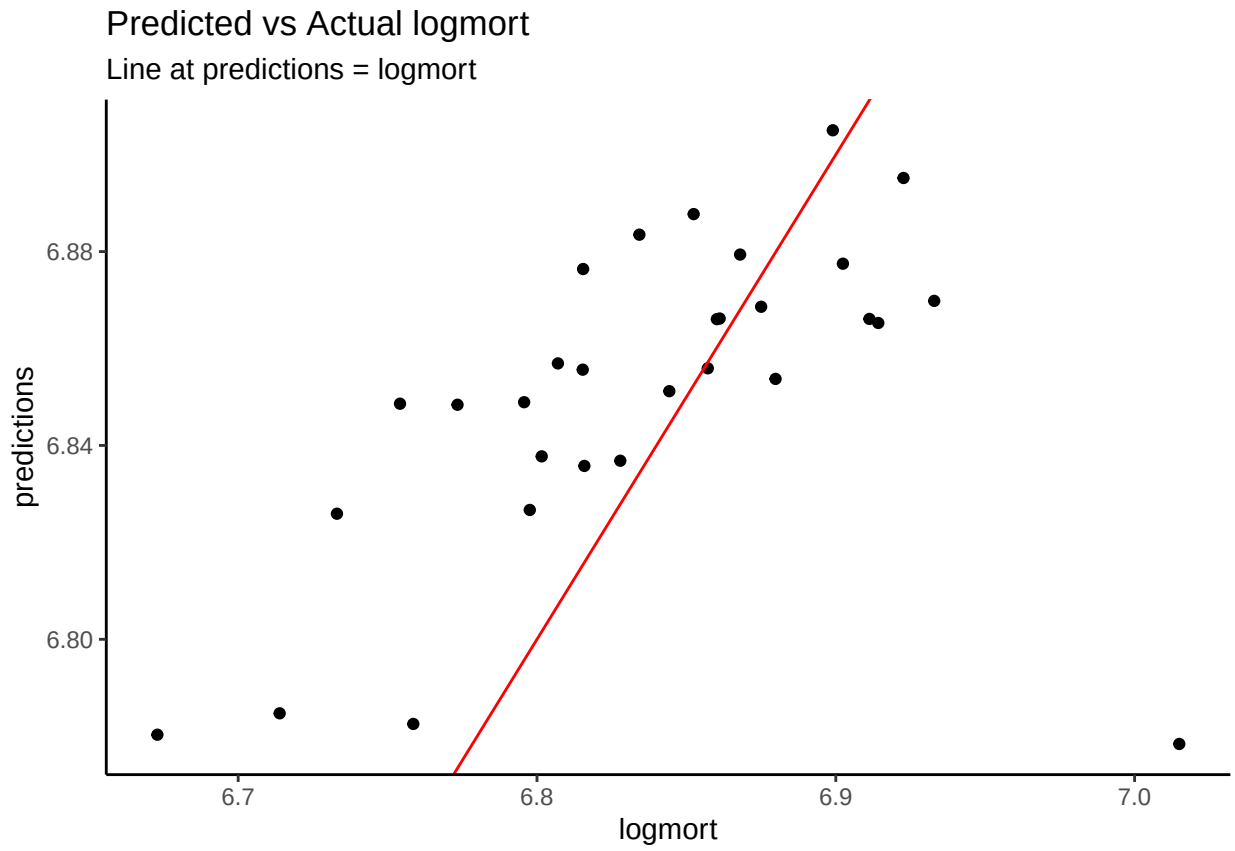
```
# taking first 30 rows of data

pol_train <- pollution %>%
  head(30)

# taking last 30 rows of data

pol_test <- pollution %>%
  tail(30)

# fitting the training model

fit_all_log_y_train <- lm(logmort ~ lognox + logso2 + loghc, data = pol_train)

# creating predictions with test data

predictions <- predict(fit_all_log_y_train, newdata = pol_test)

pred_df <- cbind(predictions, pol_test)
```

The plot below shows the predicted values of logmort vs. the actual values of logmort from the second half of the data frame. The points to the left of the line are overpredictions and the points to the right of the line are underpredictions. From this plot, it appears that the predictions are relatively close to their true values, but the model overestimated more points than underestimated.

Additionally, the Mean Squared Error of the predictions is .004, which is hard to interpret on its own without

conparisonns to other models, but also seems very small to me.

```
# plot of yhat vs y

pred_df %>%
  ggplot(aes(x = logmort, y = predictions)) +
  geom_point() +
  geom_abline(slope = 1, intercept = 0, col = "red") +
  labs(title = "Predicted vs Actual logmort",
       subtitle = "Line at predictions = logmort") +
  theme_classic()
```

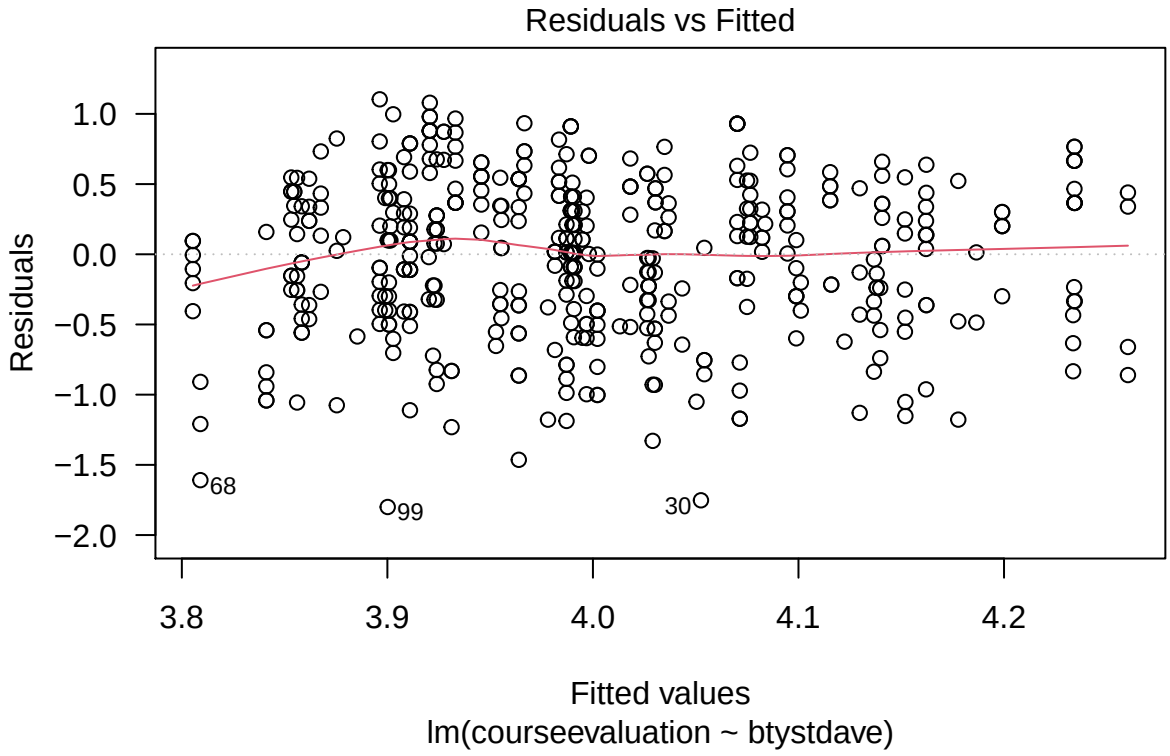



```
# calculating MSE

pred_df <- pred_df %>%
  mutate(sq_error = (predictions - logmort)^2)

MSE <- sum(pred_df$sq_error) / nrow(pred_df)
```

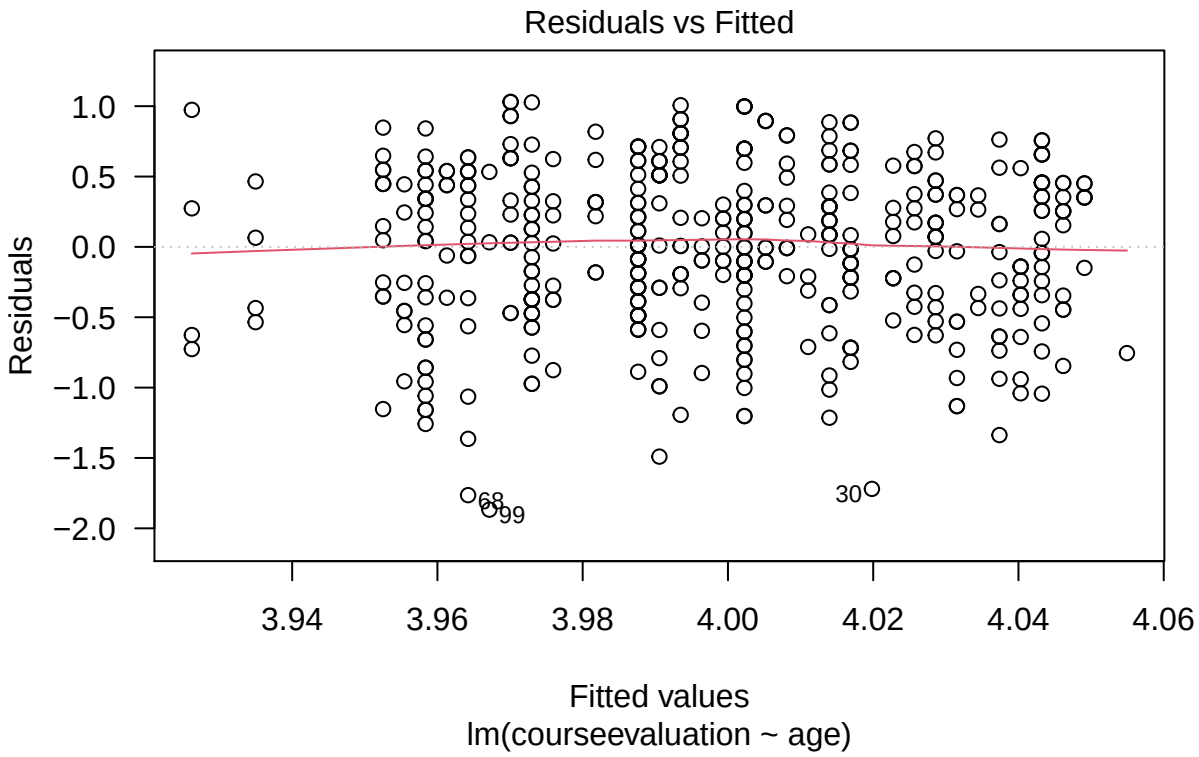
Question 8

```
# looking at btystdave. The residuals seem to be clustered more toward the
# minimum of the range.
```

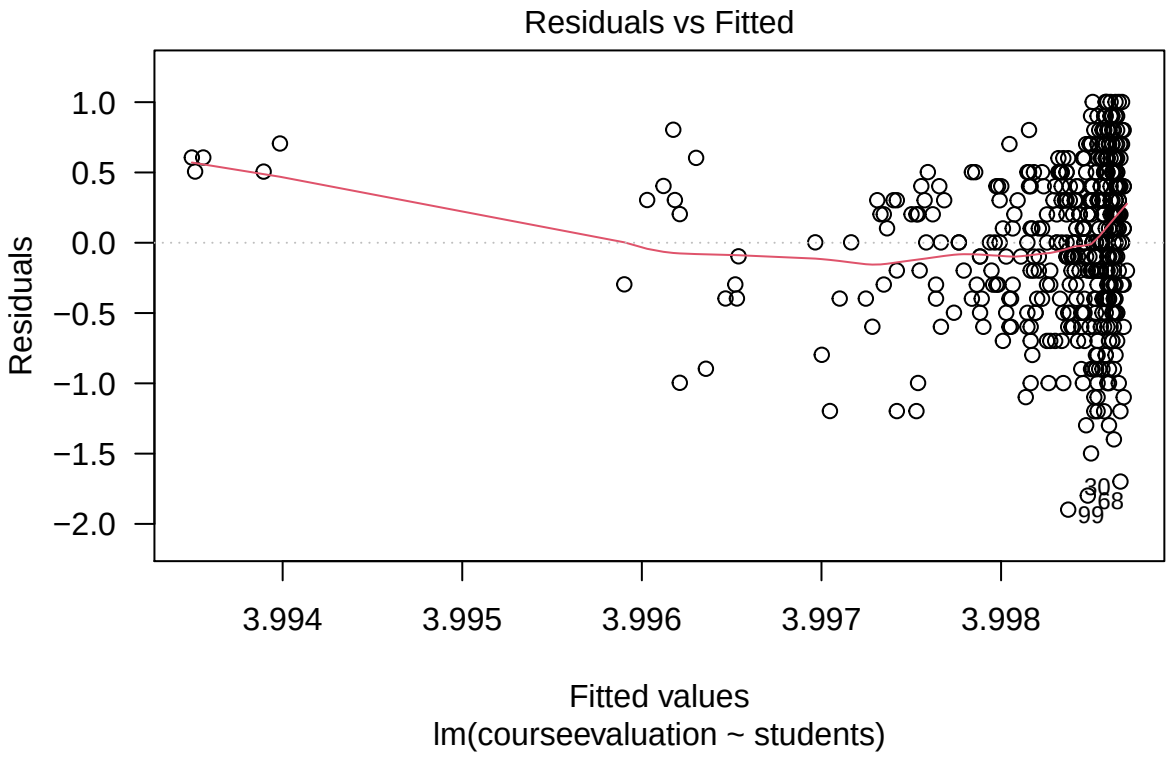
```r
fit_bty <- lm(courseevaluation ~ btystdave, data = beauty)

bty_plot <- ggplot(beauty, aes(x = btystdave, y = courseevaluation)) +
  geom_point() +
  labs(title = "Evaluations vs btystdave") +
  theme_classic()

plot(fit_bty, which = 1, las = 1, )
```



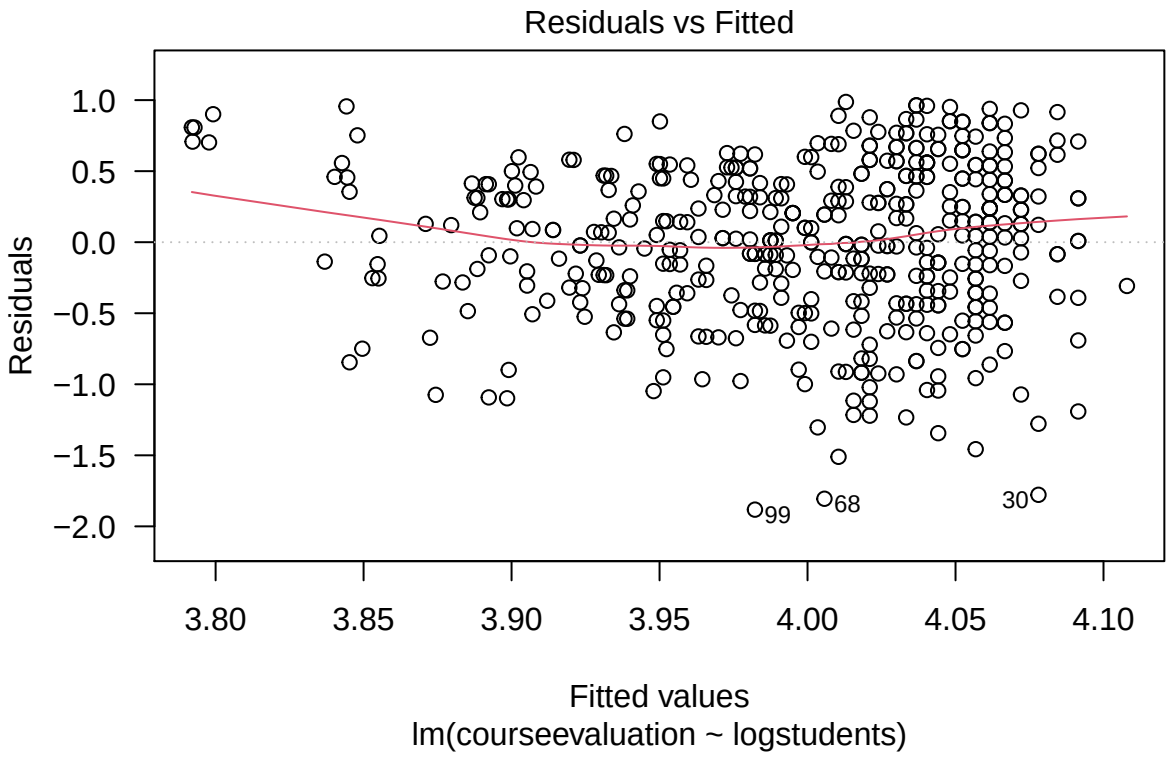

```r
# looking at age. I see quadratic transformations happen with age a lot, so
# maybe I should try it

fit_age <- lm(courseevaluation ~ age, data = beauty)

age_plot <- ggplot(beauty, aes(x = age, y = courseevaluation)) +
  geom_point() +
  labs(title = "Evaluations vs age") +
  theme_classic()

plot(fit_age, which = 1, las = 1, )
```




```
# looking at class enrollment with prediction that smaller classes have better
# evaluations. I should definitely try logarithmic transformation here.

fit_students <- lm(courseevaluation ~ students, data = beauty)

students_plot <- ggplot(beauty, aes(x = students, y = courseevaluation)) +
  geom_point() +
  labs(title = "Evaluations vs enrollment") +
  theme_classic()

plot(fit_students, which = 1, las = 1, )
```




```
# creating a logged version of beauty

beauty  <- beauty %>%
  mutate(logstudents = log(students))

fit_students_log <- lm(courseevaluation ~ logstudents, data = beauty)

students_plot_log <- ggplot(beauty, aes(x = logstudents, y = courseevaluation)) +
  geom_point() +
  labs(title = "Evaluations vs log(enrollment)") +
  theme_classic()

plot(fit_students_log, which = 1, las = 1, )
```




```
# at this point I am going to try an initial model with all of the variables and
# see how it looks. Look at the residuals, etc.

fit_all <- lm(courseevaluation ~ btystdave  + logstudents + age + female + minority,
              data =  beauty)


fit_int <- lm(courseevaluation ~ btystdave*female  + logstudents + female + minority,
              data =  beauty)

fit_int_no_min <- lm(courseevaluation ~ btystdave*female  + logstudents + female,
              data =  beauty)

stargazer(fit_all, fit_int, fit_int_no_min, type = "text")
```

```
==========================================================================
Dependent variable:
---------------------------------- courseevaluation
(1) (2) (3)
---------------------------------------------------------- btystdave 0.144*** 0.210*** 0.216***
(0.033) (0.043) (0.043)

logstudents -0.099*** -0.098*** -0.094***
(0.030) (0.030) (0.029)

age -0.004
(0.003)
```

female -0.215*** -0.205*** -0.214***
(0.053) (0.051) (0.051)

minority -0.137* -0.106
(0.073) (0.073)

btystdave:female -0.116* -0.131**
(0.064) (0.064)

Constant 4.667*** 4.469*** 4.444***
(0.189) (0.113) (0.112)

Observations 463 463 463
R2 0.095 0.097 0.093
Adjusted R2 0.085 0.087 0.085
Residual Std. Error 0.531 (df = 457) 0.530 (df = 457) 0.531 (df = 458)
F Statistic 9.547*** (df = 5; 457) 9.796*** (df = 5; 457) 11.696*** (df = 4; 458) ==========
Note: *p<0.1;* ***p<0.05;*** p<0.01