# Lab 6

Adam Shen

October 21, 2020

## Packages

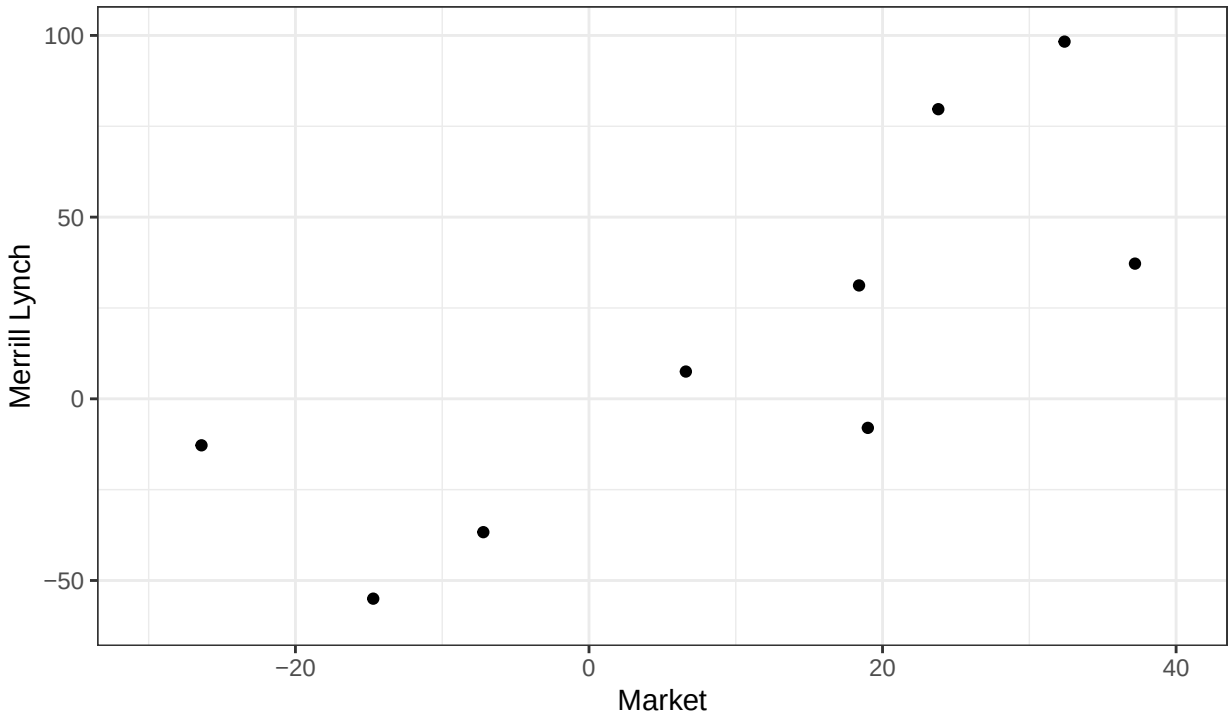
```
library(MASS)
library(car)
library(ggplot2)
library(broom)
library(dplyr)
theme_set(theme_bw())
```

## Stock data

```
capm <- read.table("./stock.txt", header=TRUE)
```

### Scatterplot

```
ggplot(capm, aes(x=Market, y=Merrill))+
  geom_point()+
  coord_cartesian(xlim=c(-30, 40), ylim=c(-60, 100))+
  labs(x="Market", y="Merrill Lynch")
```



## Fit a SLR model

```
model <- lm(Merrill ~ Market, data=capm)
summary(model)
```

```
##
## Call:
## lm(formula = Merrill ~ Market, data = capm)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -40.237 -26.037  -2.218  37.410  41.729
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -2.267     12.647  -0.179   0.8628
## Market         1.816      0.553   3.284   0.0134 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 34.2 on 7 degrees of freedom
## Multiple R-squared:  0.6064, Adjusted R-squared:  0.5502
## F-statistic: 10.79 on 1 and 7 DF,  p-value: 0.01341
```

```
anova(model)
```

```
## Analysis of Variance Table
##
## Response: Merrill
##           Df Sum Sq Mean Sq F value  Pr(>F)
## Market     1  12618 12617.7  10.786 0.01341 *
## Residuals  7   8189  1169.9
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Since this is a simple linear regression model, `anova()` is performing a test for model usefulness. The associated hypotheses are:

$$H_0 : \beta_1 = 0 \quad \text{vs} \quad H_A : \beta_1 \neq 0$$

Since the $p$-value, 0.01341, is less than 0.05, we reject the null hypothesis and conclude that the model is useful.

```
confint(model)
```

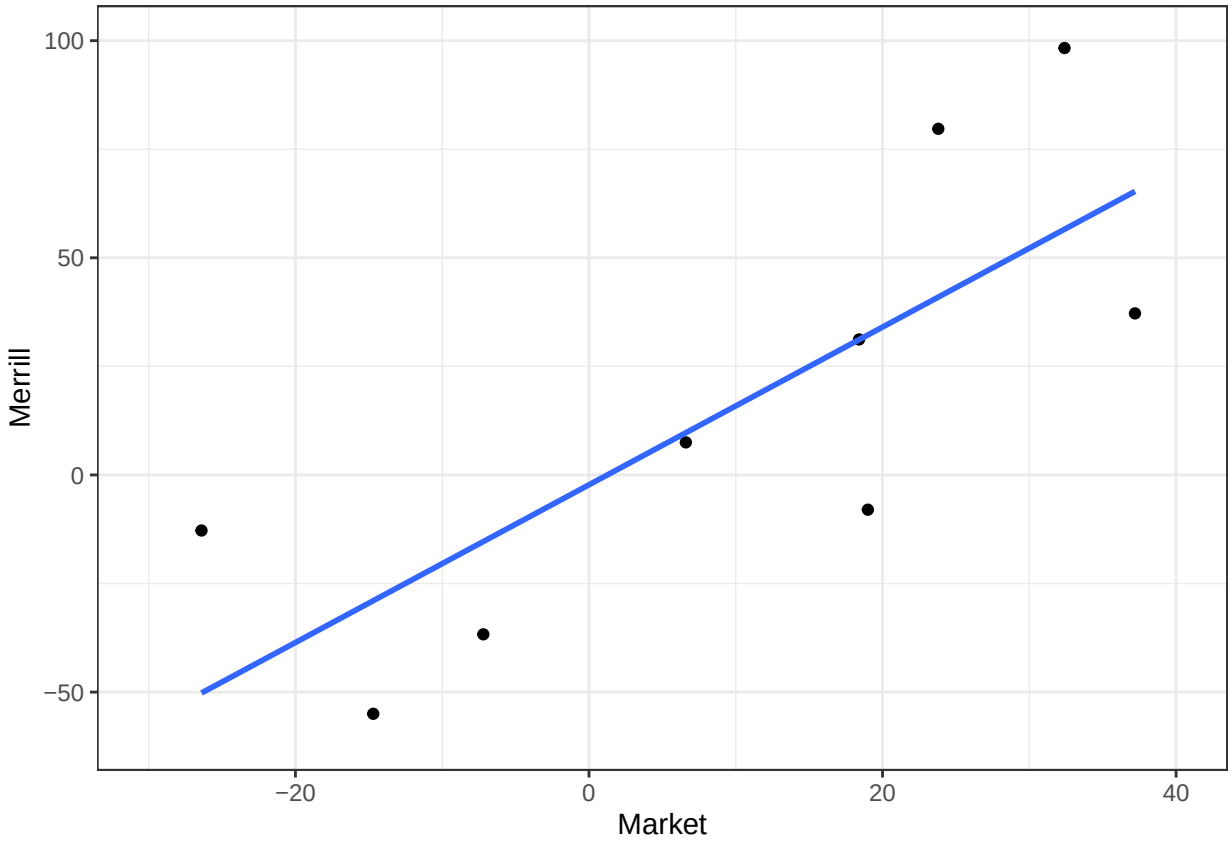
```
##                   2.5 %    97.5 %
## (Intercept) -32.1734196 27.638549
## Market        0.5084633  3.123566
```

### Visualize the fit

Moving forward, we will `broom::augment()` the model to append a column of fitted values rather than using `transform()` as in previous labs. The fitted values will be stored under a column called `.fitted`.

```
capm_aug <- augment(model)

ggplot(capm_aug, aes(x=Market))+
  geom_point(aes(y=Merrill))+
  geom_line(aes(y=.fitted), colour="#3366FF", size=1)+
  coord_cartesian(xlim=c(-30, 40), ylim=c(-60, 100))
```



## Additional model information

```
fits <- fitted(model)
res <- resid(model)
hatdiag <- hatvalues(model)
instudres <- rstandard(model)
exstudres <- rstudent(model)
cook <- cooks.distance(model)

(model_info <- data.frame(
  Merrill=capm$Merrill,
  Market=capm$Market,
  fits=round(fits, 3),
  res=round(res, 3),
  instudres=round(instudres, 4),
  hatdiag=round(hatdiag, 4),
  exstudres=round(exstudres, 4),
  cook=round(cook, 4)
  ))
```

```
##   Merrill Market    fits     res instudres hatdiag exstudres   cook
## 1    -8.0   19.0  32.237 -40.237   -1.2632  0.1328   -1.3311 0.1221
## 2   -55.0  -14.7 -28.963 -26.037   -0.8905  0.2693   -0.8756 0.1461
## 3   -12.8  -26.4 -50.210  37.410    1.4823  0.4555    1.6568 0.9191
## 4    37.2   37.2  65.288 -28.088   -0.9857  0.3059   -0.9834 0.2141
## 5    79.7   23.8  40.954  38.746    1.2372  0.1616    1.2958 0.1475
## 6   -36.7   -7.2 -15.343 -21.357   -0.6928  0.1875   -0.6645 0.0554
## 7     7.5    6.6   9.718  -2.218   -0.0689  0.1140   -0.0638 0.0003
## 8    31.2   18.4  31.147   0.053    0.0017  0.1300    0.0015 0.0000
## 9    98.3   32.4  56.571  41.729    1.4026  0.2434    1.5315 0.3165
```

We can obtain the values of most of the above columns by simply using `broom::augment()`.

```
augment(model)
```

```
## # A tibble: 9 x 8
##   Merrill Market .fitted   .resid .std.resid  .hat .sigma     .cooksd
##     <dbl>  <dbl>   <dbl>    <dbl>      <dbl> <dbl>  <dbl>       <dbl>
## 1    -8     19     32.2  -40.2      -1.26    0.133   32.5 0.122
## 2   -55    -14.7  -29.0  -26.0      -0.891   0.269   34.8 0.146
## 3   -12.8  -26.4  -50.2   37.4       1.48    0.456   30.6 0.919
## 4    37.2   37.2   65.3  -28.1      -0.986   0.306   34.3 0.214
## 5    79.7   23.8   41.0   38.7       1.24    0.162   32.7 0.148
## 6   -36.7   -7.2  -15.3  -21.4      -0.693   0.188   35.7 0.0554
## 7     7.5    6.6    9.72  -2.22     -0.0689  0.114   36.9 0.000305
## 8    31.2   18.4   31.1    0.0528    0.00165 0.130   36.9 0.000000204
## 9    98.3   32.4   56.6   41.7       1.40    0.243   31.3 0.317
```

Note that compared to the previous data frame, `augment.lm`:

- returns a column called `.std.resid` which corresponds to the internally studentized residuals (also known as standardized residuals).
- does not return the externally studentized residuals. Instead, it returns a column called `.sigma` which is the estimated residual standard error ($\sqrt{MSE}$) **when the corresponding observation is dropped**.

We can use the `.sigma` column to re-create the externally studentized residuals!

Recall that the internally studentized residuals are obtained as:

$$t_i = \frac{\hat{\varepsilon}_i}{s\sqrt{1-h_{ii}}}$$

where:

- $\hat{\varepsilon}$ is the raw residual
- $s$ is the estimated residual standard deviation, $\sqrt{MSE}$
- $h_{ii}$ is the $i^{\text{th}}$ leverage value ($i^{\text{th}}$ diagonal element of the hat matrix)

The externally studentized residuals can be obtained by swapping out $s$ for $s_{(i)}$ and computing:

$$t_{i,(i)} = \frac{\hat{\varepsilon}_i}{s_{(i)}\sqrt{1-h_{ii}}}$$

where $s_{(i)}$ is the estimated residual standard error ($\sqrt{MSE}$) when the $i^{\text{th}}$ observation is dropped. This $s_{(i)}$ is the `.sigma` column!

Adding the externally studentized residuals to our augmented output:

```
mutate(augment(model),
       ext.std.resid=.resid/(.sigma*sqrt(1-.hat)))
```

```
## # A tibble: 9 x 9
##   Merrill Market .fitted   .resid .std.resid  .hat .sigma  .cooksd ext.std.resid
##     <dbl>  <dbl>   <dbl>    <dbl>      <dbl> <dbl>  <dbl>    <dbl>         <dbl>
## 1    -8     19     32.2  -40.2      -1.26    0.133   32.5  1.22e-1      -1.33
## 2   -55    -14.7  -29.0  -26.0      -0.891   0.269   34.8  1.46e-1      -0.876
## 3   -12.8  -26.4  -50.2   37.4       1.48    0.456   30.6  9.19e-1       1.66
## 4    37.2   37.2   65.3  -28.1      -0.986   0.306   34.3  2.14e-1      -0.983
## 5    79.7   23.8   41.0   38.7       1.24    0.162   32.7  1.48e-1       1.30
## 6   -36.7   -7.2  -15.3  -21.4      -0.693   0.188   35.7  5.54e-2      -0.665
## 7     7.5    6.6    9.72  -2.22     -0.0689  0.114   36.9  3.05e-4      -0.0638
## 8    31.2   18.4   31.1    0.0528    0.00165 0.130   36.9  2.04e-7       0.00153
## 9    98.3   32.4   56.6   41.7       1.40    0.243   31.3  3.17e-1       1.53
```

Compare with original data frame that used base-R commands:

```
model_info
```

```
##   Merrill Market    fits     res instudres hatdiag exstudres   cook
## 1    -8.0   19.0  32.237 -40.237   -1.2632  0.1328   -1.3311 0.1221
## 2   -55.0  -14.7 -28.963 -26.037   -0.8905  0.2693   -0.8756 0.1461
## 3   -12.8  -26.4 -50.210  37.410    1.4823  0.4555    1.6568 0.9191
## 4    37.2   37.2  65.288 -28.088   -0.9857  0.3059   -0.9834 0.2141
## 5    79.7   23.8  40.954  38.746    1.2372  0.1616    1.2958 0.1475
## 6   -36.7   -7.2 -15.343 -21.357   -0.6928  0.1875   -0.6645 0.0554
## 7     7.5    6.6   9.718  -2.218   -0.0689  0.1140   -0.0638 0.0003
## 8    31.2   18.4  31.147   0.053    0.0017  0.1300    0.0015 0.0000
## 9    98.3   32.4  56.571  41.729    1.4026  0.2434    1.5315 0.3165
```

# Income data

```
salexp <- read.table("./income.txt", header=TRUE)
```

## Fit a SLR model

```
model <- lm(Salary ~ Experience, data=salexp)
summary(model)
```

```
##
## Call:
## lm(formula = Salary ~ Experience, data = salexp)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -17442.2  -5377.7   -542.4   4303.3  23540.2
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)     9906       2986   3.317  0.00174 **
## Experience      2203        152  14.499  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8166 on 48 degrees of freedom
## Multiple R-squared:  0.8141, Adjusted R-squared:  0.8102
## F-statistic: 210.2 on 1 and 48 DF,  p-value: < 2.2e-16
```

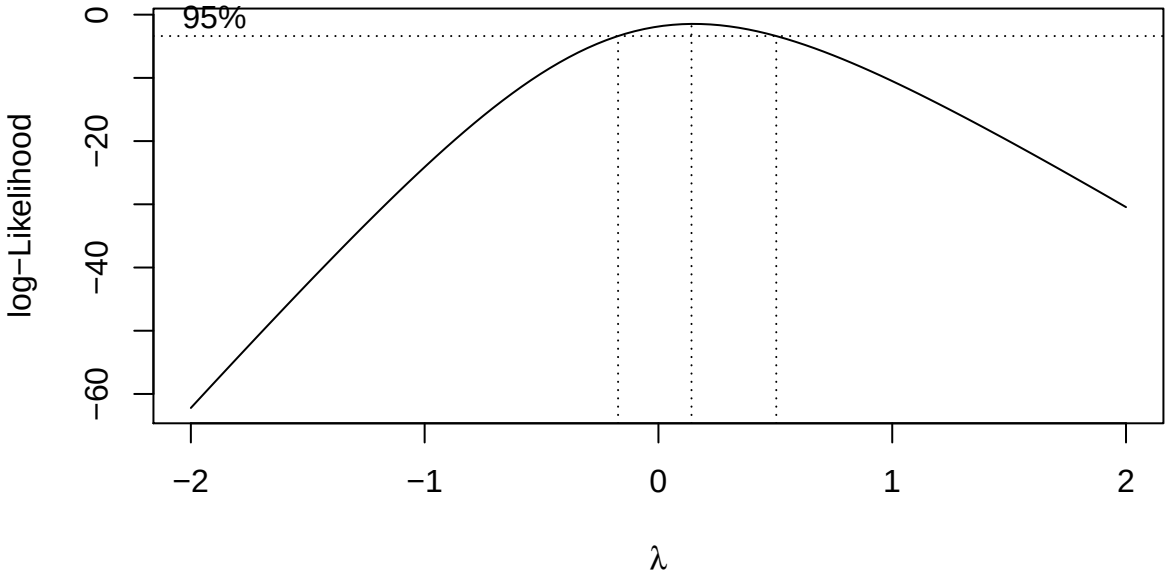
```
anova(model)
```

```
## Analysis of Variance Table
##
## Response: Salary
##            Df     Sum Sq    Mean Sq F value    Pr(>F)
## Experience  1 1.4017e+10 1.4017e+10  210.21 < 2.2e-16 ***
## Residuals  48 3.2006e+09 6.6678e+07
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

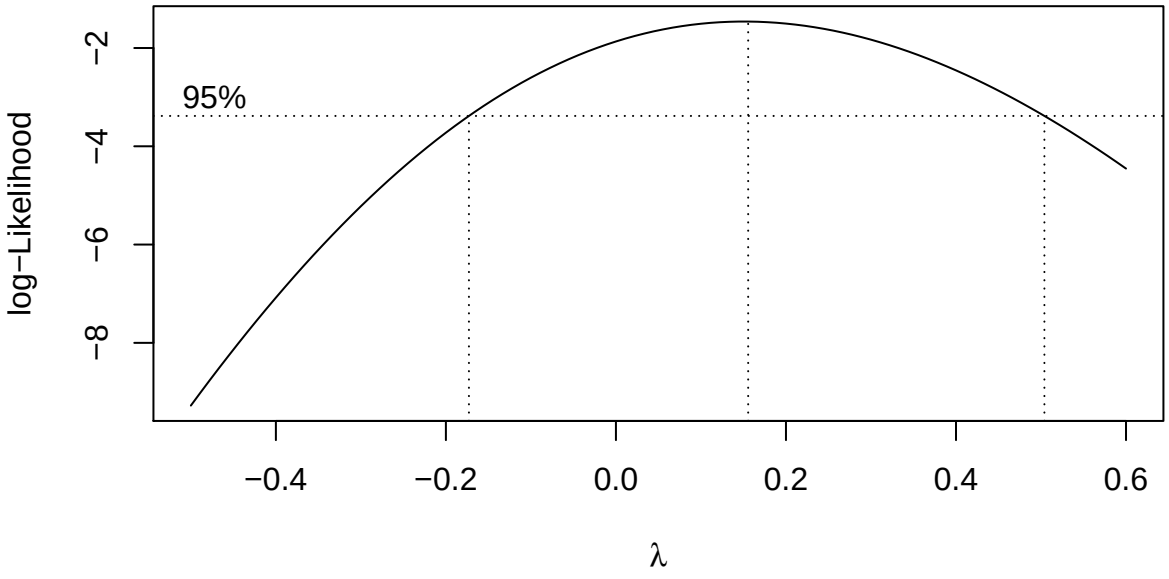
### Box-Cox transformation: finding $\lambda$

As explained in *Module 6.5*, we wish to find the value of $\lambda$ that maximizes the log-likliehood of the data, or alternatively, the value of lambda that minimizes $SSE$ of the model.

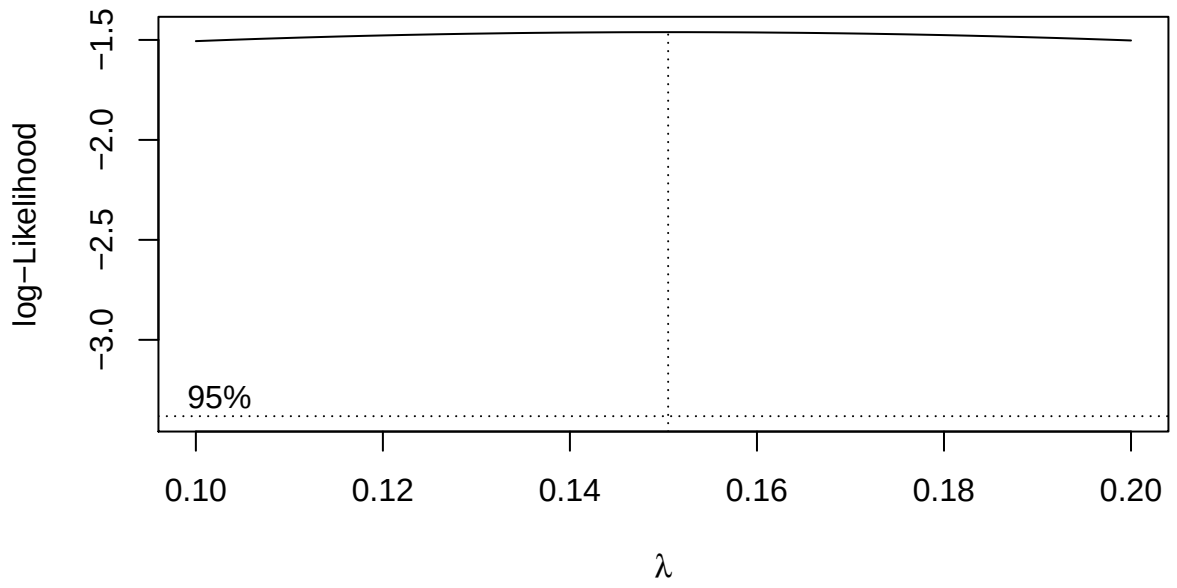
```
boxcox(model, plotit=TRUE)
```




```
boxcox(model, plotit=TRUE, lambda=seq(-0.5, 0.6, 0.1))
```



```
tr <- as.data.frame(boxcox(model, plotit=TRUE, lambda=seq(0.1, 0.2, 0.01)),
                    col.names=c("Lambda", "LogLik"))
```




Recall that the `order()` function sorts values from least to greatest, but instead of returning the sorted values, it returns the indices of the sorted values. We are looking for the index corresponding to a maximized log-likelihood.

```
order(tr$LogLik)
```

```
##   [1]   1   2 100   3  99   4  98   5  97   6  96   7  95   8  94   9  93  10
##  [19]  92  11  91  12  90  13  89  14  88  15  87  16  86  17  85  18  84  19
##  [37]  83  20  82  21  81  22  80  23  79  24  78  25  77  26  76  27  75  28
##  [55]  74  29  73  30  72  31  71  32  70  33  69  34  68  35  67  36  66  37
##  [73]  65  38  64  39  63  40  62  41  61  42  60  43  59  44  58  45  57  46
##  [91]  56  47  55  48  54  49  53  50  52  51
```

We are interested in position `51`. Let us view $\pm 5$ rows of position `51`.

```
tr[46:56,]
```

```
##       Lambda    LogLik
## 46 0.1454545 -1.461656
## 47 0.1464646 -1.461493
## 48 0.1474747 -1.461365
## 49 0.1484848 -1.461273
## 50 0.1494949 -1.461216
## 51 0.1505051 -1.461194
## 52 0.1515152 -1.461208
## 53 0.1525253 -1.461256
## 54 0.1535354 -1.461340
## 55 0.1545455 -1.461460
## 56 0.1555556 -1.461614
```

### Apply the transformation

```
lnsal <- log(salexp$Salary) # Apply natural log to `Salary`
meanlnsal <- mean(lnsal) # Compute mean of natural log of `Salary`
gmeansal <- exp(meanlnsal) # Compute geometric mean of `Salary`
lambda <- 0.15
Saltrans <- ((salexp$Salary**lambda)-1)/(lambda*(gmeansal**(lambda-1)))
tr_model <- lm(Saltrans ~ Experience, data=salexp)
```

```
summary(tr_model)
```

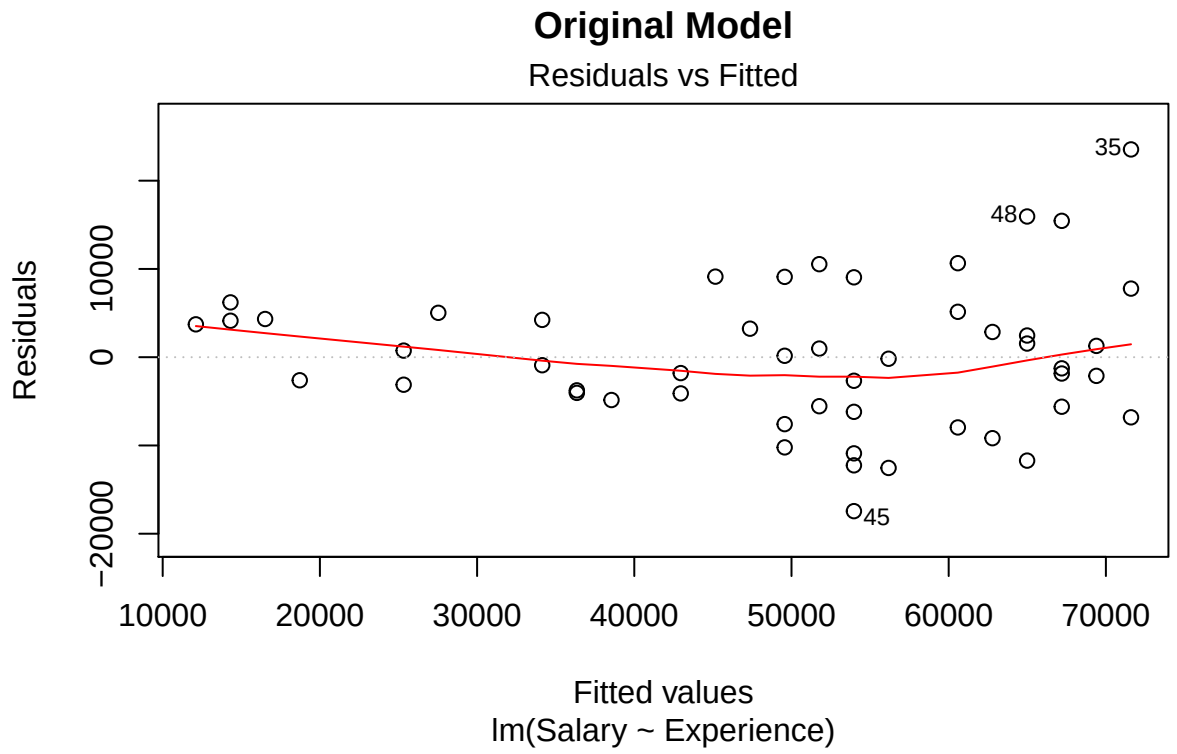
```
##
## Call:
## lm(formula = Saltrans ~ Experience, data = salexp)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15383.7  -4607.0   -303.3   3659.4  12287.0
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 201804.2     2491.9   80.98   <2e-16 ***
## Experience    2395.5      126.8   18.89   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6814 on 48 degrees of freedom
## Multiple R-squared:  0.8814, Adjusted R-squared:  0.879
## F-statistic: 356.8 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
anova(tr_model)
```

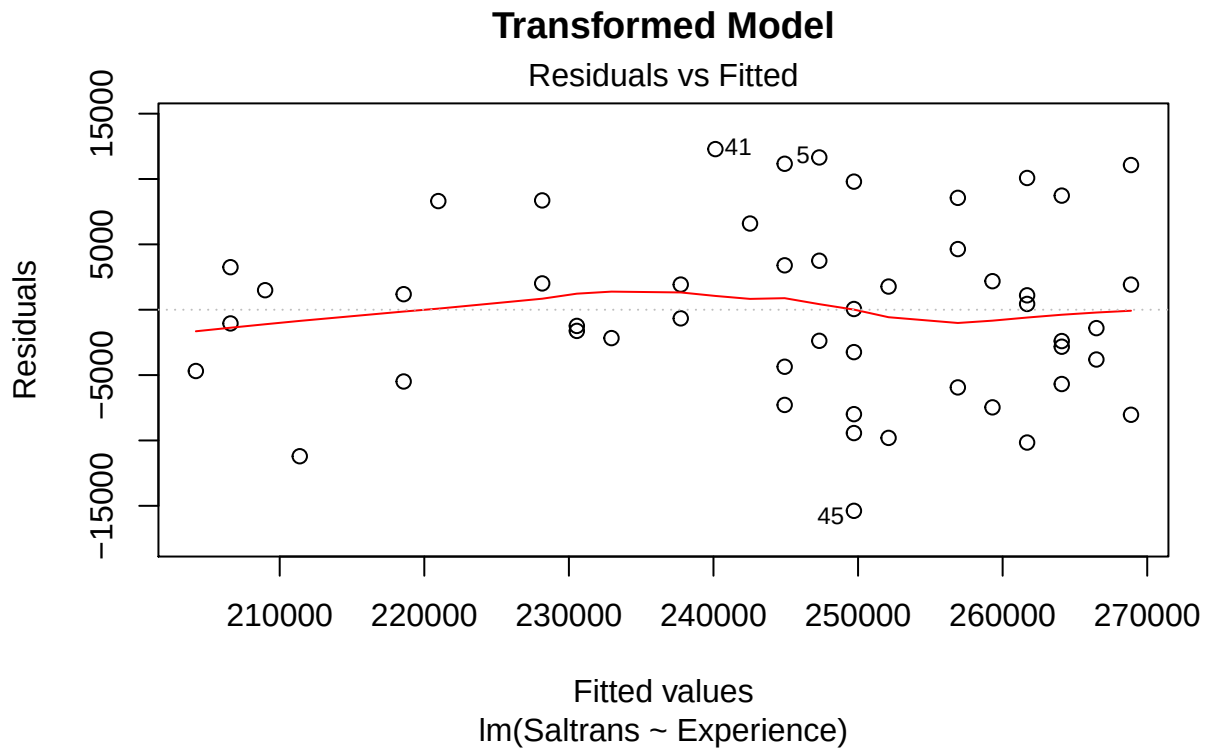
```
## Analysis of Variance Table
##
## Response: Saltrans
##            Df     Sum Sq    Mean Sq F value    Pr(>F)
## Experience  1 1.6569e+10 1.6569e+10  356.81 < 2.2e-16 ***
## Residuals  48 2.2290e+09 4.6437e+07
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Let's double check that the variance has been stabilized.

```
plot(model, which=1, main="Original Model")
```




```
plot(tr_model, which=1, main="Transformed Model")
```



# Grouped pH measurement data

## Obtaining the weights

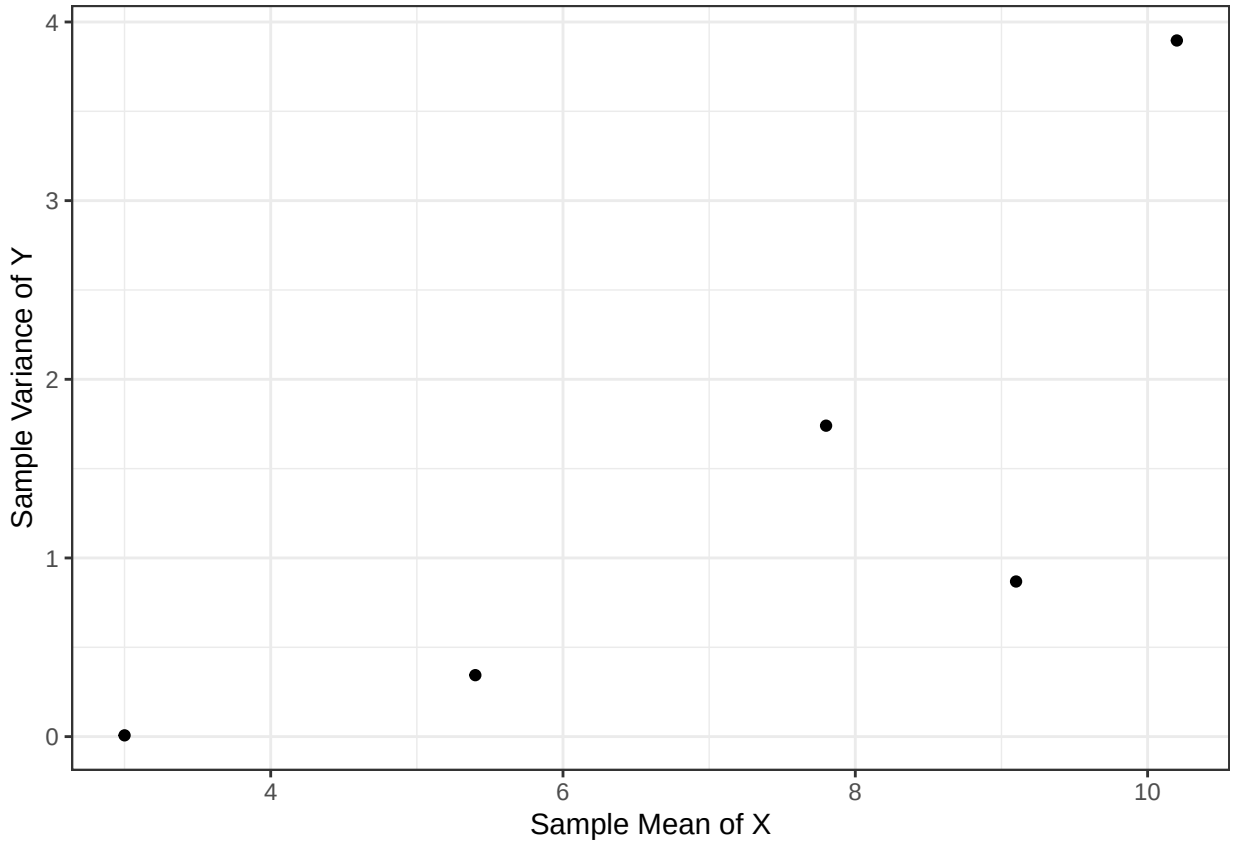
```
modelw <- read.table("./weightmodel.txt", header=TRUE)
```

To determine weights, the observations were divided into five groups with similar values in the predictor variable.

## Visualization

We wish to determine a relationship between the sample variances of the response against the sample means of the predictor.

```
ggplot(modelw, aes(x=MeanX, y=VarianceY))+
  geom_point()+
  labs(x="Sample Mean of X", y="Sample Variance of Y")
```




It looks like there is a quadratic relationship.

## Fit the model

```
modelw <- mutate(modelw,
                 MeanXsq = MeanX^2)

model <- lm(VarianceY ~ MeanX + MeanXsq, data=modelw)
summary(model)
```

```
##
## Call:
## lm(formula = VarianceY ~ MeanX + MeanXsq, data = modelw)
```

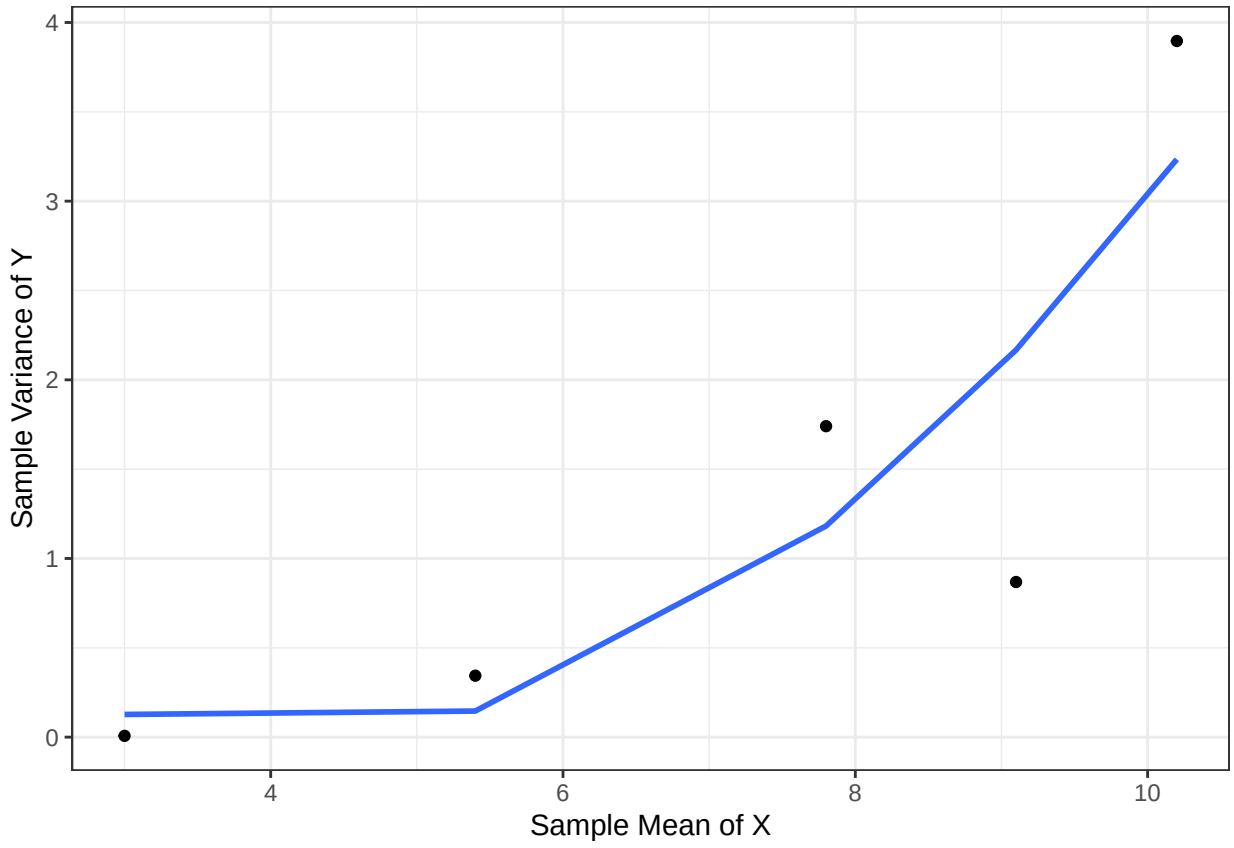
```
##
## Residuals:
##       1       2       3       4       5
## -0.1198  0.1980  0.5586 -1.2990  0.6621
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.53291    3.78395   0.405    0.725
## MeanX       -0.73343    1.28494  -0.571    0.626
## MeanXsq      0.08826    0.09666   0.913    0.458
##
## Residual standard error: 1.116 on 2 degrees of freedom
## Multiple R-squared:  0.7427, Adjusted R-squared:  0.4853
## F-statistic: 2.886 on 2 and 2 DF,  p-value: 0.2573
```

The equation of this fitted model is:

$$s_k^2 = 1.53291 - 0.73343\overline{x}_k + 0.08826\overline{x}_k^2$$

```
model_aug <- augment(model)

ggplot(model_aug, aes(x=MeanX))+
  geom_point(aes(y=VarianceY))+
  geom_line(aes(y=.fitted), colour="#3366FF", size=1)+
  labs(x="Sample Mean of X", y="Sample Variance of Y")
```



# pH measurement data

```
wls <- read.table("./weighted.txt", header=TRUE)
```

## Fit the non-weighted model

```
nw_model <- lm(MeasuredpH ~ ActualpH, data=wls)
summary(nw_model)
```

```
##
## Call:
## lm(formula = MeasuredpH ~ ActualpH, data = wls)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.0928 -0.6087 -0.0473  1.1256  2.4238
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.57895    0.67919  -0.852      0.4
## ActualpH     1.13540    0.08622  13.169 1.09e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.457 on 33 degrees of freedom
## Multiple R-squared:  0.8401, Adjusted R-squared:  0.8353
## F-statistic: 173.4 on 1 and 33 DF,  p-value: 1.089e-14
```

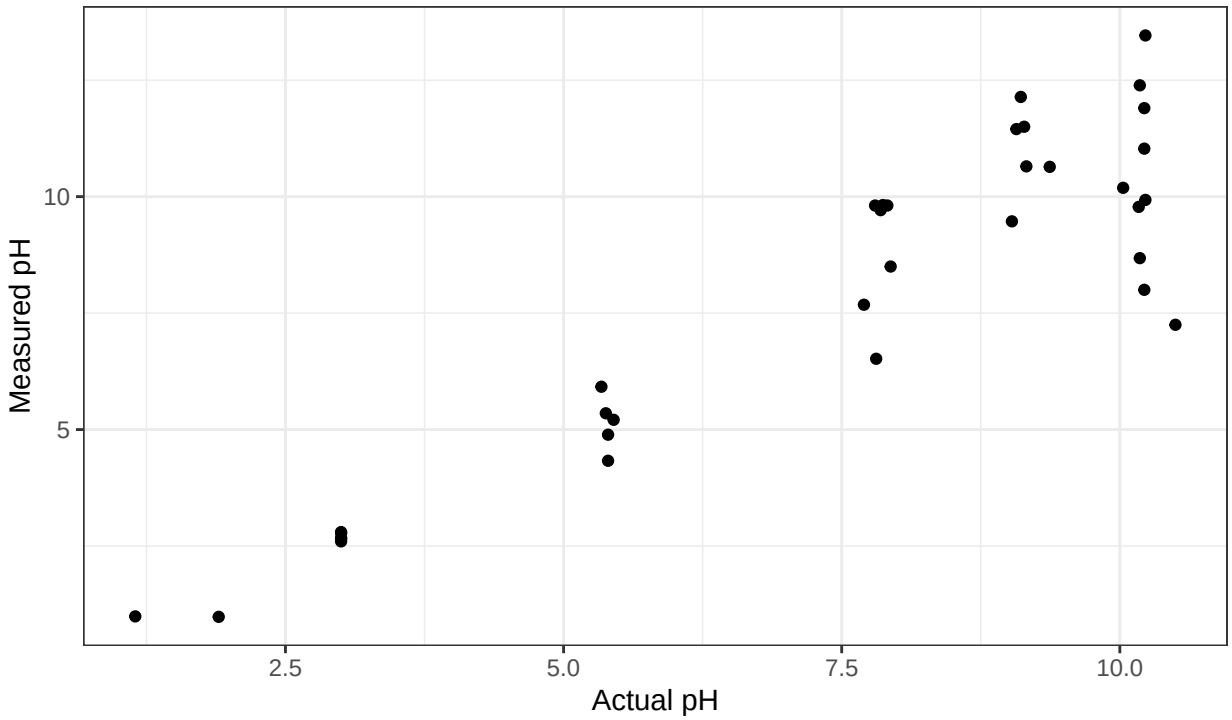
```
anova(nw_model)
```

```
## Analysis of Variance Table
##
## Response: MeasuredpH
##           Df Sum Sq Mean Sq F value    Pr(>F)
## ActualpH   1 367.95  367.95  173.42 1.089e-14 ***
## Residuals 33  70.02    2.12
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

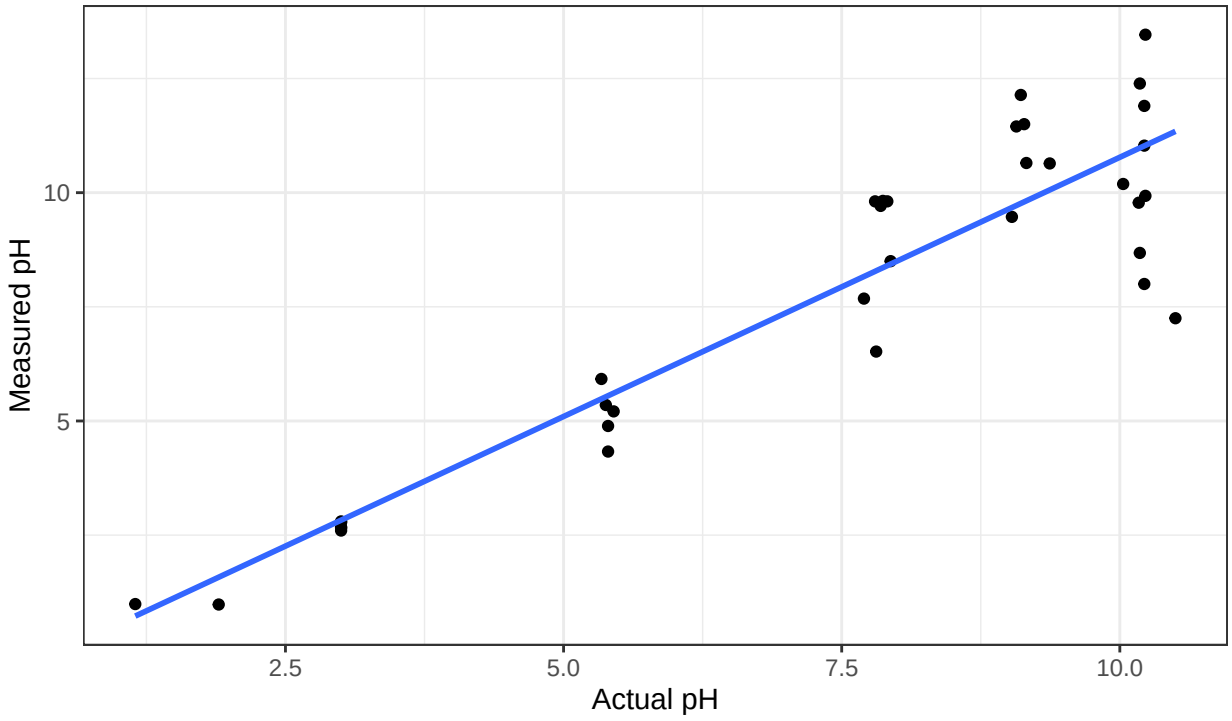
## Visualization

```
nw_model_aug <- augment(nw_model)
```

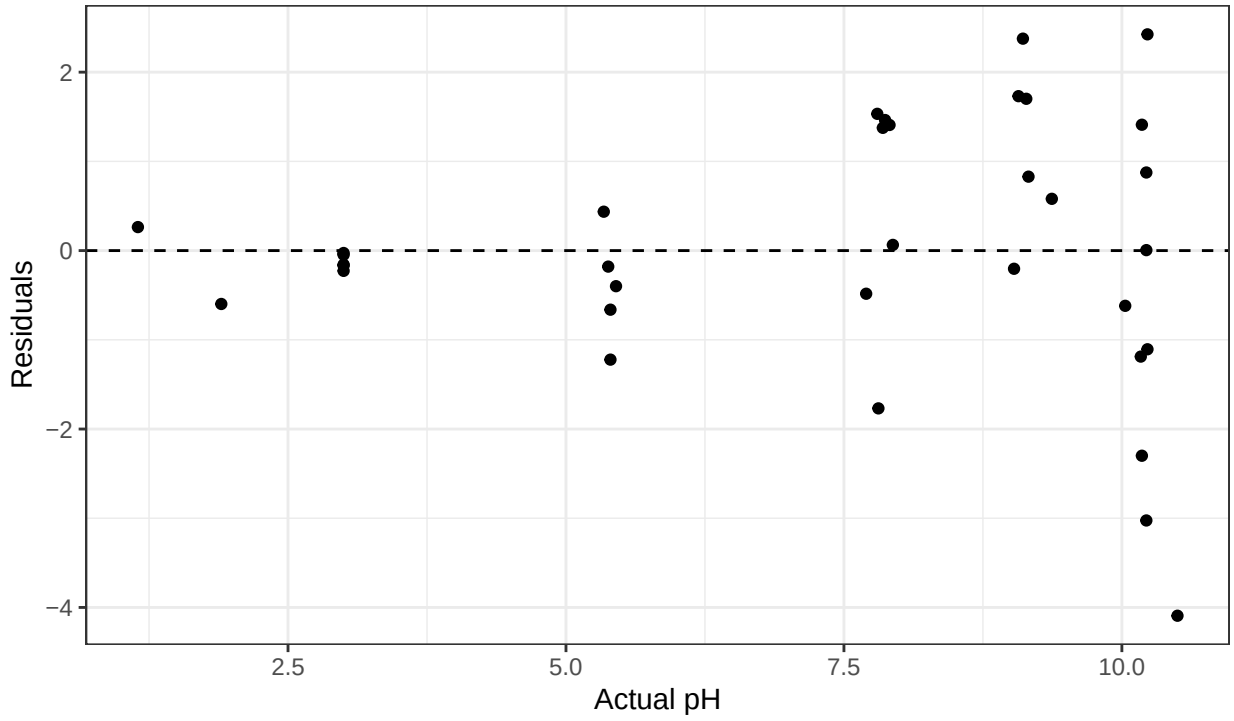
```
ggplot(nw_model_aug, aes(x=ActualpH, y=MeasuredpH))+
  geom_point()+
  labs(x="Actual pH", y="Measured pH")
```



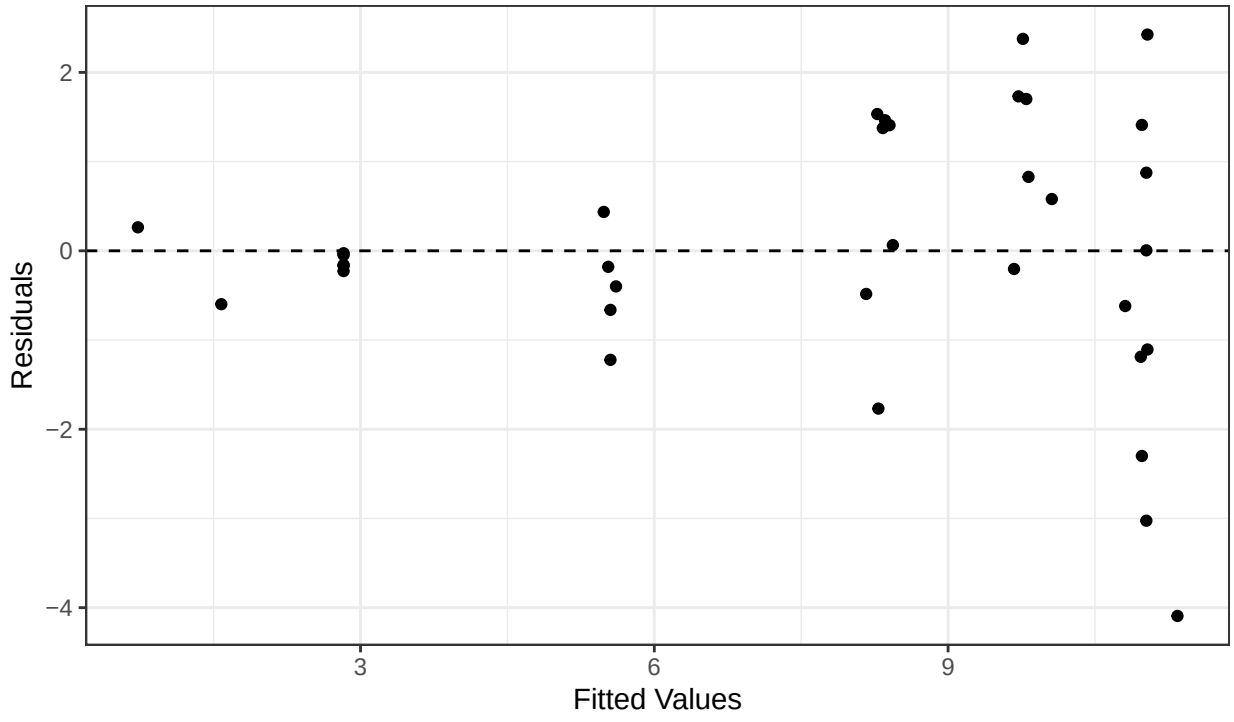

```
ggplot(nw_model_aug, aes(x=ActualpH))+
  geom_point(aes(y=MeasuredpH))+
  geom_line(aes(y=.fitted), colour="#3366FF", size=1)+
  labs(x="Actual pH", y="Measured pH")
```



```
ggplot(nw_model_aug, aes(x=ActualpH, y=.resid))+
  geom_point()+
  geom_hline(aes(yintercept=0), linetype=2)+
  labs(x="Actual pH", y="Residuals")
```




```
ggplot(nw_model_aug, aes(x=.fitted, y=.resid))+
  geom_point()+
  geom_hline(aes(yintercept=0), linetype=2)+
  labs(x="Fitted Values", y="Residuals")
```



### Compute the weights

```
wls <- mutate(wls,
              w = 1/(1.5329 - 0.73334*ActualpH + 0.0883*ActualpH^2))
wls$w
```

```
##  [1] 1.2401782 2.1818959 7.8382192 7.8382192 7.8382192 7.8382192 7.8382192
##  [8] 7.4188445 6.9773290 6.7708474 6.7708474 6.2921674 0.8916717 0.8438676
## [15] 0.8392992 0.8213905 0.8126497 0.7955805 0.7831281 0.4737313 0.4660925
## [22] 0.4586367 0.4531614 0.4495655 0.4142553 0.3117592 0.3107289 0.3066583
## [29] 0.3066583 0.3066583 0.3107289 0.2802765 0.3056531 0.3267377 0.3056531
```

### Fit the weighted model

```
w_model <- lm(MeasuredpH ~ ActualpH, weight=w, data=wls)
summary(w_model)
```

```
##
## Call:
## lm(formula = MeasuredpH ~ ActualpH, data = wls, weights = w)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -2.7869 -0.5574  0.1530  0.9825  1.6371
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.8892     0.3004   -2.96  0.00565 **
## ActualpH      1.1649     0.0594   19.61  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.137 on 33 degrees of freedom
## Multiple R-squared:  0.921,  Adjusted R-squared:  0.9186
## F-statistic: 384.5 on 1 and 33 DF,  p-value: < 2.2e-16
```

```
anova(w_model)
```

```
## Analysis of Variance Table
##
## Response: MeasuredpH
##           Df Sum Sq Mean Sq F value    Pr(>F)
## ActualpH   1 496.65  496.65  384.51 < 2.2e-16 ***
## Residuals 33  42.62    1.29
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

### Additional info from the weighted model

```
(w_model_aug <- augment(w_model))
```

```
## # A tibble: 35 x 9
##    MeasuredpH ActualpH `(weights)` .fitted   .resid .std.resid   .hat .sigma
##         <dbl>    <dbl>       <dbl>   <dbl>    <dbl>      <dbl>  <dbl>  <dbl>
##  1       0.99     1.15        1.24   0.450  0.540       0.544  0.0550   1.15
##  2       0.98     1.9         2.18   1.32  -0.344      -0.463  0.0691   1.15
##  3       2.6      3           7.84   2.61  -0.00536    -0.0143 0.145    1.15
##  4       2.67     3           7.84   2.61   0.0646      0.172  0.145    1.15
```

```
##  5       2.66     3          7.84   2.61   0.0546      0.146  0.145    1.15
##  6       2.78     3          7.84   2.61   0.175       0.465  0.145    1.15
##  7       2.8      3          7.84   2.61   0.195       0.519  0.145    1.15
##  8       5.92     5.34       7.42   5.33   0.589       1.48   0.0942   1.12
##  9       5.35     5.38       6.98   5.38  -0.0277     -0.0676 0.0897   1.15
## 10       4.33     5.4        6.77   5.40  -1.07       -2.57   0.0876   1.03
## # ... with 25 more rows, and 1 more variable: .cooksd <dbl>
```

## Compare non-weighted and weighted models

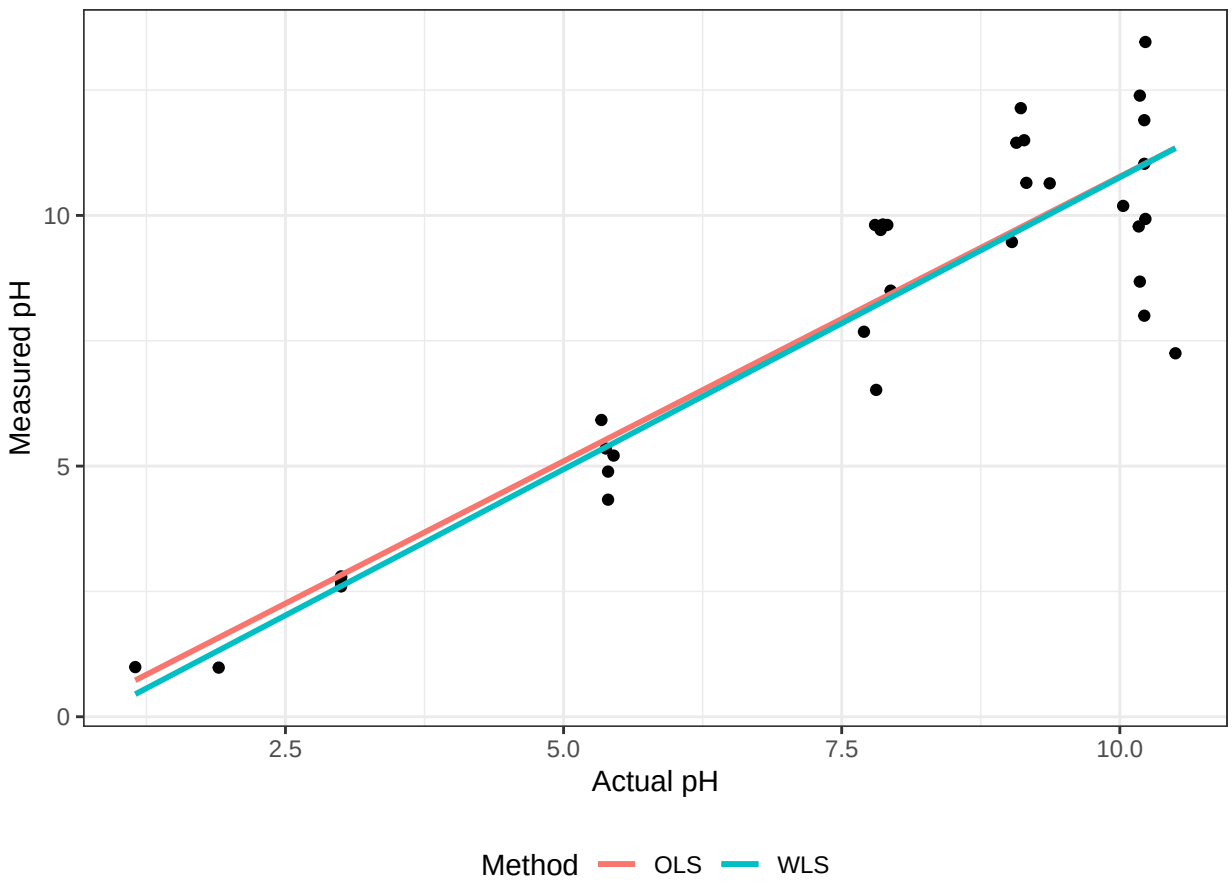
### Data prep

```
# Create new variables indicating method used
nw_model_aug <- mutate(nw_model_aug,
                       Method = "OLS")

w_model_aug <- mutate(w_model_aug,
                      Method = "WLS")
```

### Make the plot

```
ggplot(data=NULL, aes(x=ActualpH))+
  geom_point(data=wls, aes(y=MeasuredpH))+
  geom_line(data=nw_model_aug, aes(y=.fitted, colour=Method), size=1)+
  geom_line(data=w_model_aug, aes(y=.fitted, colour=Method), size=1)+
  labs(x="Actual pH", y="Measured pH")+
  theme(legend.position="bottom")
```



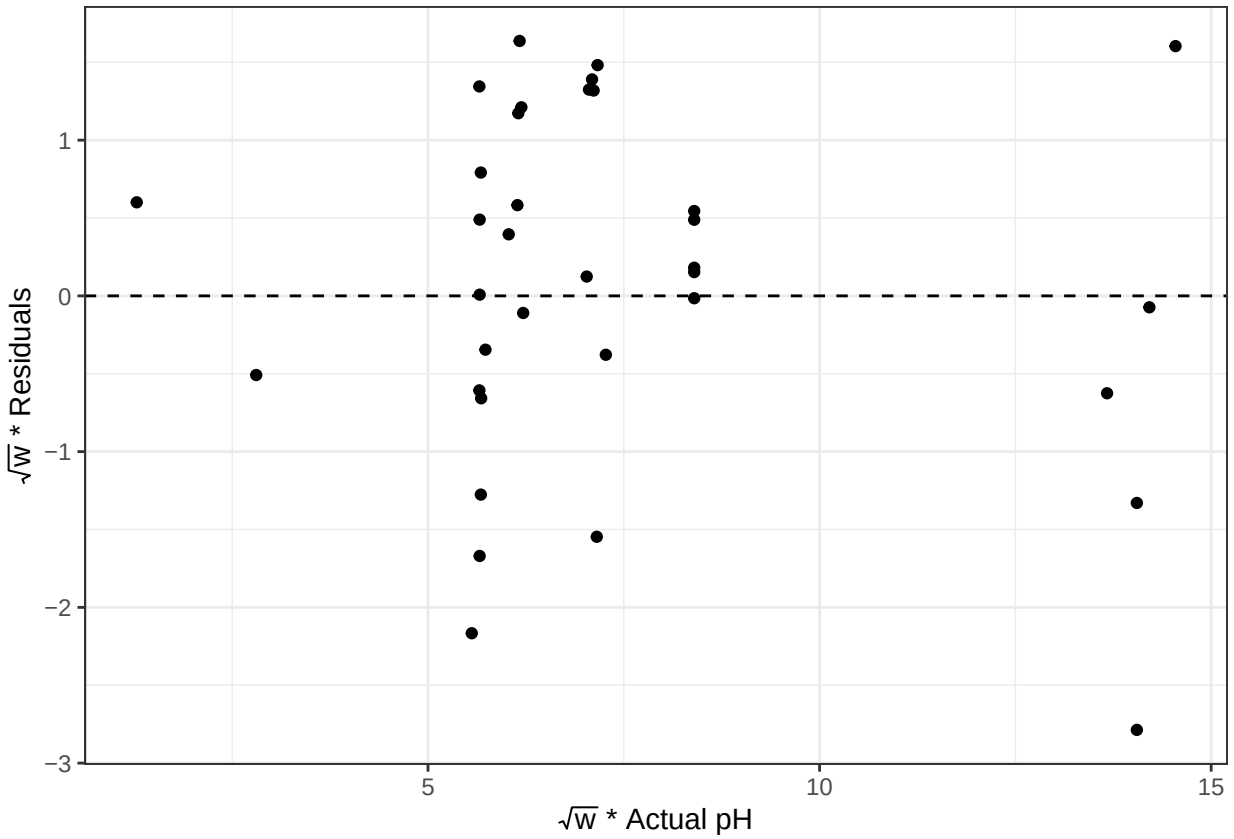
## Diagnostics

### Data prep

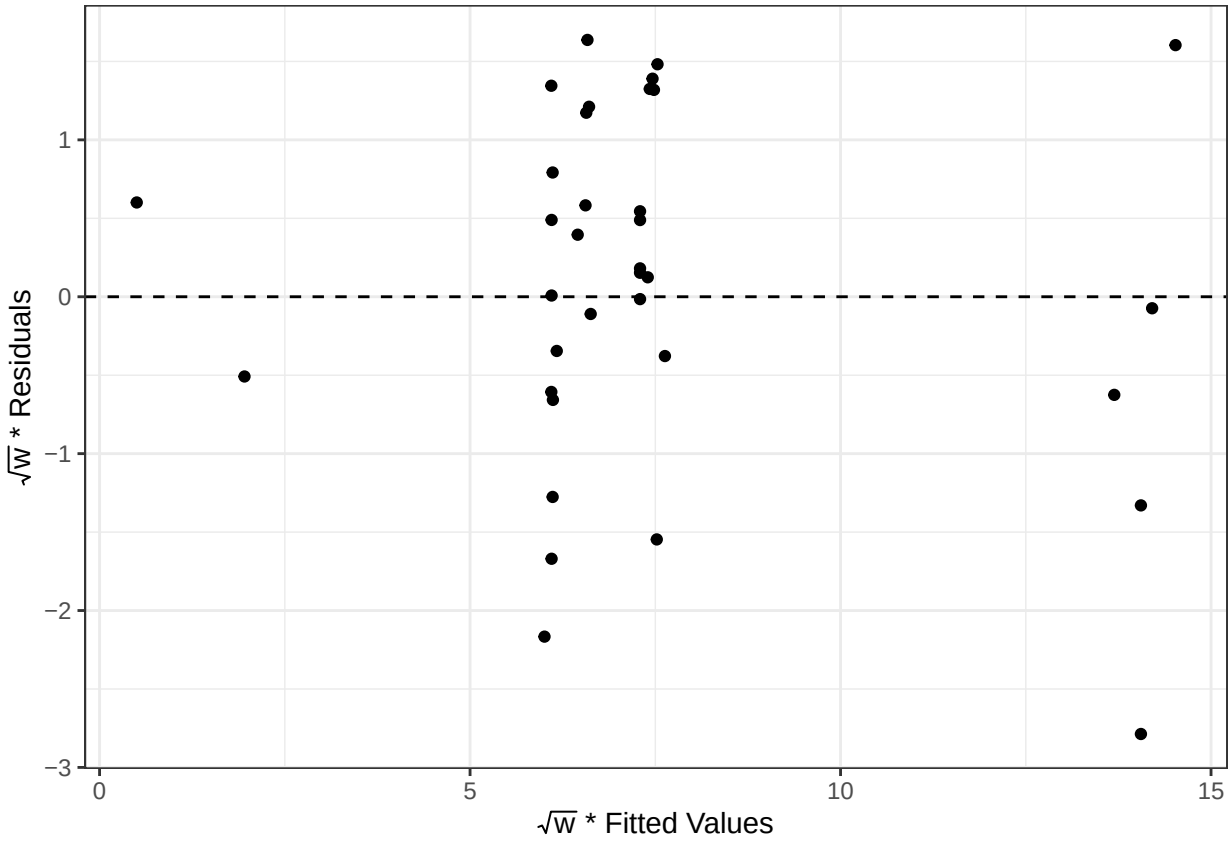
```
w_model_aug <- mutate(w_model_aug,
                      rtw = sqrt(`(weights)`),
                      rtwActualpH = rtw * ActualpH,
                      rtwMeasuredpH = rtw * MeasuredpH,
                      rtwRes = rtw * .resid,
                      rtwFits = rtw * .fitted)
```

### Plots

```
ggplot(w_model_aug, aes(x=rtwActualpH, y=rtwRes))+
  geom_point()+
  geom_hline(aes(yintercept=0), linetype=2)+
  labs(x=expression(sqrt("w") ~ "* Actual pH"),
       y=expression(sqrt("w") ~ "* Residuals"))
```



```
ggplot(w_model_aug, aes(x=rtwFits, y=rtwRes))+
  geom_point()+
  geom_hline(aes(yintercept=0), linetype=2)+
  labs(x=expression(sqrt("w") ~ "* Fitted Values"),
       y=expression(sqrt("w") ~ "* Residuals"))
```




We no longer have the issue of a non-constant error term variance.

# Inventory data

```
tseries <- read.table("./computer.txt", header=TRUE)
```

## Fit SLR

```
model <- lm(Firm ~ Industry, data=tseries)
summary(model)
```

```
##
## Call:
## lm(formula = Firm ~ Industry, data = tseries)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.91277 -0.67136  0.09514  0.53886  1.80259
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -7.739      7.175  -1.079    0.299
## Industry      53.953      3.520  15.329 3.82e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9543 on 14 degrees of freedom
## Multiple R-squared:  0.9438, Adjusted R-squared:  0.9398
## F-statistic:   235 on 1 and 14 DF,  p-value: 3.818e-10
```

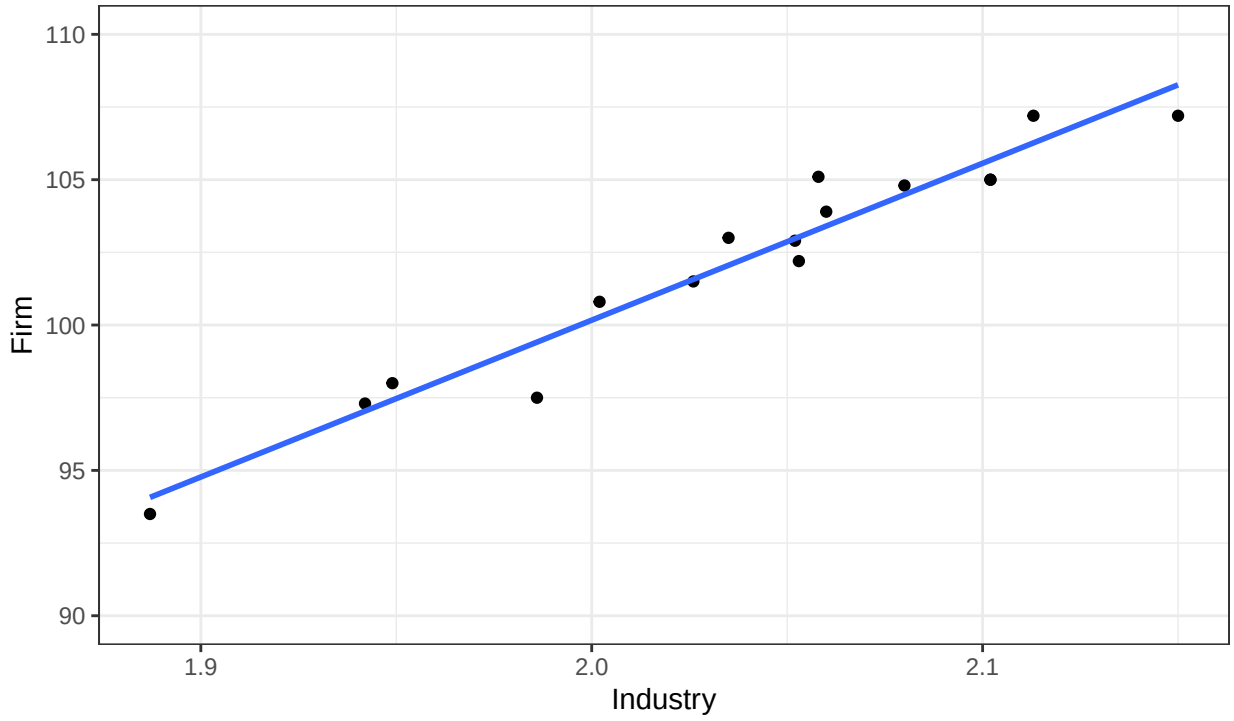
```
anova(model)
```

```
## Analysis of Variance Table
##
## Response: Firm
##           Df Sum Sq Mean Sq F value    Pr(>F)
## Industry   1 214.00 213.995  234.98 3.818e-10 ***
## Residuals 14  12.75   0.911
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

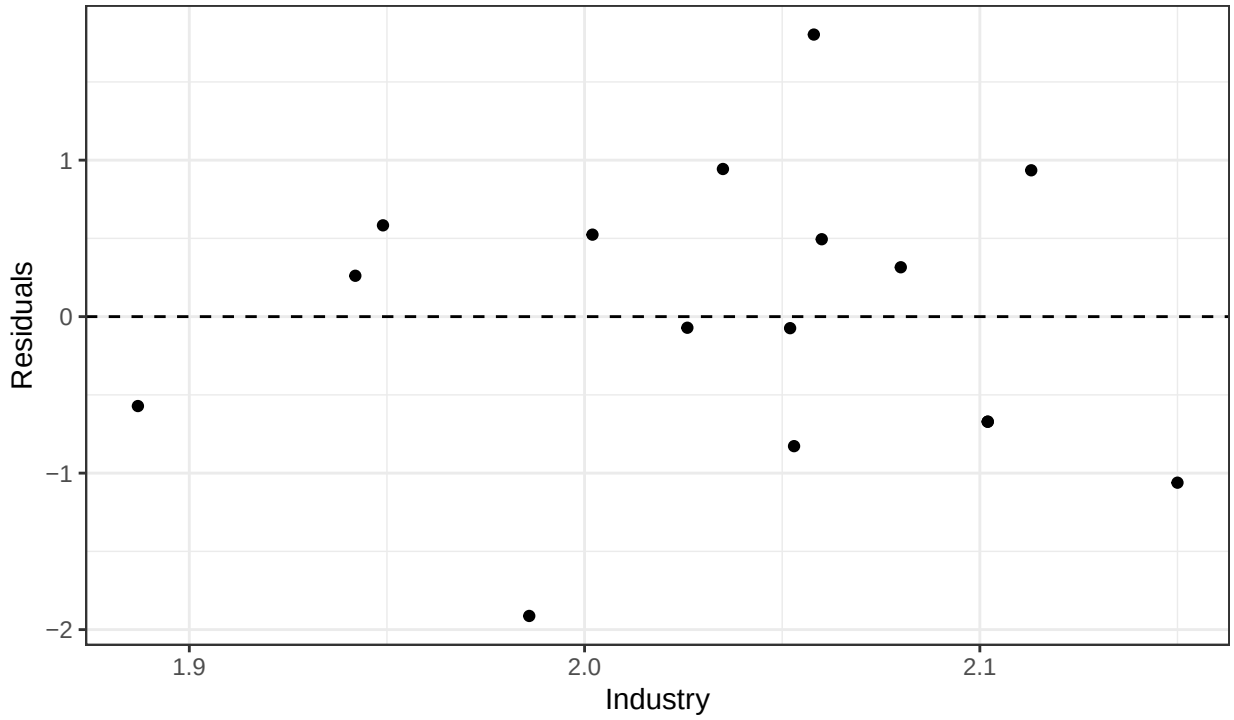
## Diagnostic plots

```
model_aug <- augment(model)
```

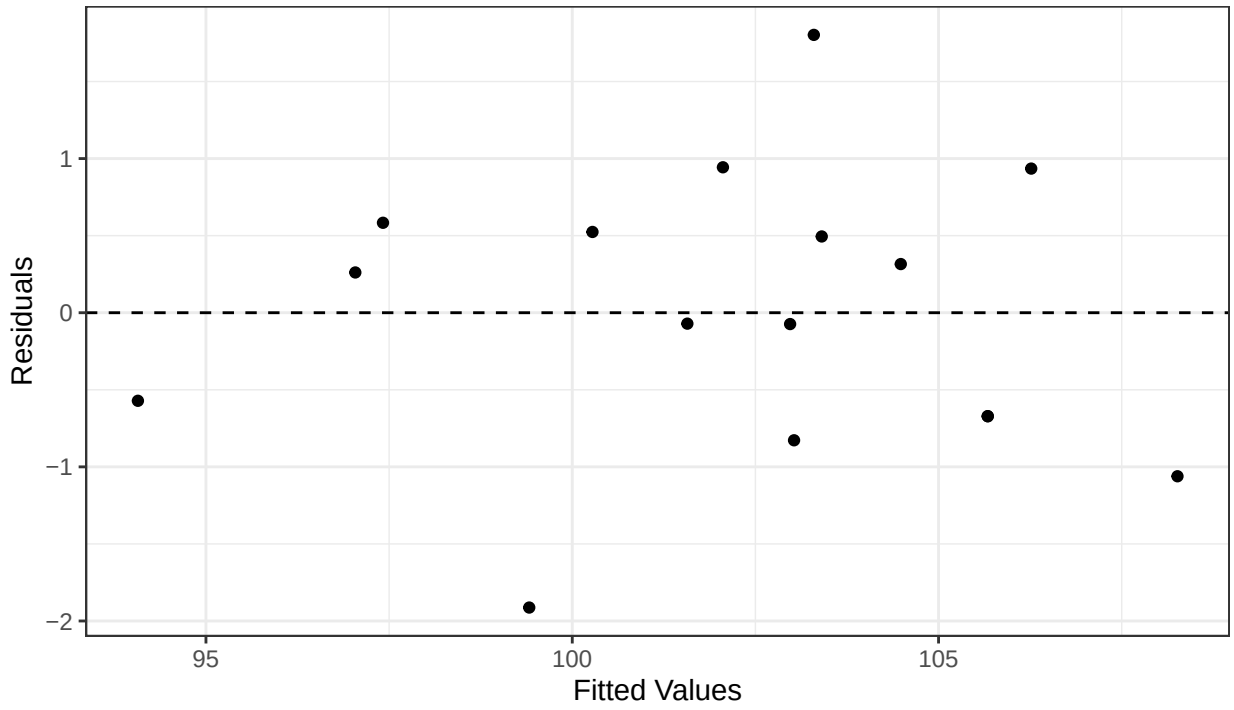
```
ggplot(model_aug, aes(x=Industry))+
  geom_point(aes(y=Firm))+
  geom_line(aes(y=.fitted), colour="#3366FF", size=1)+
  coord_cartesian(ylim=c(90, 110))
```



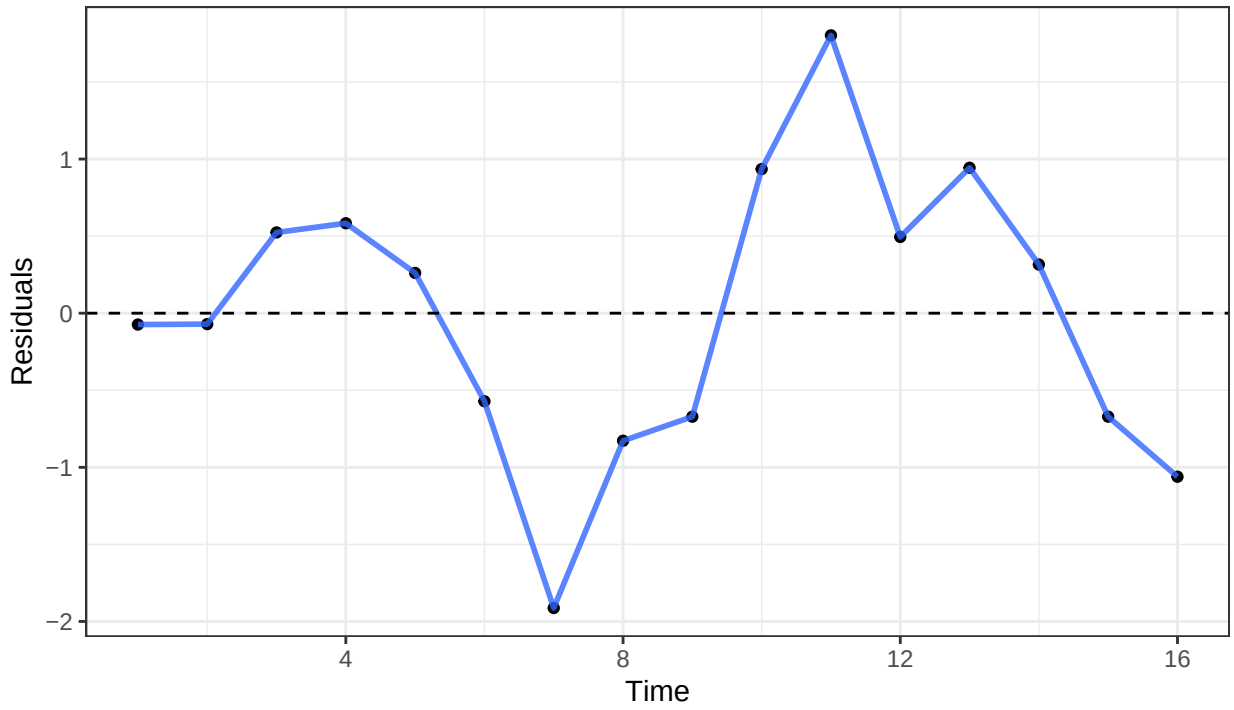

```
ggplot(model_aug, aes(x=Industry, y=.resid))+
  geom_point()+
  geom_hline(aes(yintercept=0), linetype=2)+
  labs(y="Residuals")
```



```r
ggplot(model_aug, aes(x=.fitted, y=.resid))+
  geom_point()+
  geom_hline(aes(yintercept=0), linetype=2)+
  labs(x="Fitted Values", y="Residuals")
```

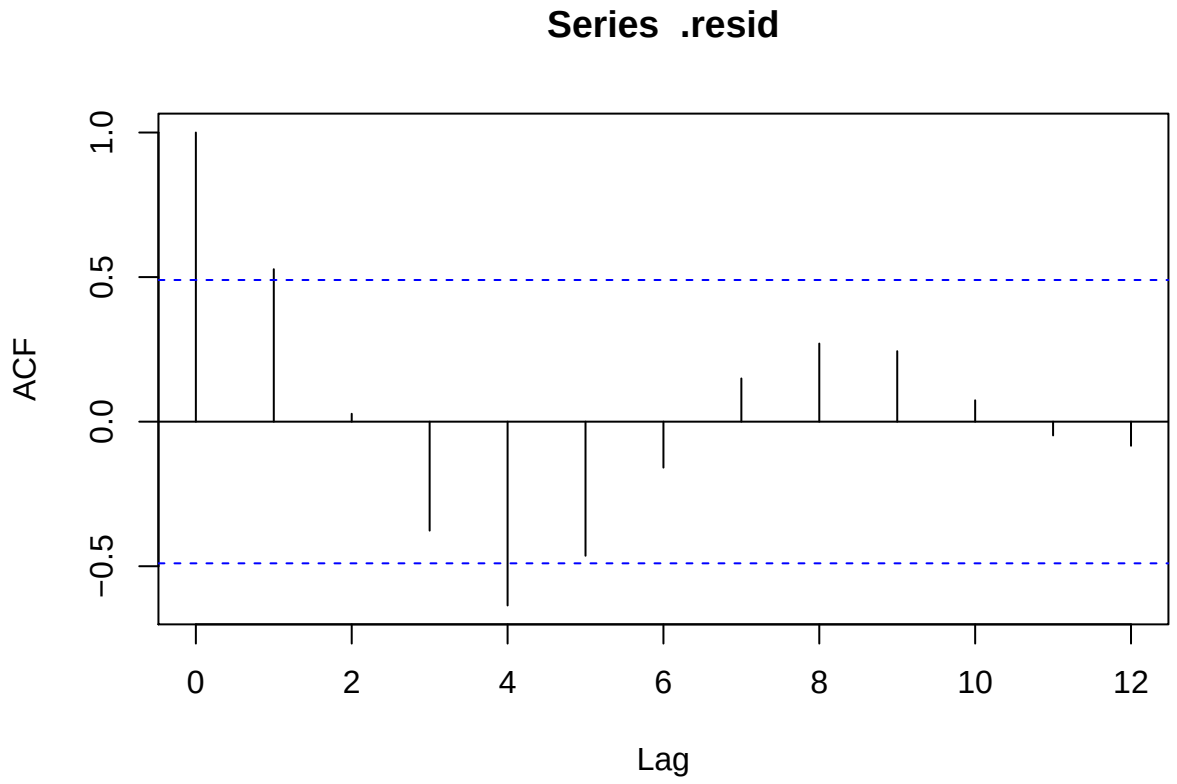



```r
ggplot(model_aug, aes(x=1:nrow(model_aug), y=.resid))+
  geom_point()+
  geom_line(colour="#3366FF", alpha=0.8, size=1)+
  geom_hline(aes(yintercept=0), linetype=2)+
  labs(x="Time", y="Residuals")
```



### Check for autocorrelation (visually)

```
with(model_aug, acf(.resid, lag.max=12))
```




### Check for autocorrelation (hypothesis test)

```
set.seed(10)
dwt(model, alternative=c("positive"))
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1       0.5273293     0.8566024   0.003
##  Alternative hypothesis: rho > 0
```

A test for positive autocorrelation has hypotheses:

$$H_0 : \rho = 0 \quad \text{vs} \quad H_A : \rho > 0$$

With $\alpha = 0.05$, $n = 16 \approx 15$, and $p = 1$, we have

$$d_L = 1.08 \quad d_U = 1.36$$

- $D = 0.8566024$
- $D \not> d_U = 1.36$
- $D < d_L = 1.08$

We reject the null hypothesis at the 5% level of significance. We conclude that the error terms are positively autocorrelated.