Model Diagnostics and Remedial Measures in Simple Linear Regression

The linear regression model involves several assumptions. Among them are:

1. The errors, for each fixed value of x, have mean 0. This implies that the relationship as specified in the regression equation is appropriate.
2. The errors, for each fixed value of x, have constant variance. That is, the variation in the errors is theoretically the same regardless of the value of x (or $\hat{y}$).
3. The errors are independent.
4. The errors, for each fixed value of x, follow a normal distribution.

To assess assumptions 1 and 2, we can examine scatterplots of:

- y versus x.
- residuals versus fitted values, $\hat{y}$, or x.

Assumption 3 is assessed with an autocorrelation (ACF) plot of the residuals.

Assumption 4 is assessed with a normal probability plot, and is considered the least crucial of the assumptions.

We will see how to generate the relevant graphical displays to help us assess whether the assumptions are met.

For this tutorial, we will go over the electric utility example from the textbook. The textbook describes the data as:

An electric utility is interested in developing a model relating peak-hour demand (y) to total energy usage during the month (x). This is an important planning problem because while most customers pay directly for energy usage (in kilowatt-hours), the generation system must be large enough to meet the maximum demand imposed.

Download the data file, `electricity.csv`, from Collab and read the data in. Also load the `tidyverse` package.

```
library(tidyverse)
```

```
Data<-read.csv("electricity.csv", header=TRUE)
head(Data)
```

```
##   Customer x..kWh. y..kW.
## 1        1     679   0.79
## 2        2     292   0.44
## 3        3    1012   0.56
## 4        4     493   0.79
## 5        5     582   2.70
## 6        6    1156   3.64
```

Notice there is an extra column for the observation number, and the names of the columns are long and complicated. So we remove the first column and rename the 2nd and 3rd columns

```
##remove first column
Data<-Data[,-1]
##rename the remaining 2 columns
names(Data)<-c("Usage","Demand")
head(Data)
```

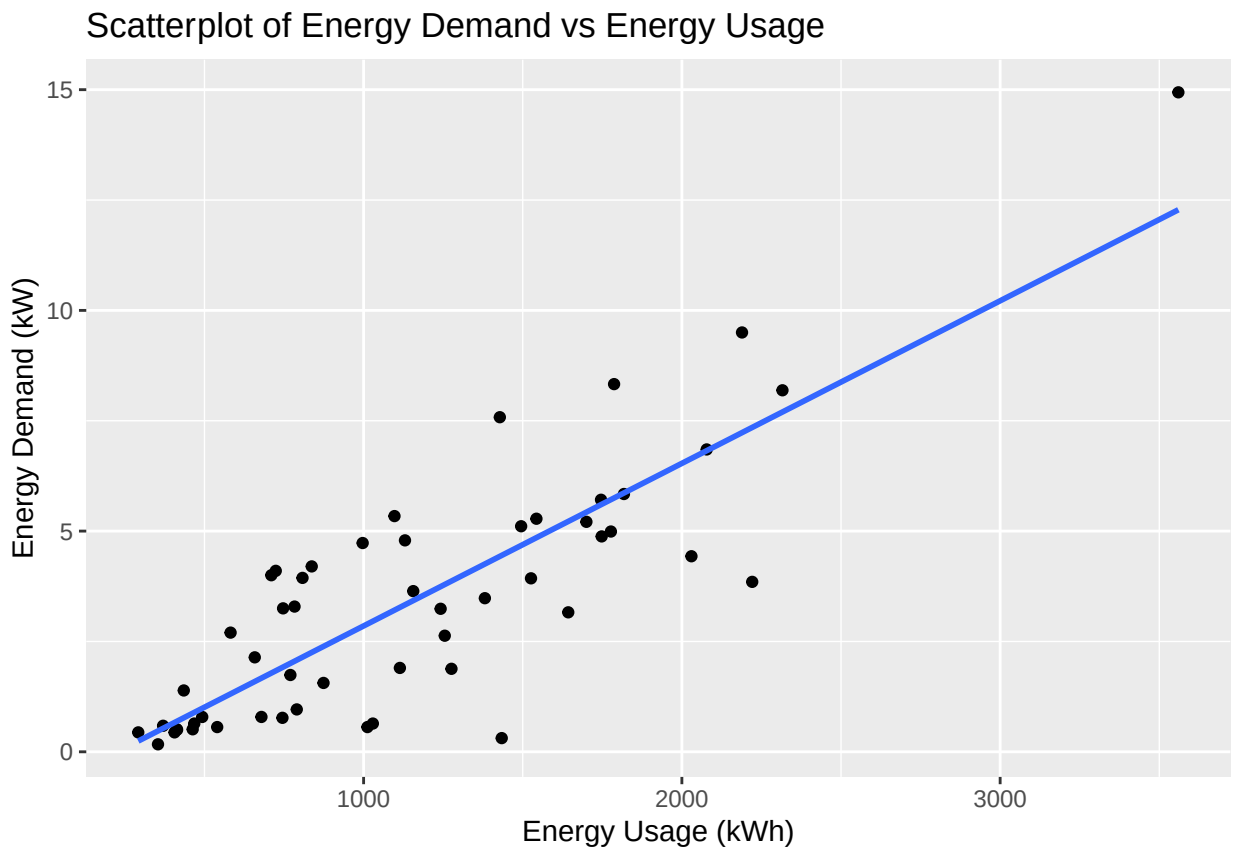
```
##   Usage Demand
## 1   679   0.79
## 2   292   0.44
## 3  1012   0.56
## 4   493   0.79
## 5   582   2.70
## 6  1156   3.64
```

1. Model Diagnostics with a Scatterplot

We can use a scatterplot of the response variable against the predictor to assess assumptions 1 and 2.

```
ggplot(Data, aes(x=Usage,y=Demand))+
  geom_point()+
  geom_smooth(method = "lm", se=FALSE)+
  labs(x="Energy Usage (kWh)",
       y="Energy Demand (kW)",
       title="Scatterplot of Energy Demand vs Energy Usage")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Notice we added an extra layer `geom_smooth` to the scatterplot. This layer overlays the regression line of `Demand` against `Usage` on our plot to aid in visualizing the assumptions.

To assess assumption 1, the data points should be evenly scattered on both sides of the regression line, as we move from left to right. We see this in the scatterplot, so assumption 1 is met.

To assess assumption 2, the vertical spread of the data points should be constant as we move from left to right. The spread seems to be increasing, so assumption 2 is not met.

Fairly often, when assessing regression assumptions, a residual plot is easier to visualize than a scatterplot.

Out of curiosity, to create a scatterplot using the `plot()` function from base R

```
plot(Data$Usage, Data$Demand,
     xlab="Energy Usage (kWh)",
     ylab="Energy Demand (kW)",
     main="Scatterplot of Energy Demand vs Energy Usage")
abline(lm(Demand~Usage, data=Data))
```

The `abline()` function overlays the regression line on the scatterplot.

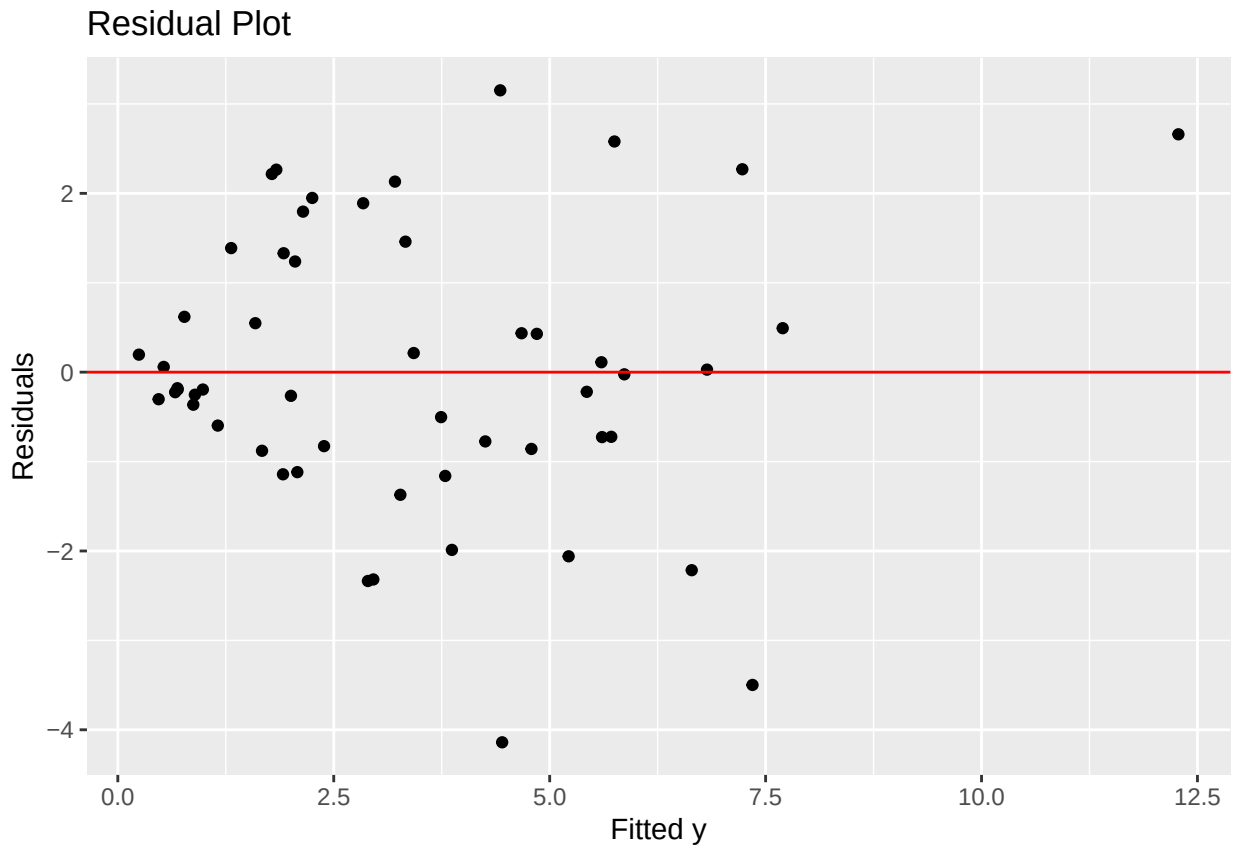
2. Model Diagnostics with a Residual Plot

A residual plot has $\hat{y}$ on the x-axis and the residuals, e, on the y-axis. So we fit the regression model using `lm()`, and then store the fitted values and residuals and add them to the data frame

```
##Fit a regression model
result<-lm(Demand~Usage, data=Data)
##store fitted y & residuals
yhat<-result$fitted.values
res<-result$residuals
##add to data frame
Data<-data.frame(Data,yhat,res)
```

Then, we create the residual plot

```
ggplot(Data, aes(x=yhat,y=res))+
  geom_point()+
  geom_hline(yintercept=0, color="red")+
  labs(x="Fitted y",
       y="Residuals",
       title="Residual Plot")
```



Notice we added a layer `geom_hline`. This overlays a red horizontal line where the y-axis is 0, to aid in assessing assumptions 1 and 2.

To assess assumption 1, the residuals should be evenly scattered on both sides of the red line, as we move from left to right. We see this in the residual plot, so assumption 1 is met.

To assess assumption 2, the vertical spread of the residuals should be constant as we move from left to right. The spread seems to be increasing, so assumption 2 is not met.

To create the residual plot using `plot()`

```
plot(yhat,res,
     xlab="Fitted y",
     ylab="Residuals",
     main="Residual Plot")
abline(h=0, col="red")
```

Notice that we obtained the same conclusions with the scatterplot and the residual plot.

Now that we know that assumption 2 is not met, we need to transform the response variable.

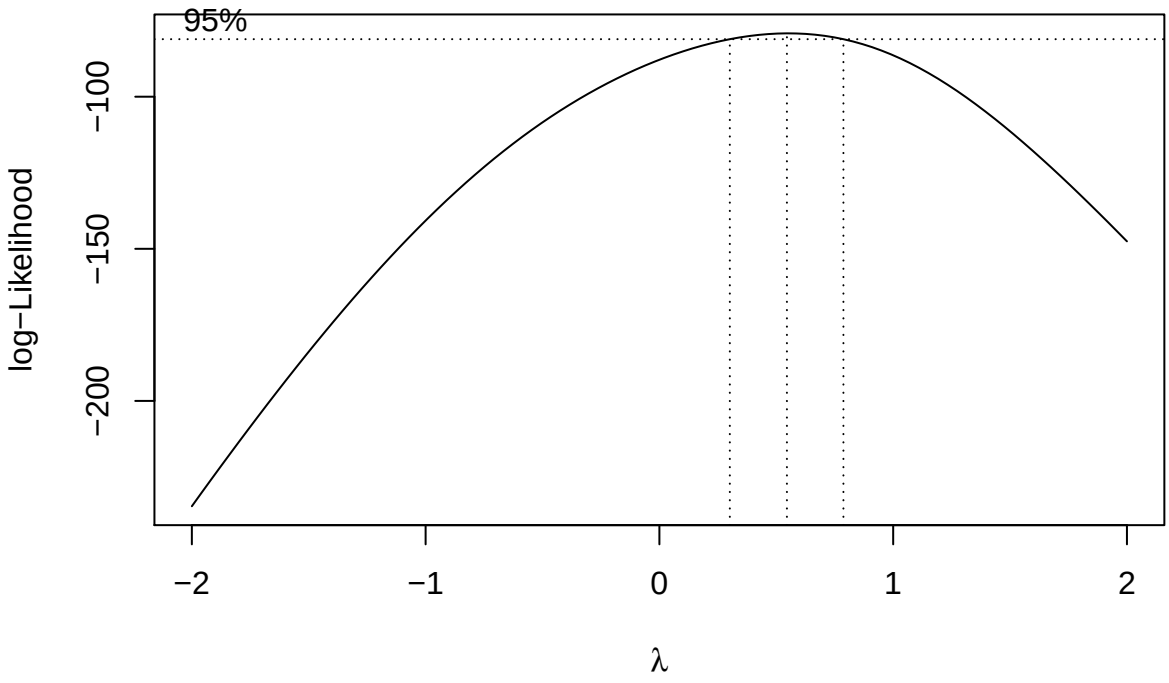
3. Box Cox Transformation on y

The Box Cox plot can be used to decide how exactly to transform the response variable. The transformation takes the form $y^* = y^\lambda$, with the value of λ to be chosen. If $\lambda = 0$, we perform a log transformation.

We will use the `boxcox()` function from the `MASS` package.

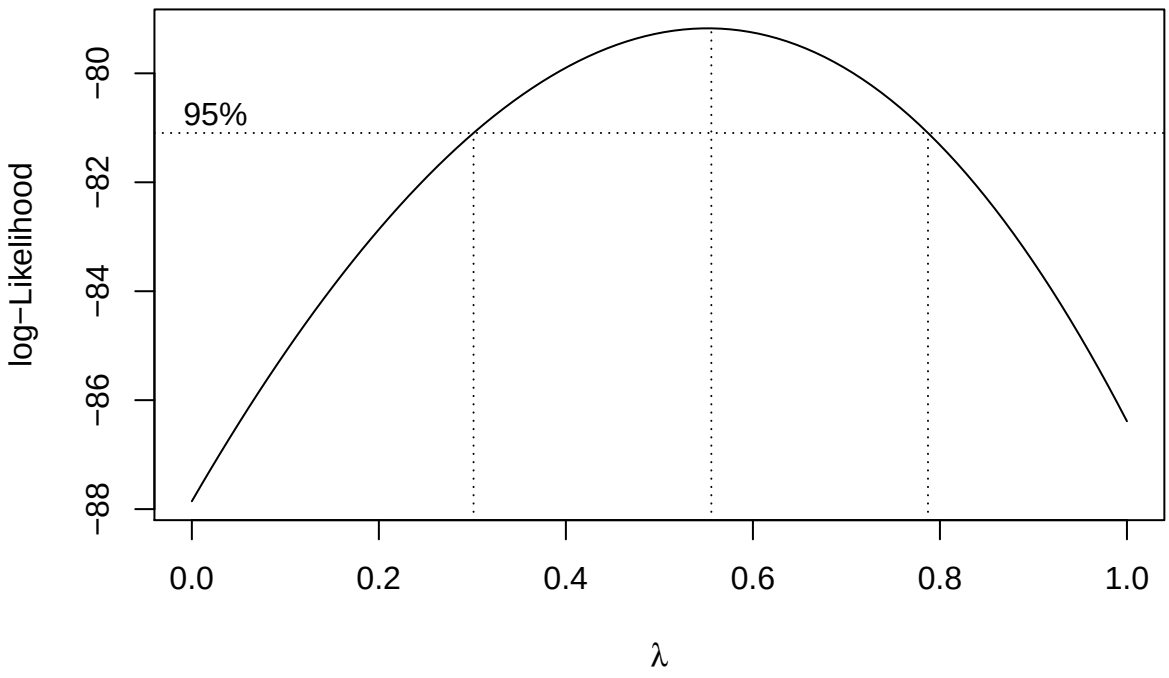
```
library(MASS)
```

```
boxcox(result)
```



We want to choose a value of λ between the 3 dotted vertical lines. We can "zoom in" on the plot to have a better idea about the value of λ we can use

```
boxcox(result, lambda = seq(0, 1, 1/10))
```



We can choose $\lambda = 0.5$, to square root the response variable to get $y^* = y^{0.5}$. We regress y^* against x

```
ystar<-(Data$Demand)^0.5
Data<-data.frame(Data,ystar)
result.ystar<-lm(ystar~Usage, data=Data)
```

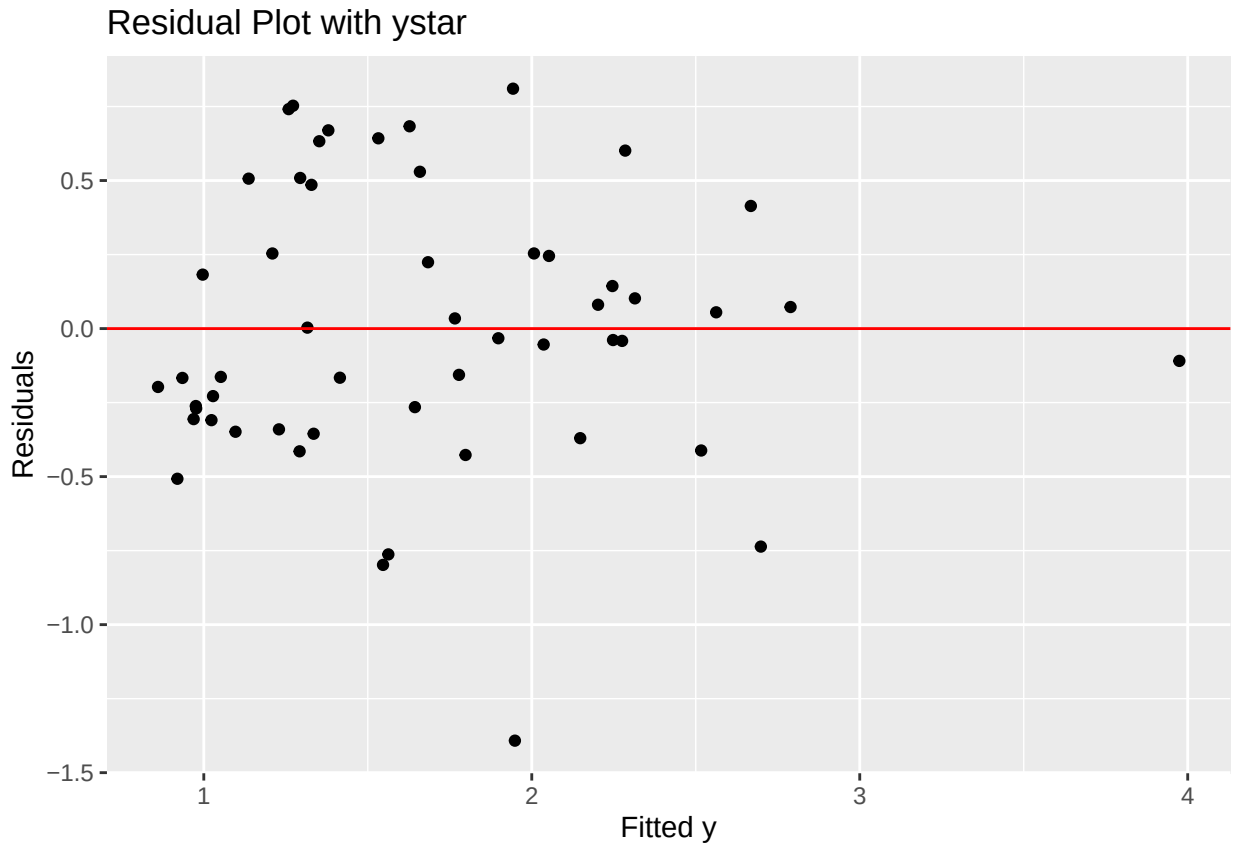
And create the residual plot just like earlier

```
##store fitted y & residuals
yhat2<-result.ystar$fitted.values
res2<-result.ystar$residuals

##add to data frame
Data<-data.frame(Data,yhat2,res2)

##residual plot with ystar
ggplot(Data, aes(x=yhat2,y=res2))+
  geom_point()+
  geom_hline(yintercept=0, color="red")+
  labs(x="Fitted y",
```

```
       y="Residuals",
       title="Residual Plot with ystar")
```



We need to reassess assumptions 1 and 2.

The residuals should be evenly scattered on both sides of the red horizontal line, as we move from left to right. We see this in the residual plot, so assumption 1 is met.

The vertical spread of the residuals should be constant as we move from left to right. The spread seems to be constant, so assumption 2 is met.

So we no longer need to transform either variable. If you want to, you can create a Box Cox plot for this new regression, just to confirm we do not need to transform the response anymore

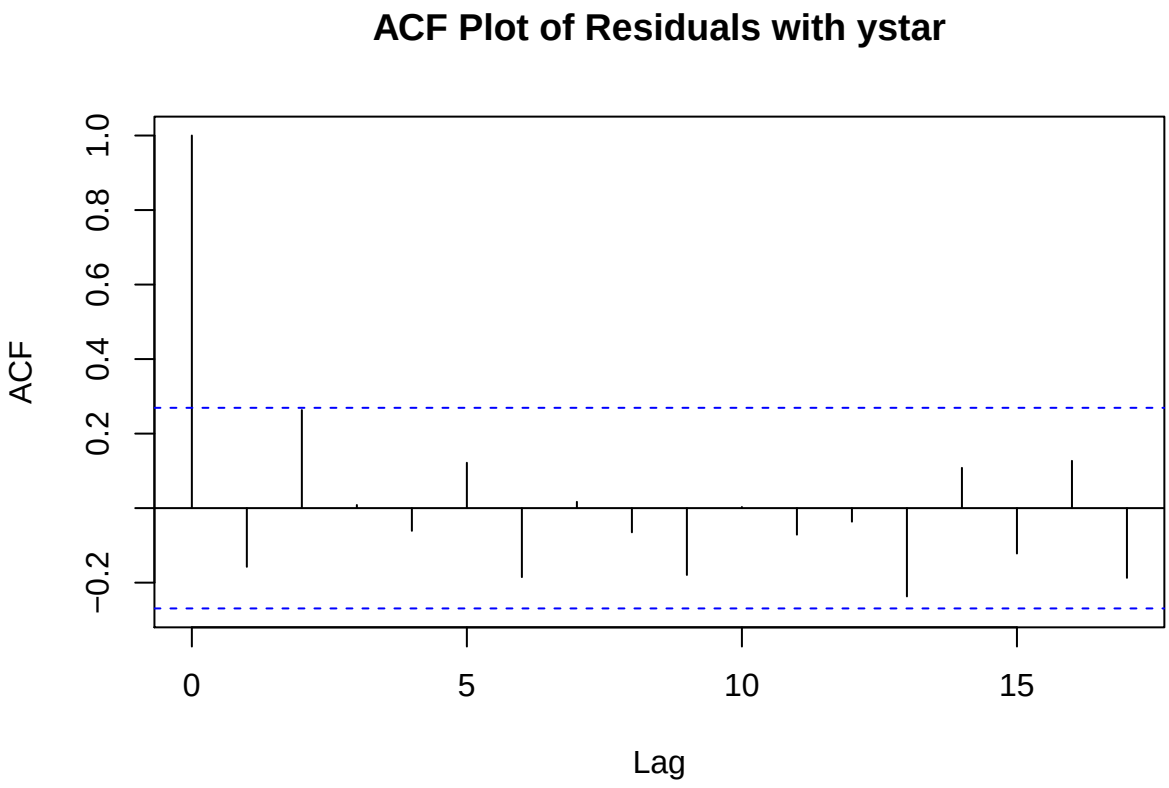
```
boxcox(result.ystar)
```

4. ACF Plot of Residuals

Now that we are satisfied that assumptions 1 and 2 are met, we need to verify that assumptions 3 and 4 are met.

For assumption 3, the independence of the errors, we use an autocorrelation function (ACF) plot. The ACF plot assesses if the residuals are correlated or not. The `acf()` function is used

```
acf(res2, main="ACF Plot of Residuals with ystar")
```



The ACF at lag 0 is always 1, by definition (the correlation of the vector of residuals with itself). Insignificant ACFs at lag 1 and greater indicate that the residuals are uncorrelated, so we have no evidence that assumption 3 is not met.

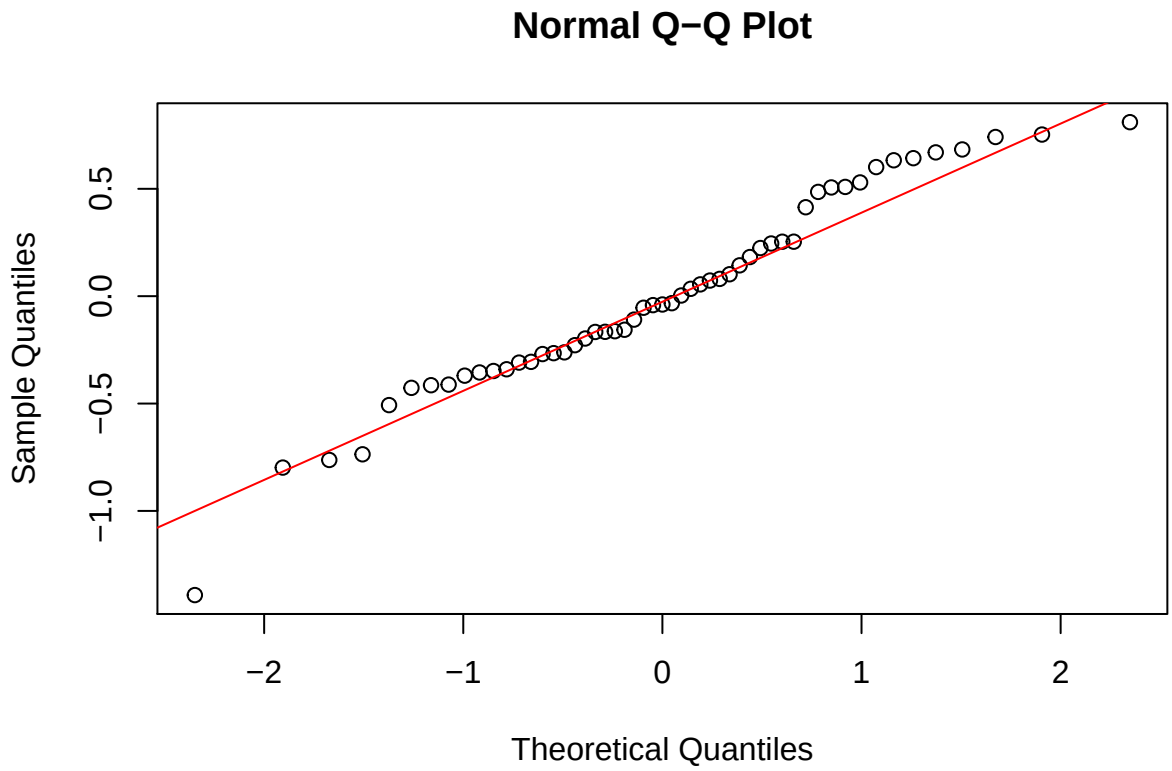
A by product of assumption 3 is that the observations are independent. So this assumption usually deals with how the observations were sampled. If they were randomly selected, we expect assumption 3 to hold. If there is already some dependence or structure in the observations, we expect assumption 3 to fail, and we need to consider models that do not assume independence in the observations.

5. QQ Plot of Residuals

To assess assumption 4, we produce a QQ plot (sometimes called a normal probability plot) of the residuals. The `qqnorm()` function produces a plot of the sample quantiles against the

theoretical quantiles. The red line produced by `qqline()` allows us to see how the sample quantiles match their theoretical values.

```
qqnorm(res2)
qqline(res2, col="red")
```



In this QQ plot, the sample quantiles fall pretty close to the red line, so assumption 4, the normality of the errors, is met.

Do note that the normality assumption is the least important assumption. Regression models are fairly robust to deviations from this assumption.

So our regression equation is

```
result.ystar
```

```
##
## Call:
## lm(formula = ystar ~ Usage, data = Data)
##
## Coefficients:
## (Intercept)        Usage
##   0.5822259    0.0009529
```

$y^* = 0.582 + 0.001x$, where $y^* = \sqrt{y}$.