Regression Diagnostics, Part 1

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \qquad \epsilon_i \sim N(0, \sigma^2)$$

$$\hat{\boldsymbol{\beta}} = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{Y} \qquad E[\hat{\boldsymbol{\beta}}] = \boldsymbol{\beta} \qquad Var[\hat{\boldsymbol{\beta}}] = \boldsymbol{\sigma}^2\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}$$

Thus far, we have assumed that the model errors follow a normal distribution with mean 0 and constant variance σ^2. This has allowed us to derive the distribution of tests about one parameter (t-tests) or tests about nested models (F-tests). If these assumptions are not true, we can still carry out these tests, however the results of these tests will not be valid. (Garbage In, Garbage Out)

We will look at a number of ways of checking these two assumptions,

- **Normality of Errors**
  - Histogram of Residuals
  - Q-Q Plot
  - Shapiro–Wilk Test
- **Constant Variance**
  - Fitted vs Residual Plot
  - Breusch–Pagan Test

Additional information can be found in Chapter 6 of Faraway's *Linear Models with R*. (Or Chapter 7 of *Practical Regression and ANOVA Using R* which can be found on Compass.)

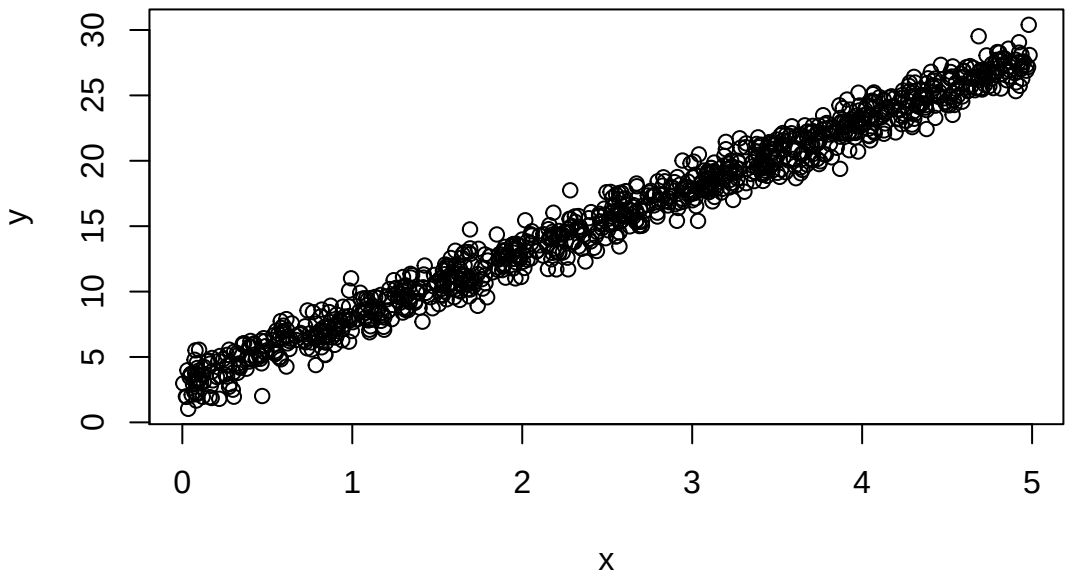
Let's consider the following true model,

$$Y_i = 3 + 5x_i + \epsilon_i \qquad \epsilon_i \sim N(0, 1)$$

We can simulate 1000 observation from this model in `R`,

```
x <- runif(1000)*5
y <- 3 + 5*x + rnorm(1000,0,1)
```

```
plot(x,y)
```




```
fit <- lm(y ~ x)
summary(fit)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3234 -0.6661 -0.0037  0.7021  3.3468
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.99001    0.06582   45.43   <2e-16 ***
## x            4.99735    0.02241  222.99   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.007 on 998 degrees of freedom
## Multiple R-squared:  0.9803, Adjusted R-squared:  0.9803
## F-statistic: 4.973e+04 on 1 and 998 DF,  p-value: < 2.2e-16
```

Note, Rs estimated coefficients using the simulated data are very close to the truth.

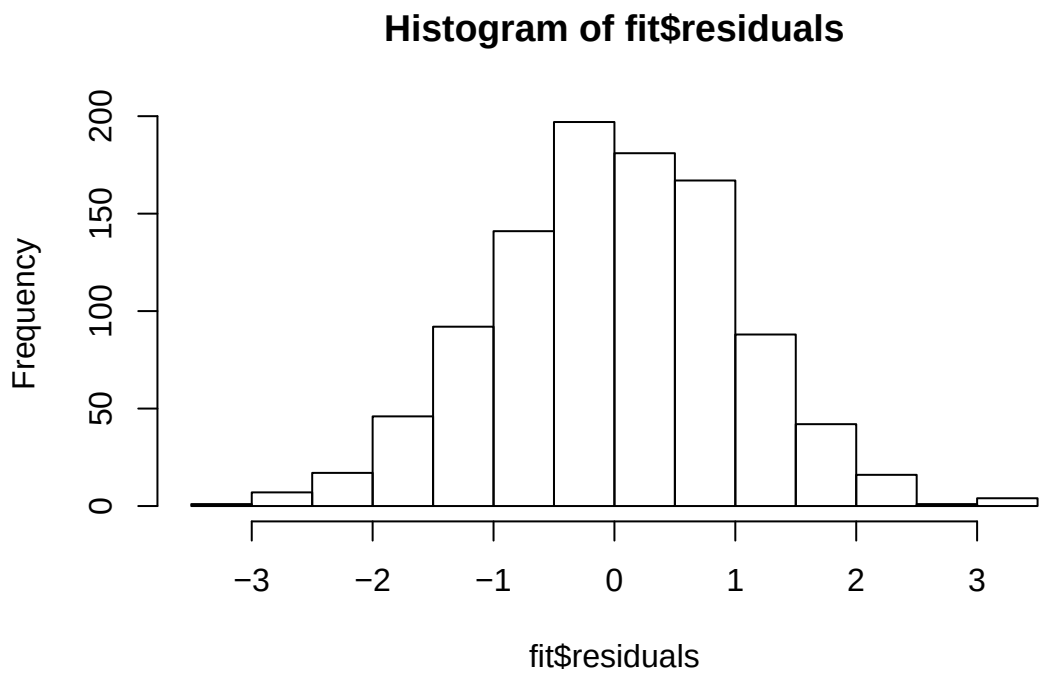
Normality of Errors

Histogram of Residuals

The first way to check for normality of the errors, is to look at a histogram of the residuals. (Note, since we obviously can't observe the true errors, we will use the residuals to estimate their distribution.)

$$e_i = y_i - \hat{y}_i$$

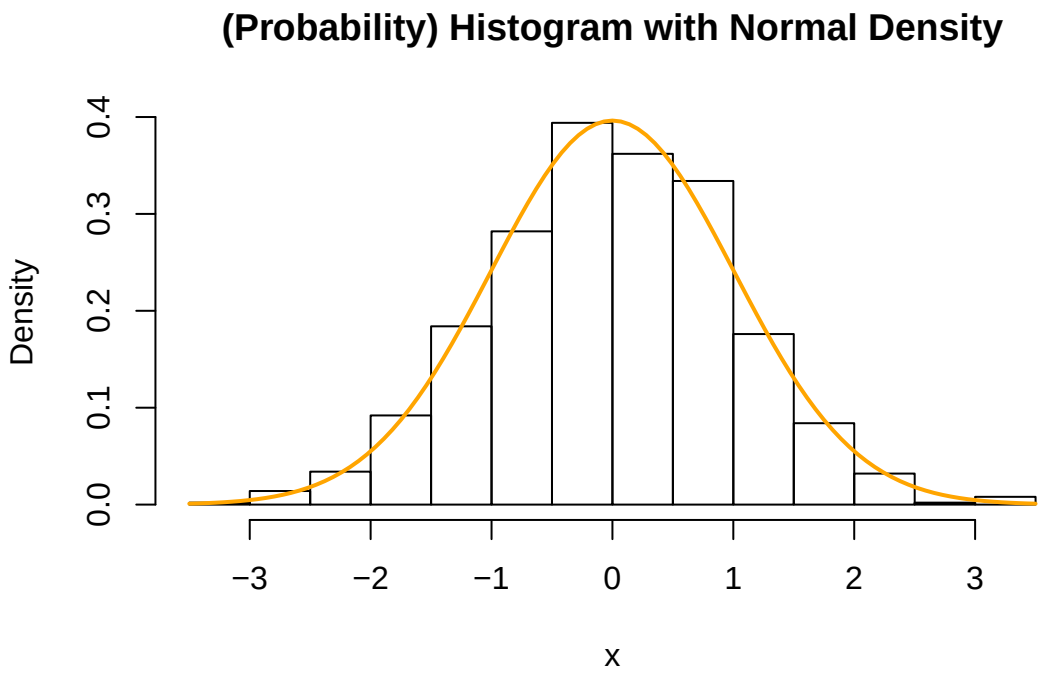
```
hist(fit$residuals)
```



In class, we wrote a nice function that adds an estimated density curve, using the sample mean and sample variance of our simulated data.

```
histNorm <- function(x, densCol = "darkblue"){
  m <- mean(x)
  std <- sqrt(var(x))
  h <- max(hist(x,plot=FALSE)$density)
  d <- dnorm(x, mean=m, sd=std)
  maxY <- max(h,d)
  hist(x, prob=TRUE,
     xlab="x", ylim=c(0, maxY),
     main="(Probability) Histogram with Normal Density")
  curve(dnorm(x, mean=m, sd=std),
      col=densCol, lwd=2, add=TRUE)
}
```

```
histNorm(fit$residuals, "orange")
```



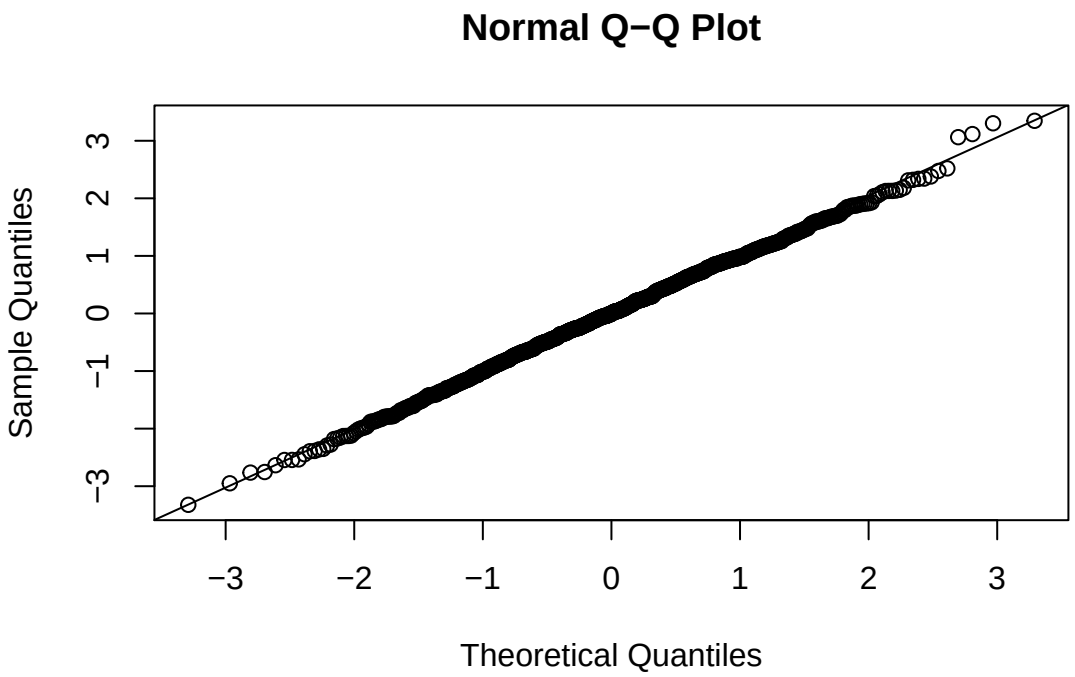
Indeed this histogram suggests these residuals could be from a normal distribution. (But we already knew that, since we're simulating observations that way.) However, in general a histogram can only tell us so much, and it is only a first step.

Q-Q Plot

Another more powerful visual method of assessing normality of the errors is a Q-Q Plot.

In `R` this is easy to make,

```
qqnorm(fit$residuals)
qqline(fit$residuals)
```



If the points of the plot do not closely follow a straight line, this would suggest that the data do not come from a normal distribution.

The calculations required to create the plot vary depending on the implementation, but essentially the y-axis is the sorted data, and the x-axis is the values we would expect if the data did come from a normal distribution.

The Wikipedia page for Normal probability plots gives details on how this is implemented in `R` if you are interested. Wikipedia: Normal probability plot

Shapiro–Wilk Test

Histograms and Q-Q Plots give a nice visual representation of the residuals distribution, however if we are interested in formal testing, there are a number of options available. A commonly used test is the Shapiro–Wilk test, which is implemented in `R`.

```
shapiro.test(fit$residuals)
```

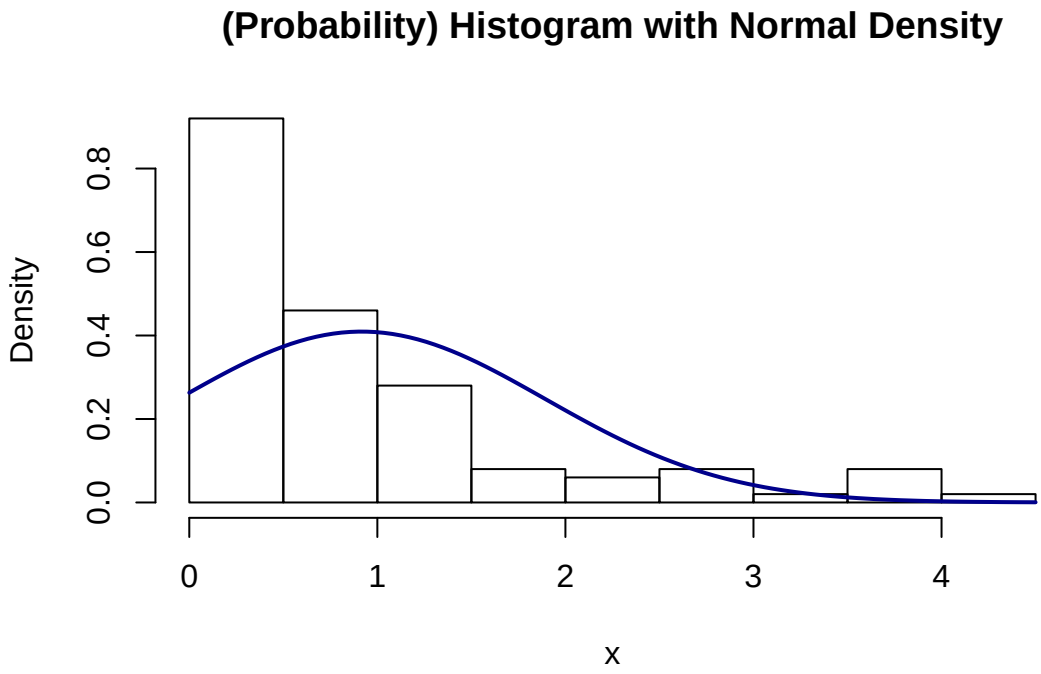
```
##
##  Shapiro-Wilk normality test
##
## data:  fit$residuals
## W = 0.9989, p-value = 0.8079
```

This gives us the value of the test statistic and its p-value. The null hypothesis assumes the data follow a normal distribution, thus a small p-value indicates we believe there is only a small probability the data follow a normal distribution.

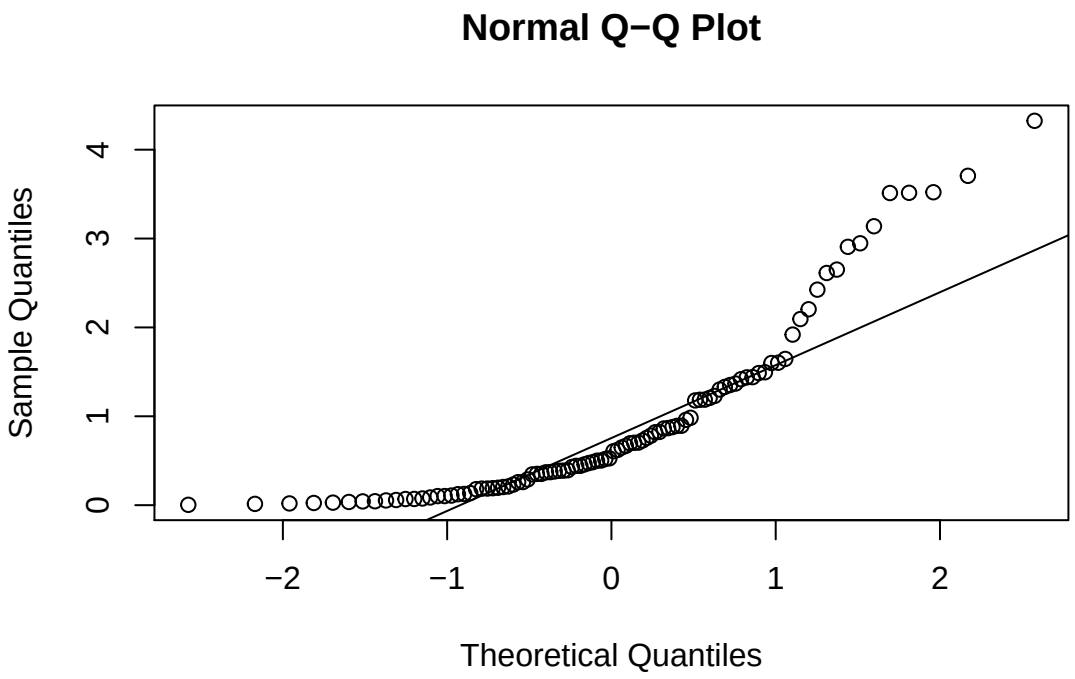
For details, see: Wikipedia: Shapiro–Wilk test

Let's quickly look at some non-normal data, specifically we'll simulate data from an exponential distribution.

```
y <- rexp(100)
histNorm(y)
```




```
qqnorm(y)
qqline(y)
```




```
shapiro.test(y)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  y
## W = 0.81, p-value = 5.099e-10
```

Here, the histogram is clearly not normal, the Q-Q plot is far from a line, and the Shapiro–Wilk test very clearly rejects the null hypothesis that the data are normal.

Solutions for Non-Normality

There are a number of ways of dealing with non-normal errors, including,

- Bootstrap
- Generalized Linear Models (GLMs)
- Non-Parametric Regression

Time permitting we will look at examples of each of these.

Constant Variance

In addition to assuming the errors follow a normal distribution, we also have assumed that their variance is constant. (Homoscedastic.) Non-constant variance, heteroscedasticity, can happen in a number of ways. Sometimes it is written as

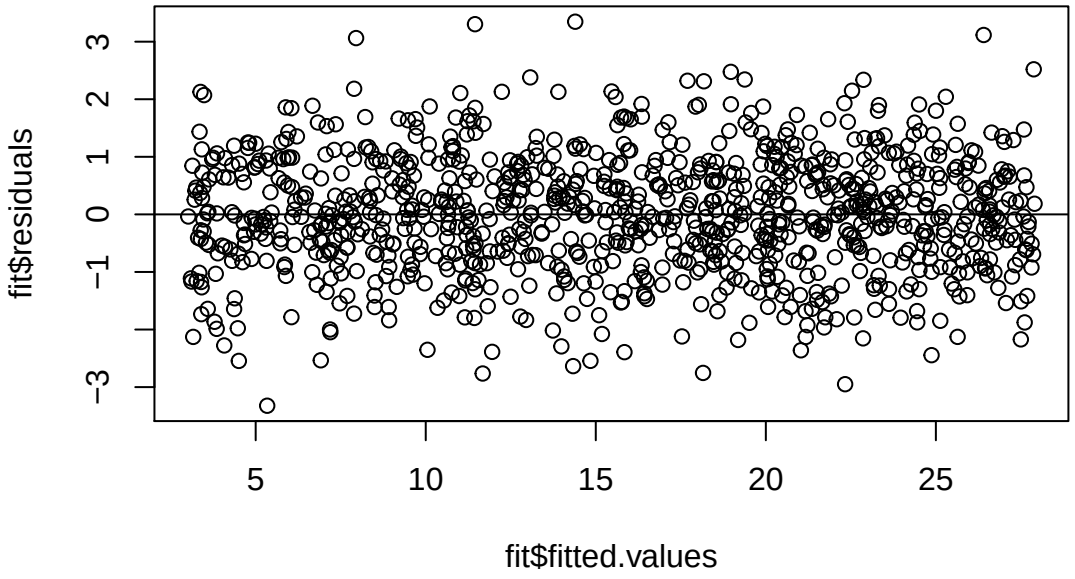
$$\epsilon_i \sim N(0, \sigma_i^2)$$

which indicates that each observation has a different variance.

Fitted vs Residual Plot

Frequently, when variance is non-constant, certain patterns will appear in the residuals. To look for these patterns we will plot fitted versus residuals.

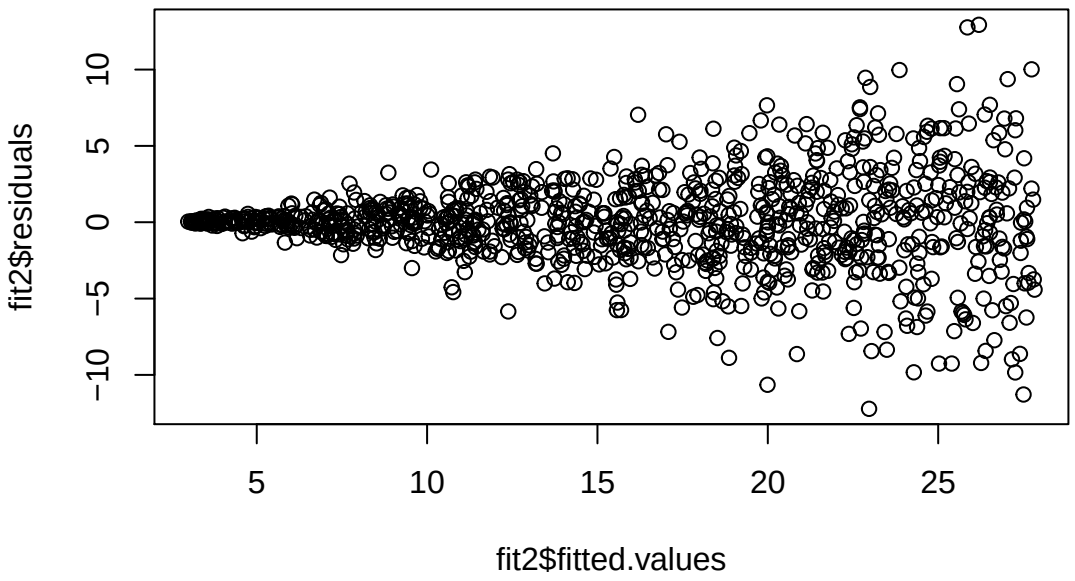
```
plot(fit$fitted.values,fit$residuals)
abline(h = 0)
```



Here, we don't see any patterns in the variance of the residuals, it does indeed seem constant.

If we simulate new data without constant variance, we can see this very clearly in the fitted versus residuals plot.

```
y2 <- 3 + 5*x + rnorm(1000,0,x)
fit2 <- lm(y2 ~ x)
plot(fit2$fitted.values,fit2$residuals)
```



Here we see a rather obvious pattern of increasing variance.

Breusch–Pagan Test

If we would like a more formal test about constant variance, we can use the Breusch–Pagan test. This is not implemented by default in `R`, but we can install a package which will allow us to use it. The Breusch–Pagan test's null hypothesis is homoscedasticity of the regression model, the alternative being a heteroscedastic model.

```
install.packages("lmtest")
```

```
library(lmtest)
```

```
bptest(fit)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  fit
## BP = 0.295, df = 1, p-value = 0.587
```

```
bptest(fit2)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  fit2
## BP = 168.8792, df = 1, p-value < 2.2e-16
```

Here the test does not reject the null for our model with constant variance, and does indeed reject for our model with increasing variance.

For details, see: Wikipedia: Breusch–Pagan test

Solutions for Non-Constant Error Variance

There are a number of ways of dealing with non-constant error variance, including,

- Transformations
- Weighted Least Squares

We will look at transformations in the future. WLS will be left for STAT 425.

Miscellaneous Code

The following code is additional examples, possibly seen in class. Output does not appear in this document.

```
x <- rnorm(1000)
histNorm(x)
histNorm(x, "orange")
qqnorm(x)
qqline(x)
shapiro.test(x)

x <- rnorm(100, mean = 5, sd = 0.1)
histNorm(x)
qqnorm(x)
qqline(x)
shapiro.test(x)

x <- rexp(100)
histNorm(x)
qqnorm(x)
qqline(x)
shapiro.test(x)

x <- rt(100, df=3)
histNorm(x)
qqnorm(x)
qqline(x)
shapiro.test(x)

x <- rbinom(1000, 10000, 0.25)
histNorm(x)
qqnorm(x)
qqline(x)
shapiro.test(x)
```

```r
pvals <- rep(0,10000)
for(i in 1:10000){
  x <- rbinom(1000, 10000, 0.25)
  pvals[i] <- shapiro.test(x)$p.value
}

sum(pvals < 0.05) /  length(pvals)
```