PSTAT 126

Lab 6

Roupen Khanjian

Spring 2021

```r
library(faraway) # Functions and Datasets for Books by Julian Faraway
library(alr4) # Data to Accompany Applied Linear Regression 4th Edition
library(tidyverse) # Easily Install and Load the 'Tidyverse'
library(Lahman) # Sean 'Lahman' Baseball Database
```

Contents

Transformations

- See Chapter 7 of Faraway book

Can we predict At bats from GIDP (Grounded into double plays)?

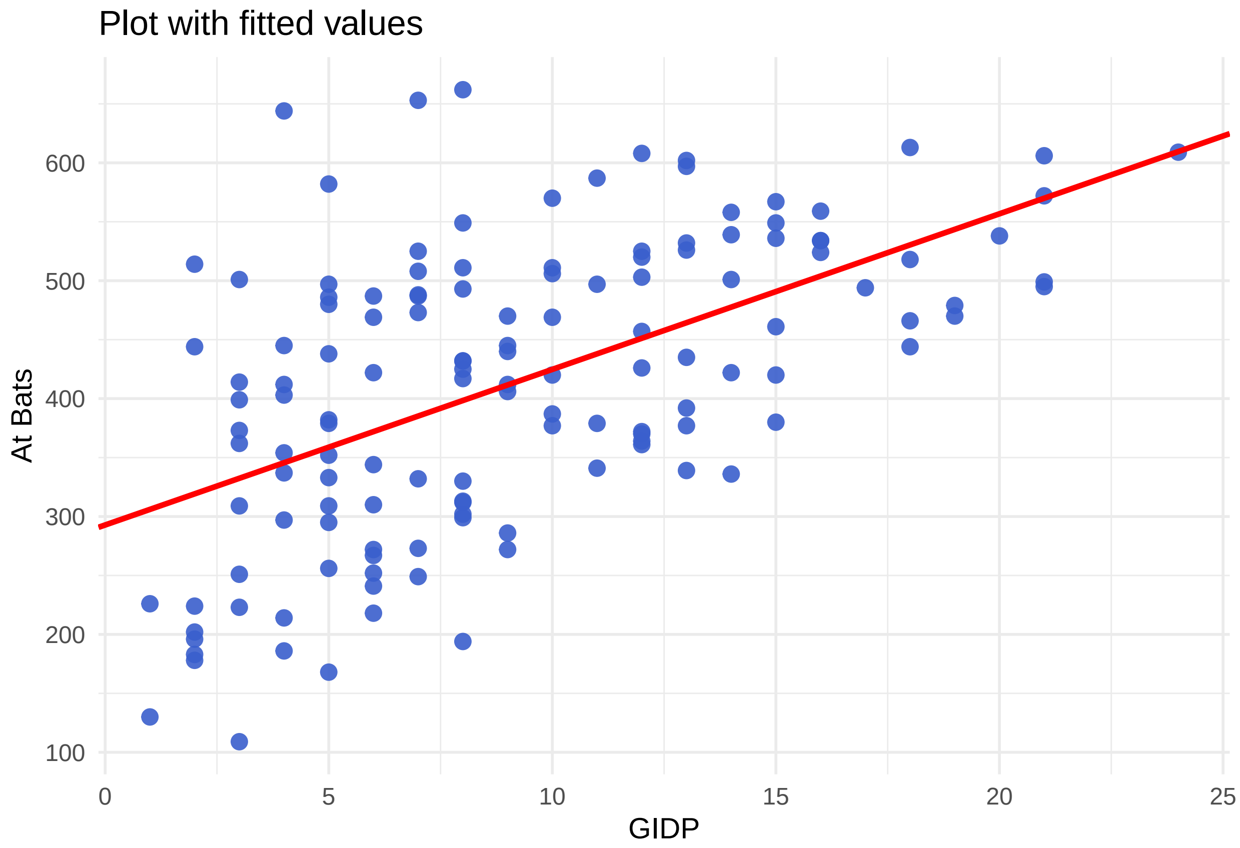
```r
df3 <- Batting %>%
  filter(yearID == "2017" & # stats from 2017
           lgID == "NL" & # only from the NL
           G > 70 & # Only players who have played more than 70 games
           SB != 0) # Only players that have at least stolen 1 base.

dim(df3)
```

```
## [1] 137  22
```

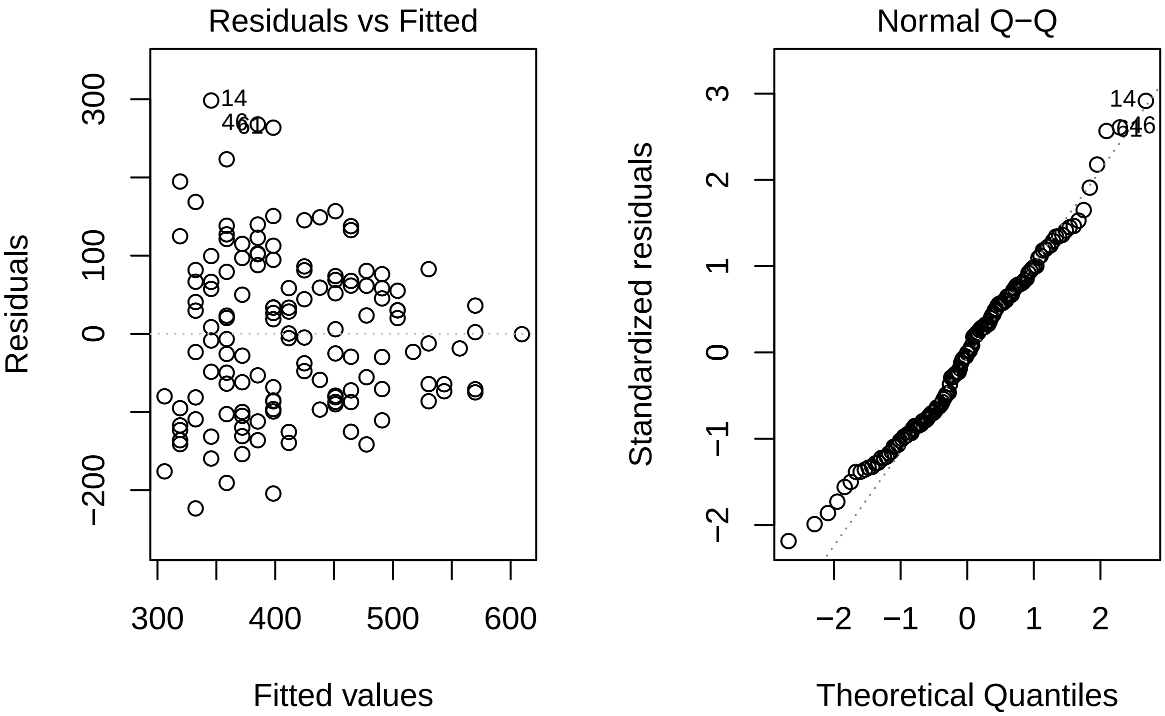
```r
model_3 <- lm(AB ~ GIDP,
          data = df3)

summary(model_3)
```

```
##
## Call:
## lm(formula = AB ~ GIDP, data = df3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -223.420  -81.420   -0.541   68.813  298.383
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  292.832     17.984  16.283  < 2e-16 ***
## GIDP          13.196      1.706   7.734 2.15e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 103.1 on 135 degrees of freedom
## Multiple R-squared:  0.3071, Adjusted R-squared:  0.3019
## F-statistic: 59.82 on 1 and 135 DF,  p-value: 2.15e-12
```

```
ggplot(data = df3) +
  geom_point(aes(x = GIDP, y = AB), color = "royalblue3",
             alpha = 0.9, size = 2.4) +
  geom_abline(aes(intercept = coef(model_3)[1],
                  slope = coef(model_3)[2]),
                  color = "red",
                  size = 1) +
  labs(x = "GIDP",
       y = "At Bats",
       title = "Plot with fitted values") +
  theme_minimal()
```

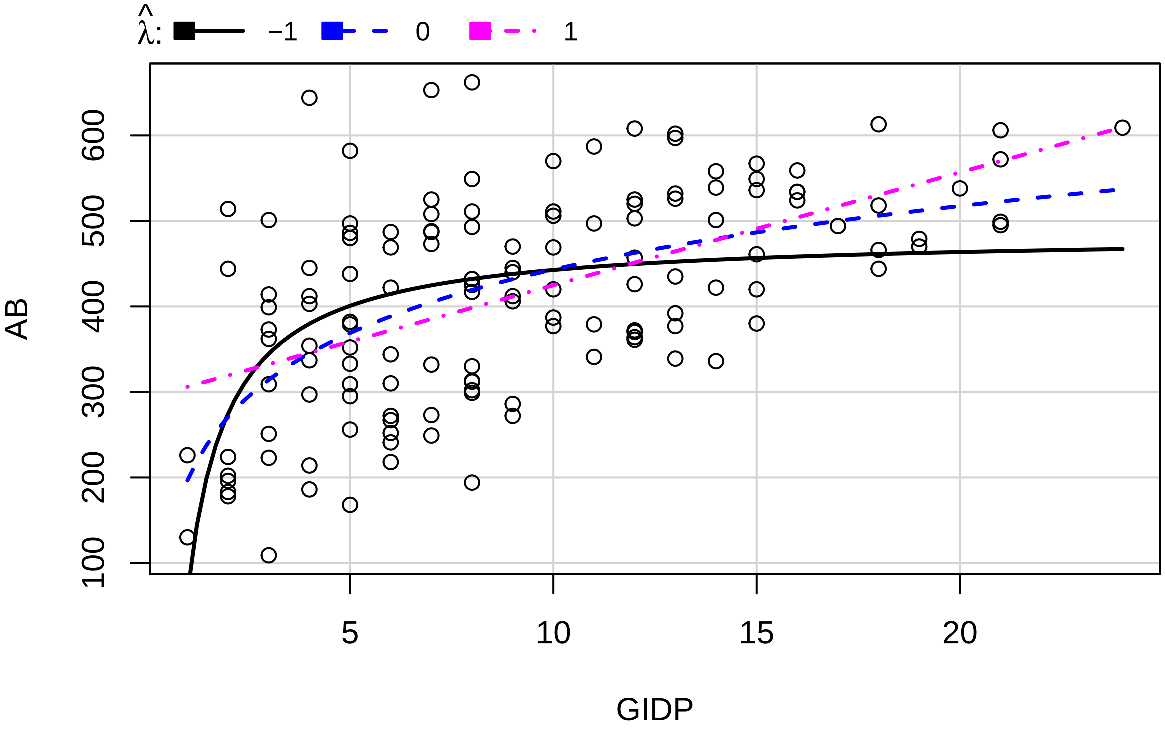


```
par(mfrow = c(1,2))
plot(model_3, which = 1 , add.smooth = F)
plot(model_3, which = 2)
```



- Possible skewness of residuals and heteroscedasticity

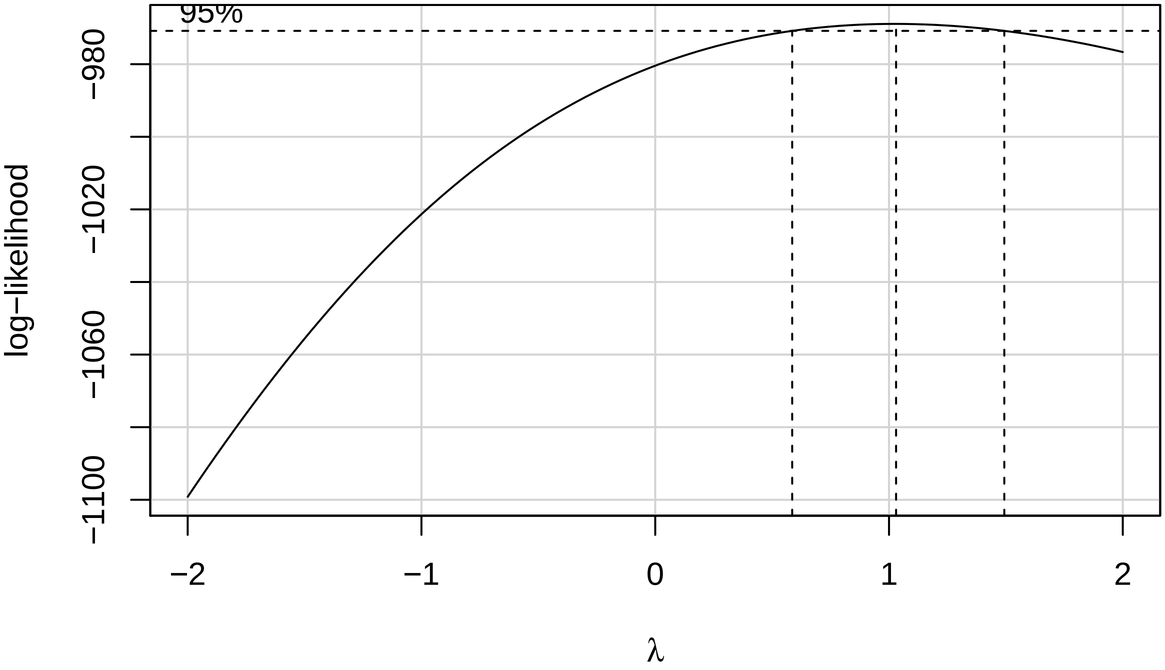
```r
par(mfrow = c(1,1))
invTranPlot(AB ~ GIDP, data = df3,
            lambda = c(-1, 0, 1), optimal = FALSE)
```



```
##   lambda     RSS
## 1     -1 1533496
## 2      0 1389698
## 3      1 1434801
```

Would chose to log transform predictor variable according to above plot. Remember to conduct diagnostic checks again after transforming either the response or predictor(s) variable.

```r
bc <- boxCox(lm(AB ~ log(GIDP),
          data = df3))
```



```r
bc$x[which.max(bc$y)]
```

```
## [1] 1.030303
```

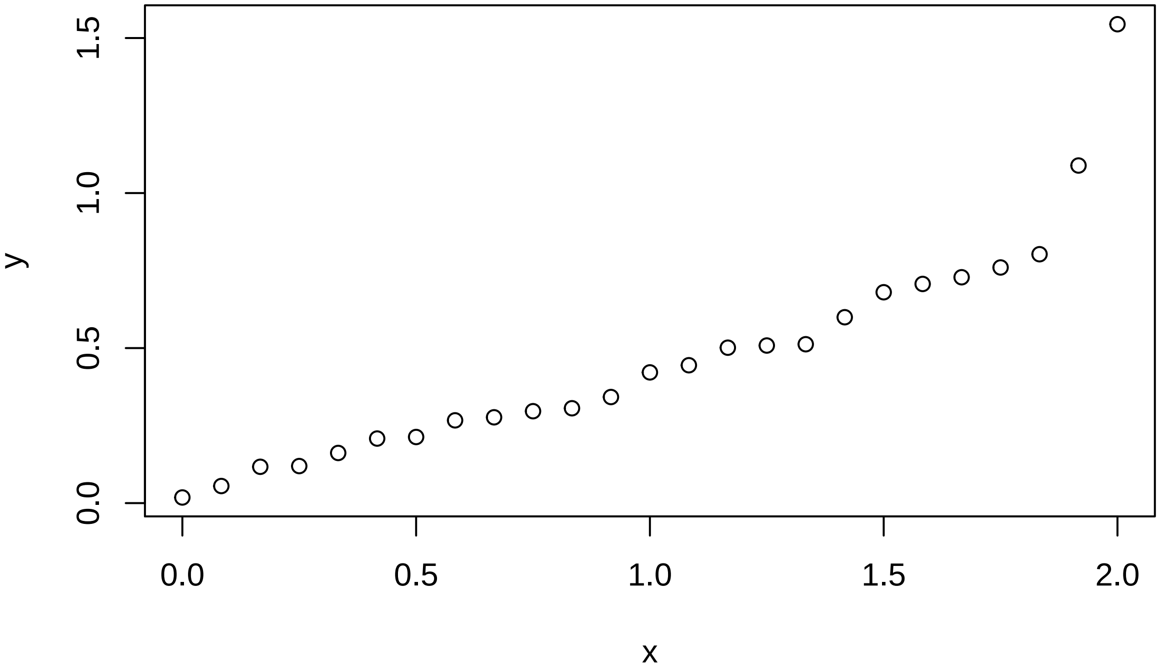
Since $\lambda = 1.030303$ is very close to 1, and 1 is in the 95% confidence interval, we choose to not transform the response variable.

Another Box-Cox Transformation example

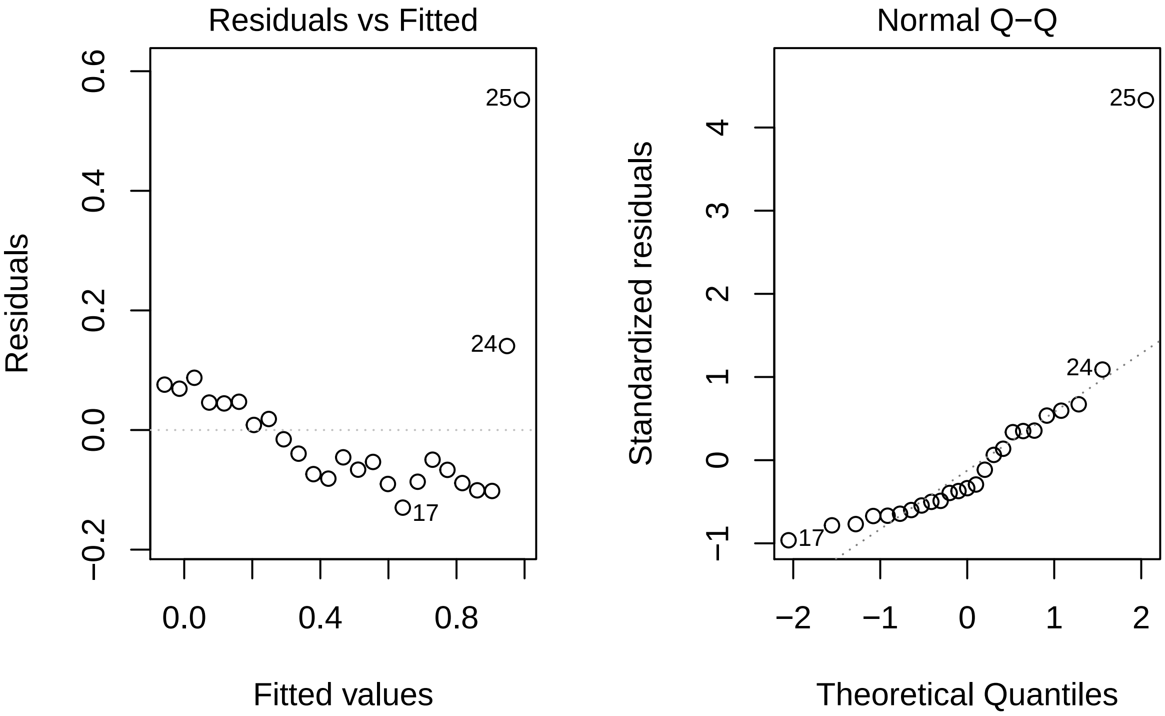
```r
set.seed(71)
y <- sort(rexp(25, rate = 2)) # Response
x <- seq(0,2,length.out = 25) # Predictor

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

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.12976 -0.08133 -0.04566  0.04591  0.55241
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.05772    0.05375  -1.074    0.294
## x            0.52493    0.04607  11.393 6.18e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1384 on 23 degrees of freedom
## Multiple R-squared:  0.8495, Adjusted R-squared:  0.8429
## F-statistic: 129.8 on 1 and 23 DF,  p-value: 6.179e-11
```

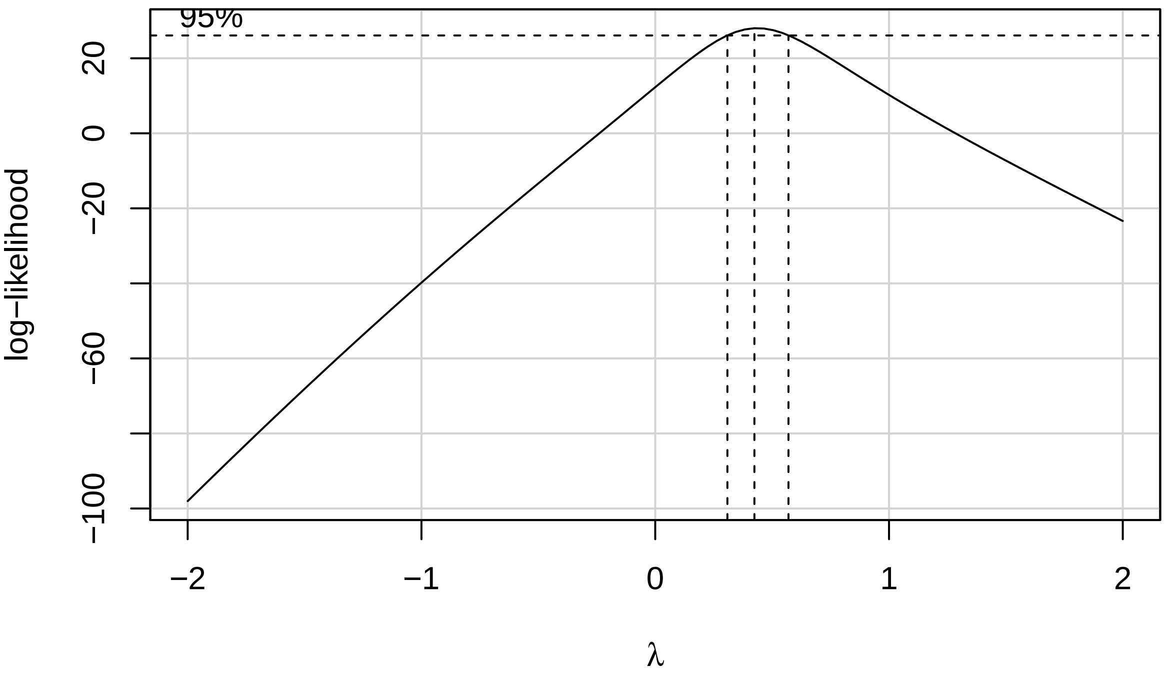
```
plot(x,y)
```



```
par(mfrow = c(1,2))
plot(model_bc, which = 1 , add.smooth = F)
plot(model_bc, which = 2)
```



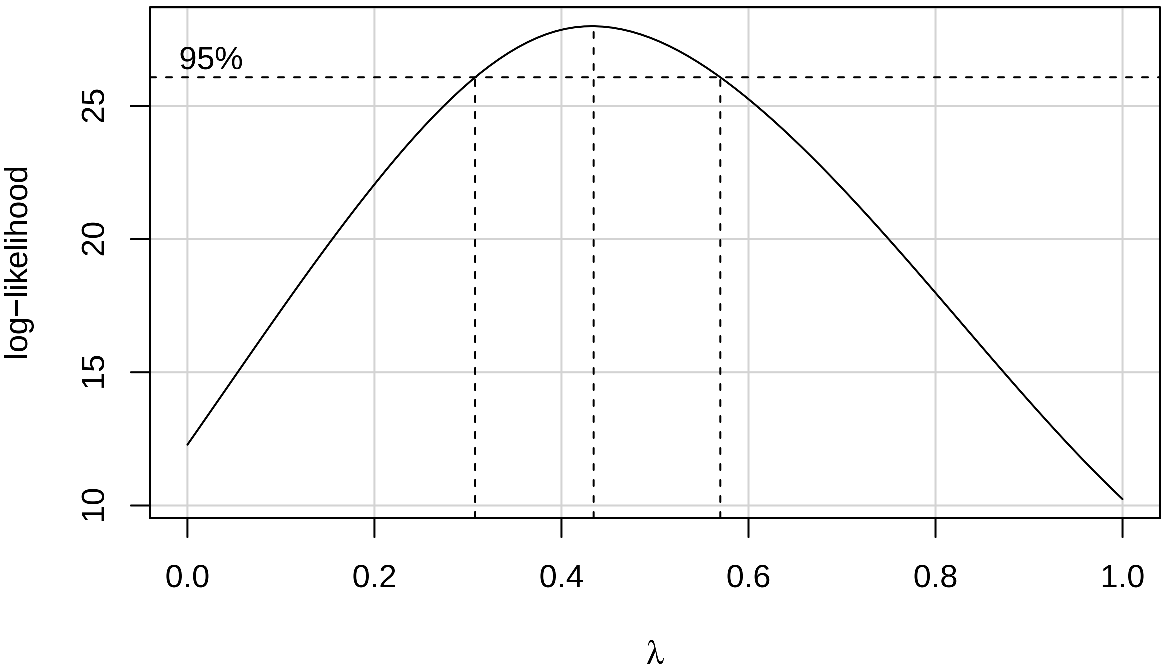
```r
bc <- boxCox(model_bc)
```



```r
bc$x[which.max(bc$y)]
```

```
## [1] 0.4242424
```

```r
bc <- boxCox(model_bc, lambda = seq(0,1,by = 0.2))
```



- Would probably choose a square root transformation on the response variable since 0.5 is within the 95% confidence interval.

Adding polynomial terms to our model with the I() function

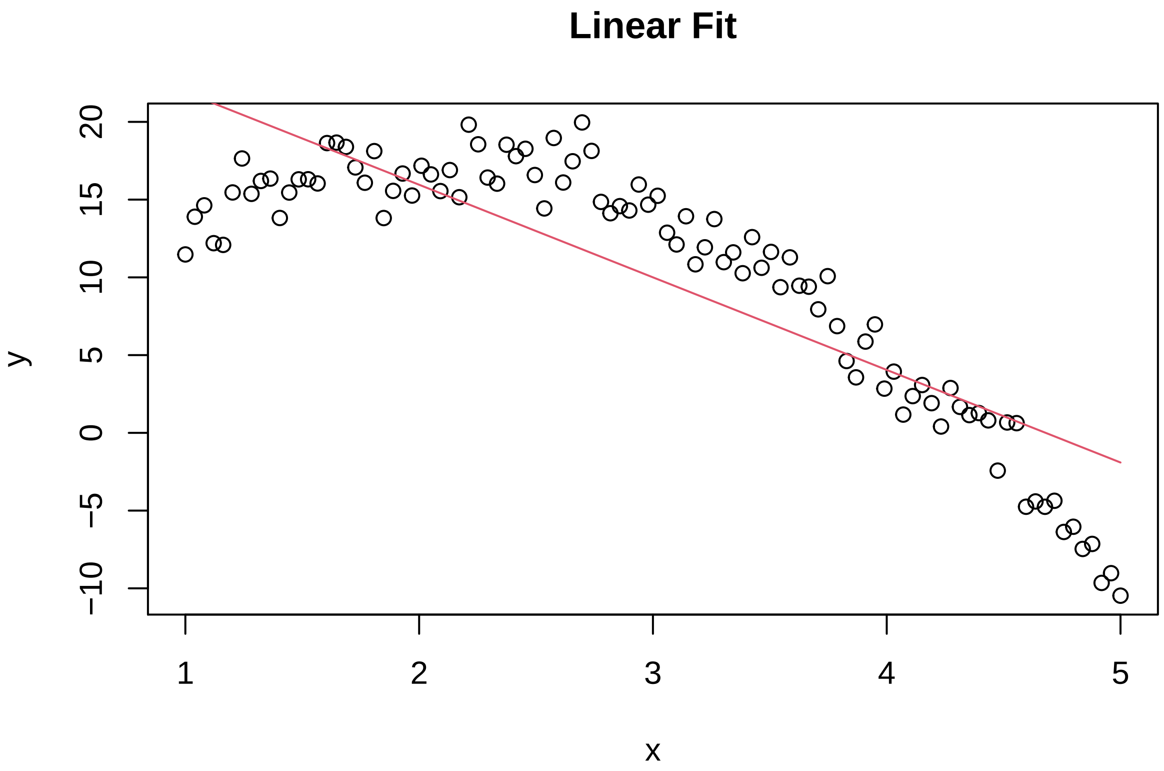
- Chapter 9.4 in Faraway (page 139)

```
par(mfrow = c(1,1))
n <- 100
x <-  seq(1, 5, length = n)
y <-  5 + 12 * x - 3 * x ^ 2 +
  rnorm(n, mean = 0, sd = sqrt(2))

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

Simulated data

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.4361  -2.7457   0.3167   3.3706   8.1573
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  27.8644     1.1566   24.09   <2e-16 ***
## x            -5.9542     0.3593  -16.57   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.191 on 98 degrees of freedom
## Multiple R-squared:  0.737,  Adjusted R-squared:  0.7343
## F-statistic: 274.6 on 1 and 98 DF,  p-value: < 2.2e-16
```

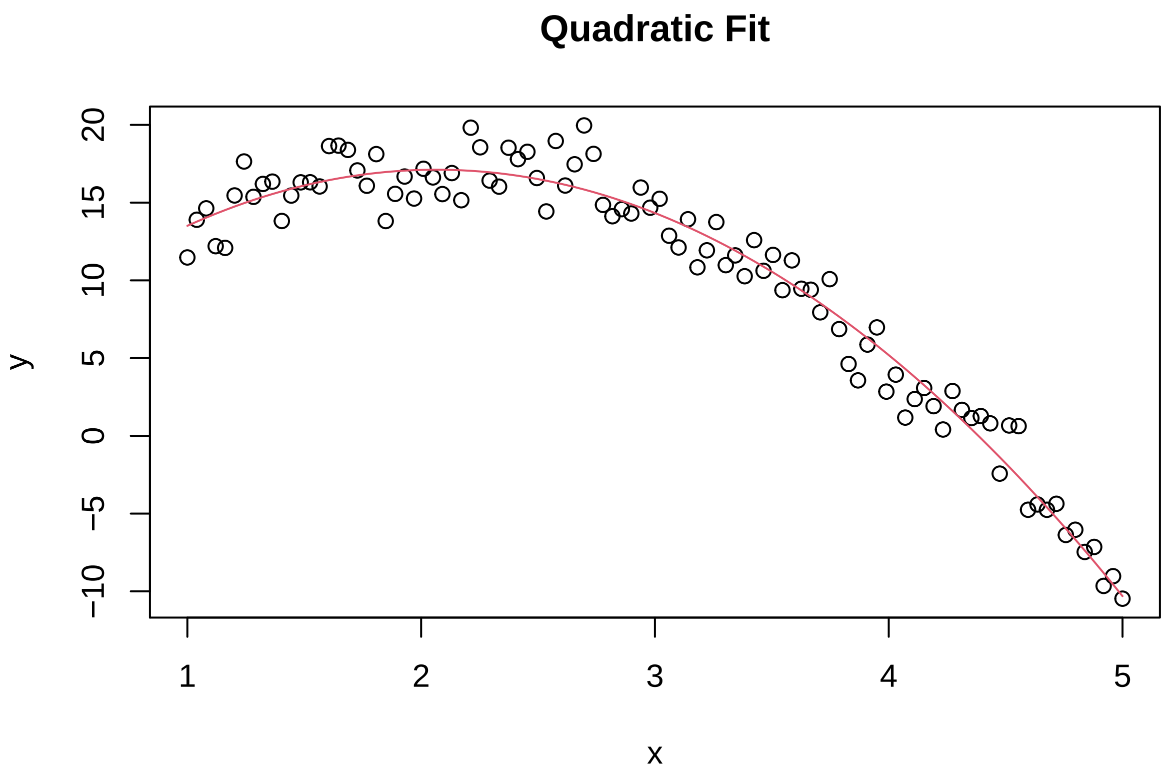
```
yhat <-  fitted(fit)

plot(x, y, main = 'Linear Fit')
lines(x, yhat, col = 2)
```



```r
fit_2 <-  lm(y ~ x + I(x ^ 2))
summary(fit_2)
```

```
##
## Call:
## lm(formula = y ~ x + I(x^2))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.1895 -0.8896 -0.1248  1.0824  4.1203
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.5498     1.0382   3.419  0.00092 ***
## x            13.1417     0.7577  17.345  < 2e-16 ***
## I(x^2)       -3.1827     0.1244 -25.581  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.513 on 97 degrees of freedom
## Multiple R-squared:  0.966,  Adjusted R-squared:  0.9653
## F-statistic:  1380 on 2 and 97 DF,  p-value: < 2.2e-16
```

```r
yhat_2 <-  fitted(fit_2)

plot(x, y, main = 'Quadratic Fit')
lines(x, yhat_2, col = 2)
```


Quadratic Fit
y
20
15
10
5
0
−5
−10
x
1
2
3
4
5

Data from faraway book.

```
head(savings, 4)
```

```
##              sr pop15 pop75     dpi ddpi
## Australia 11.43 29.35  2.87 2329.68 2.87
## Austria   12.07 23.32  4.41 1507.99 3.93
## Belgium   13.17 23.80  4.43 2108.47 3.82
## Bolivia    5.75 41.89  1.67  189.13 0.22
```

- sr = savings rate - personal saving divided by disposable income.
- ddpi = percent growth rate of per-capita disposable income in dollars.

```
summary(lm(sr ~ ddpi,savings))
```

```
## 
## Call:
## lm(formula = sr ~ ddpi, data = savings)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -8.5535 -3.7349  0.9835  2.7720  9.3104 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)   7.8830     1.0110   7.797 4.46e-10 ***
## ddpi          0.4758     0.2146   2.217   0.0314 *  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 4.311 on 48 degrees of freedom
## Multiple R-squared:  0.0929, Adjusted R-squared:  0.074 
## F-statistic: 4.916 on 1 and 48 DF,  p-value: 0.03139
```

```
summary(lm(sr ~ ddpi+I(ddpi^2),savings))
```

```
## 
## Call:
## lm(formula = sr ~ ddpi + I(ddpi^2), data = savings)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -8.5601 -2.5612  0.5546  2.5735  7.8080 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  5.13038    1.43472   3.576 0.000821 ***
## ddpi         1.75752    0.53772   3.268 0.002026 ** 
## I(ddpi^2)   -0.09299    0.03612  -2.574 0.013262 *  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 4.079 on 47 degrees of freedom
## Multiple R-squared:  0.205,  Adjusted R-squared:  0.1711 
## F-statistic: 6.059 on 2 and 47 DF,  p-value: 0.004559
```

```
summary(lm(sr ~ ddpi+I(ddpi^2)+I(ddpi^3),savings))
```

```
##
## Call:
## lm(formula = sr ~ ddpi + I(ddpi^2) + I(ddpi^3), data = savings)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.5571 -2.5575  0.5616  2.5756  7.7984
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.145e+00  2.199e+00   2.340   0.0237 *
## ddpi         1.746e+00  1.380e+00   1.265   0.2123
## I(ddpi^2)   -9.097e-02  2.256e-01  -0.403   0.6886
## I(ddpi^3)   -8.497e-05  9.374e-03  -0.009   0.9928
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.123 on 46 degrees of freedom
## Multiple R-squared:  0.205,  Adjusted R-squared:  0.1531
## F-statistic: 3.953 on 3 and 46 DF,  p-value: 0.01369
```

- Even if lower order terms are not statistically significant, want to keep them in the model.

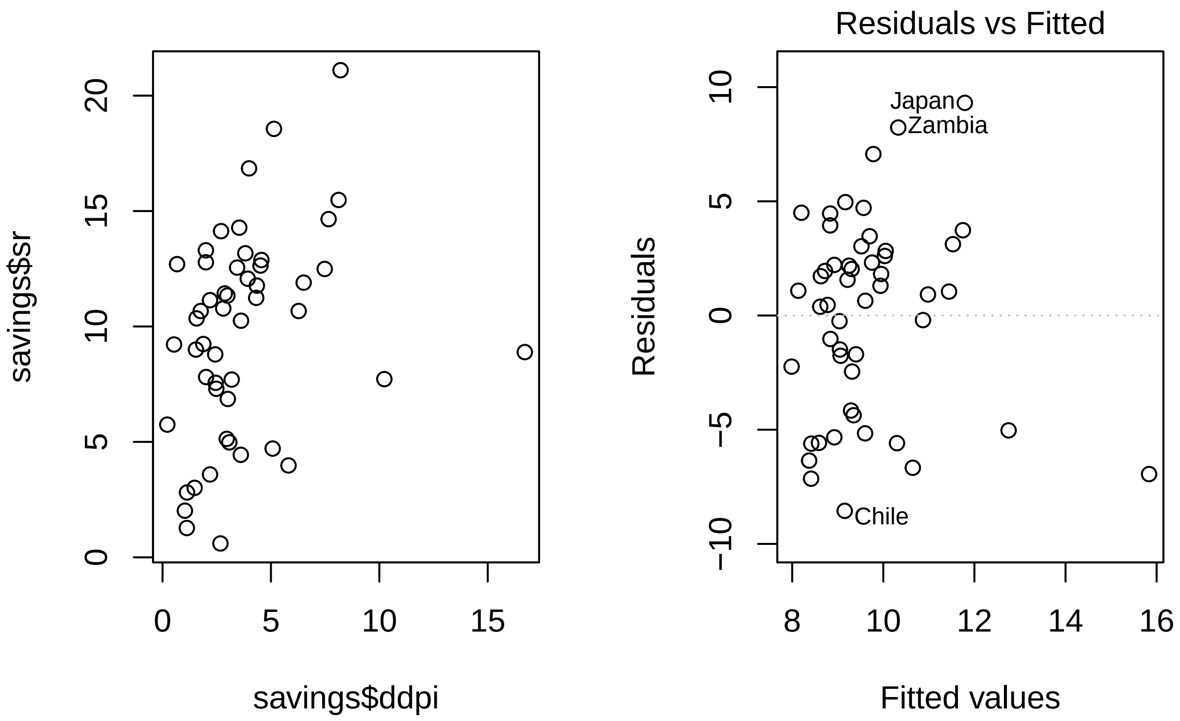
```
lmod <- lm(Species ~ Area + Elevation + Nearest + Scruz + Adjacent, data = gala)
summary(lmod)
```

p-values revisited

```
##
## Call:
## lm(formula = Species ~ Area + Elevation + Nearest + Scruz + Adjacent,
##     data = gala)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -111.679  -34.898   -7.862   33.460  182.584
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.068221  19.154198   0.369 0.715351
## Area        -0.023938   0.022422  -1.068 0.296318
## Elevation    0.319465   0.053663   5.953 3.82e-06 ***
## Nearest      0.009144   1.054136   0.009 0.993151
## Scruz       -0.240524   0.215402  -1.117 0.275208
## Adjacent    -0.074805   0.017700  -4.226 0.000297 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 60.98 on 24 degrees of freedom
## Multiple R-squared:  0.7658, Adjusted R-squared:  0.7171
## F-statistic:  15.7 on 5 and 24 DF,  p-value: 6.838e-07
```

Partial F-tests with Polynomial Regression

```
model_1 <- lm(sr ~ ddpi,savings)

par(mfrow = c(1,2))
plot(savings$ddpi, savings$sr)
plot(model_1, which = 1, add.smooth = F)
```



```
model_2 <- lm(sr ~ ddpi+I(ddpi^2),savings)

anova(model_1, model_2)
```

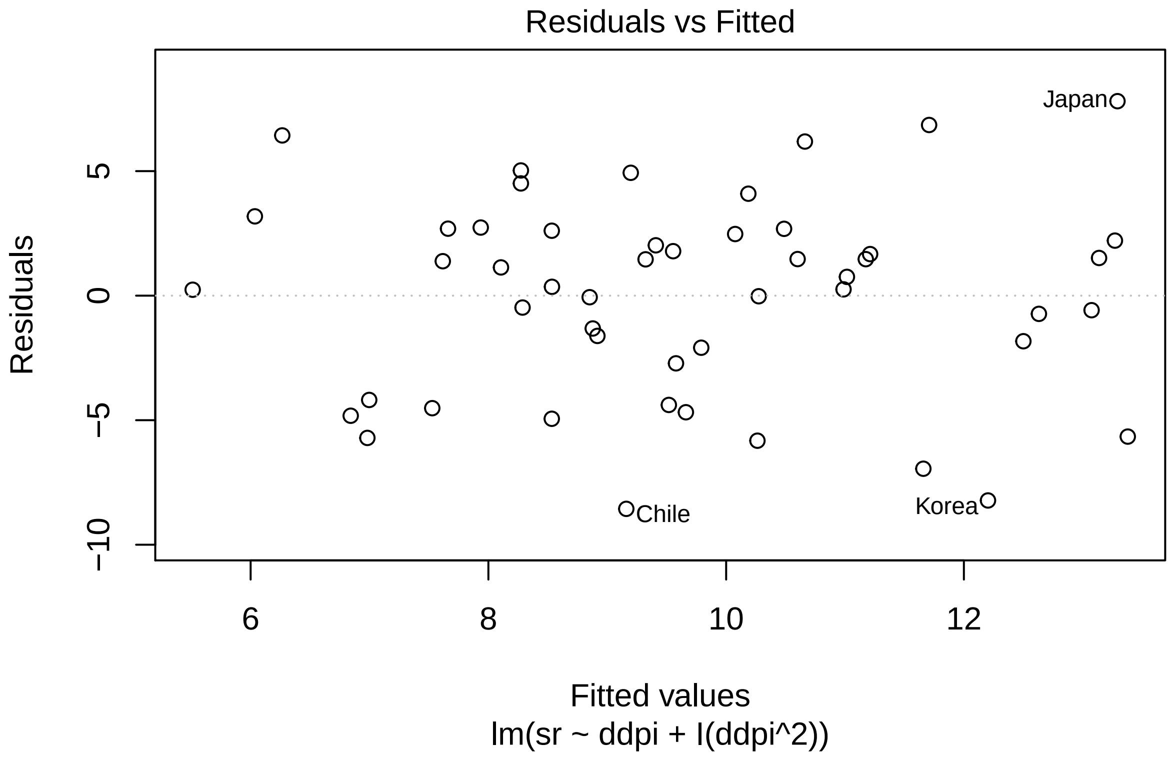
```
## Analysis of Variance Table
##
## Model 1: sr ~ ddpi
## Model 2: sr ~ ddpi + I(ddpi^2)
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1     48 892.25
## 2     47 782.01  1    110.25 6.6261 0.01326 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(model_2)
```

```
##
## Call:
## lm(formula = sr ~ ddpi + I(ddpi^2), data = savings)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.5601 -2.5612  0.5546  2.5735  7.8080
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.13038    1.43472   3.576 0.000821 ***
## ddpi         1.75752    0.53772   3.268 0.002026 **
## I(ddpi^2)   -0.09299    0.03612  -2.574 0.013262 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 4.079 on 47 degrees of freedom
## Multiple R-squared:  0.205,  Adjusted R-squared:  0.1711
## F-statistic: 6.059 on 2 and 47 DF,  p-value: 0.004559
```

```
par(mfrow = c(1,1))
plot(model_2, which = 1, add.smooth = F)
```



```
model_3 <- lm(sr ~ ddpi+I(ddpi^2)+I(ddpi^3), savings)
summary(model_3)
```

```
##
## Call:
## lm(formula = sr ~ ddpi + I(ddpi^2) + I(ddpi^3), data = savings)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.5571 -2.5575  0.5616  2.5756  7.7984
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.145e+00  2.199e+00   2.340   0.0237 *
## ddpi         1.746e+00  1.380e+00   1.265   0.2123
## I(ddpi^2)   -9.097e-02  2.256e-01  -0.403   0.6886
## I(ddpi^3)   -8.497e-05  9.374e-03  -0.009   0.9928
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.123 on 46 degrees of freedom
## Multiple R-squared:  0.205,  Adjusted R-squared:  0.1531
## F-statistic: 3.953 on 3 and 46 DF,  p-value: 0.01369
```

```
anova(model_1, model_3)
```

```
## Analysis of Variance Table
##
## Model 1: sr ~ ddpi
## Model 2: sr ~ ddpi + I(ddpi^2) + I(ddpi^3)
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1     48 892.25
## 2     46 782.01  2    110.25 3.2426 0.04815 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(model_2, model_3)
```

```
## Analysis of Variance Table
##
## Model 1: sr ~ ddpi + I(ddpi^2)
## Model 2: sr ~ ddpi + I(ddpi^2) + I(ddpi^3)
##   Res.Df    RSS Df Sum of Sq     F Pr(>F)
## 1     47 782.01
## 2     46 782.01  1 0.0013968 1e-04 0.9928
```