

# Regression Models: Assignment 1

Daniel Alonso

November 24th, 2020

Importing libraries

```
library(dplyr)
library(MuMIn)
library(MASS)
library(leaps)
```

## Exercise 1

### Simulation

```
# terms
sim = list()
epsilon = list()
cosine_term = list()

# betas estimation
beta1_est = list()
beta2_est = list()

# variables
phi = 50
beta1 = 3
beta2 = 3

# simulation
for (j in 1:1000) {
  vals = c()
  eps = c()
  cos_t = c()
  for (i in 1:100) {
    eps = c(eps, rnorm(1, mean=0, sd=1))
    cos_t = c(cos_t, cos(i/10 + phi))
    run = beta1 + beta2*cos_t[i] + eps[i]
    vals = c(vals, run)
  }
  sim[[j]] = vals
  epsilon[[j]] = eps
  cosine_term[[j]] = cos_t
  beta1_est[[j]] = mean(sim[[j]]) - beta2*mean(cosine_term[[j]]) - mean(epsilon[[j]])
  beta2_est[[j]] = (mean(sim[[j]]) - beta1 - mean(epsilon[[j]]))/mean(cosine_term[[j]])
}
```

## Exercise 2

### Importing the data

```
d <- data.frame(read.table('../data/index.txt', header=TRUE))
```

```
X = d$PovPct  
Y = d$Brth15to17  
beta1 = cov(X, Y)/var(X)  
beta0 = mean(Y) - beta1*mean(X)
```

```
beta1  
#> [1] 1.373345  
beta0  
#> [1] 4.267293
```

### Exercise 3

First we have the log-likelihood function for  $\beta$  and  $\sigma^2$

$$l(\sigma^2|X) = \sum_{i=1}^n \log\left(\frac{1}{\sqrt{2\pi\sigma^2}} - \frac{(Y_i - (\beta_0 + \beta_1 x_{i1} + \dots + \beta_k x_{ik}))^2}{2\sigma^2}\right)$$

$$\propto -\frac{n}{2} \log(\sigma^2) - \frac{(Y - X\beta)'(Y - X\beta)}{2\sigma^2}$$

Differentiating the second expression:

$$\frac{\partial l}{\partial \sigma} \left( -\frac{n}{2} \log(\sigma^2) - \frac{(Y - X\beta)'(Y - X\beta)}{2\sigma^2} \right) = 0$$

We get:

$$-\frac{n}{2} \left( \frac{1}{\sigma^2} \right) (2\sigma) - (Y - X\beta)'(Y - X\beta) * (-2)(2\sigma^{-3}) = 0$$

We reduce the expression further:

$$-\frac{n}{\sigma} + \frac{(Y - X\beta)'(Y - X\beta)}{\sigma^3} = 0$$

We multiply both sides by  $\sigma^3$  and we get:

$$-n\sigma^2 + (Y - X\beta)'(Y - X\beta) = 0$$

And solving for  $\sigma^2$  we get:

$$\hat{\sigma}^2 = \frac{(Y - X\beta)'(Y - X\beta)}{n}$$

Which is our maximum likelihood estimator for  $\sigma^2$

### Exercise 4

## Exercise 5

```
bodyfat <- data.frame(read.table('../data/bodyfat.txt', header=TRUE))
modall <- lm(hwfat ~., data = bodyfat)
summary(modall)
#>
#> Call:
#> lm(formula = hwfat ~ ., data = bodyfat)
#>
#> Residuals:
#>      Min       1Q   Median       3Q      Max
#> -6.162 -1.858 -0.464  2.502  8.177
#>
#> Coefficients:
#>              Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 13.29370     9.63027   1.380  0.1718
#> age         -0.32893     0.32158  -1.023  0.3098
#> ht          -0.06731     0.16051  -0.419  0.6762
#> wt          -0.01365     0.02591  -0.527  0.5999
#> abs          0.37142     0.08837   4.203 7.55e-05 ***
#> triceps      0.38743     0.13761   2.815  0.0063 **
#> subscap      0.11405     0.14193   0.804  0.4243
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 3.028 on 71 degrees of freedom
#> Multiple R-squared:  0.8918, Adjusted R-squared:  0.8827
#> F-statistic: 97.54 on 6 and 71 DF,  p-value: < 2.2e-16
```

The sum of residuals is zero:

```
residuals <- sum(resid(modall))
```

The sum of the observed data is equal to the sum of the fitted values

```
Y_hat <- predict(modall, bodyfat[1:length(names(bodyfat))-1])
sum(bodyfat$hwfat) - sum(Y_hat)
#> [1] 4.547474e-13
```

The residuals are orthogonal to the predictors

```
sum(residuals*bodyfat[1:length(names(bodyfat))-1])
#> [1] -3.077268e-10
```

The residuals are orthogonal to the fitted values

```
sum(residuals*Y_hat)
#> [1] -1.568657e-11
```

## Exercise 6

We use `regsubsets` to find the best model combinations by adjusted  $R^2$

```
options(na.action = "na.fail")
modall <- lm(hwfat ~., data = bodyfat)
combs <- leaps::regsubsets(bodyfat[,1:6],bodyfat[,7])
summary(combs)
#> Subset selection object
#> 6 Variables (and intercept)
#>      Forced in Forced out
#> age          FALSE      FALSE
#> ht           FALSE      FALSE
#> wt           FALSE      FALSE
#> abs          FALSE      FALSE
#> triceps      FALSE      FALSE
#> subscap      FALSE      FALSE
#> 1 subsets of each size up to 6
#> Selection Algorithm: exhaustive
#>      age ht wt abs triceps subscap
#> 1 ( 1 ) " " " " " " "*" " " " "
#> 2 ( 1 ) " " " " " " "*" "*" " "
#> 3 ( 1 ) "*" " " " " " "*" "*" " "
#> 4 ( 1 ) "*" "*" " " " "*" "*" " "
#> 5 ( 1 ) "*" " " " "*" "*" "*" "*"
#> 6 ( 1 ) "*" "*" "*" "*" "*" "*" "
```

And their corresponding  $R^2$  values:

```
summary(combs)$adjr2
#> [1] 0.8409068 0.8801014 0.8849817 0.8846381 0.8840129 0.8826699
```

We can see the best model is the one with the following  $R^2$

```
max(summary(combs)$adjr2)
#> [1] "NULL"
```

Which corresponds to the model which uses the variables **age**, **ht**, **abs** and **triceps**.

We can also do this same calculation using the function `dredge` (albeit less efficiently):

```
combs <- dredge(modall, extra = "R^2")
print("best model")
#> [1] "best model"
combs[combs$"R^2" == max(combs$"R^2")]
#> Global model call: lm(formula = hwfat ~ ., data = bodyfat)
#> ---
#> Model selection table
#>      (Intrc)      abs      age      ht      sbscp      trcps      wt      R^2 df logLik
#> 64      13.29 0.3714 -0.3289 -0.06731 0.1141 0.3874 -0.01365 0.8918 8 -193.43
#>      AICc delta weight
#> 64 404.9 5.58 1
#> Models ranked by AICc(x)
```

## Exercise 7

We know that:

$$\begin{aligned} SST &= \sum_{i=1}^n (y_i - \bar{y})^2 \\ &= \sum_{i=1}^n (y_i - \hat{y}_i + \hat{y}_i - \bar{y})^2 \\ &= \sum_{i=1}^n (y_i - \hat{y}_i)^2 + 2 \sum_{i=1}^n (y_i - \hat{y}_i)(\hat{y}_i - \bar{y}) + \sum_{i=1}^n (\hat{y}_i - \bar{y})^2 \\ &= SSE + SSR + 2 \sum_{i=1}^n (y_i - \hat{y}_i)(\hat{y}_i - \bar{y}) \end{aligned}$$

No we must prove that:

$$2 \sum_{i=1}^n (y_i - \hat{y}_i)(\hat{y}_i - \bar{y}) = 0$$

So then we have:

$$\sum_{i=1}^n (y_i - \hat{y}_i)(\hat{y}_i - \bar{y}) = \sum_{i=1}^n (y_i - \hat{y}_i)\hat{y}_i - \sum_{i=1}^n (y_i - \hat{y}_i)\bar{y} = 0$$

Because we know that:

$$\sum_{i=1}^n (y_i - \hat{y}_i)\hat{y}_i = 0$$

Given that the residuals must be orthogonal to the fitted values.

And:

$$\sum_{i=1}^n (y_i - \hat{y}_i)\bar{y} = 0$$

Because the sum of the observed data is equal to the sum of the fitted values:

$$\sum_{i=1}^n y_i = \sum_{i=1}^n \hat{y}_i$$

## Exercise 8

We define a list with all the models excluding, in each one, a single variable.

```
models <- list()
vars <- c("age", "ht", "wt", "abs", "triceps", "subscap")
models[[1]] <- update(modall, ~.-age)
models[[2]] <- update(modall, ~.-ht)
models[[3]] <- update(modall, ~.-wt)
models[[4]] <- update(modall, ~.-abs)
models[[5]] <- update(modall, ~.-triceps)
models[[6]] <- update(modall, ~.-subscap)
```

We run ANOVA with both the models without each variable and the main model including all the other variables.

We can see the pvalues for the ANOVA where each specific variable was excluded:

```
anovas <- list()
pvalues <- c()
amount_of_vars <- length(names(bodyfat))-1
for (i in 1:amount_of_vars) {
  anovas[[i]] <- anova(models[[i]], modall)
  pvalues <- c(pvalues, sum(anovas[[i]][2, "Pr(>F)"]))
}
for (i in 1:length(vars)) {
  print(paste("excluding: ", vars[i], ": ", pvalues[i], sep=""))
}
#> [1] "excluding: age: 0.30983932449522"
#> [1] "excluding: ht: 0.67622546378066"
#> [1] "excluding: wt: 0.599878887504826"
#> [1] "excluding: abs: 7.54898491342447e-05"
#> [1] "excluding: triceps: 0.00630111253287972"
#> [1] "excluding: subscap: 0.424314507846979"
```

Then we compare with summary:

```
summary(modall)[4]
#> $coefficients
#>
#> Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 13.29369860 9.63026704 1.3804081 1.717917e-01
#> age -0.32893403 0.32157778 -1.0228755 3.098393e-01
#> ht -0.06730905 0.16050751 -0.4193514 6.762255e-01
#> wt -0.01365183 0.02590783 -0.5269385 5.998789e-01
#> abs 0.37141976 0.08836595 4.2032001 7.548985e-05
#> triceps 0.38742647 0.13761017 2.8153912 6.301113e-03
#> subscap 0.11405213 0.14192779 0.8035927 4.243145e-01
```

And we can see we get the same pvalues in the summary. Therefore viewing the summary can be a much faster version of performing such testing.

as a result we get that the least meaningful variable (the variable that explains the lowest variance of the model) is the variable ht (height) followed by the variable wt (weight).

## Exercise 9

Given that  $E[\hat{Y}|X_h] = \hat{Y}_h \sim N(X_h\beta, \sigma^2 X_h(X'X)^{-1}X_h')$

$$\Rightarrow \hat{y}_h \pm t_{n-(k+1), \frac{\alpha}{2}} * \hat{\sigma} \sqrt{h_{ii}}$$

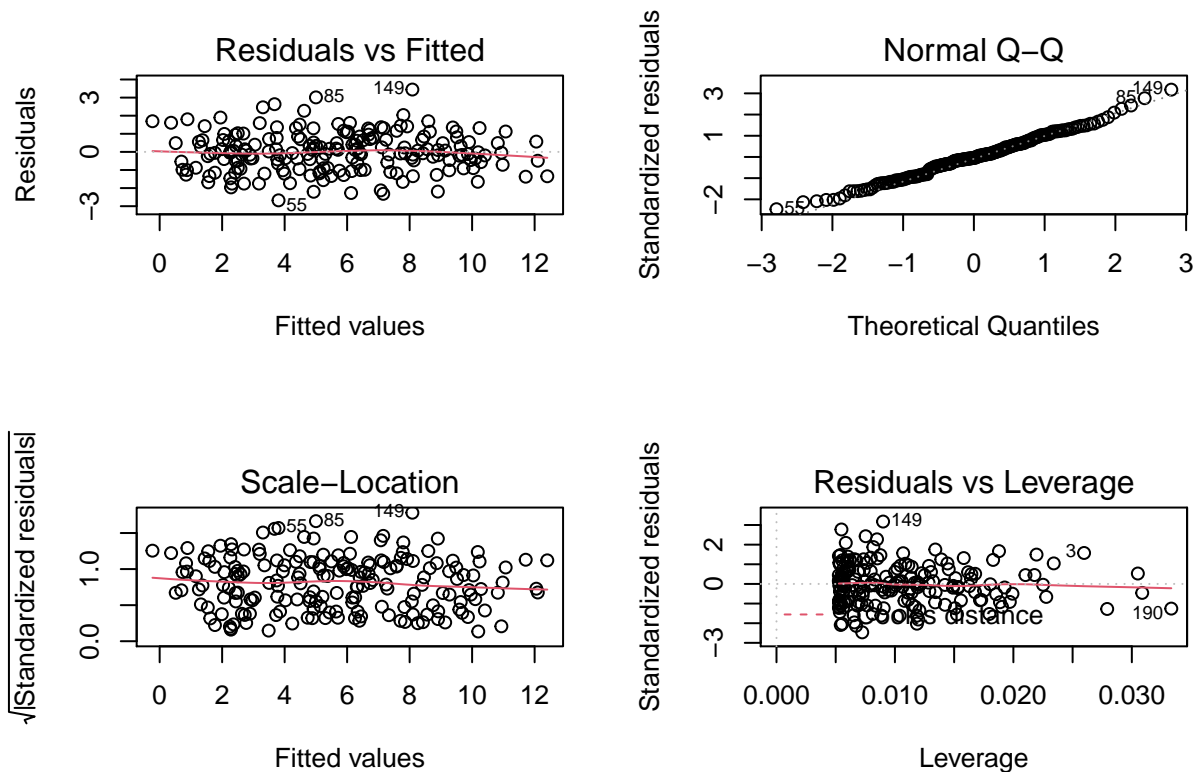
where  $h_{ii}$  is the diagonal of our  $H$  matrix.

is our expression for the  $(1 - \alpha)\%$  confidence interval for  $\hat{Y}_h$  when  $\sigma^2$  is unknown.

## Exercise 10

```
minmax_scaler <- function(x) {
  return((x-min(x))/(max(x)-min(x)))
}
```

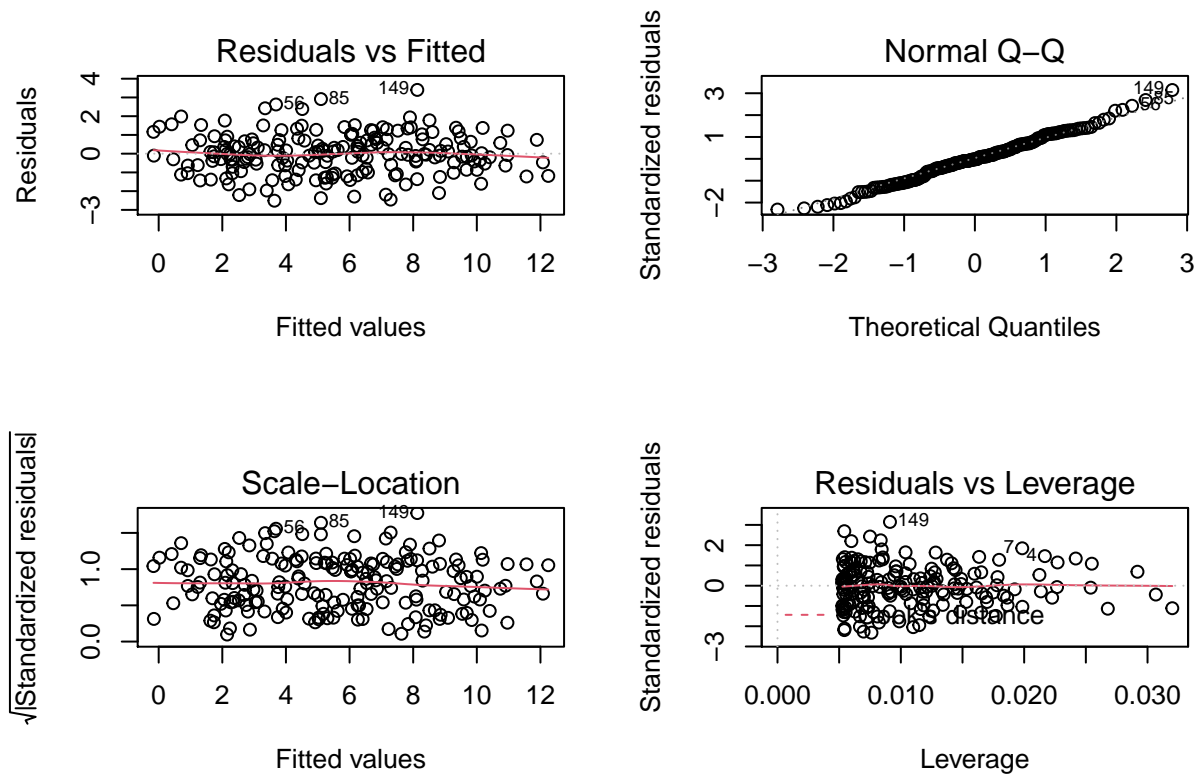
```
transform <- data.frame(read.table('../data/Transform_V2.txt', header=TRUE))
trm1 <- lm(y ~ sqrt(x2+1), data=transform)
trm2 <- lm(y ~ I(x3^(2)), data=transform)
par(mfrow=c(2,2))
plot(trm1)
```



We can see that the square root transformation is appropriate for the  $X_2$  variable



```
par(mfrow=c(2,2))
plot(trm2)
```

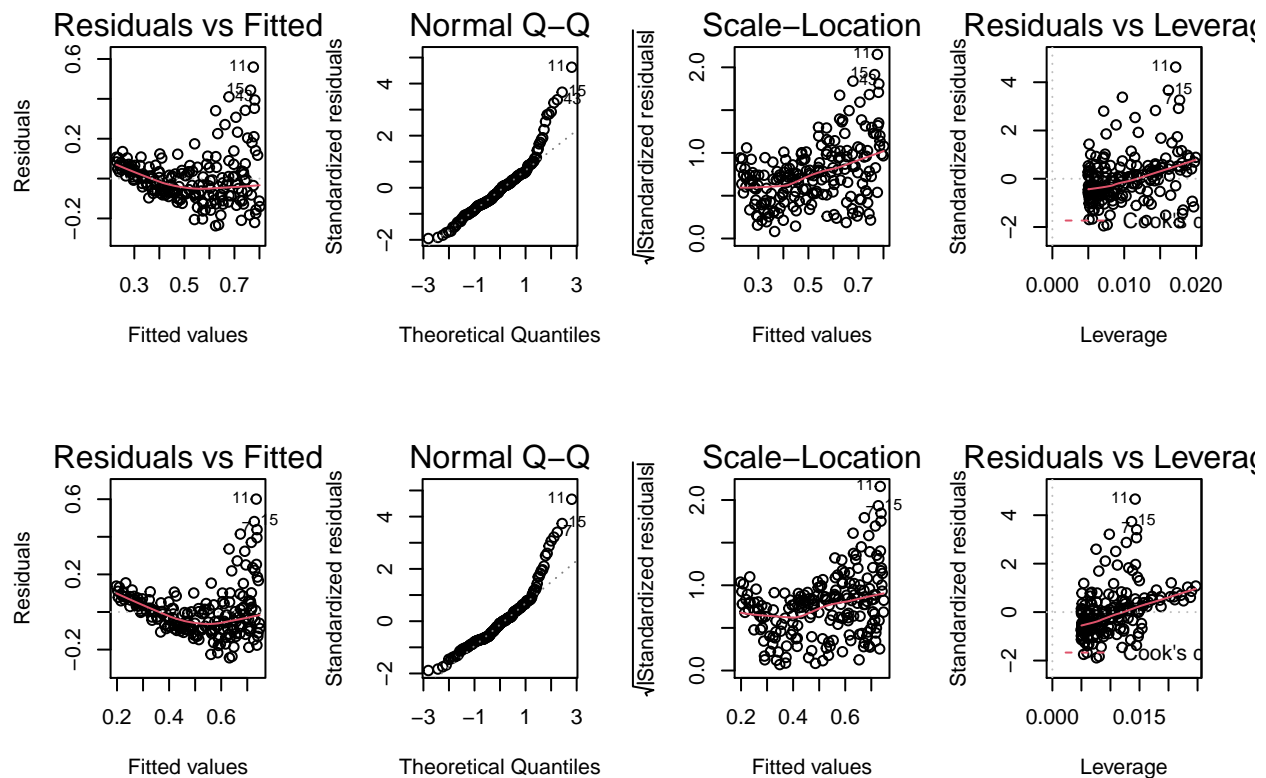


We can see that the  $X^2$  transformation is appropriate for the  $X_3$  variable

## Exercise 11

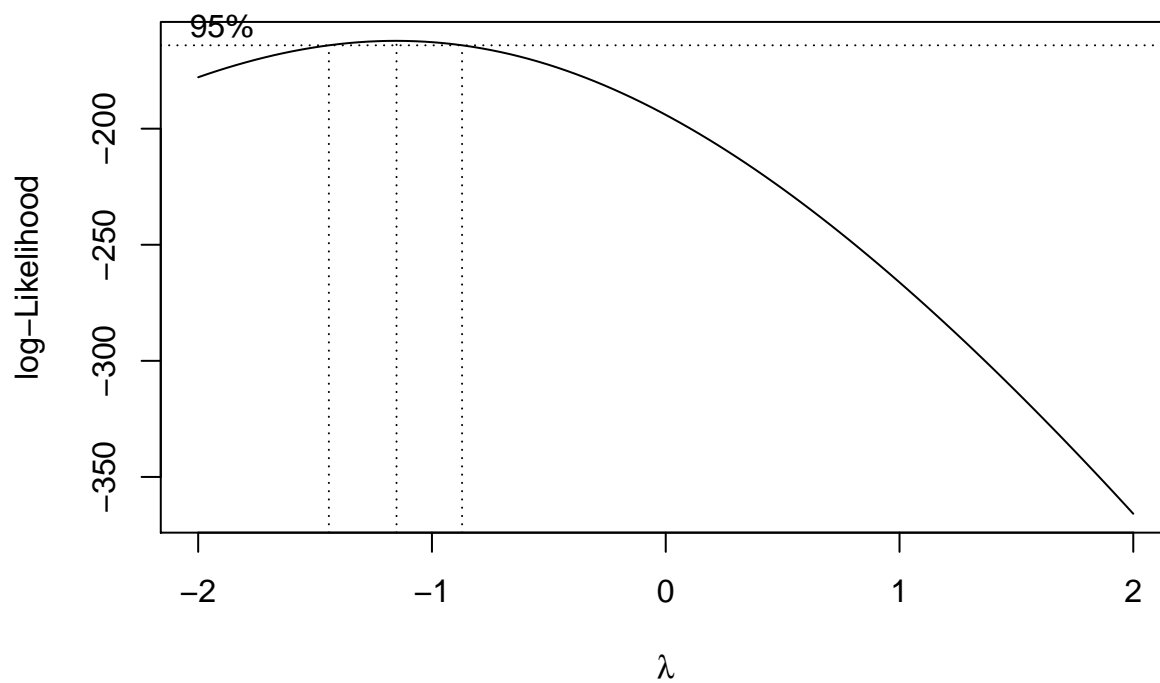
```
bxcx_transf <- function(x, lambda) {
  if (lambda == 0) {
    return(log(x))
  } else {
    return((x^(lambda)-1)/lambda)
  }
}
```

```
transform2 <- data.frame(read.table('../data/Transform2_V2.txt', header=TRUE))
trm1 <- lm(y2 ~ x1, data=transform2)
trm2 <- lm(y2 ~ x2, data=transform2)
par(mfrow=c(2,4))
plot(trm1)
plot(trm2)
```

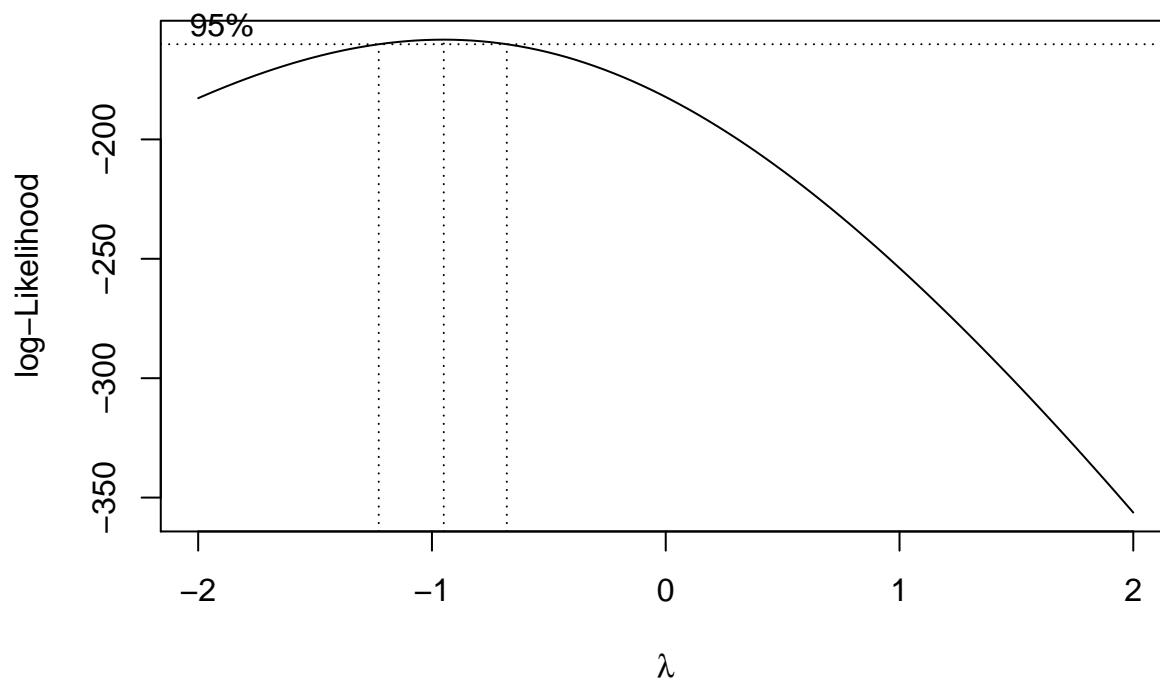


Neither variable has constant variance, therefore we apply a boxcox transformation to both  $X_2$  and  $X_3$ .

```
boxcox(trm2)
```

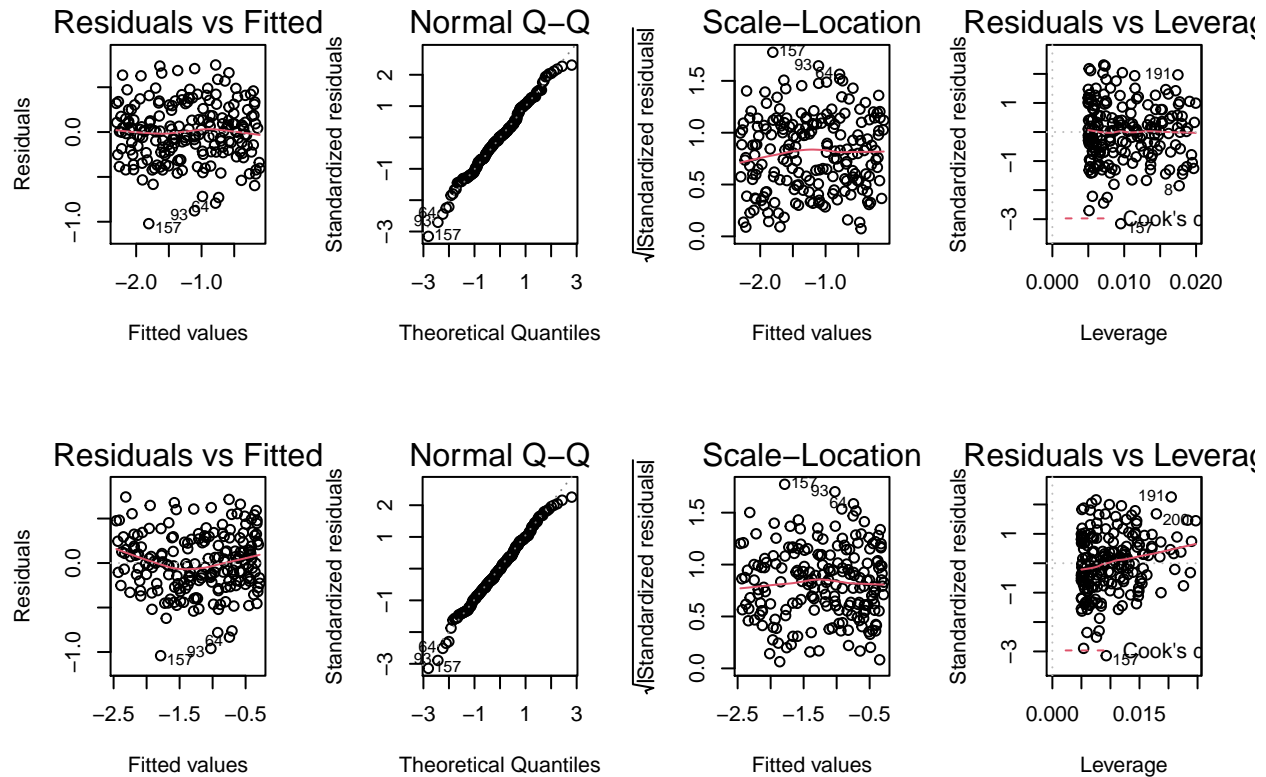


```
boxcox(trm1)
```



As  $-1$  is in the confidence interval for both models, we will apply a boxcox transformation using  $\lambda = -1$ .

```
transform2 <- data.frame(read.table('../data/Transform2_V2.txt', header=TRUE))
trm1 <- lm(bxcx_transf(y2,-1) ~ x1, data=transform2)
trm2 <- lm(bxcx_transf(y2,-1) ~ x2, data=transform2)
par(mfrow=c(2,4))
plot(trm1)
plot(trm2)
```



Here we see the situation has improved significantly and we now have constant variance.