HW4_ssingh478

Chapter7: Exercise 5

For the prostate data, fit a model with lpsa as the response and the other variables as predictors ## (a) Compute and comment on the condition numbers:

Answer:

There are 6 condition numbers which are larger than 30 which indicates that there might be correlations between predictors and there be more than just one combination of predictors.

```
data(prostate,package='faraway')
lmod75<-lm(lpsa~.,data=prostate)
x <- model.matrix(lmod75)[,-1]
e <- eigen(t(x) %*% x)
e$val
```

```
## [1] 4.790826e+05 6.190704e+04 2.109042e+02 1.756329e+02 6.479853e+01
## [6] 4.452379e+01 2.023914e+01 8.093145e+00
```

```
sqrt(e$val[1]/e$val)
```

```
## [1]   1.00000   2.78186  47.66094  52.22787  85.98499 103.73114 153.85414
## [8] 243.30248
```

(b) Compute and comment on the correlations between the predictors.

Answer:

The correlation matrix shows that are many predictors which are strongly correlated.

```
round(cor(prostate[,-9]),2)
```

```
##         lcavol lweight  age  lbph   svi   lcp gleason pgg45
## lcavol    1.00    0.19 0.22  0.03  0.54  0.68    0.43  0.43
## lweight   0.19    1.00 0.31  0.43  0.11  0.10    0.00  0.05
## age       0.22    0.31 1.00  0.35  0.12  0.13    0.27  0.28
## lbph      0.03    0.43 0.35  1.00 -0.09 -0.01    0.08  0.08
## svi       0.54    0.11 0.12 -0.09  1.00  0.67    0.32  0.46
## lcp       0.68    0.10 0.13 -0.01  0.67  1.00    0.51  0.63
## gleason   0.43    0.00 0.27  0.08  0.32  0.51    1.00  0.75
## pgg45     0.43    0.05 0.28  0.08  0.46  0.63    0.75  1.00
```

(c) Compute the variance inflation factors

```
require(faraway)
```

```
## Loading required package: faraway
```

```
## Warning: package 'faraway' was built under R version 3.4.4
```

```
vif(x)
```

```
##   lcavol  lweight      age     lbph      svi      lcp  gleason    pgg45
## 2.054115 1.363704 1.323599 1.375534 1.956881 3.097954 2.473411 2.974361
```

Chapter7: Exercise 6

a)Is the predictor Lactic statistically significant in this model?

b)Give the R command to extract the p-value for the test of βlactic = 0. Hint:look at summary()$coef

```
data("cheddar")
lmod76<-lm(taste~.,data=cheddar)

summary(lmod76)$coef[4,4]
```

```
## [1] 0.03107948
```

```
##p-value for Blactic=0.03107948
```

Answer:

Yes, as p-value=0.031<0.05, hence Lactic is significant.

c)

Add normally distributed errors to Lactic with mean zero and standard deviation 0.01 and refit the model. Now what is the p-value for the previous test?

```
ched<-as.data.frame(cheddar)
chedt<-as.data.frame(cheddar)
chedt$Lactic<-chedt$Lactic+rnorm(30,0,0.01)
lmod76b<-lm(taste~.,data=chedt)
summary(lmod76b)$coef[4,4]
```

```
## [1] 0.0315838
```

```
##p-value for Blactic=0.03360579
```

Answer:

p-value for Blactic=0.03360579

d)

Repeat this same calculation of adding errors to Lactic 1000 times within for loop. Save the p-values into a vector. Report on the average p-value. Does this much measurement error make a qualitative difference to the conclusions?

```
p=c()
for (i in 1:1000)
{
chedt$Lactic<-ched$Lactic+rnorm(30,0,0.01)
lmod76t<-lm(taste~.,data=chedt)
summary(lmod76t)$coef[4,4]

p[i]=summary(lmod76t)$coef[4,4]
}
p_avg=mean(p)
p_avg
```

```
## [1] 0.03143328
```

```
##p-mean value for Blactic=0.03360579
```

Answer:

p-value doesn't vary much. Hence, Lactic is still significant. There is no qualitative difference to the conclusions

e)

Repeat the previous question but with a standard deviation of 0.1. Does this much measurement error make an important difference?

```
p=c()
for (i in 1:1000)
{
chedt$Lactic<-ched$Lactic+rnorm(30,0,0.1)
lmod76t<-lm(taste~.,data=chedt)
summary(lmod76t)$coef[4,4]

p[i]=summary(lmod76t)$coef[4,4]
}
p_avg=mean(p)
p_avg
```

```
## [1] 0.06843327
```

```
##p-mean value for Blactic=0.06951663
```

Answer:

Now p-value is greater than 0.05, which means Lactic will become insignificant and it makes an important difference.

Chapter 8: Exercise 1

a)Fit a regression model Lab ~ Field. Check for non-constant variance

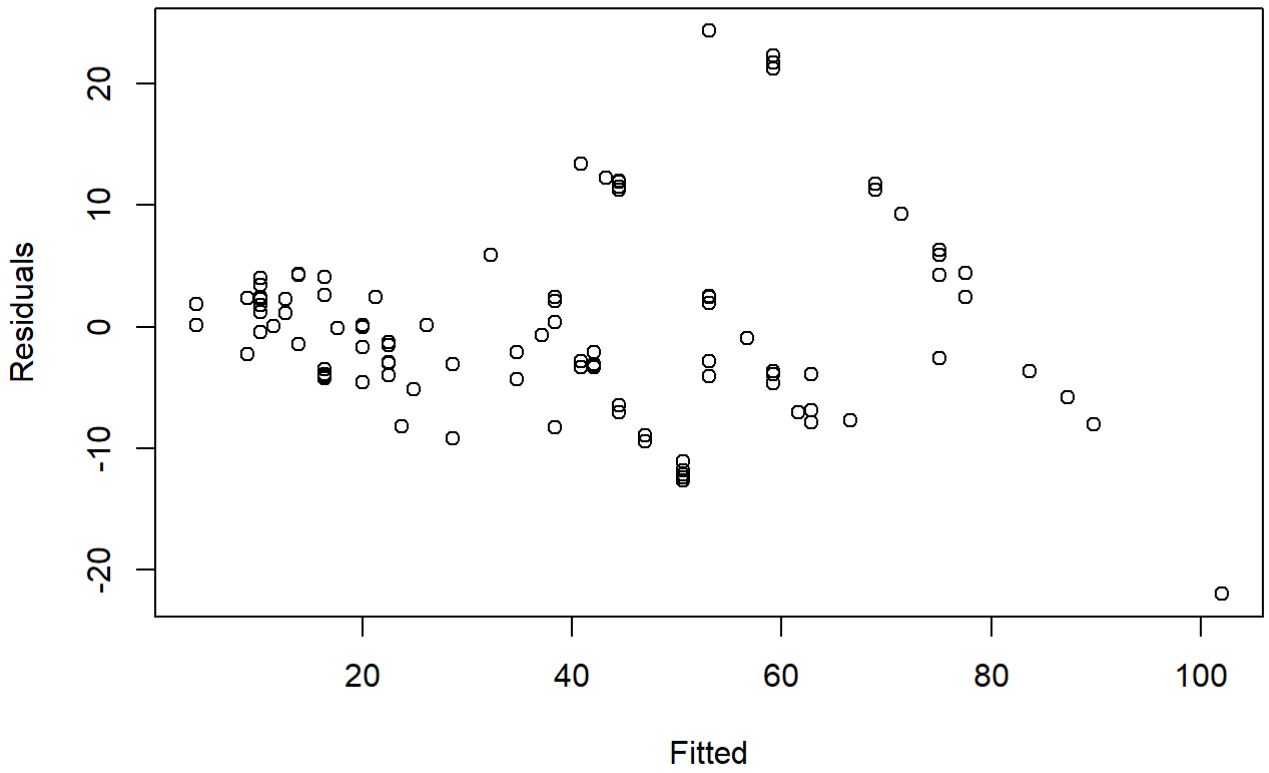
Answer:

Pipeline data is loaded from the 'faraway' package. A linear model is fitted to model the Lab response against the Field predictor. Residuals are plotted against Fitted values to check for non-constant variance. As seen in the plot, there indeed is a non=constant variance.

```
data(pipeline,package='faraway')
lmod<-lm(Lab~Field,data=pipeline)
summary(lmod)
```

```
##
## Call:
## lm(formula = Lab ~ Field, data = pipeline)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -21.985  -4.072  -1.431   2.504  24.334
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.96750    1.57479  -1.249    0.214
## Field        1.22297    0.04107  29.778   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.865 on 105 degrees of freedom
## Multiple R-squared:  0.8941, Adjusted R-squared:  0.8931
## F-statistic: 886.7 on 1 and 105 DF,  p-value: < 2.2e-16
```

```
plot(fitted(lmod),residuals(lmod),xlab="Fitted",ylab="Residuals")
```



b)

Regress log(varlab) on log(meanfield) to estimate a0 and a1. (You might choose to remove the last point.) Use this to determine appropriate weights in a WLS fit of Lab on Field. Show the regression summary.

Answer:

Splitting Field into 12 groups of size nine. Next the variance of Lab within the group is plotted with mean of the Field within each group. A logarithmic relationship is assumed and parameters are estimated. a0=-1.9352 and a1=1.6707. Next, the weights in the linear model are modified for every preictor with weights varying as 1/Field^(a1)

```
pipeline<-as.data.frame(pipeline)

i <- order(pipeline$Field)
npipe <- pipeline[i,]
ff <- gl(12,9)[-108]
meanfield <- unlist(lapply(split(npipe$Field,ff),mean))
varlab <- unlist(lapply(split(npipe$Lab,ff),var))

lmod2<-lm(log(varlab[-12])~log(meanfield[-12]))
summary(lmod2)
```

```
##
## Call:
## lm(formula = log(varlab[-12]) ~ log(meanfield[-12]))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.00477 -0.42268  0.05989  0.37854  0.93815
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -1.9352     1.0929  -1.771 0.110403
## log(meanfield[-12])   1.6707     0.3296   5.070 0.000672 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.657 on 9 degrees of freedom
## Multiple R-squared:  0.7406, Adjusted R-squared:  0.7118
## F-statistic:  25.7 on 1 and 9 DF,  p-value: 0.0006723
```

```
#a0=-1.9352,a1=1.6707

lmod3<-lm(Lab~Field,data=pipeline,weights=1/Field^(1.6707))

summary(lmod3)
```

```
##
## Call:
## lm(formula = Lab ~ Field, data = pipeline, weights = 1/Field^(1.6707))
##
## Weighted Residuals:
##      Min       1Q   Median       3Q      Max
## -0.66245 -0.25532 -0.09474  0.22675  1.03651
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.05531    0.69766  -1.513    0.133
## Field        1.18963    0.03401  34.984   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3742 on 105 degrees of freedom
## Multiple R-squared:  0.921,  Adjusted R-squared:  0.9202
## F-statistic:  1224 on 1 and 105 DF,  p-value: < 2.2e-16
```

c)

An alternative to weighting is transformation. Find transformations on Lab and/or Field so that in the transformed scale the relationship is approximately linear with constant variance. You may restrict your choice of transformation to square root, log and inverse.

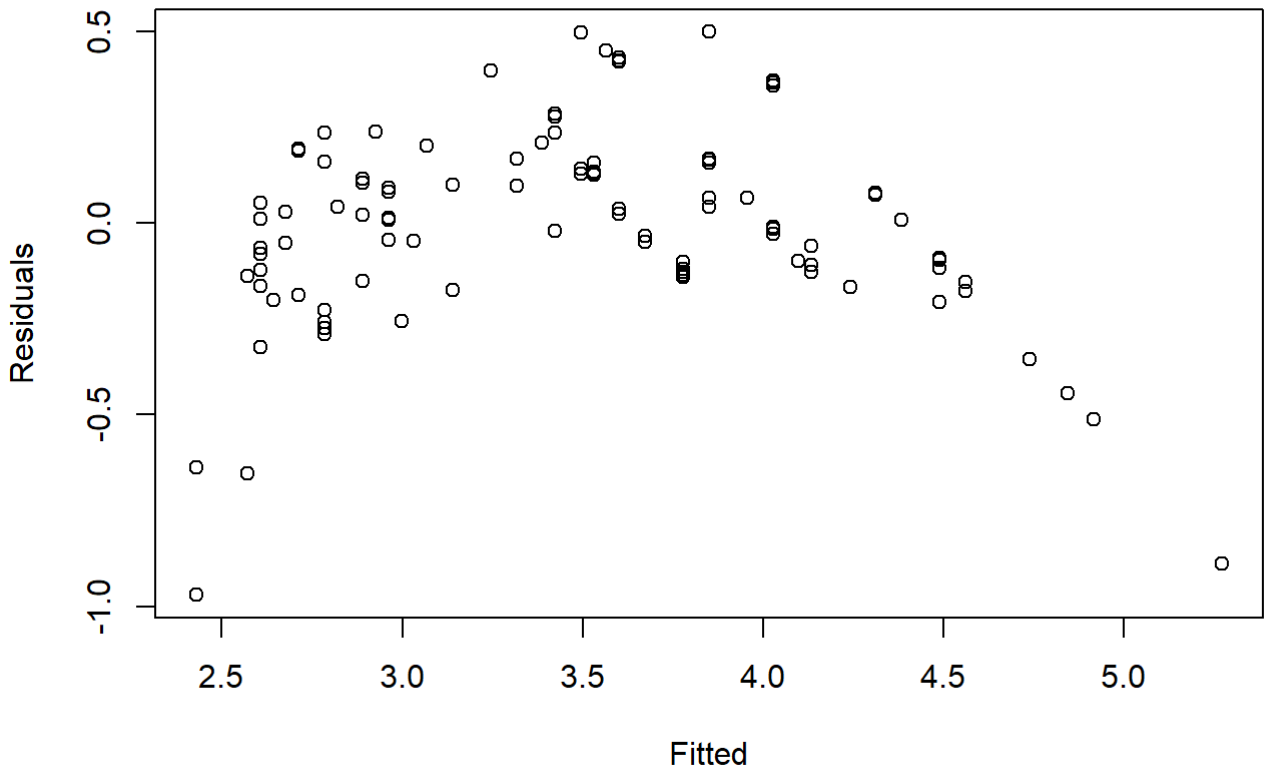
Answer:

The transformation log(Lab) vs log(Field) produces the linear model which has a R-squared value of 0.93 and Residual Plotted shows a constant variance.

```
lmodc1<-lm(log(Lab)~Field,data=pipeline)
summary(lmodc1)
```

```
##
## Call:
## lm(formula = log(Lab) ~ Field, data = pipeline)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.97034 -0.13220  0.00857  0.15797  0.49898
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.251322   0.052269   43.07   <2e-16 ***
## Field       0.035526   0.001363   26.06   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.261 on 105 degrees of freedom
## Multiple R-squared:  0.8661, Adjusted R-squared:  0.8648
## F-statistic: 679.2 on 1 and 105 DF,  p-value: < 2.2e-16
```

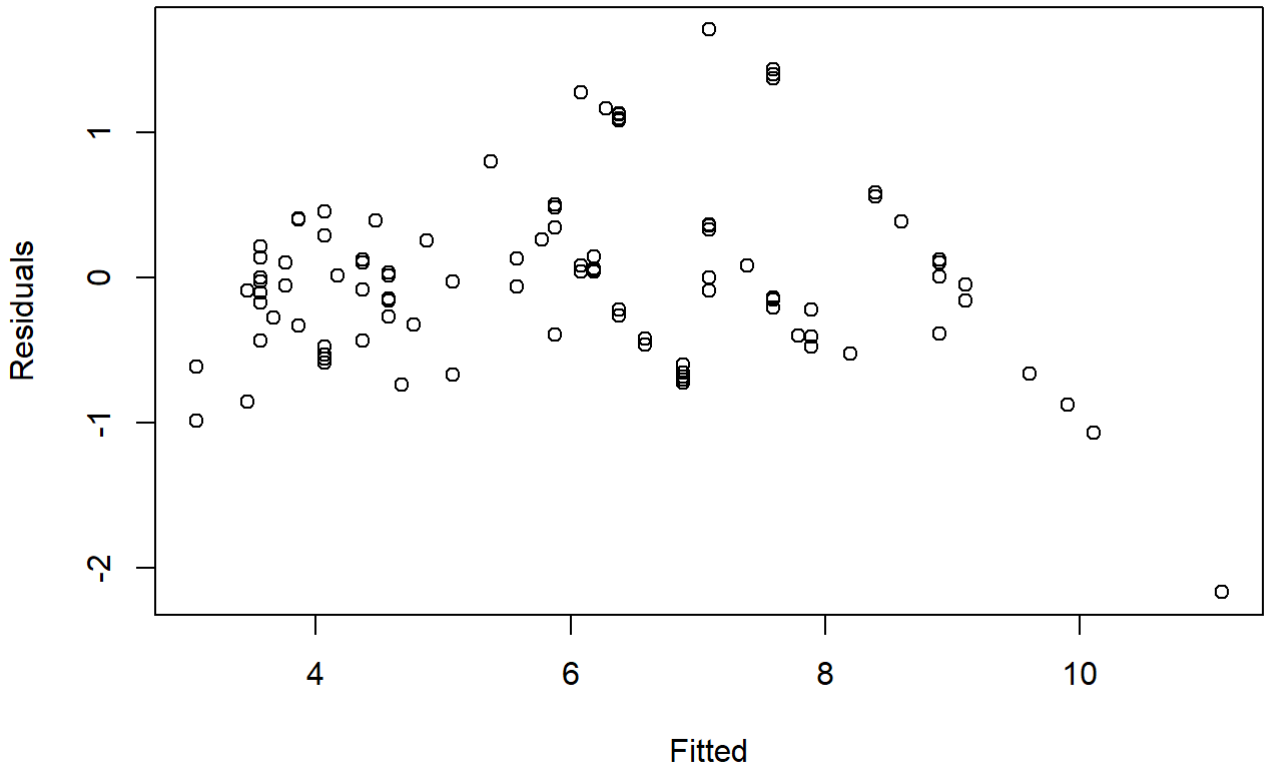
```
plot(fitted(lmodc1),residuals(lmodc1),xlab="Fitted",ylab="Residuals")
```



```
lmodc2<-lm(sqrt(Lab)~Field,data=pipeline)
summary(lmodc2)
```

```
##
## Call:
## lm(formula = sqrt(Lab) ~ Field, data = pipeline)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.1688 -0.4060 -0.0514  0.2766  1.7103
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.558536   0.122530   20.88   <2e-16 ***
## Field       0.100642   0.003195   31.50   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6119 on 105 degrees of freedom
## Multiple R-squared:  0.9043, Adjusted R-squared:  0.9034
## F-statistic: 991.9 on 1 and 105 DF,  p-value: < 2.2e-16
```

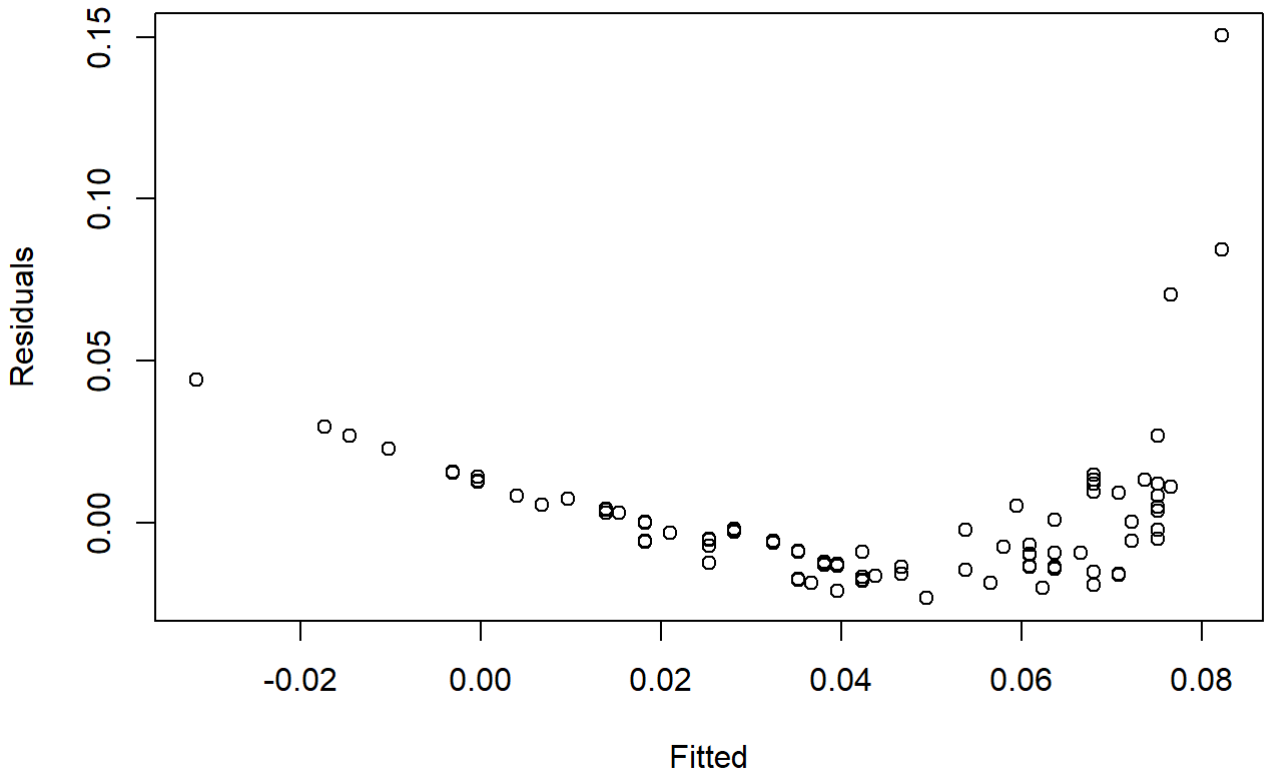
```
plot(fitted(lmodc2),residuals(lmodc2),xlab="Fitted",ylab="Residuals")
```




```
lmodc3<-lm((1/Lab)~Field,data=pipeline)
summary(lmodc3)
```

```
##
## Call:
## lm(formula = (1/Lab) ~ Field, data = pipeline)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.023296 -0.012976 -0.005380  0.005615  0.150379
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.0892885  0.0044632   20.00   <2e-16 ***
## Field       -0.0014220  0.0001164  -12.22   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02229 on 105 degrees of freedom
## Multiple R-squared:  0.587,  Adjusted R-squared:  0.5831
## F-statistic: 149.2 on 1 and 105 DF,  p-value: < 2.2e-16
```

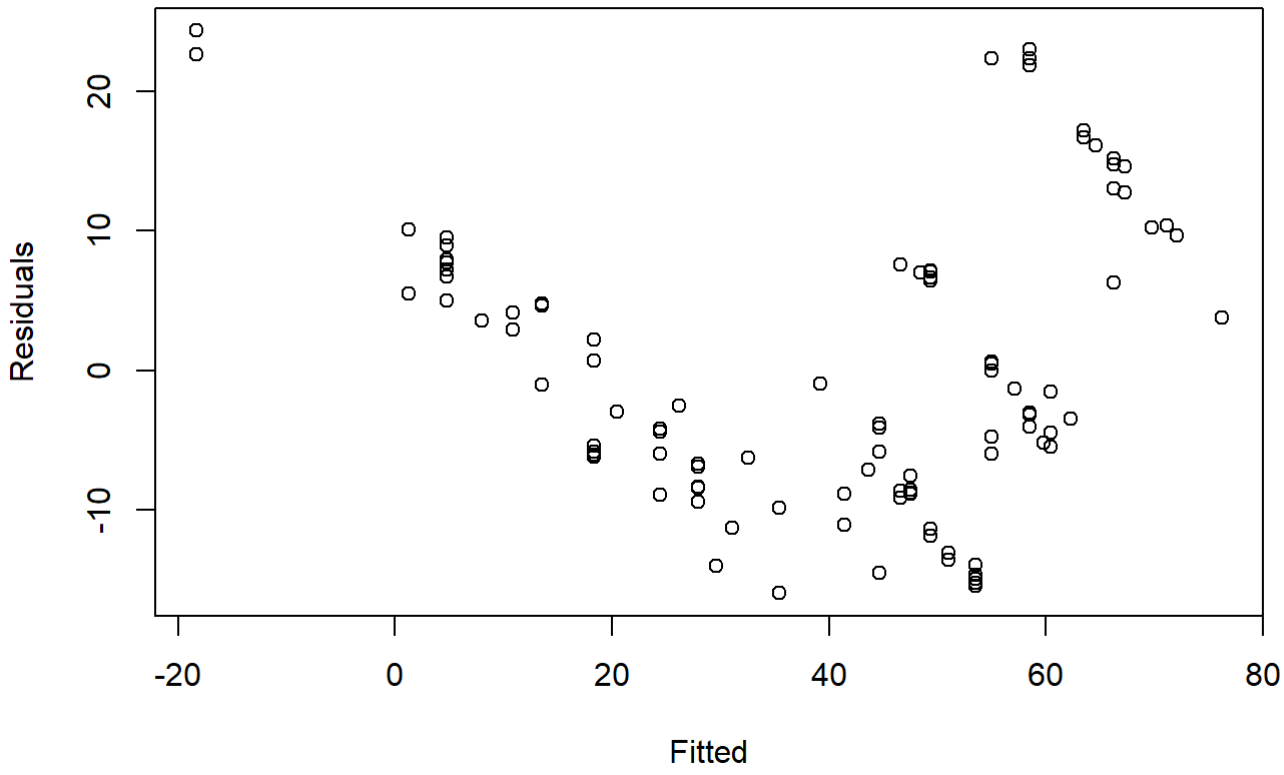
```
plot(fitted(lmodc3),residuals(lmodc3),xlab="Fitted",ylab="Residuals")
```




```
lmodc4<-lm((Lab)~log(Field),data=pipeline)
summary(lmodc4)
```

```
##
## Call:
## lm(formula = (Lab) ~ log(Field), data = pipeline)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.984  -8.435  -3.022   7.089  24.342
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -72.068      5.272  -13.67   <2e-16 ***
## log(Field)    33.382      1.554   21.48   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.41 on 105 degrees of freedom
## Multiple R-squared:  0.8146, Adjusted R-squared:  0.8129
## F-statistic: 461.4 on 1 and 105 DF,  p-value: < 2.2e-16
```

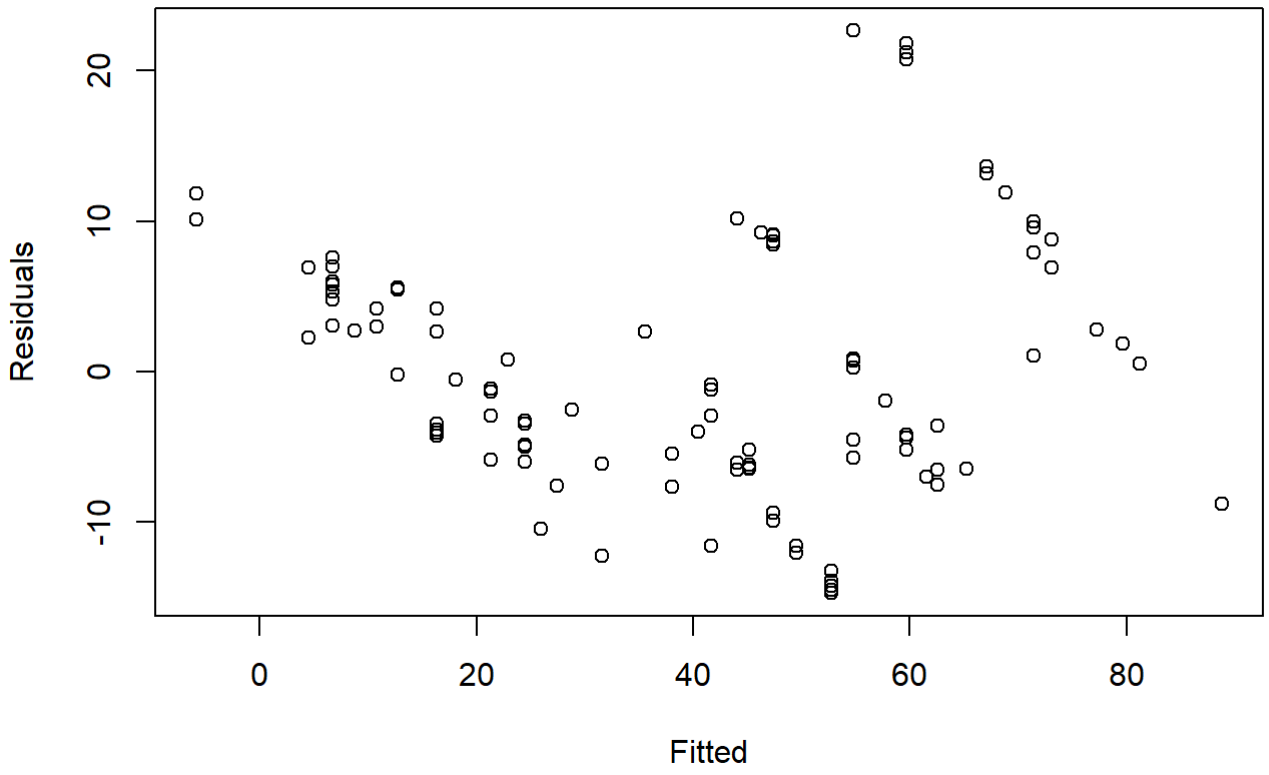
```
plot(fitted(lmodc4),residuals(lmodc4),xlab="Fitted",ylab="Residuals")
```




```
lmodc5<-lm((Lab)~sqrt(Field),data=pipeline)
summary(lmodc5)
```

```
##
## Call:
## lm(formula = (Lab) ~ sqrt(Field), data = pipeline)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.706  -5.843  -1.343   5.538  22.652
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -36.1385     2.8573  -12.65   <2e-16 ***
## sqrt(Field)  13.5486     0.4931   27.48   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.446 on 105 degrees of freedom
## Multiple R-squared:  0.8779, Adjusted R-squared:  0.8767
## F-statistic:   755 on 1 and 105 DF,  p-value: < 2.2e-16
```

```
plot(fitted(lmodc5),residuals(lmodc5),xlab="Fitted",ylab="Residuals")
```




```
lmodc6<-lm((Lab)~(1/Field),data=pipeline)
summary(lmodc6)
```

```
##
## Call:
## lm(formula = (Lab) ~ (1/Field), data = pipeline)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -34.799 -20.749  -1.099  16.451  42.801
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   39.099      2.326   16.81   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 24.06 on 106 degrees of freedom
```

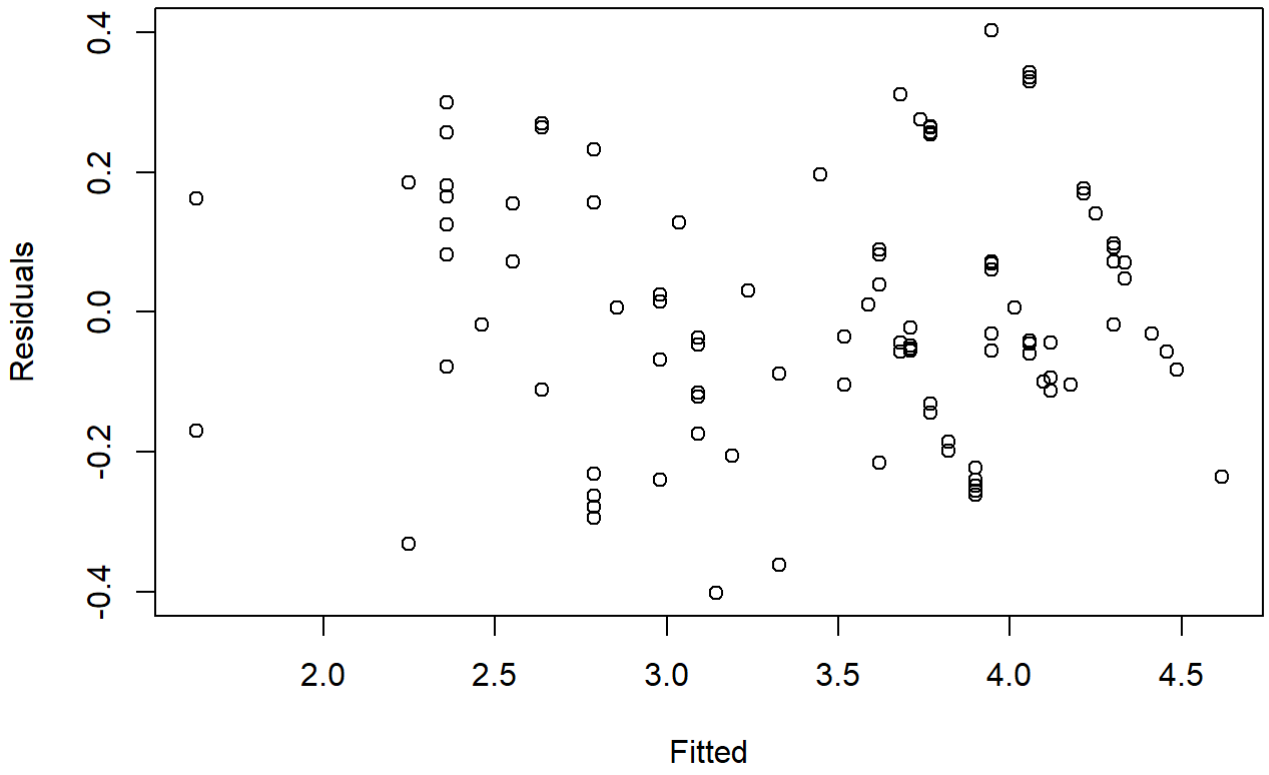
```
plot(fitted(lmodc6),residuals(lmodc6),xlab="Fitted",ylab="Residuals")
```

```
## Warning in plot.window(...): relative range of values = 29 * EPS, is small
## (axis 1)
```


```
lmodc7<-lm(log(Lab)~log(Field),data=pipeline)
summary(lmodc7)
```

```
##
## Call:
## lm(formula = log(Lab) ~ log(Field), data = pipeline)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.40212 -0.11853 -0.03092  0.13424  0.40209
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.06849    0.09305  -0.736    0.463
## log(Field)   1.05483    0.02743  38.457   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1837 on 105 degrees of freedom
## Multiple R-squared:  0.9337, Adjusted R-squared:  0.9331
## F-statistic:  1479 on 1 and 105 DF,  p-value: < 2.2e-16
```

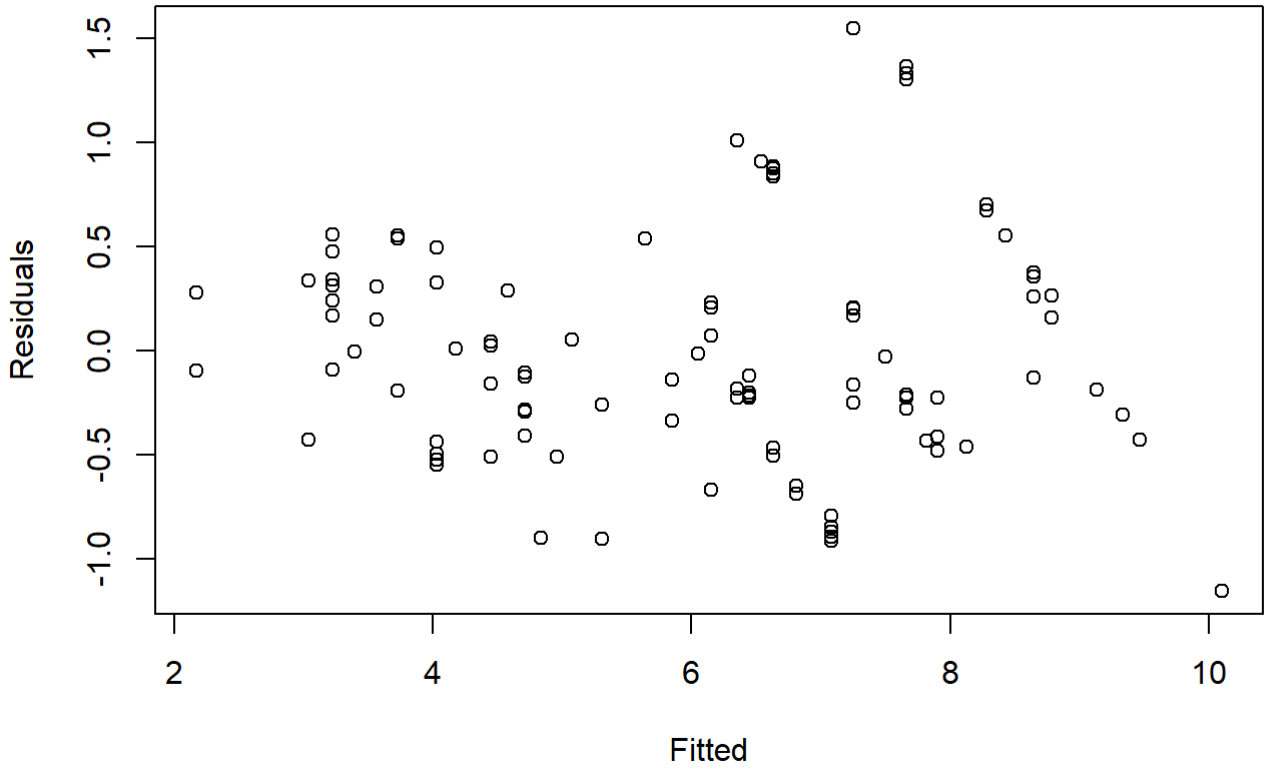
```
plot(fitted(lmodc7),residuals(lmodc7),xlab="Fitted",ylab="Residuals")
```




```
lmodc8<-lm(sqrt(Lab)~sqrt(Field),data=pipeline)
summary(lmodc8)
```

```
##
## Call:
## lm(formula = sqrt(Lab) ~ sqrt(Field), data = pipeline)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.1570 -0.4125 -0.1209  0.3098  1.5481
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.36773    0.18815  -1.954   0.0533 .
## sqrt(Field)  1.13553    0.03247  34.973   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5561 on 105 degrees of freedom
## Multiple R-squared:  0.9209, Adjusted R-squared:  0.9202
## F-statistic:  1223 on 1 and 105 DF,  p-value: < 2.2e-16
```

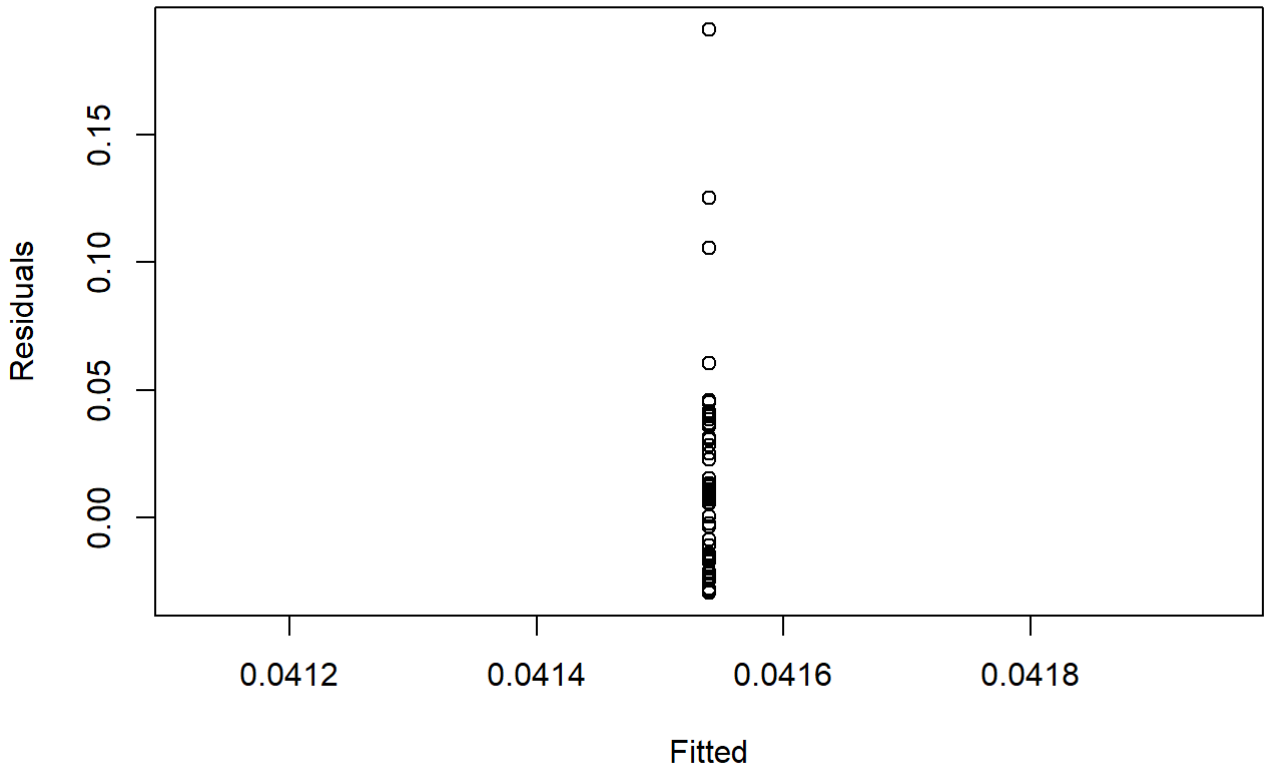
```
plot(fitted(lmodc8),residuals(lmodc8),xlab="Fitted",ylab="Residuals")
```




```
lmodc9<-lm((1/Lab)~(1/Field),data=pipeline)
summary(lmodc9)
```

```
##
## Call:
## lm(formula = (1/Lab) ~ (1/Field), data = pipeline)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.02933 -0.02354 -0.01522  0.01296  0.19102
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.041540   0.003337   12.45   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03452 on 106 degrees of freedom
```

```
plot(fitted(lmodc9),residuals(lmodc9),xlab="Fitted",ylab="Residuals")
```



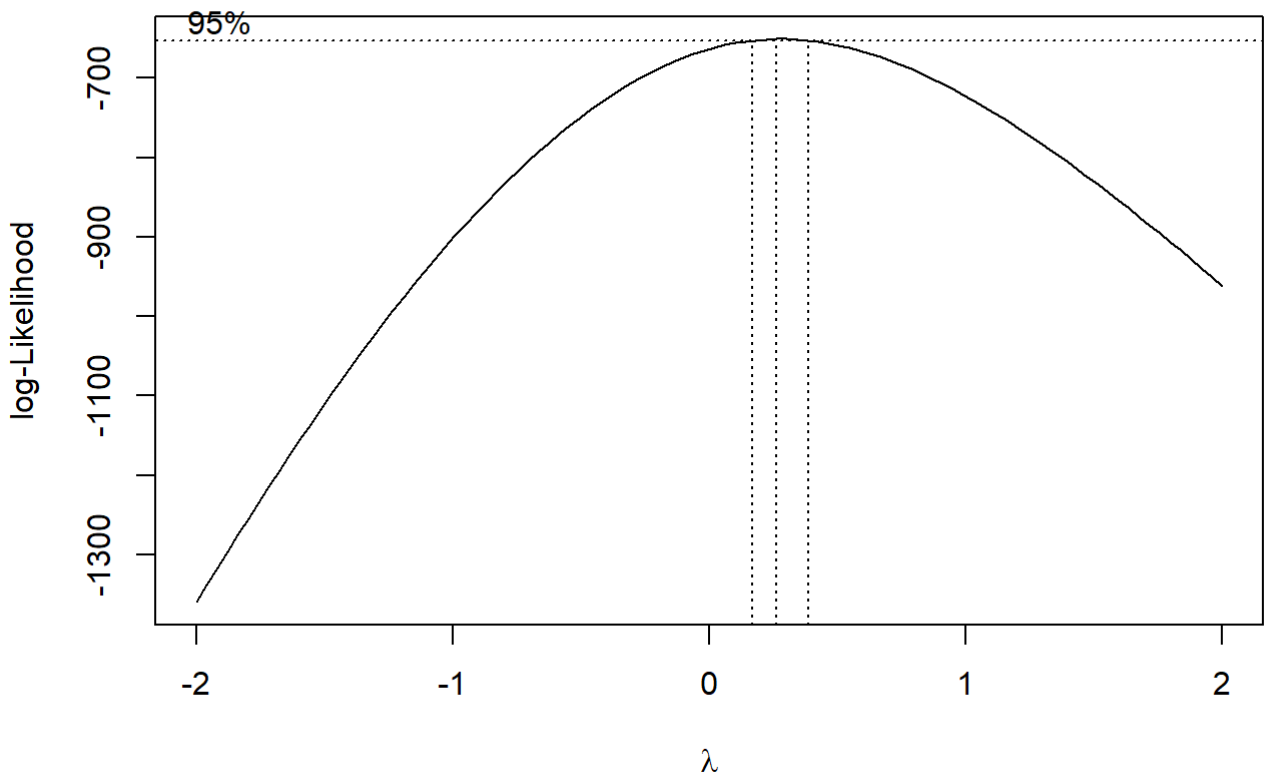
Chapter 9: Exercise 3

Using the ozone data, fit a model with O3 as the response and temp, humidity and ibh as predictors. Use the Box–Cox method to determine the best transformation on the response.

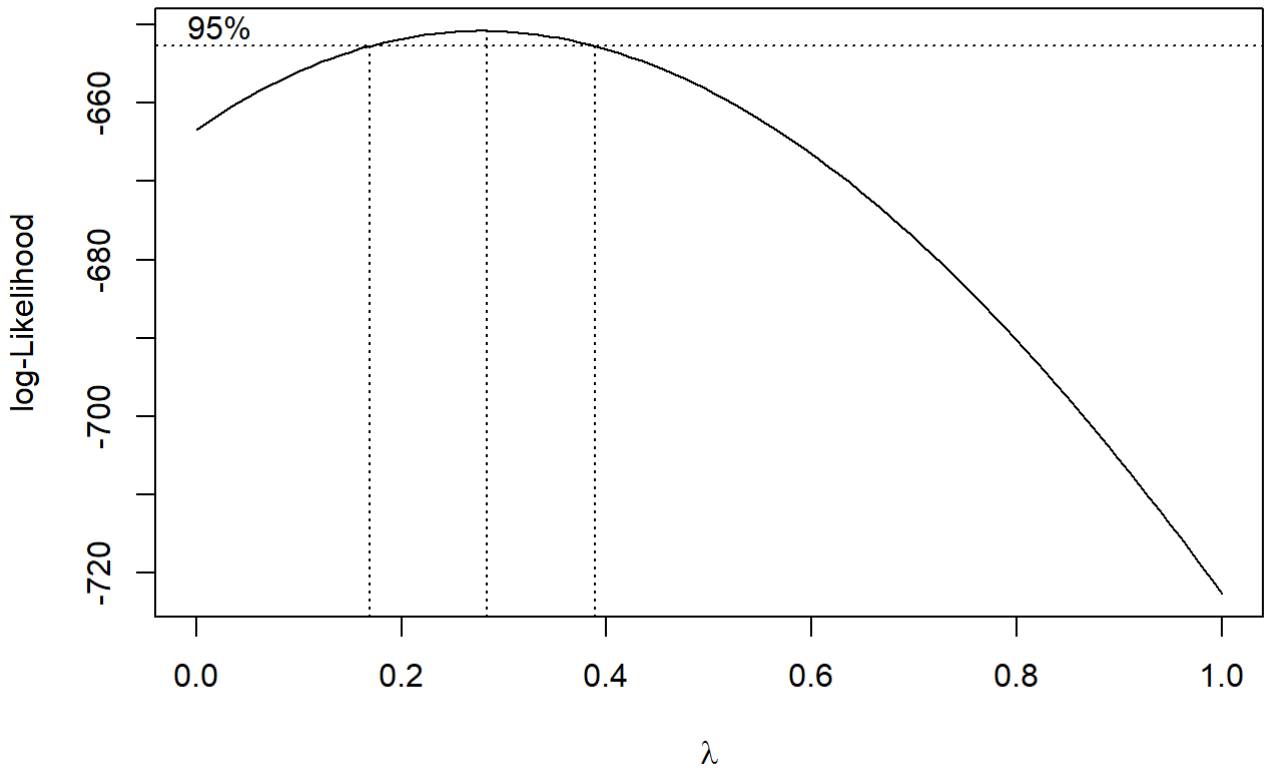
Answer:

First the package, MASS, which contains boxcox transformation function is loaded. Then the data is read and a linear model is fit. BoxCox transformation is made and the resulting log-likelihood ratios are plotted. a second plot is drawn to zoom in on the plot. It is quite clear that the value of lambda which makes the likelihood maximum and hence is the best transformation is between 0.2 and 0.4 and around 0.3. Hence the most appropriate transform on the response is the cuberoot of the transform.

```
library(MASS)
data(ozone,package='faraway')
lmod<-lm(O3~temp+humidity+ibh,data=ozone)
boxcox(lmod,plot=T)
```




```
boxcox(lmod,plot=T,lambda=seq(0,1,by=0.05))
```



Chapter 9: Exerise 4

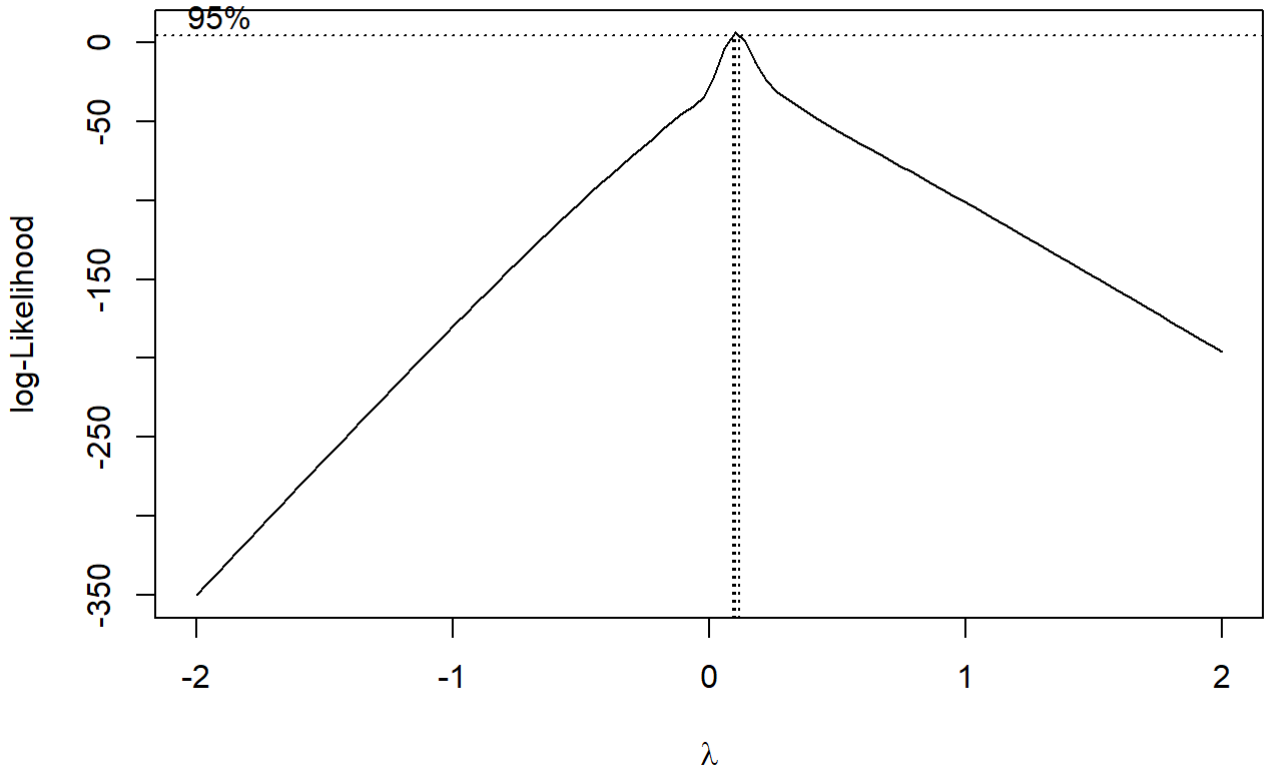
Use the pressure data to fit a model with pressure as the response and temperature as the predictor using transformations to obtain a good fit.

Answer:

```
data(pressure)
lmod9e4<-lm(pressure~temperature,data=pressure)
summary(lmod9e4)
```

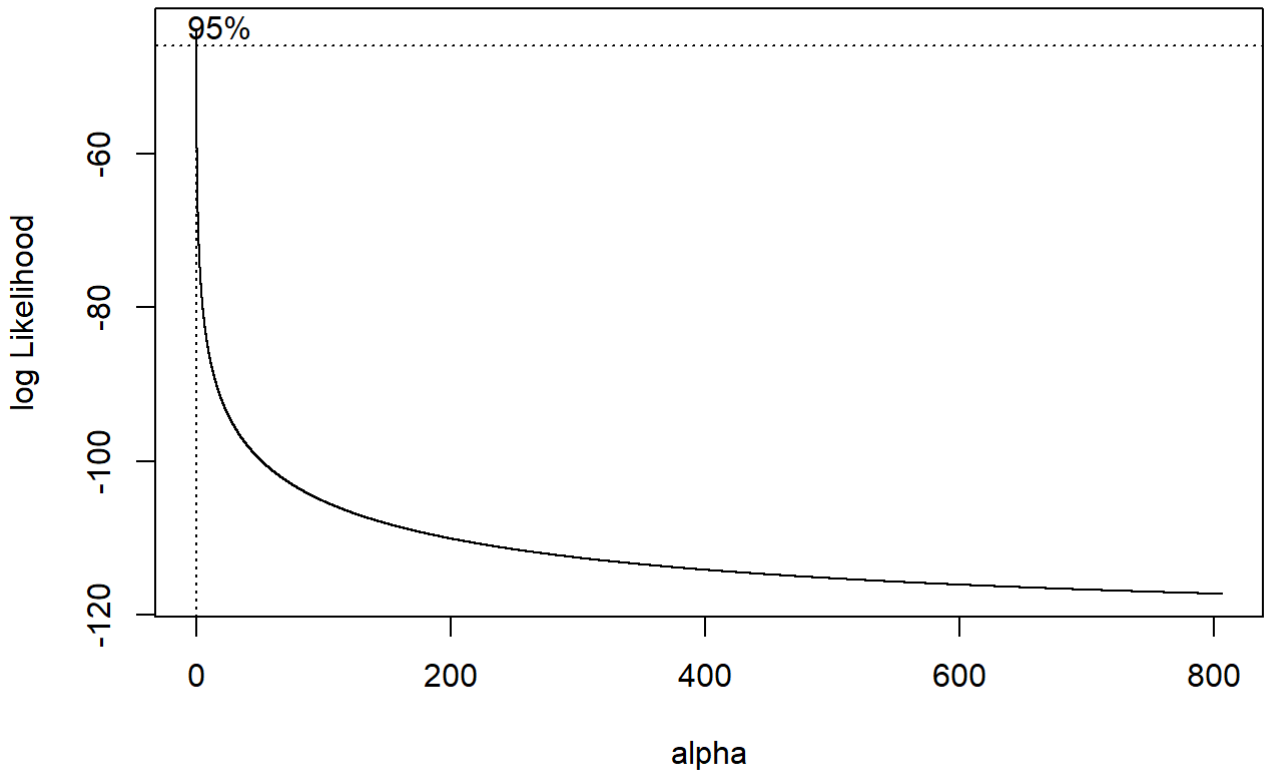
```
##
## Call:
## lm(formula = pressure ~ temperature, data = pressure)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -158.08 -117.06  -32.84   72.30  409.43
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -147.8989    66.5529  -2.222 0.040124 *
## temperature    1.5124     0.3158   4.788 0.000171 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 150.8 on 17 degrees of freedom
## Multiple R-squared:  0.5742, Adjusted R-squared:  0.5492
## F-statistic: 22.93 on 1 and 17 DF,  p-value: 0.000171
```

```
boxcox(lmod9e4,plotit=T)
```



According to this plot using box-cosx transformation with lambda=1 seems like a good transformation.

```
logtrans(lmod9e4,plotit=TRUE, alpha=seq(-min(pressure$temperature)+0.001,806,by=0.01))
```



This transformation suggests that log on the response will be a better tranformation as alpha=0 in this case. As Box Cox transformation also tends to log(response) when lambda tends to 0 as seems to be happening when we apply boxcox to this model is this means log(pressure) vs temp will be a good fit.

```
lmod9e42<-lm(log(pressure)~temperature,data=pressure)

summary(lmod9e42)
```

```
##
## Call:
## lm(formula = log(pressure) ~ temperature, data = pressure)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.4491 -0.6876  0.2866  0.8716  1.1365
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6.068144   0.483831  -12.54 5.10e-10 ***
## temperature  0.039792   0.002296   17.33 3.07e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.096 on 17 degrees of freedom
## Multiple R-squared:  0.9464, Adjusted R-squared:  0.9433
## F-statistic: 300.3 on 1 and 17 DF,  p-value: 3.07e-12
```

As can be seen, the model has improved on every parameter.