STAT 504 - Homework 4

Due date: Thursday, March 14th. Submit your homework solutions to the course Canvas page. Please submit the output and plots, but not your R code unless the quesiton specifically asks for it. Total possible points: 24 points.

1. (2 points) Assessing model diagnostic plots requires experience. Often it is difficult to decide whether a deviation is a systematic one (i.e. needing correction) or a random one (i.e. just variability in the data). Experience can be gained by performing model diagnostics on problems where it is known whether the model assumptions hold or do not hold. This allows us to identify the naturally occurring variability in the results.

   In the following we simulate one predictor `xx` and four responses `yy.a`, `yy.b`, `yy.c`, and `yy.d`.

```
> set.seed(21)
> n <- 100
> xx <- 1:n
> yy.a <- 2+1*xx+rnorm(n)
> yy.b <- 2+1*xx+rnorm(n)*(xx)
> yy.c <- 2+1*xx+rnorm(n)*(1+xx/n)
> yy.d <- cos(xx*pi/(n/2)) + rnorm(n)
```

   Fit four simple linear regression models using `xx` as the predictor.

   (a) (2 points) For each model, create a scatter plot with the regression line, plot the four standard residual plots and the plot containing Cook's distance. Decide for each model which of the assumptions are fulfilled and which ones are violated. Verify your claims with the construction of the responses.

   Instead of `plot.lm()` you can also use the function `resplot()` provided on Canvas in resplot.R. The function `resplot()` uses resampling to visualize whether a model violation is present. You can also try changing the set.seed() in function resplot.

   Solution:

   From the plots below we can derive the following:

   .a Model assumptions seem valid.

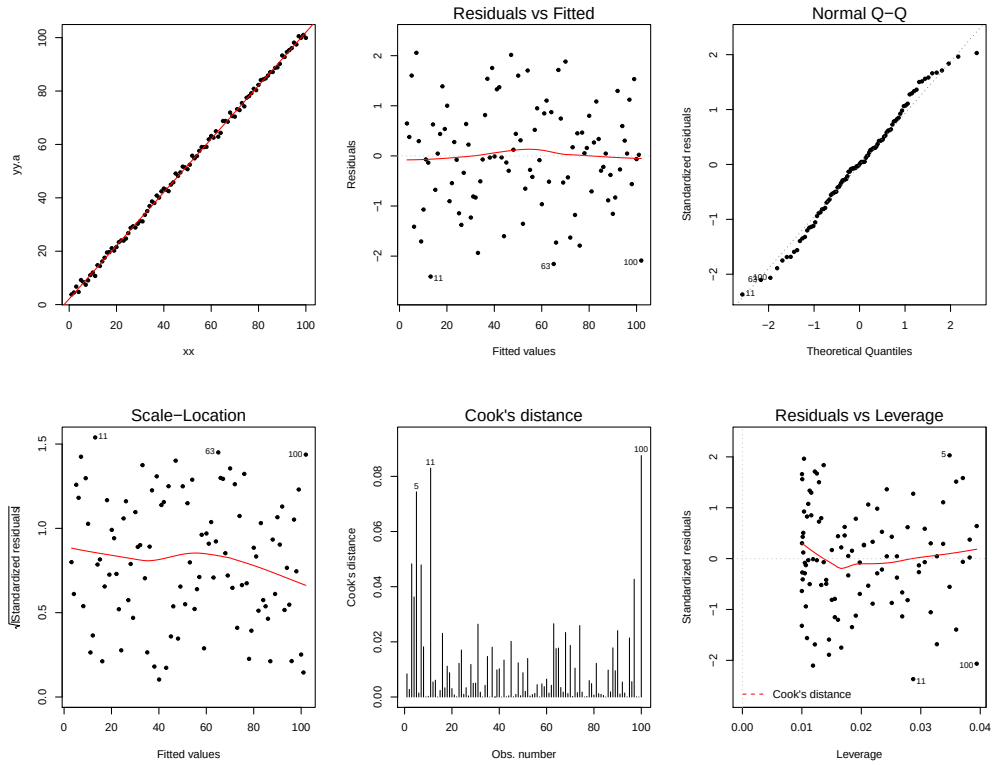
   .b Model contains strong non-constant variance.

   .c Variance slightly non-constant.

   .d Non-linear model (linear model shows systematic error).

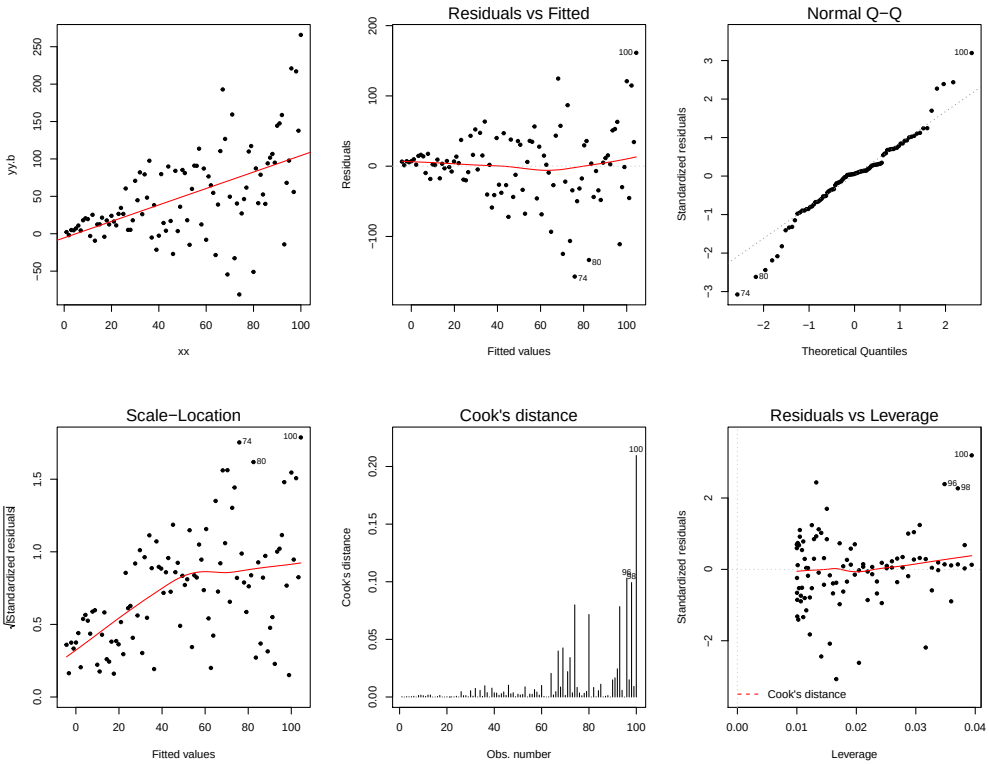
   (.5 Points for each answer above.)

```
> ## yy.a: scatter plots, residuals and Cook's Distance
> par(mfrow=c(2,3))
> plot(yy.a ~ xx, pch=20)
> abline(fit <- lm(yy.a ~ xx), col="red")
> plot(fit,1:5,pch=20)
```



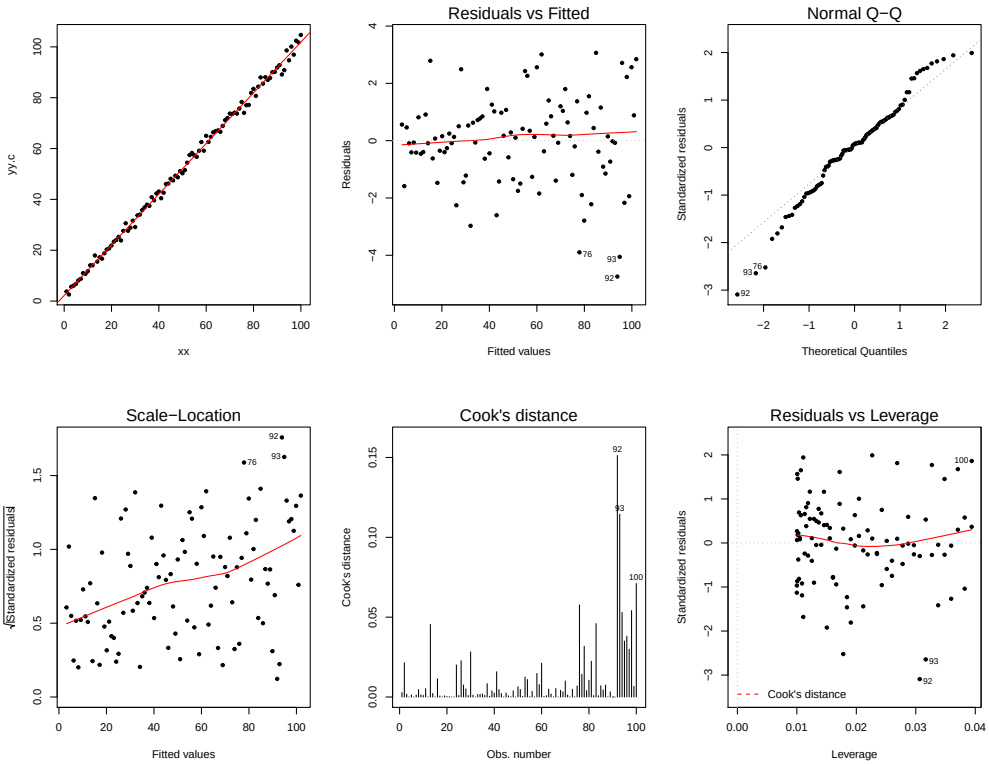
`yy.a`: For the first model the residual plots look perfect. Only in the plot containing Cook's distance, there are a few values that are slightly larger than the rest. These are the observations with the smallest/largest x-values. However, since those values are far from 0.5, there is no problem.

```
> ## yy.b: scatter plots, residuals and Cook's Distance
> par(mfrow=c(2,3))
> plot(yy.b ~ xx, pch=20)
> abline(fit <- lm(yy.b ~ xx), col="red")
> plot(fit,1:5,pch=20)
```



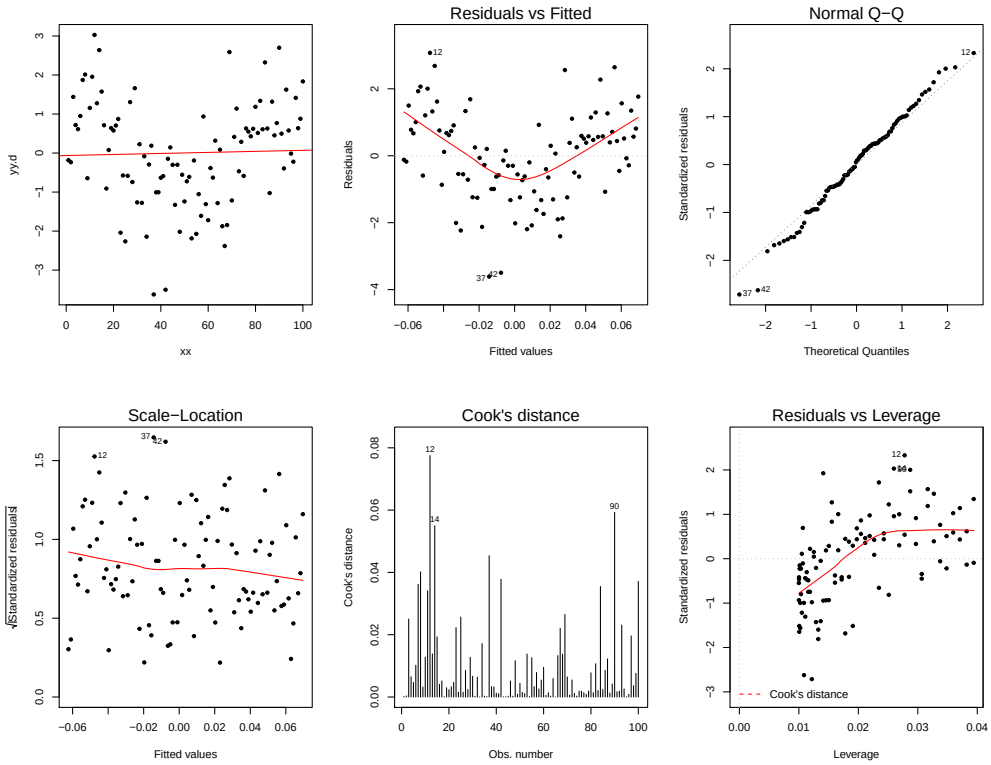
`yy.b`: In case of the second model, we see the increasing variance with the magnitude of the fitted values in the Tukey-Anscombe-Plot. The Normal plot shows a violation of the normality assumption, even though the errors do follow a Normal distribution per definition. However, the variance is not constant which also needs to be fulfilled for the Normal plot (so that the points follow a straight line). So the violation stems from the fact that the variance is not constant. In the scale-location plot we can also see the increase in the variance. There are no leverage points nor influential data points – even though the points with large observation numbers have larger values of Cook's distance.

```
> ## yy.c: scatter plots, residuals and Cook's Distance
> par(mfrow=c(2,3))
> plot(yy.c ~ xx, pch=20)
> abline(fit <- lm(yy.c ~ xx), col="red")
> plot(fit,1:5,pch=20)
```



`yy.c`: For the third model, the analysis is similar as in case of the second model. This is the case because the model violations are similar. The model violation is less accentuated than in the previous example.

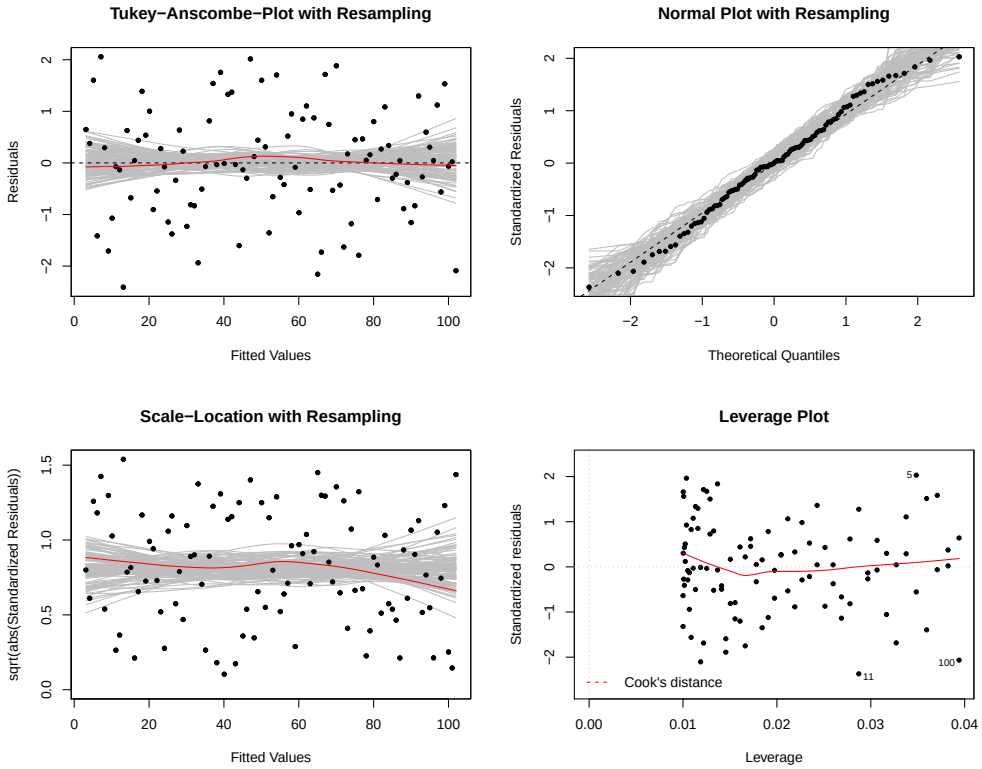
```
> ## yy.d: scatter plots, residuals and Cook's Distance
> par(mfrow=c(2,3))
> plot(yy.d ~ xx, pch=20)
> abline(fit <- lm(yy.d ~ xx), col="red")
> plot(fit,1:5,pch=20)
```



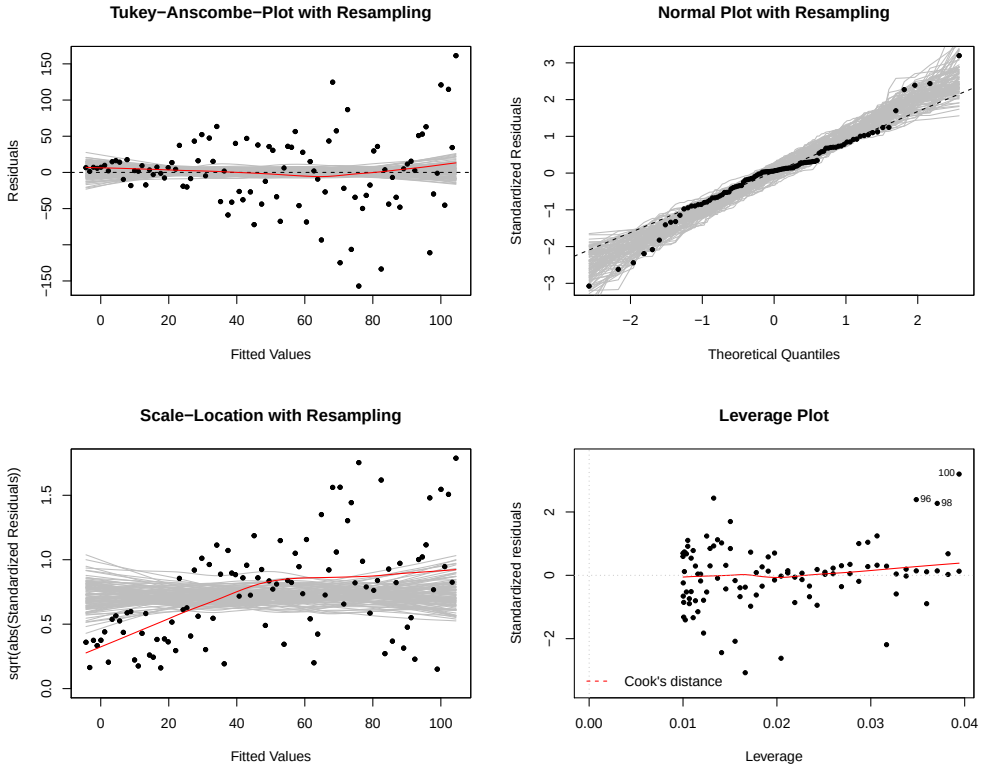
`yy.d`: In case of the fourth model, the systematic error can be easily detected in the Tukey-Anscombe plot since it exhibits a U-shaped pattern. The Normal plot and the scale-location plot do not show any abnormalities. There are no influential data points but the smoother deviates from the horizon in the leverage plot. This is the case because the points with large leverage (i.e. points at the border of this simple regression) have systematically positive residuals.

Solution:

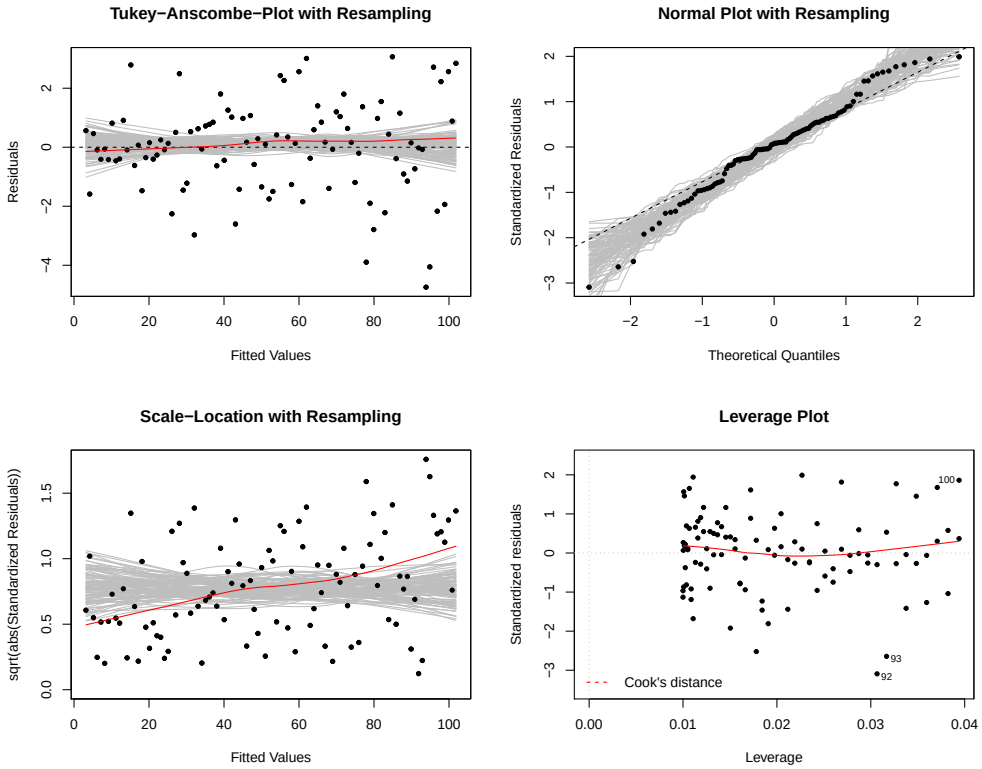
```
> ## source function (needs to be in your working directory)
> source("resplot.R")
> ## yy.a: residual plots with resampling
> par(mfrow=c(2,2))
> fit <- lm(yy.a ~ xx)
> resplot(fit)
```



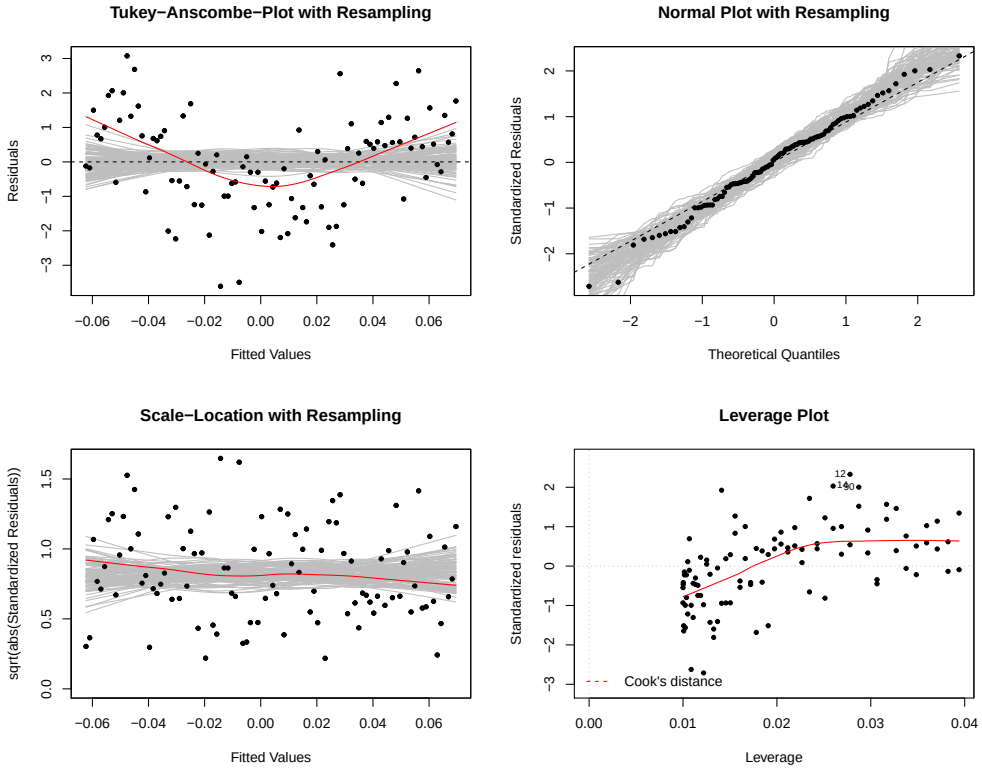

```
> ## yy.b: residual plots with resampling
> fit <- lm(yy.b ~ xx)
> resplot(fit)
```




```
> ## yy.c: residual plots with resampling
> fit <- lm(yy.c ~ xx)
> resplot(fit)
```




```
> ## yy.d: residual plots with resampling
> fit <- lm(yy.d ~ xx)
> resplot(fit)
```



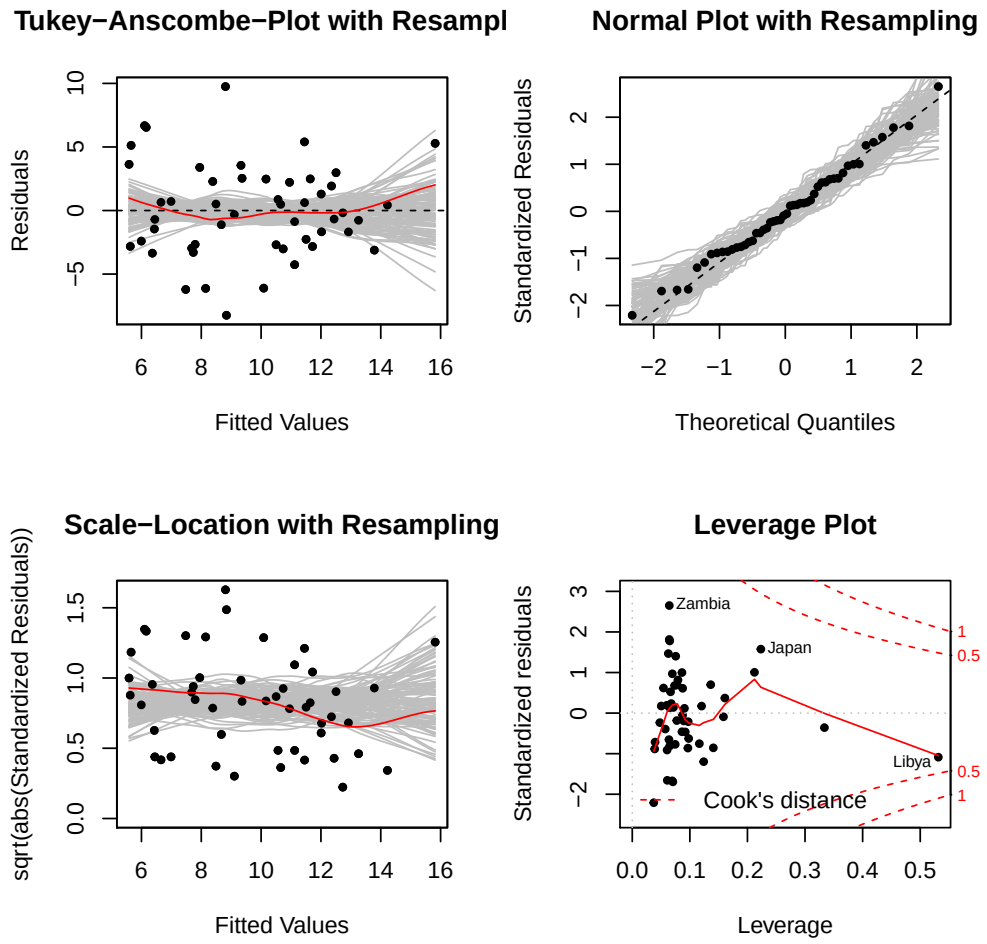
As you can see from the plots, the function does a good job in detecting the three model violations. Additionally, it does not make a mistake "in the other direction", either. I.e. the smoother does not lie outside of the gray area in cases where the model assumptions are fulfilled. In other words, there are no "false positives".

2. (8 points) In this exercise, we would like to analyze how much savings differ between countries. The data set `savings` in R package `faraway` contains 50 observations. For each country the values are averaged over the entire population and the period 1960 - 1970. The variables have the following meanings:
`sr` : proportion of the available income that is saved
`pop15` : proportion of the population that is younger than 15 years
`pop75` : proportion of the population that is older than 75 years
`dpi` : per capita income
`ddpi` : growth rate of `dpi`

(a) (1 point) Fit the model `sr` $\sim$ `pop15 + pop75 + dpi + ddpi`. Do a residual analysis.

Solution:

```
> ## load data
> library(faraway)
> data(savings)
> ## model without transformations
> fit <- lm(sr ~ pop15 + pop75 + dpi + ddpi, data=savings)
> ## residuals and Cook's Distance
> par(mfrow=c(2,2))
> resplot(fit)
```

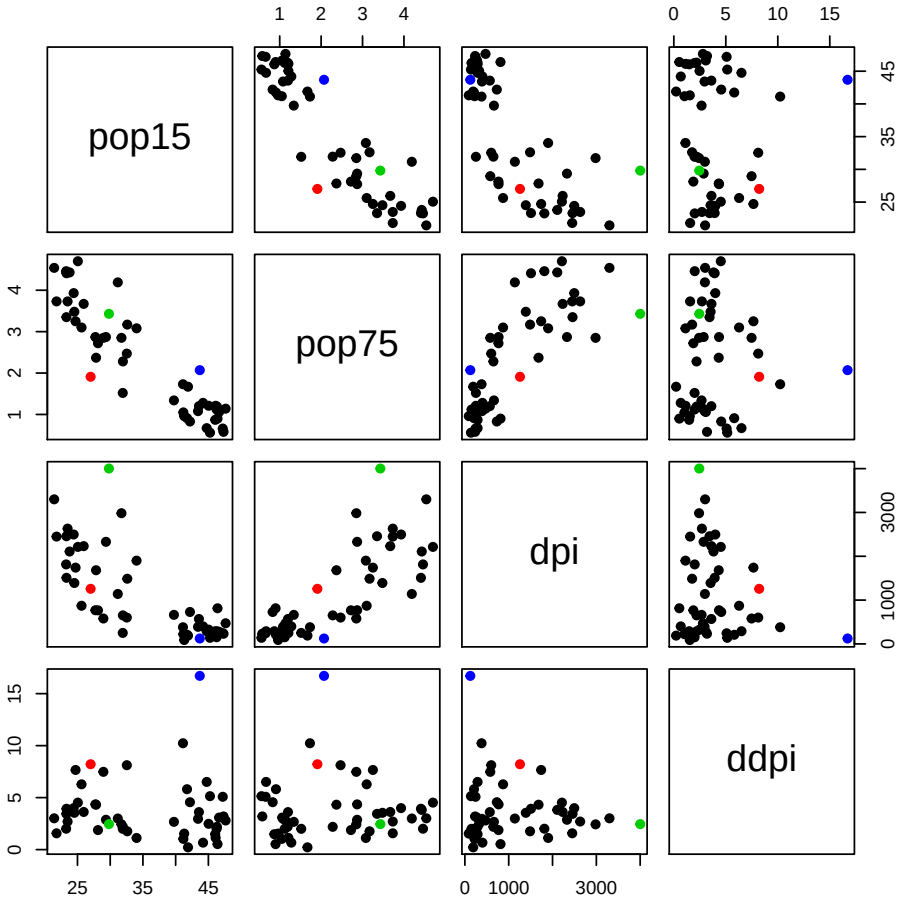


The assumptions seem to be satisfied. There is no violation of constant variance or non-linearities in the TA plot. The Normal QQ-plot also looks satisfactory. There are a few points with large leverage but none of these points is influential as Cook's distance is smaller than 0.5 for all points. **(1 Point)**

(b) (2 points) Identify the three observations having the largest leverage. What countries do these correspond to? How do these points differ from the remaining data points?

Solution:

```
> ## observations with the largest leverage
> sort(hatvalues(fit), decreasing=TRUE)[1:3]

        Libya United States         Japan
    0.5314568     0.3336880     0.2233099

> weli <- which(rownames(savings) %in% c("Libya", "United States", "Japan"))
> farb <- rep(1,nrow(savings))
> farb[weli] <- c(2,3,4)
> pairs(savings[,-1], pch=19, col=farb) ## Japan (red), USA (green), Libya (blue)
```



The three countries with the largest leverage are Libya, the USA, and Japan. **(1 Point)** To simplest way to see why these points have extraordinary predictor configurations is to plot pairwise scatter plots.

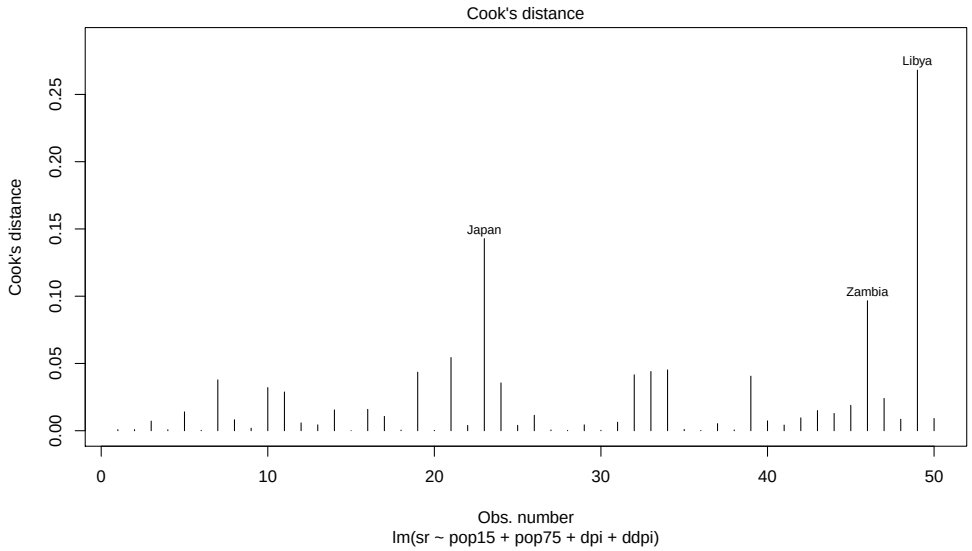
In the plots, Japan corresponds to red, USA to green and Libya to blue. The latter has a very low value of `dpi` but a very large value of `ddpi`. The USA have the largest `dpi` value and a relatively large proportion for `pop75`. Japan, on the other hand, lies at the border in several scatter plots but is not extraordinary with respect to a single feature. **(1 Point)**

(c) (1 point) Remove the data point with the largest Cook's distance from the analysis. To what extent do the results change? Consider the summary of the new model and the new residual plots.

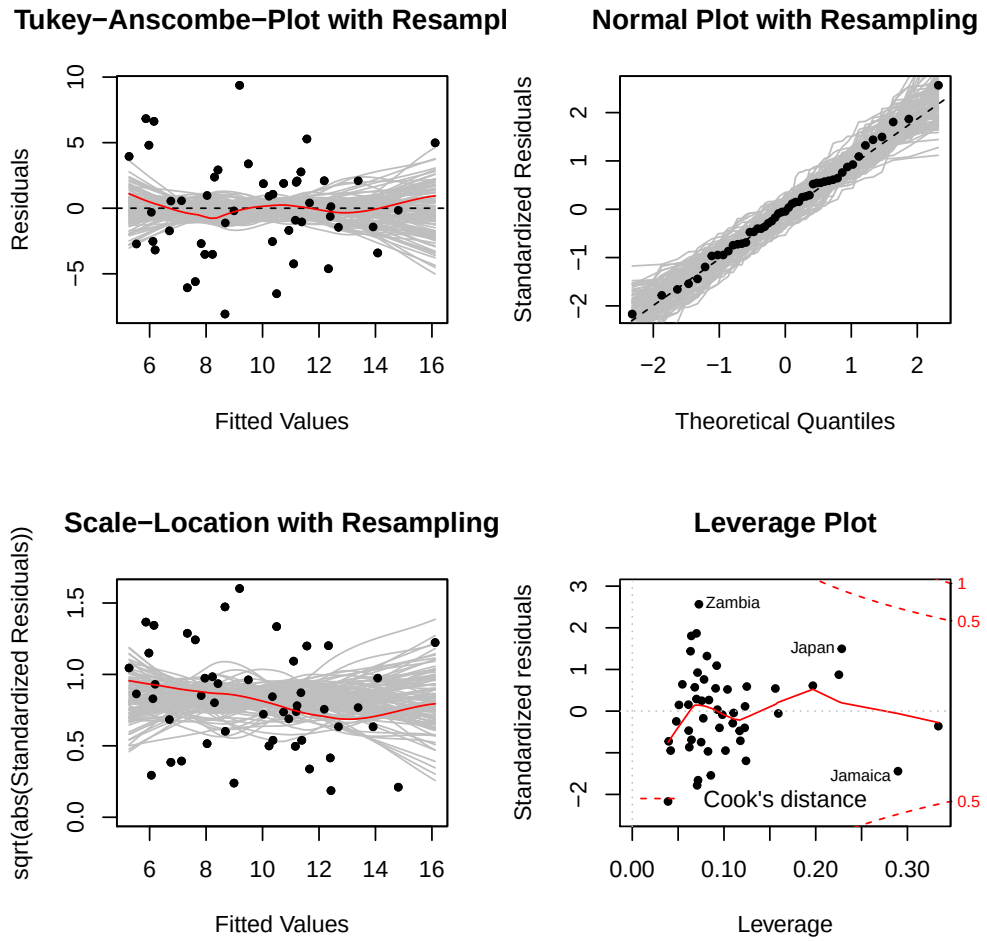
Solution:

```
> ## analysis without data point with largest Cook's distance
> plot(fit, which=4) ## exclude Libya
> weli <- which(rownames(savings)=="Libya")
```

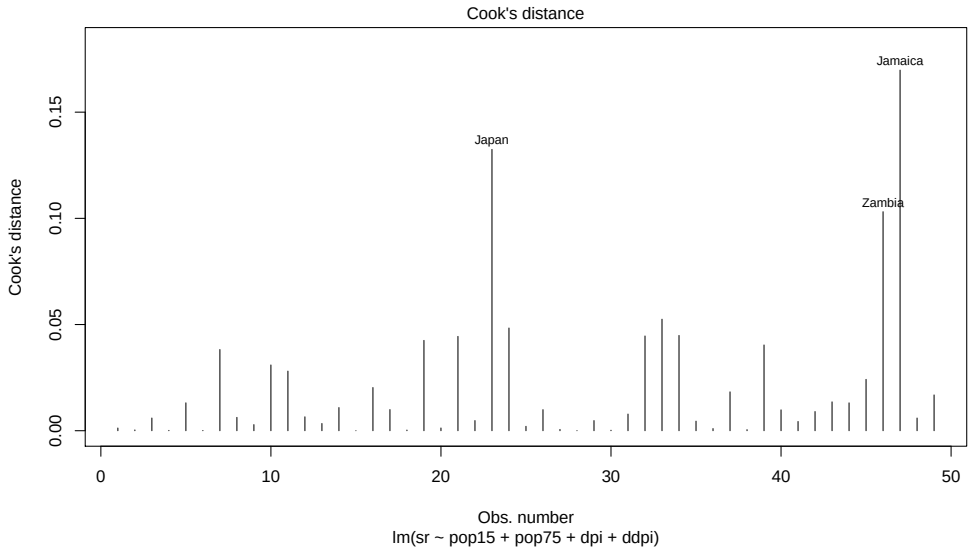
```
> fit1 <- lm(sr ~ pop15 + pop75 + dpi + ddpi, data=savings[-weli,])
> ## comparison of the estimated coefficient
> coef(fit); coef(fit1)


  (Intercept)         pop15         pop75           dpi          ddpi
28.5660865407 -0.4611931471 -1.6914976767 -0.0003369019  0.4096949279


  (Intercept)         pop15         pop75           dpi          ddpi
24.5240459788 -0.3914401268 -1.2808669233 -0.0003189001  0.6102790264


> summary(fit); summary(fit1)


Call:
lm(formula = sr ~ pop15 + pop75 + dpi + ddpi, data = savings)

Residuals:
    Min      1Q  Median      3Q     Max
-8.2422 -2.6857 -0.2488  2.4280  9.7509

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 28.5660865  7.3545161   3.884 0.000334 ***
pop15       -0.4611931  0.1446422  -3.189 0.002603 **
pop75       -1.6914977  1.0835989  -1.561 0.125530
dpi         -0.0003369  0.0009311  -0.362 0.719173
ddpi         0.4096949  0.1961971   2.088 0.042471 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.803 on 45 degrees of freedom
Multiple R-squared:  0.3385,        Adjusted R-squared:  0.2797
F-statistic: 5.756 on 4 and 45 DF,  p-value: 0.0007904


Call:
lm(formula = sr ~ pop15 + pop75 + dpi + ddpi, data = savings[-weli,
    ])

Residuals:
    Min      1Q  Median      3Q     Max
-8.0699 -2.5408 -0.1584  2.0934  9.3732

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 24.5240460  8.2240263   2.982  0.00465 **
pop15       -0.3914401  0.1579095  -2.479  0.01708 *
pop75       -1.2808669  1.1451821  -1.118  0.26943
dpi         -0.0003189  0.0009293  -0.343  0.73312
ddpi         0.6102790  0.2687784   2.271  0.02812 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.795 on 44 degrees of freedom
Multiple R-squared:  0.3554,        Adjusted R-squared:  0.2968
F-statistic: 6.065 on 4 and 44 DF,  p-value: 0.0005617
```




```
> par(mfrow=c(2,2))
> resplot(fit1)
```




```
> par(mfrow=c(1,1))
> plot(fit1, 4, pch=20)
```

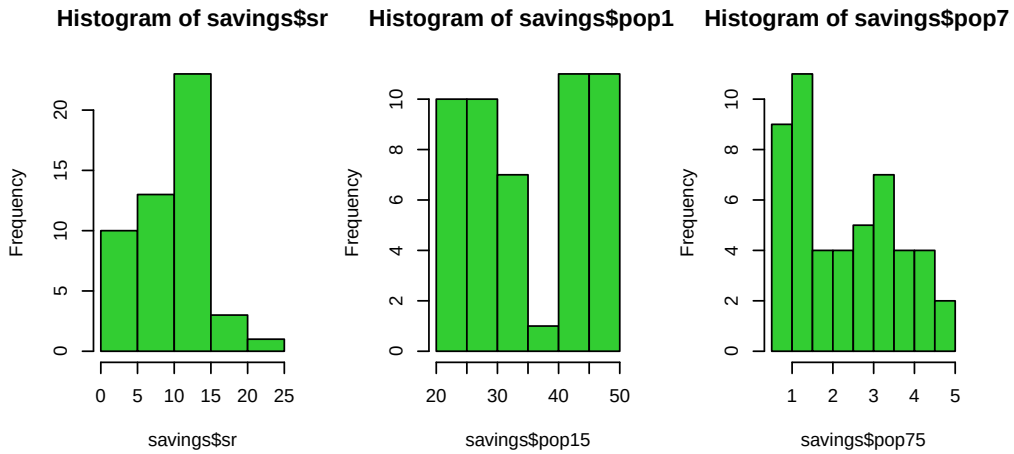


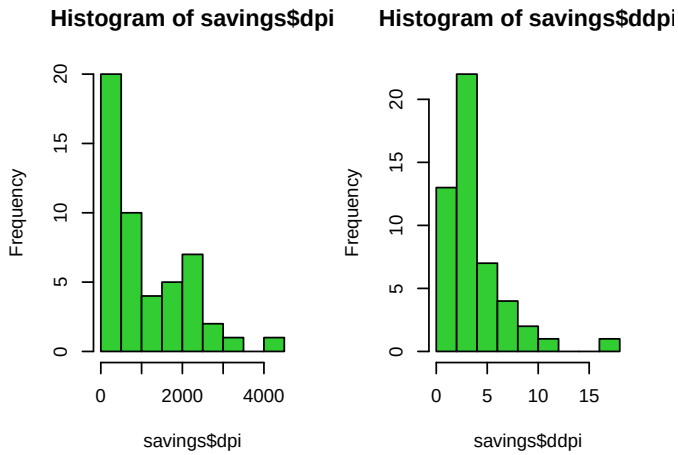
The results only change slightly. The coefficients have similar magnitudes and the same predictors are significant. Also the residual analysis does not yield entirely new insights. This is not surprising as we have seen that Libya does not have a large influence, even though its leverage is high. **(1 Point)**

(d) (4 points) Now consider variable transformations. Plot the histograms of the individual variables and the pairs plot. Use the Box-Cox method to decide whether transforming the response makes sense. Decide on possible variable transformations, fit the corresponding models and analyze the residuals. Finally, decide whether transforming any variables contributes to a better model.
Hint: Use slide 36 from Model Diagnostics I lecture as a reference for possible transformations of predictors. Use slide 33 as a reference to plot the response against a transformed predictor.

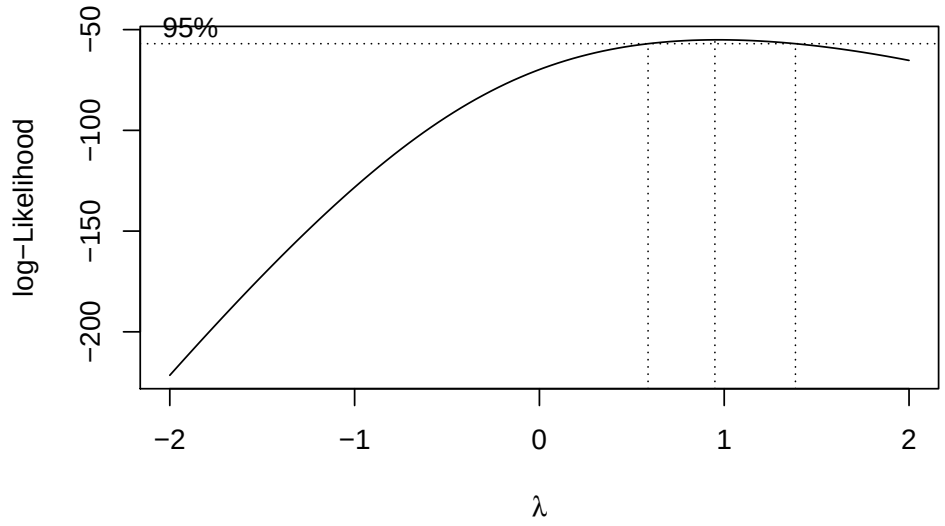
Solution:

```
> ## consider additional models
> par(mfrow=c(2,3))
> hist(savings$sr, col="limegreen")
> hist(savings$pop15, col="limegreen")
> hist(savings$pop75, col="limegreen")
> hist(savings$dpi, col="limegreen")
> hist(savings$ddpi, col="limegreen")
```

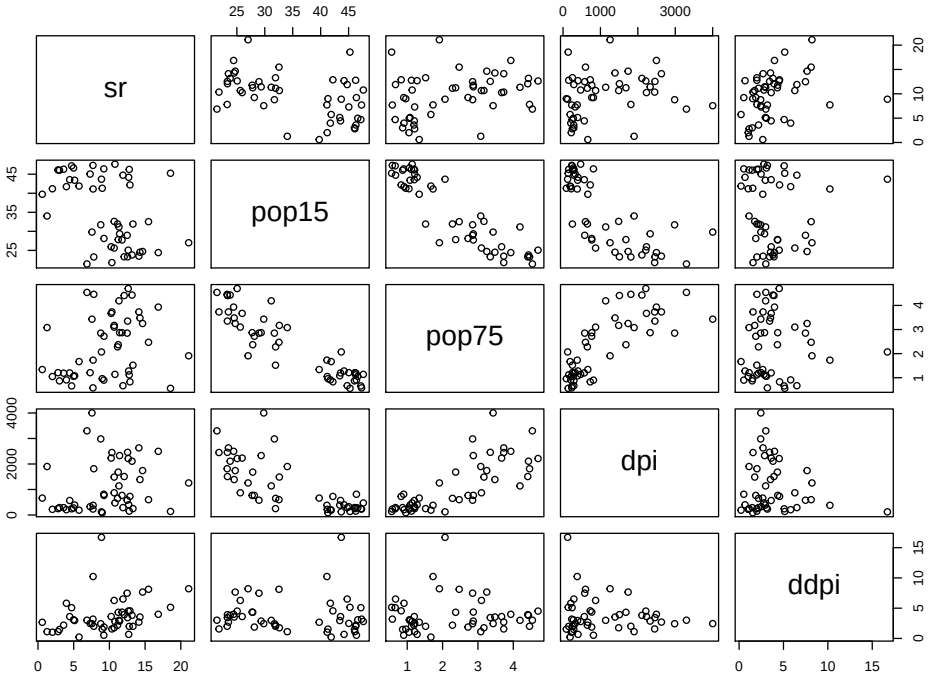





```
> library(MASS)
> boxcox(fit1)
```

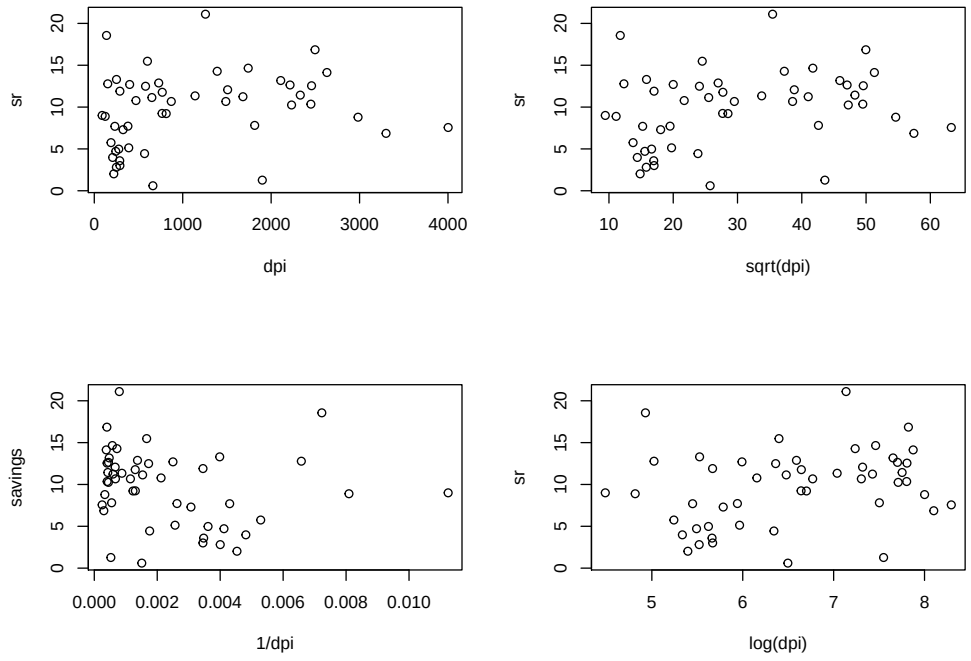



```
> pairs(savings)
```



Based on the Box-Cox result, we should not transform the response `sr`. **(1 Point)** Predictors `dpi` and `ddpi` are both right-skewed and correlated (based on the data description).
Let's consider some possible transformations of `dpi` and `ddpi`.

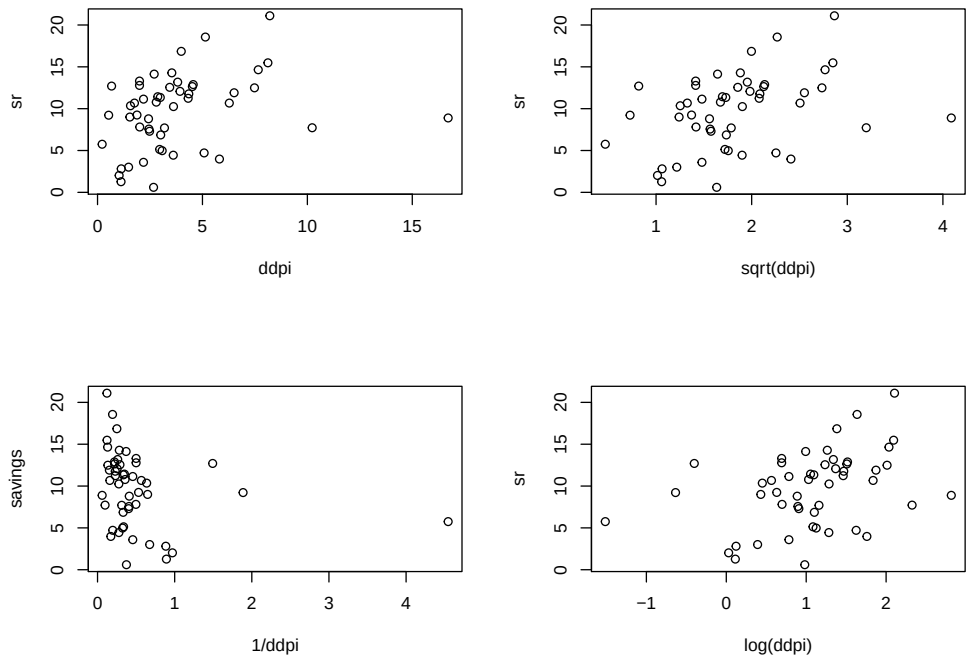
```
> par(mfrow=c(2,2))
> plot(sr~dpi,data=savings)
> plot(sr~sqrt(dpi),data=savings)
> invdpi <- 1/savings$dpi
> plot(savings$sr~invdpi,ylab="savings",xlab="1/dpi")
> plot(sr~log(dpi),data=savings)
```



None of the plots show a clear linear relationship of sr and dpi. Perhaps the log transformed or the square root transformed dpi seem the most reasonable. **(1 Point)**

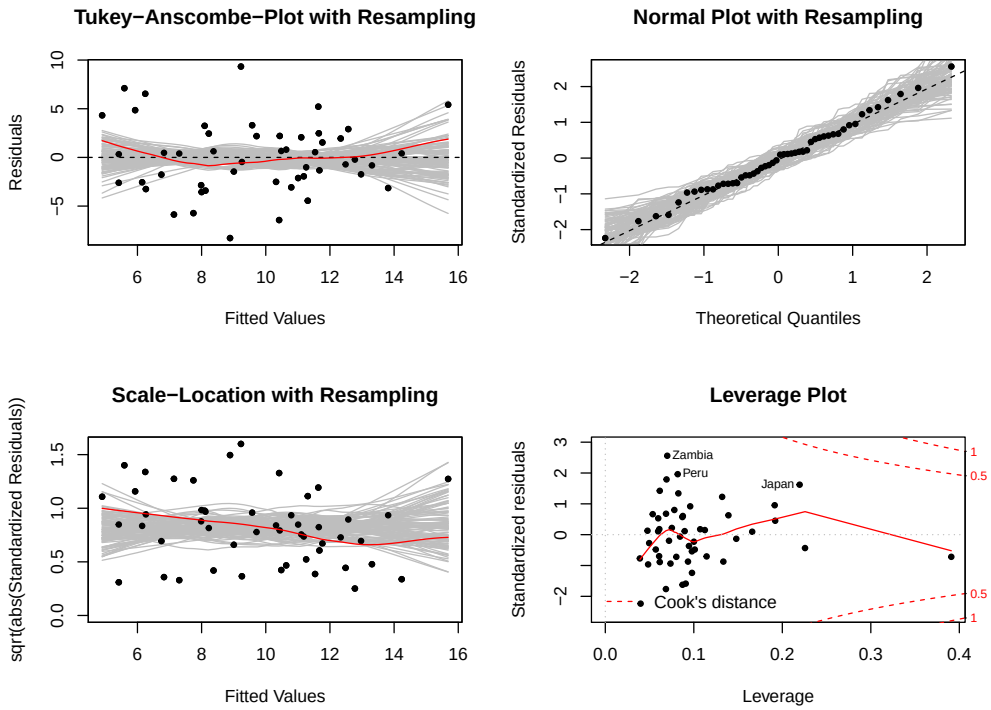
Let's perform the same analysis on `ddpi`.

```
> par(mfrow=c(2,2))
> plot(sr~ddpi,data=savings)
> plot(sr~sqrt(ddpi),data=savings)
> invddpi <- 1/savings$ddpi
> plot(savings$sr~invddpi,ylab="savings",xlab="1/ddpi")
> plot(sr~log(ddpi),data=savings)
```

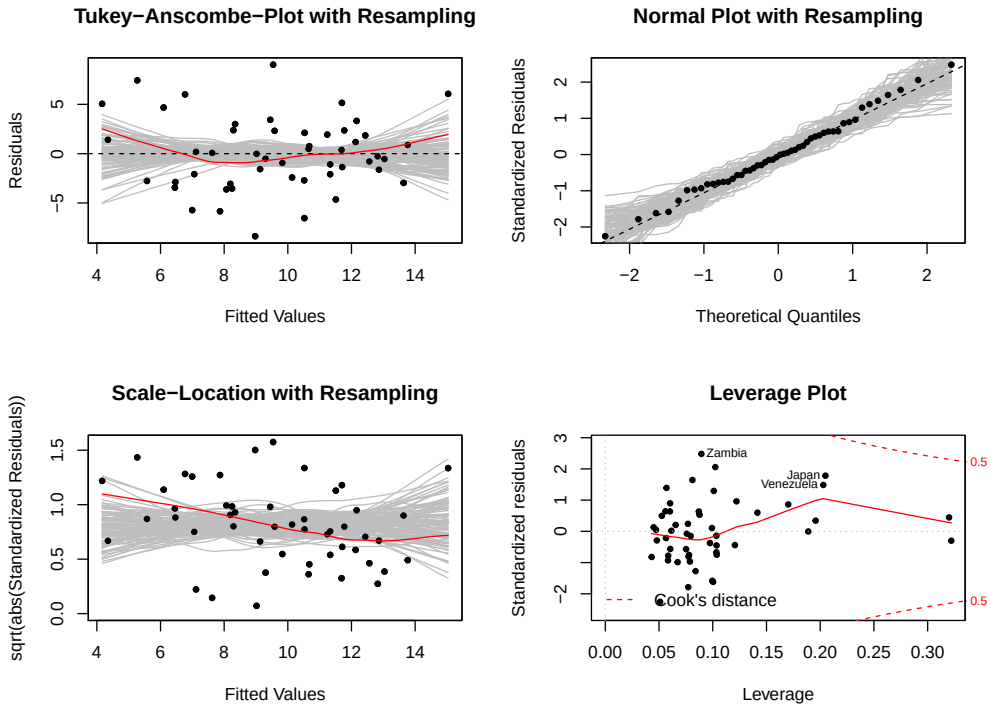


A square root or a log transform seem to imptove the situation slightly. But there is no big difference compared to untransformed ddpi. **(1 Point)** We shall experiment with two different models. In the first one we transform ddpi and dpi using the square root transform. In the second one we transform ddpi and dpi using the log transform.

```
> ## consider additional models : 1
> fit2 <- lm(sr ~ pop15 + pop75 + sqrt(dpi) + sqrt(ddpi), data=savings)
> resplot(fit2)
```




```
> ## consider additional models : 2
> fit3 <- lm(sr ~ pop15 + pop75 + log(dpi) + log(ddpi), data=savings)
> resplot(fit3)
```



The residual plots do not look better compared to the untransformed case. Therefore, we will use the orignal model without transformations. **(1 Point)**

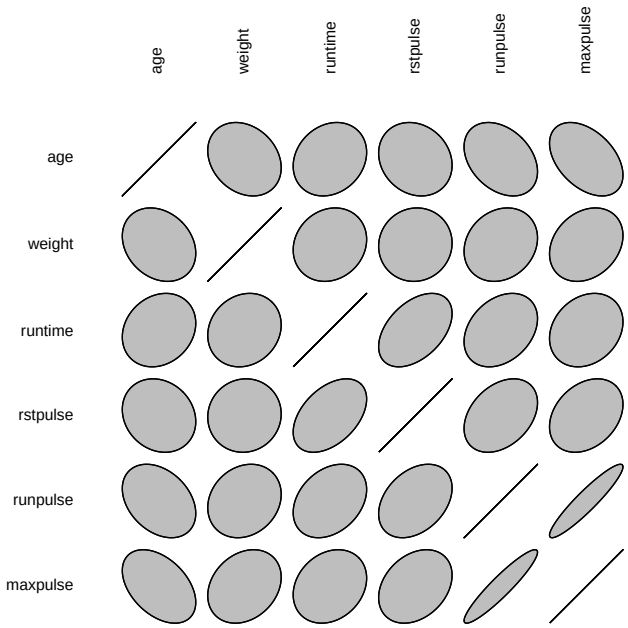
3. (8 points) The file `fitness.rda` (on Canvas) contains measurements of a fitness test for 31 patients. The target variable `oxy` is the rate of oxygen consumption which was measured with a complicated and expensive procedure.

Predictors are `age`, `weight`, `runtime` (running time), `rstpulse` (resting pulse), `runpulse` (running/active pulse) and `maxpulse` (maximal pulse).

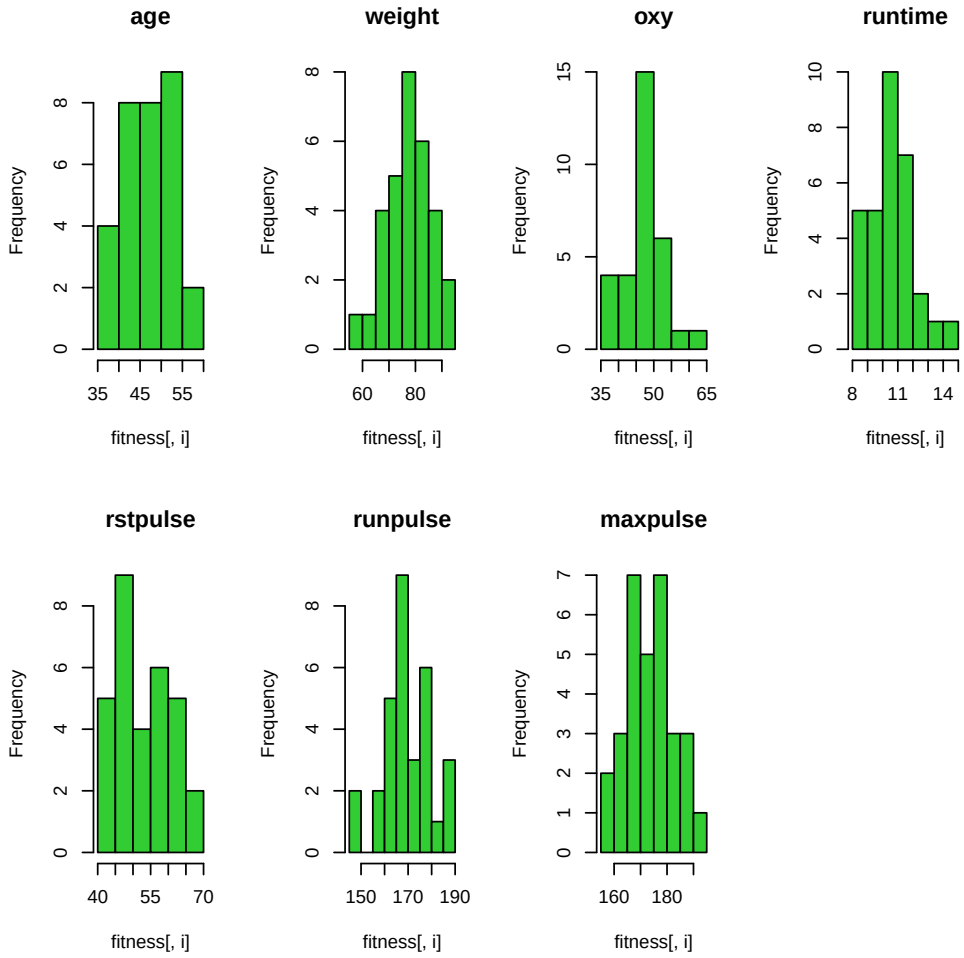
(a) (3 points) Analyze the data. What transformations are necessary (consider both response and predictors)? Are there any other problems? What can you say about the pairwise correlations between the predictors?

Solution:

```
> ## load data
> load("fitness.rda")
> ## fit model
> library(MASS)
> fit <- lm(oxy ~ ., data=fitness)
> boxcox(fit)
```

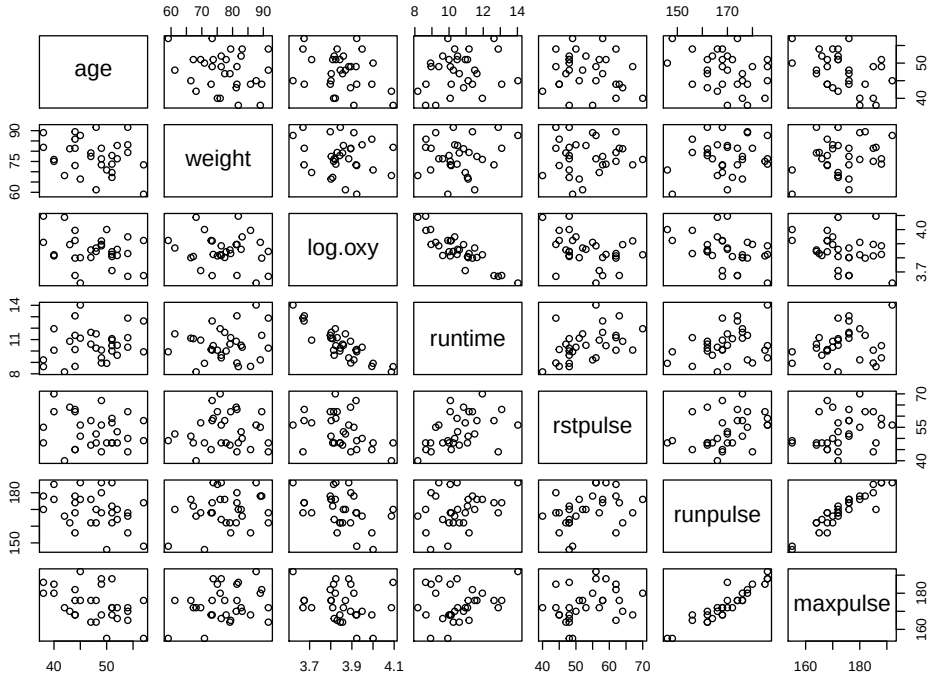



```
> ## analyze the variables
> par(mfrow=c(2,4))
> for (i in 1:7) hist(fitness[,i], col="limegreen", main=names(fitness)[i])
```

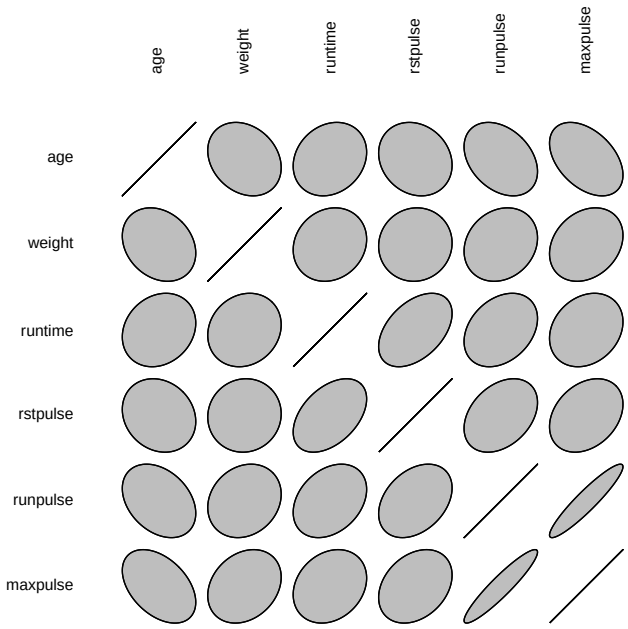


The Box-Cox procedure recommends log transforming the response. **(1 Point)** As you can see from the histograms, there are no variables that are strongly skewed to the right and/or have a relative scale with a large range of values. Therefore, we will not apply any transformations to the predictors. **(1 Point)**

```
> fitness.new <- data.frame(age=fitness$age, weight=fitness$weight, log.oxy=log(fitness$oxy),
+                           runtime=fitness$runtime, rstpulse=fitness$rstpulse,
+                           runpulse=fitness$runpulse,
+                           maxpulse=fitness$maxpulse)
> pairs(fitness.new)
```




```
> par(mfrow=c(1,1))
> library(ellipse)
> plotcorr(cor(fitness[,-3]), cex.lab = 0.75, mar = c(1,1,1,1))
```



As we can see from the above plots, there is a strong positive correlation between the running pulse and the maximal pulse. The remaining variables do not show strong pairwise correlations. **(1 Point)**

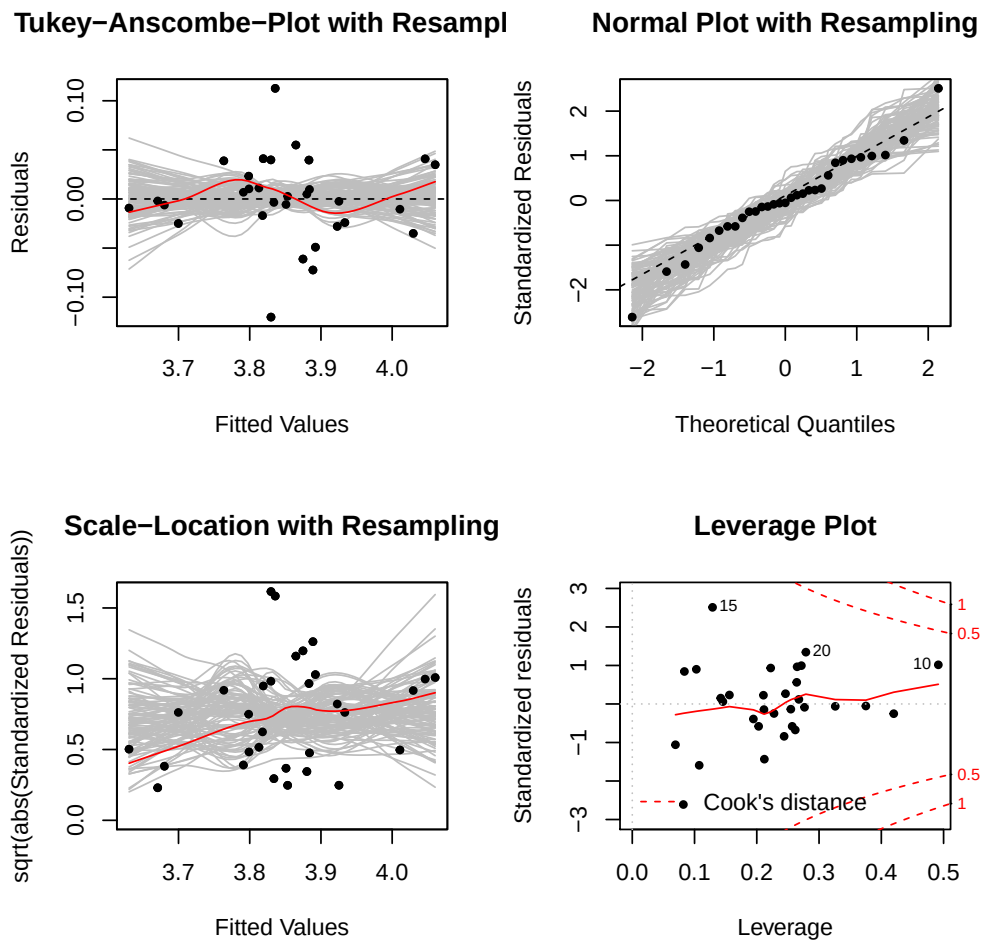
(b) (1 point) Fit a model containing all predictors after applying potentially necessary transformations. Perform a residual analysis and ensure that the model does not show a systematic error or any other model violation.

Solution:

```
> ## fit model
> fit <- lm(log.oxy ~ ., data=fitness.new)
> summary(fit)
```

```
Call:
lm(formula = log.oxy ~ ., data = fitness.new)

Residuals:
      Min        1Q    Median        3Q       Max
-0.120310 -0.020447 -0.002011  0.029030  0.112639

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  5.0465013  0.2574943  19.599 2.84e-16 ***
age         -0.0044171  0.0020727  -2.131   0.0435 *
weight      -0.0014717  0.0011334  -1.299   0.2064
runtime     -0.0584035  0.0079836  -7.315 1.48e-07 ***
rstpulse     0.0000586  0.0013713   0.043   0.9663
runpulse    -0.0065699  0.0024882  -2.640   0.0143 *
maxpulse     0.0049480  0.0028337   1.746   0.0936 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.0481 on 24 degrees of freedom
Multiple R-squared:  0.8516,        Adjusted R-squared:  0.8145
F-statistic: 22.95 on 6 and 24 DF,  p-value: 7.749e-09
```

```
> ## fit model
> source("resplot.R")
> resplot(fit)
```



There are no systematic errors. There are two large residuals, one of which is positive, the other one is negative. The assumption of constant variance is potentially violated. The normality assumption seems to be satisfied. While the residual plots do not look perfect we could consider the model assumptions to be satisfied to a sufficient degree. **(1 Point)**

(c) (1 point) Check whether there is high multicollinearity by computing the VIFs.
Hint: `library(faraway); vif(fit)`

Solution:

```
> ## multicollinearity
> library(faraway)
> vif(fit)

     age   weight  runtime rstpulse runpulse maxpulse
1.512836 1.155329 1.590868 1.415589 8.437274 8.743848
```

The VIFs of `runpulse` and `maxpulse` indicate the presence of multicollinearity. This is not surprising given the large pairwise correlation between running pulse and maximal pulse. **(1 Point)**

(d) (2 points) Address the multicollinearity problem by using different methods:

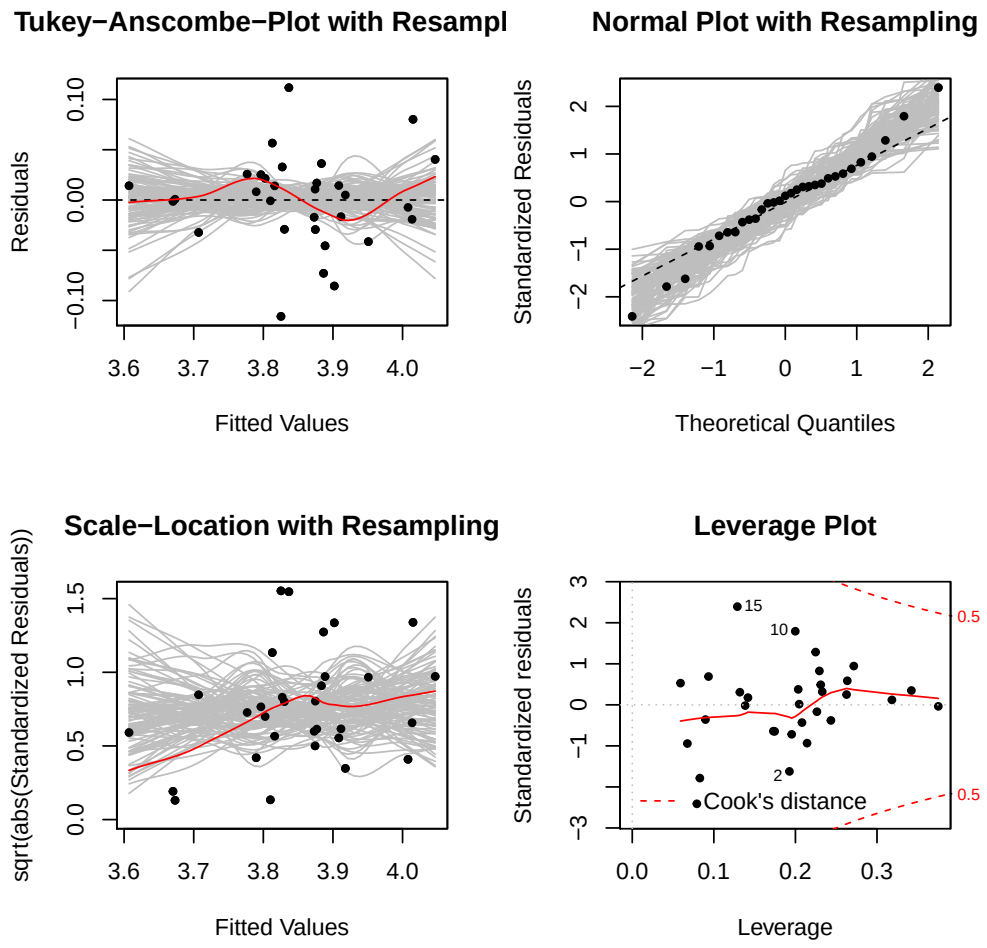
(i) Amputation, i.e. leave out redundant variables.

(ii) Create new variables that are not collinear.

Solution: (i) Amputation

```
> ## fitted values
> f.o <- fitted(fit)
```

```
> ## Amputation - leave out maxpulse
> fit <- lm(log.oxy ~ age + weight + runtime + rstpulse + runpulse, data=fitness.new)
> resplot(fit)
```



Since the high multicollinearity stems from the large pairwise correlation between running pulse and maximal pulse, one of these two variables should be excluded from the model. We recommend to leave out the maximal pulse due to background knowledge.

```
> vif(fit)
```

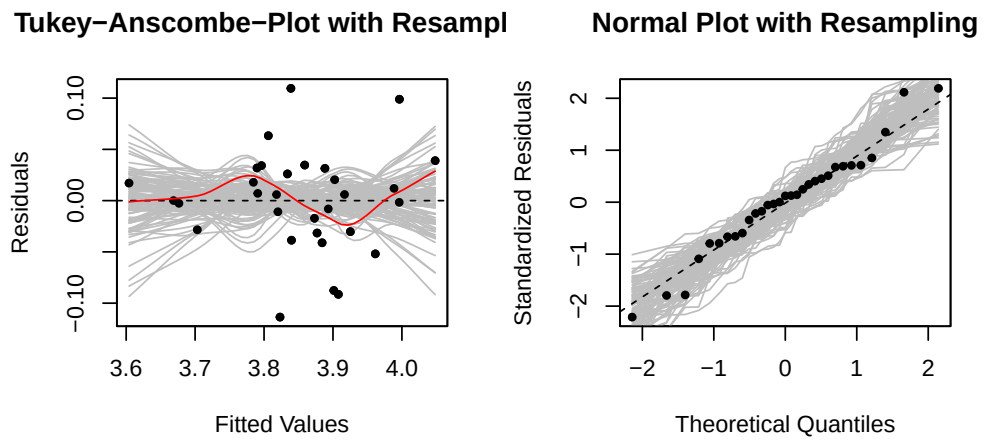
```
     age   weight  runtime rstpulse runpulse
1.408289 1.116150 1.578518 1.413545 1.388799
```

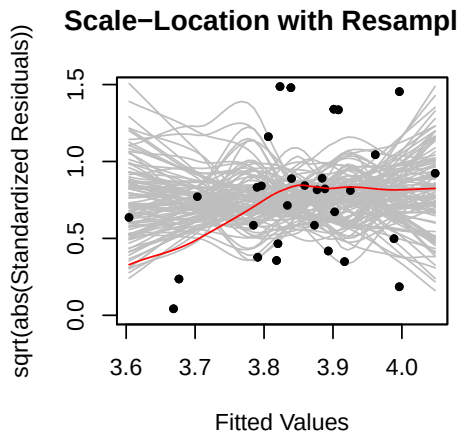
The resulting model does not show a systematic error and there is no high multicollinearity. **(1 Point)**

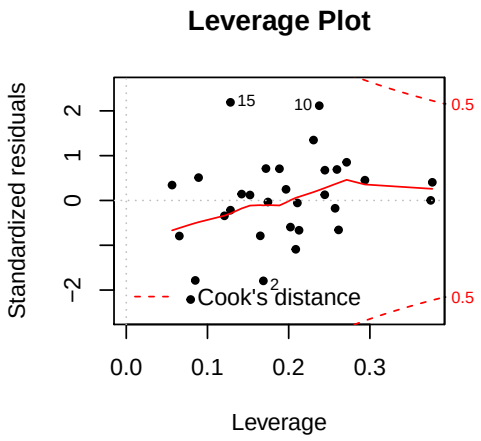
It is also ok to choose to leave out `runpulse`. In that case:

```
> ## Amputation - leave out maxpulse
> fit <- lm(log.oxy ~ age + weight + runtime + rstpulse + maxpulse, data=fitness.new)
> resplot(fit)
> vif(fit)
```

```
     age   weight  runtime rstpulse maxpulse
1.497522 1.130227 1.529564 1.402388 1.439261
```

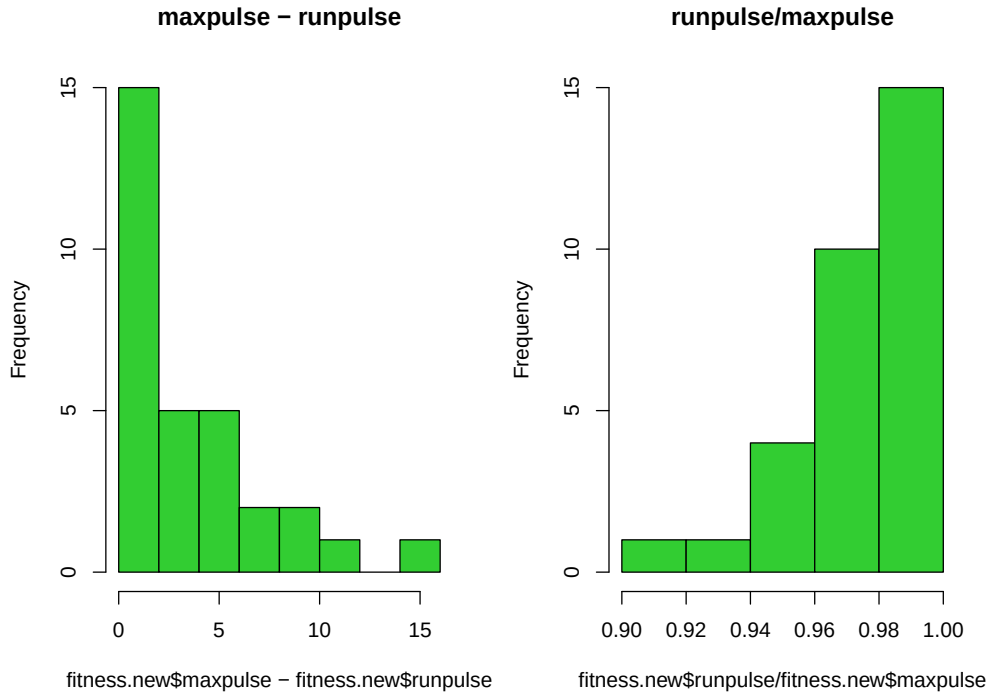






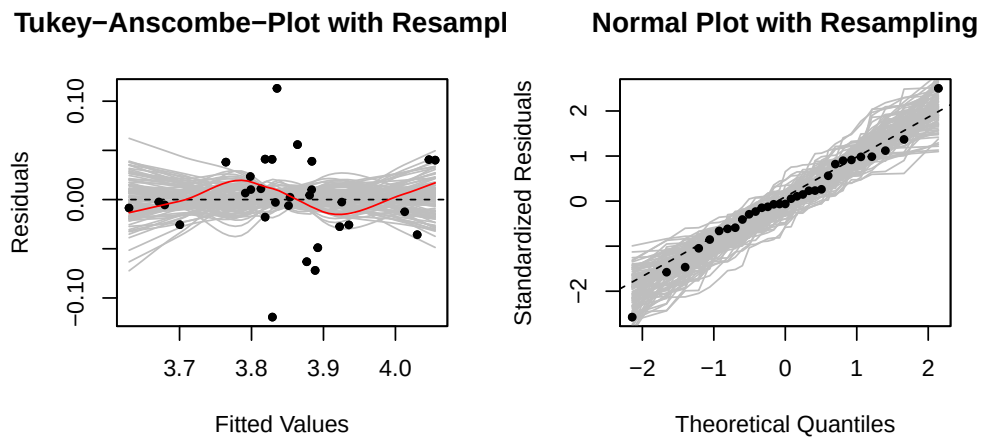
(ii) Creating new variables

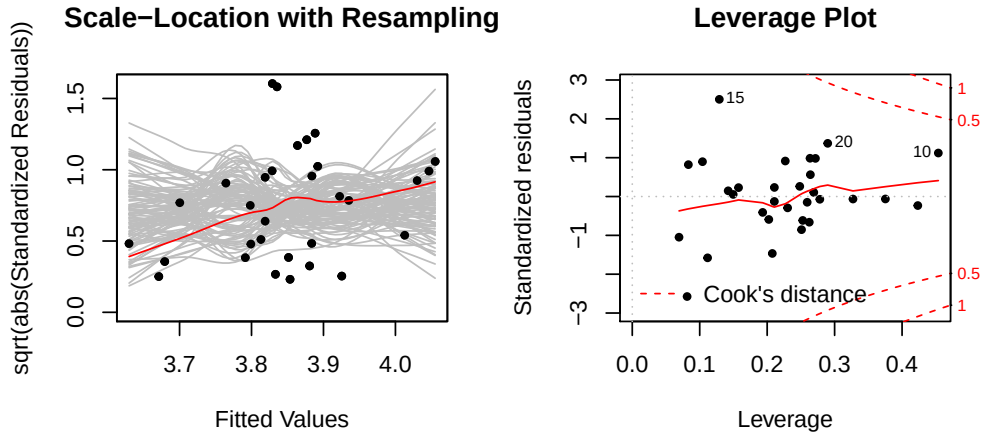
```
> ## transformation
> par(mfrow=c(1,2))
> hist(fitness.new$maxpulse-fitness.new$runpulse, col="limegreen", main = "maxpulse - runpulse")
> hist(fitness.new$runpulse/fitness.new$maxpulse, col="limegreen", main = "runpulse/maxpulse")
> my.fitness <- fitness.new[,-7]
> my.fitness$intensity <- fitness.new$runpulse/fitness.new$maxpulse
```



Either `runpulse` or `maxpulse` needs to be adjusted. We leave the running pulse in the model and substitute the maximal pulse by either `maxpulse-runpulse` or `runpulse/maxpulse`. Since the latter shows less skew in the histogram, we choose to use the quotient. (It is also ok to use `maxpulse-runpulse`.)

```
> fit <- lm(log.oxy ~ ., data=my.fitness)
> resplot(fit)
```






```
> vif(fit)

      age    weight   runtime  rstpulse  runpulse intensity
 1.500884  1.152036  1.594347  1.414005  1.961997  1.615894
```

(1 Point)

(e) (1 point) Use the model from subquestion **e)** (i) and perform a backward elimination using the BIC in order to reduce the set of predictors and just include those that are strictly necessary.
Which predictors are chosen by this procedure?

Solution:

```
> ## backward elimination using the p-values
> fit <- lm(log.oxy ~ age + weight + runtime + rstpulse + runpulse, data=fitness.new)
> step(fit,direction="backward", k=log(31))

Start:  AIC=-171.76
log.oxy ~ age + weight + runtime + rstpulse + runpulse

            Df Sum of Sq      RSS     AIC
- rstpulse   1  0.000001 0.062583 -175.19
- weight     1  0.002286 0.064868 -174.08
<none>                  0.062582 -171.76
- runpulse   1  0.015335 0.077916 -168.40
- age        1  0.016675 0.079256 -167.87
- runtime    1  0.130089 0.192670 -140.33
```

```
Step:  AIC=-175.19
log.oxy ~ age + weight + runtime + runpulse

           Df Sum of Sq      RSS     AIC
- weight    1  0.002304 0.064887 -177.51
<none>                  0.062583 -175.19
- runpulse  1  0.015744 0.078327 -171.67
- age       1  0.017486 0.080069 -170.99
- runtime   1  0.160899 0.223482 -139.17

Step:  AIC=-177.51
log.oxy ~ age + runtime + runpulse

           Df Sum of Sq      RSS     AIC
<none>                  0.064887 -177.51
- age       1  0.015506 0.080393 -174.30
- runpulse  1  0.016197 0.081083 -174.03
- runtime   1  0.172107 0.236994 -140.78

Call:
lm(formula = log.oxy ~ age + runtime + runpulse, data = fitness.new)

Coefficients:
(Intercept)          age      runtime     runpulse
   5.178370    -0.004910    -0.060900    -0.002638
```

This procedure chooses predictors `age` and `runtime` and `runpulse` as the final model.
If the amputed model from e) (i) included maxpulse instead of runpulse:

```
> ## backward elimination using the p-values
> fit <- lm(log.oxy ~ age + weight + runtime + rstpulse + maxpulse, data=fitness.new)
> step(fit,direction="backward", k=log(31))

Start:  AIC=-167.56
log.oxy ~ age + weight + runtime + rstpulse + maxpulse

           Df Sum of Sq      RSS     AIC
- rstpulse  1  0.000105 0.071763 -170.95
- weight    1  0.001956 0.073614 -170.16
- maxpulse  1  0.006258 0.077916 -168.40
<none>                  0.071658 -167.56
- age       1  0.013427 0.085085 -165.67
- runtime   1  0.147674 0.219332 -136.32

Step:  AIC=-170.95
log.oxy ~ age + weight + runtime + maxpulse

           Df Sum of Sq      RSS     AIC
- weight    1  0.001875 0.073638 -173.58
- maxpulse  1  0.006563 0.078327 -171.67
<none>                  0.071763 -170.95
- age       1  0.013559 0.085323 -169.02
- runtime   1  0.191868 0.263631 -134.05

Step:  AIC=-173.58
log.oxy ~ age + runtime + maxpulse

           Df Sum of Sq      RSS     AIC
- maxpulse  1  0.007445 0.081083 -174.03
```

```
<none>                  0.073638 -173.58
- age       1  0.012175 0.085814 -172.28
- runtime   1  0.202041 0.275679 -136.10

Step:  AIC=-174.03
log.oxy ~ age + runtime

          Df Sum of Sq     RSS     AIC
- age      1  0.006042 0.08713 -175.24
<none>                 0.08108 -174.03
- runtime  1  0.261536 0.34262 -132.79

Step:  AIC=-175.24
log.oxy ~ runtime

          Df Sum of Sq     RSS     AIC
<none>                 0.08713 -175.24
- runtime  1   0.28698 0.37411 -133.50

Call:
lm(formula = log.oxy ~ runtime, data = fitness.new)

Coefficients:
(Intercept)      runtime
     4.5983      -0.0705
```

This procedure only chooses predictor `runtime` in the final model.
(1 Point)

4. (6 points) In a study on infection risk controlling in US hospitals a sample from 113 hospitals (on Canvas) contains the following variables:

`id`	randomly assigned ID of the hospital
`length`	average duration of hospital stay (in days)
`age`	average age of patients (in years)
`inf`	average infection risk (in percent)
`cult`	number of bacteorological tests per asymptomatic patient x 100
`xray`	number of X-rays per asymptomatic patient x 100
`beds`	number of beds
`school`	university hospital 1=yes 2=no
`region`	geographical region 1=NE 2=N 3=S 4=W
`pat`	average number of patients a day
`nurs`	number of full-employed, trained nurses
`serv`	percentage of available services from a fixed list of 35 references

Using the variables `age`, `inf`, `region`, `beds`, `pat`, `nurs` as predictors and `length` as response variable, perform a linear regression analysis and find an optimal model by following the next instructions:

Solution:

```
> load("senic.rda")
> senic <- senic[,c("length", "age", "inf", "region", "beds", "pat", "nurs")]
```

(a) (1 point) Check the correlations between the continuous predictors. Which of them are problematic and why? Is there an intuitive explanation of this problem? Combine some of the predictors to improve the situation.
Hint: Use `pat.bed = pat/bed` and `pat.nurs = pat/nurs`

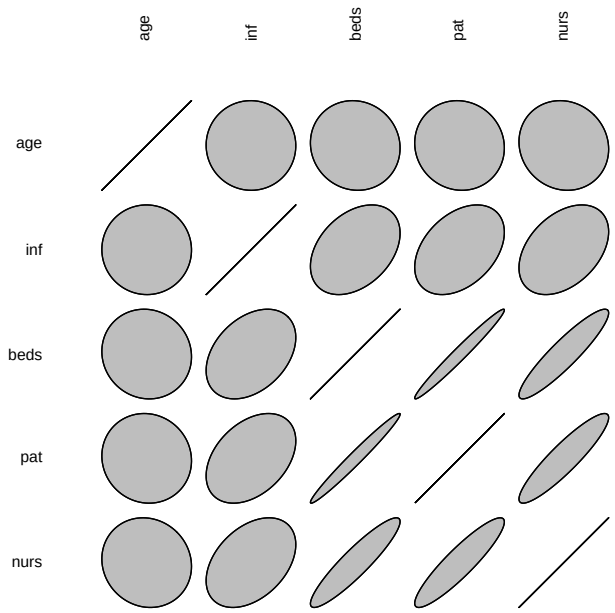
Solution: We check the correlations between the continuous predictors:

```
> indices_categorical_vars <- which(is.element(colnames(senic), c("length", "region")))
> cor(senic[, -indices_categorical_vars])
```

```
              age          inf        beds         pat        nurs
age   1.000000000 -0.006266807 -0.05882316 -0.05477467 -0.08294462
inf  -0.006266807  1.000000000  0.36917855  0.39070521  0.40291139
beds -0.058823160  0.369178549  1.00000000  0.98099774  0.91550415
pat  -0.054774667  0.390705214  0.98099774  1.00000000  0.90789698
nurs -0.082944616  0.402911390  0.91550415  0.90789698  1.00000000
```

Graphical illustration of the correlations:

```
>     library(ellipse)
>     plotcorr(cor(senic[, -indices_categorical_vars]), cex.lab = 0.75, mar = c(1,1,1,1))
```



We see that `beds`, `pat` and `nurs` are strongly correlated. We expected this because they all can be seen as measures of the size of a hospital. We will leave the variable `pat` unmodified because it is definitely a key factor to take into account when `length` is the response variable. We change the others to solve the high-correlation problem without having to take them out of the model. For this, we will substitute `beds` by `pat/beds` and `nurs` by `pat/nurs`.
Before combining the variables, we check if `beds` and `nurs` contain zeroes:

```
> any(senic$beds == 0)
```

```
[1] FALSE
```

```
> any(senic$nurs == 0)
```

```
[1] FALSE
```

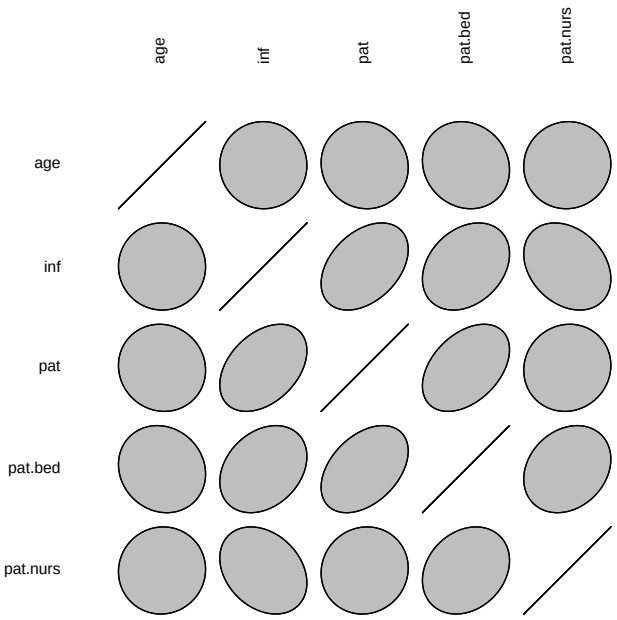
Now we combine the variables and check the correlations again.

```
> senic.02 <- data.frame(length=senic$length, age=senic$age, inf=senic$inf,
+ region=senic$region, pat=senic$pat, pat.bed=senic$pat/senic$beds,
+ pat.nurs=senic$pat/senic$nurs)
> cor(senic.02[,-indices_categorical_vars])
```

```
                 age          inf         pat    pat.bed    pat.nurs
age       1.000000000 -0.006266807 -0.05477467 -0.1096058  0.02695459
inf      -0.006266807  1.000000000  0.39070521  0.2897338 -0.28598480
pat      -0.054774667  0.390705214  1.00000000  0.4151079  0.05659985
pat.bed  -0.109605797  0.289733778  0.41510791  1.0000000  0.22893307
pat.nurs  0.026954588 -0.285984796  0.05659985  0.2289331  1.00000000
```

Graphical illustration of the correlations after modifying some variables:

```
>     plotcorr(cor(senic.02[,-indices_categorical_vars]), cex.lab = 0.75, mar = c(1,1,1,1))
```

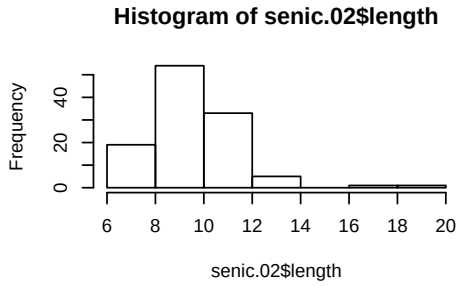


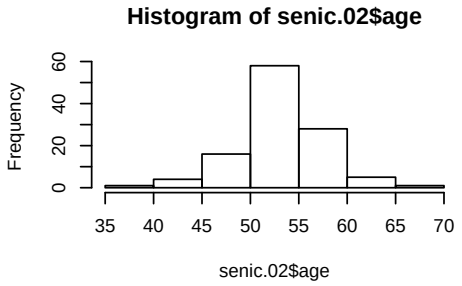
The correlations were strongly reduced and we still have some information about the variables `beds` and `nurs`. **(1 Point)**

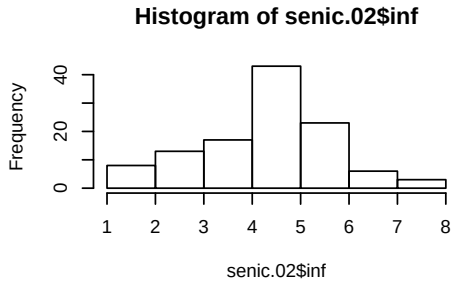
(b) (3 points) Perform the necessary transformations on the predictors and response.

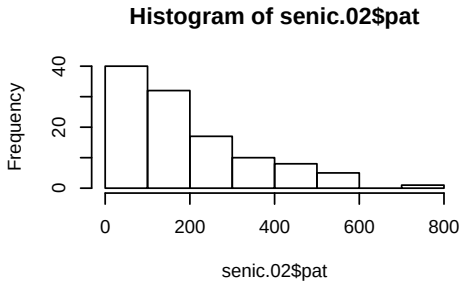
Solution: First, we take a look at the histogram of the predictors before doing transformations:

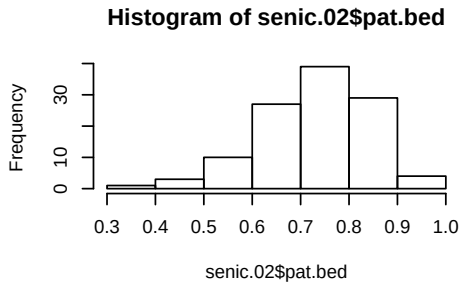
```
>     par(mfrow=c(3,2))
>     hist(senic.02$length)
>     hist(senic.02$age)
>     hist(senic.02$inf)
>     hist(senic.02$pat)
>     hist(senic.02$pat.bed)
>     hist(senic.02$pat.nurs)
```

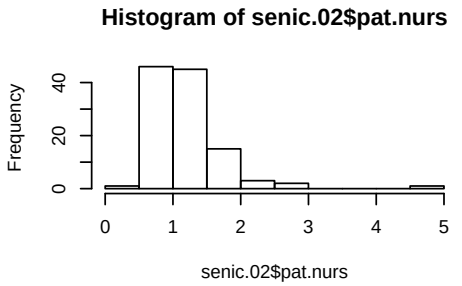




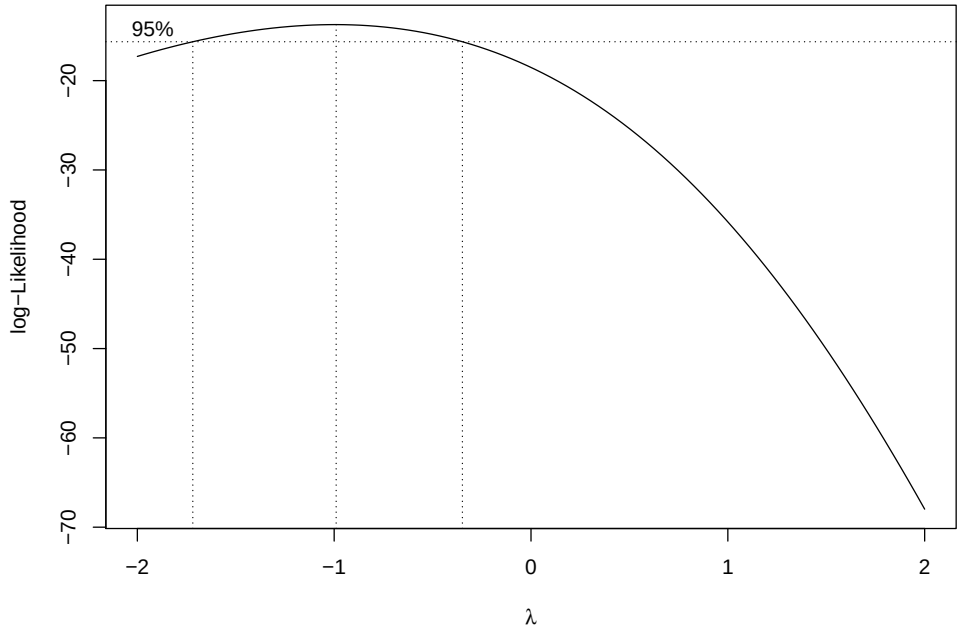









```
> fit.02 <- lm(length ~ age + inf + region + pat + pat.bed + pat.nurs, data=senic.02)
>  boxcox(fit.02)
```



The variables `length`, `pat` and `pat.nurs` may need to be transformed. According to the Box-Cox procedure, we should consider the inverse of `length`. **(1 Point)**

We check for zeroes in `pat` and `length`:

```
> any(senic.02$length == 0)
```

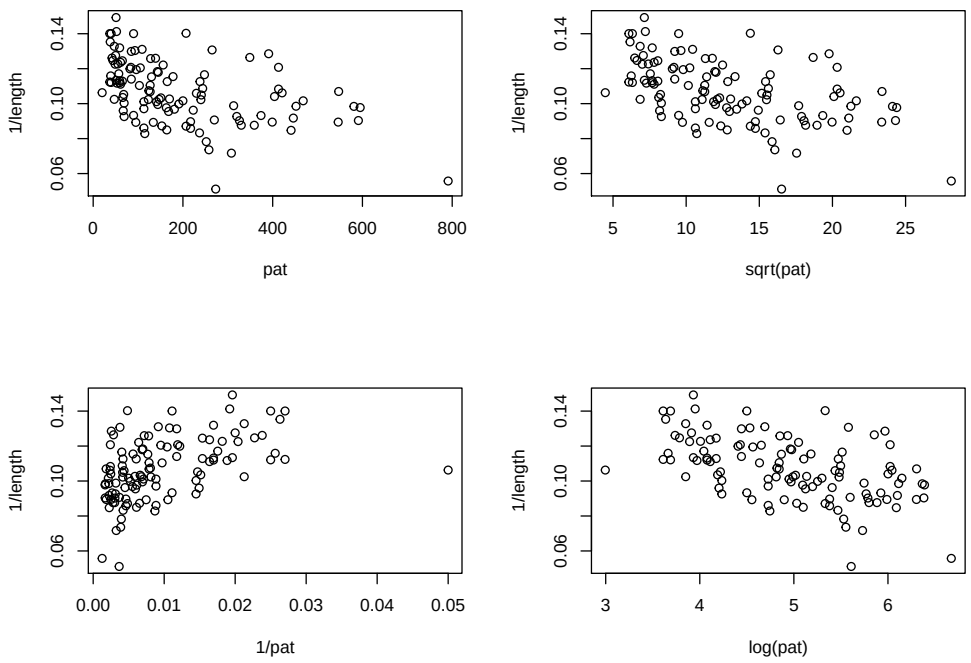
```
[1] FALSE
```

```
> any(senic.02$pat == 0)
```

```
[1] FALSE
```

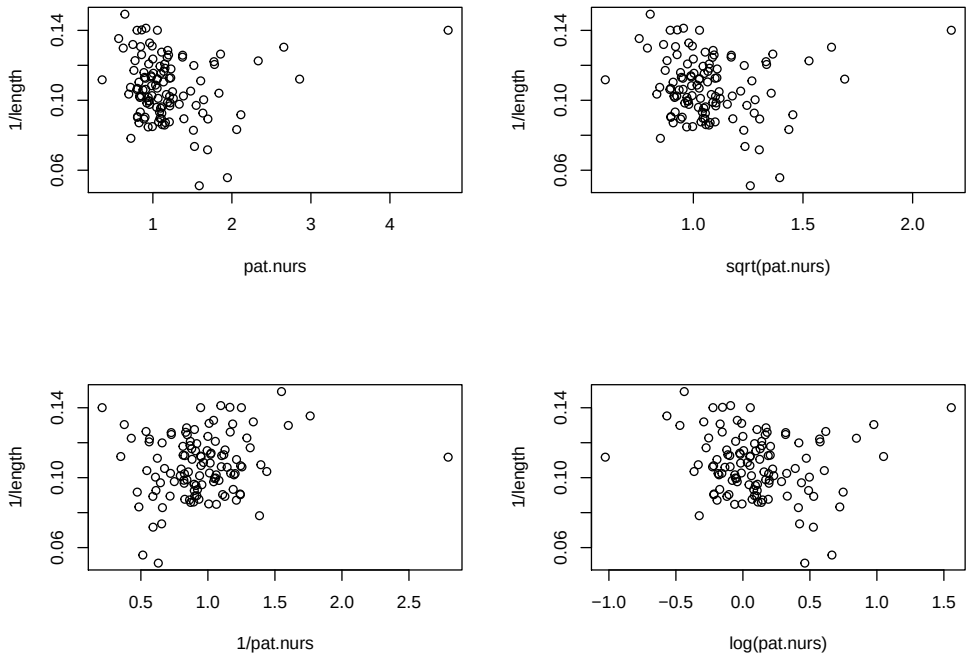
Now we want to see which transformations seem appropriate for `pat` and `pat.nurs`.

```
> senic.03 <- senic.02
> senic.03$length <- 1/(senic.02$length)
> par(mfrow=c(2,2))
> plot(length~pat,data=senic.03,ylab="1/length")
>   plot(length~sqrt(pat),data=senic.03,ylab="1/length")
>   patinv <- 1/senic.03$pat
>   plot(senic.03$length~patinv,xlab="1/pat",ylab="1/length")
>   plot(length~log(pat),ylab="1/length",data=senic.03)
```



The log transformation seems to work best for `pat`. **(1 Point)**

```
> senic.03$pat <- log(senic.03$pat)
> par(mfrow=c(2,2))
> plot(length~pat.nurs,data=senic.03,ylab="1/length")
>   plot(length~sqrt(pat.nurs),data=senic.03,ylab="1/length")
>   patnursinv <- 1/senic.03$pat.nurs
>   plot(senic.03$length~patnursinv,xlab="1/pat.nurs",ylab="1/length")
>   plot(length~log(pat.nurs),ylab="1/length",data=senic.03)
```



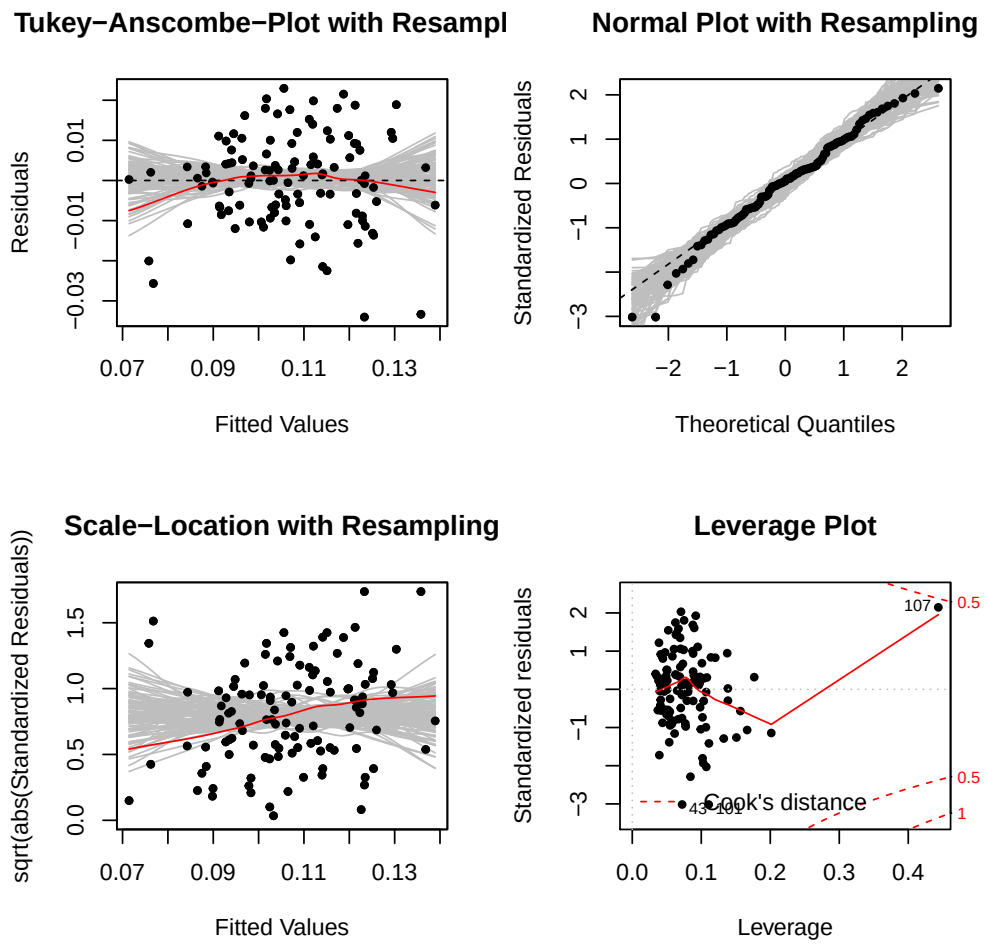
There is no clear best option for transforming the variable `pat.nurs`, so we will leave it untransformed. **(1 Point)**

(c) (1 point) Fit a linear regression using the transformed variables and perform a residual analysis.

Solution:

We fit a linear regression:

```
>     fit.03 <- lm(length ~ age + inf + region + pat + pat.bed + pat.nurs, data=senic.03)
>     summary(fit.03)

Call:
lm(formula = length ~ age + inf + region + pat + pat.bed + pat.nurs,
    data = senic.03)

Residuals:
      Min        1Q    Median        3Q       Max
-0.034047 -0.006701  0.000585  0.007527  0.022954

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.1962072  0.0179234  10.947  < 2e-16 ***
age         -0.0007350  0.0002584  -2.844 0.005358 **
inf         -0.0050216  0.0010504  -4.781 5.77e-06 ***
regionN      0.0060531  0.0031545   1.919 0.057743 .
regionS      0.0106961  0.0030751   3.478 0.000738 ***
regionW      0.0223063  0.0040193   5.550 2.20e-07 ***
pat         -0.0051220  0.0017975  -2.849 0.005281 **
pat.bed     -0.0129301  0.0125064  -1.034 0.303591
pat.nurs    -0.0015660  0.0023915  -0.655 0.514033
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.01172 on 104 degrees of freedom
Multiple R-squared:  0.6006,        Adjusted R-squared:  0.5699
F-statistic: 19.55 on 8 and 104 DF,  p-value: < 2.2e-16

>     par(mfrow=c(2,2))
>     resplot(fit.03)
```



From the summary we see that `pat.bed` is not statistically significant and a variable selection is necessary (see next question).

From the model diagnostics plots we note that there are three outliers, i.e. observations 47, 101, and 112. However, since their Cook's distance is below 0.5, they don't significantly influence our fit and we proceed with our analysis. The assumptions of linearity and constant variance seem to be satisfied. The QQ-plot does not look perfect but we can also accept the normality assumption. **(1 Point)**

(d) (1 point) Perform a forward selection using the AIC criterion. Thus, start with the empty model and use the `step`. Which predictors are in the final model?

Solution: Forward selection:

```
> fit.empty <- lm(length ~ 1, data=senic.03)
>     scp   <- list(lower=~1, upper=~age + inf + region + pat + pat.bed + pat.nurs)

>     fit.F <- step(fit.empty, scope=scp, direction="forward")

Start:  AIC=-908.53
length ~ 1

            Df Sum of Sq      RSS     AIC
+ inf        1 0.0107331 0.025045 -946.84
+ pat        1 0.0098106 0.025967 -942.75
+ region     3 0.0103495 0.025428 -941.12
+ pat.bed    1 0.0070595 0.028718 -931.37
+ age        1 0.0008214 0.034957 -909.16
<none>                   0.035778 -908.53
+ pat.nurs   1 0.0000124 0.035766 -906.57
```

```
Step:  AIC=-946.84
length ~ inf

           Df Sum of Sq      RSS     AIC
+ region    3 0.0078812 0.017164 -983.54
+ pat       1 0.0032392 0.021806 -960.49
+ pat.bed   1 0.0031837 0.021861 -960.20
+ pat.nurs  1 0.0011966 0.023848 -950.37
+ age       1 0.0008590 0.024186 -948.78
<none>                  0.025045 -946.84

Step:  AIC=-983.54
length ~ inf + region

           Df  Sum of Sq      RSS     AIC
+ pat       1 0.00153832 0.015625 -992.15
+ age       1 0.00071995 0.016444 -986.38
+ pat.bed   1 0.00070603 0.016458 -986.28
+ pat.nurs  1 0.00032862 0.016835 -983.72
<none>                  0.017164 -983.54

Step:  AIC=-992.15
length ~ inf + region + pat

           Df  Sum of Sq      RSS     AIC
+ age       1 0.00107364 0.014552 -998.19
<none>                  0.015625 -992.15
+ pat.nurs  1 0.00014851 0.015477 -991.23
+ pat.bed   1 0.00012337 0.015502 -991.04

Step:  AIC=-998.19
length ~ inf + region + pat + age

           Df  Sum of Sq      RSS     AIC
<none>                  0.014552 -998.19
+ pat.bed   1 0.00020225 0.014349 -997.77
+ pat.nurs  1 0.00011430 0.014437 -997.08
```

```
>     summary(fit.F)

Call:
lm(formula = length ~ inf + region + pat + age, data = senic.03)

Residuals:
      Min        1Q    Median        3Q       Max
-0.035075 -0.007422  0.000321  0.007428  0.023581

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.1877873  0.0166557  11.275  < 2e-16 ***
inf         -0.0049082  0.0009888  -4.964 2.66e-06 ***
regionN      0.0065222  0.0031190   2.091 0.038908 *
regionS      0.0106331  0.0030490   3.487 0.000711 ***
regionW      0.0240796  0.0037932   6.348 5.52e-09 ***
pat         -0.0060785  0.0016373  -3.712 0.000329 ***
age         -0.0007175  0.0002565  -2.797 0.006134 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
Residual standard error: 0.01172 on 106 degrees of freedom
Multiple R-squared:  0.5933,        Adjusted R-squared:  0.5703
F-statistic: 25.77 on 6 and 106 DF,  p-value: < 2.2e-16
```

The function chooses to exclude predictors pat.bed and pat.nurs from the final model. **(1 Point)**