Laboratorio con `R` - 2

Metodi e Modelli per l'Inferenza Statistica - Ing. Matematica - a.a. 2023-24

Topics:

- Analisi dei punti influenti

- Collinearità e non linearità

- Trasformazione di variabili

0. Required packages

```
library( car )
library( ellipse )
library( leaps )
library(MASS)
library( GGally)
library(BAS)
library(faraway)
library(rgl)
library(corrplot)
```

1. Linear regression (refresh).

1.a Let's start from the linear model we built in the Laboratory 1. Upload `faraway` library and the dataset `savings`, an economic dataset on 50 different countries. These data are averages over 1960-1970 (to remove business cycle or other short-term fluctuations).

```
# import data
savings = read.table(file='savings.txt', header=T)

# Dimensioni
dim(savings)
## [1] 50  5
```

We have 50 observations (50 countries) with 5 attributes each.

Look at the main statistics for each covariate:

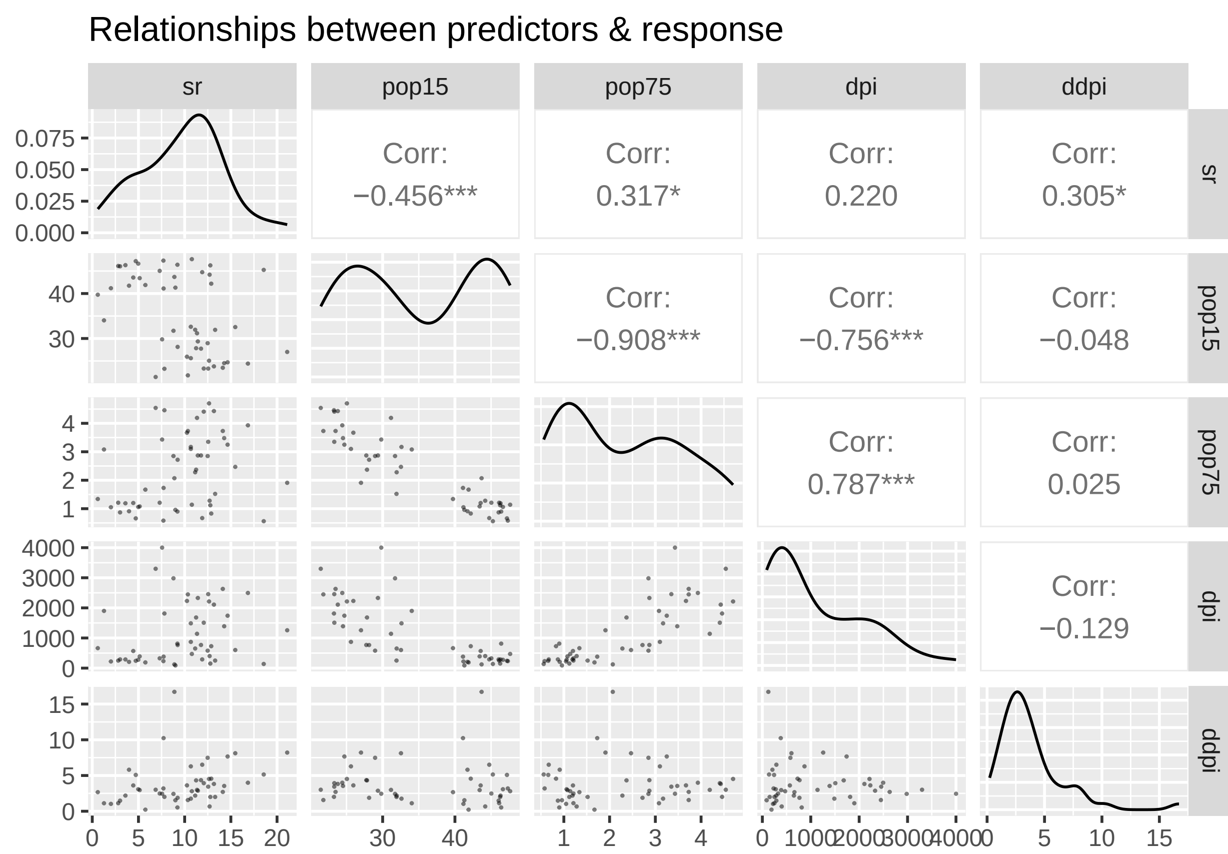
```
# a brief description of each columns
summary(savings)
##        sr             pop15           pop75            dpi
##  Min.   : 0.600   Min.   :21.44   Min.   :0.560   Min.   :  88.94
##  1st Qu.: 6.970   1st Qu.:26.21   1st Qu.:1.125   1st Qu.: 288.21
##  Median :10.510   Median :32.58   Median :2.175   Median : 695.66
##  Mean   : 9.671   Mean   :35.09   Mean   :2.293   Mean   :1106.76
##  3rd Qu.:12.617   3rd Qu.:44.06   3rd Qu.:3.325   3rd Qu.:1795.62
##  Max.   :21.100   Max.   :47.64   Max.   :4.700   Max.   :4001.89
##       ddpi
```

```
##   Min.   : 0.220
##   1st Qu.: 2.002
##   Median : 3.000
##   Mean   : 3.758
##   3rd Qu.: 4.478
##   Max.   :16.710

str(savings)
## 'data.frame':    50 obs. of  5 variables:
##  $ sr   : num  11.43 12.07 13.17 5.75 12.88 ...
##  $ pop15: num  29.4 23.3 23.8 41.9 42.2 ...
##  $ pop75: num  2.87 4.41 4.43 1.67 0.83 2.85 1.34 0.67 1.06 1.14 ...
##  $ dpi  : num  2330 1508 2108 189 728 ...
##  $ ddpi : num  2.87 3.93 3.82 0.22 4.56 2.43 2.67 6.51 3.08 2.8 ...
```

1.b Visualize the data and fit the complete linear model, in which `sr` is the outcome of interest.

solution For visualizing the data, we can plot the pairs or ggpairs. It is useful also for making an idea about the relationship between the covariates.

```
ggpairs(data = savings, title ="Relationships between predictors & response",
        lower = list(continuous=wrap("points", alpha = 0.5, size=0.1)))
```



Secondly, we fit the complete linear model (Lab 1) and look at the summary of the estimated coefficients.

```
g = lm( sr ~ pop15 + pop75 + dpi + ddpi, data = savings )
#g = lm( sr ~ ., savings )
```

```r
summary( g )
##
## 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

gs = summary( g )
```

2. Diagnostics: detecting influential points

The goal of diagnostics consists in detecting possible influential points in a sample. In general, an influential point is one whose removal from the dataset would cause a large change in the fit. Influential points are outliers and leverages (in italiano, punti leva). The definitions of outliers and leverages can overlap. A possible definition of outlier is 'a point that does not fit the chosen model'. On the other hand, a leverage is 'a point that significantly affects the estimates of the model'. It is immediate to see that often an outlier is also a leverage point.

By 'influential', we mean that :

1. estimated coefficients with or without an observation significantly changes: $\hat{\beta} - \hat{\beta}_{-i}$
2. fitted values with or without an observation significantly changes: $x^T(\hat{\beta} - \hat{\beta}_{-i}) = \hat{y} - \hat{y}_{-i}$ Anyway, these are hard measures to judge in the sense that the scale varies between datasets.

There are several approaches for identifying influential points in a sample, such as:

- **a. Leverages (projection matrix)**
- **b. Standardized Residuals**
- **c. Studentized Residuals**
- **d. Cook's Distance**

a. Leverages

Investigate possible leverages among data. Leverages are defined as the diagonal elements of H matrix:

$$H = X(X^T X)^{-1} X^T$$

such that $\hat{y} = Hy$.

solution We can compute diagonal elements of H matrix with two different functions:

```r
X = model.matrix( g )
X
##                 (Intercept) pop15 pop75     dpi  ddpi
## Australia                 1 29.35  2.87 2329.68  2.87
## Austria                   1 23.32  4.41 1507.99  3.93
## Belgium                   1 23.80  4.43 2108.47  3.82
## Bolivia                   1 41.89  1.67  189.13  0.22
## Brazil                    1 42.19  0.83  728.47  4.56
## Canada                    1 31.72  2.85 2982.88  2.43
## Chile                     1 39.74  1.34  662.86  2.67
## China                     1 44.75  0.67  289.52  6.51
## Colombia                  1 46.64  1.06  276.65  3.08
## Costa Rica                1 47.64  1.14  471.24  2.80
## Denmark                   1 24.42  3.93 2496.53  3.99
## Ecuador                   1 46.31  1.19  287.77  2.19
## Finland                   1 27.84  2.37 1681.25  4.32
## France                    1 25.06  4.70 2213.82  4.52
## Germany                   1 23.31  3.35 2457.12  3.44
## Greece                    1 25.62  3.10  870.85  6.28
## Guatamala                 1 46.05  0.87  289.71  1.48
## Honduras                  1 47.32  0.58  232.44  3.19
## Iceland                   1 34.03  3.08 1900.10  1.12
## India                     1 41.31  0.96   88.94  1.54
## Ireland                   1 31.16  4.19 1139.95  2.99
## Italy                     1 24.52  3.48 1390.00  3.54
## Japan                     1 27.01  1.91 1257.28  8.21
## Korea                     1 41.74  0.91  207.68  5.81
## Luxembourg                1 21.80  3.73 2449.39  1.57
## Malta                     1 32.54  2.47  601.05  8.12
## Norway                    1 25.95  3.67 2231.03  3.62
## Netherlands               1 24.71  3.25 1740.70  7.66
## New Zealand               1 32.61  3.17 1487.52  1.76
## Nicaragua                 1 45.04  1.21  325.54  2.48
## Panama                    1 43.56  1.20  568.56  3.61
## Paraguay                  1 41.18  1.05  220.56  1.03
## Peru                      1 44.19  1.28  400.06  0.67
## Philippines               1 46.26  1.12  152.01  2.00
## Portugal                  1 28.96  2.85  579.51  7.48
## South Africa              1 31.94  2.28  651.11  2.19
## South Rhodesia            1 31.92  1.52  250.96  2.00
## Spain                     1 27.74  2.87  768.79  4.35
## Sweden                    1 21.44  4.54 3299.49  3.01
## Switzerland               1 23.49  3.73 2630.96  2.70
## Turkey                    1 43.42  1.08  389.66  2.96
## Tunisia                   1 46.12  1.21  249.87  1.13
## United Kingdom            1 23.27  4.46 1813.93  2.01
## United States             1 29.81  3.43 4001.89  2.45
## Venezuela                 1 46.40  0.90  813.39  0.53
## Zambia                    1 45.25  0.56  138.33  5.14
## Jamaica                   1 41.12  1.73  380.47 10.23
## Uruguay                   1 28.13  2.72  766.54  1.88
## Libya                     1 43.69  2.07  123.58 16.71
## Malaysia                  1 47.20  0.66  242.69  5.08
```

```
## attr(,"assign")
## [1] 0 1 2 3 4
```

```
lev = hat( X )
lev
##  [1] 0.06771343 0.12038393 0.08748248 0.08947114 0.06955944 0.15840239
##  [7] 0.03729796 0.07795899 0.05730171 0.07546780 0.06271782 0.06372651
## [13] 0.09204246 0.13620478 0.08735739 0.09662073 0.06049212 0.06008079
## [19] 0.07049590 0.07145213 0.21223634 0.06651170 0.22330989 0.06079915
## [25] 0.08634787 0.07940290 0.04793213 0.09061400 0.05421789 0.05035056
## [31] 0.03897459 0.06937188 0.06504891 0.06425415 0.09714946 0.06510405
## [37] 0.16080923 0.07732854 0.12398898 0.07359423 0.03964224 0.07456729
## [43] 0.11651375 0.33368800 0.08628365 0.06433163 0.14076016 0.09794717
## [49] 0.53145676 0.06523300
# oppure
lev = hatvalues( g )
lev
##      Australia        Austria        Belgium        Bolivia         Brazil
##     0.06771343     0.12038393     0.08748248     0.08947114     0.06955944
##         Canada          Chile          China       Colombia     Costa Rica
##     0.15840239     0.03729796     0.07795899     0.05730171     0.07546780
##        Denmark        Ecuador        Finland         France        Germany
##     0.06271782     0.06372651     0.09204246     0.13620478     0.08735739
##         Greece      Guatamala       Honduras        Iceland          India
##     0.09662073     0.06049212     0.06008079     0.07049590     0.07145213
##        Ireland          Italy          Japan          Korea     Luxembourg
##     0.21223634     0.06651170     0.22330989     0.06079915     0.08634787
##          Malta         Norway    Netherlands    New Zealand      Nicaragua
##     0.07940290     0.04793213     0.09061400     0.05421789     0.05035056
##         Panama       Paraguay           Peru    Philippines       Portugal
##     0.03897459     0.06937188     0.06504891     0.06425415     0.09714946
##   South Africa South Rhodesia          Spain         Sweden    Switzerland
##     0.06510405     0.16080923     0.07732854     0.12398898     0.07359423
##         Turkey        Tunisia United Kingdom  United States      Venezuela
##     0.03964224     0.07456729     0.11651375     0.33368800     0.08628365
##         Zambia        Jamaica        Uruguay          Libya       Malaysia
##     0.06433163     0.14076016     0.09794717     0.53145676     0.06523300
```

Alternatively, we can compute H manually and then extract its diagonal elements:

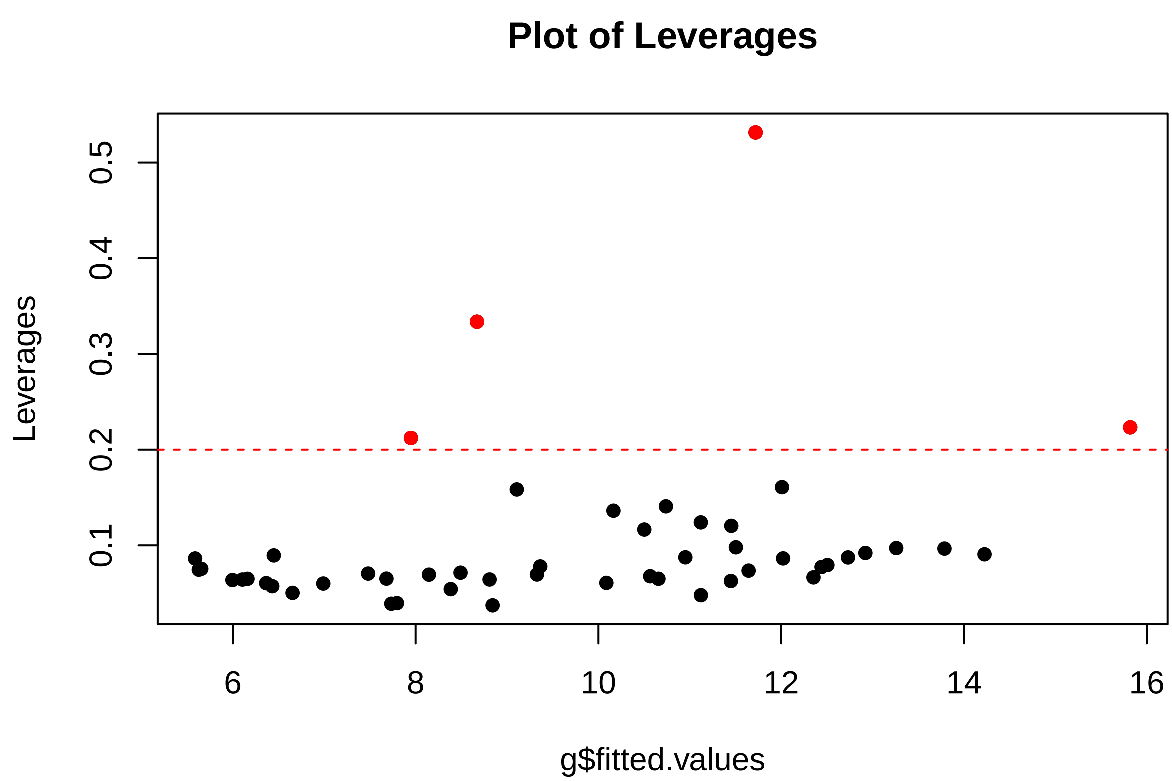
```
#manually
H = X %*% solve( t( X ) %*% X ) %*% t( X )
lev = diag( H )

sum(lev) # verifica: sum_i hat( x )_i = p = r + 1
## [1] 5
```

REMARK The trace of the H matrix (sum of the diagonal elements of a matrix) is equal to the rank of X matrix, which is p (number of covariates $r + 1$ for the intercept), assuming that covariates are all linearly independent and $p < n$. This is the size of the vectorial subspace generated by the linear combinations of the columns of X. The geometric interpretation of the Ordinary Least Square linear regression (OLS) states that H acts on $\mathbf{y}$ (vector of outcomes) by projecting it on the former subspace. The final output is $\hat{\mathbf{y}}$.

Rule of thumb: Given a point $\hat{h}_{ii}$ diagonal element of H, the i-th observation is a leverage if:

$$\hat{h}_{ii} > 2 \cdot \frac{p}{n}$$

```
p = g$rank # p = 5
n = dim(savings)[1] # n = 50


plot( g$fitted.values, lev, ylab = "Leverages", main = "Plot of Leverages",
      pch = 16, col = 'black' )

abline( h = 2 * p/n, lty = 2, col = 'red' )

watchout_points_lev = lev[ which( lev > 2 * p/n  ) ]
watchout_ids_lev = seq_along( lev )[ which( lev > 2 * p/n ) ]

points( g$fitted.values[ watchout_ids_lev ], watchout_points_lev, col = 'red', pch = 16 )
```




```
lev [ lev >  2 * 5 / 50 ]
##       Ireland         Japan United States         Libya
##     0.2122363     0.2233099     0.3336880     0.5314568
sum( lev [ lev >  2 * 5 / 50 ] )
## [1] 1.300691
```

Fit the model without leverages.

solution

```
gl = lm( sr ~ pop15 + pop75 + dpi + ddpi, savings, subset = ( lev < 0.2 ) )
summary( gl )
##
## Call:
## lm(formula = sr ~ pop15 + pop75 + dpi + ddpi, data = savings,
##     subset = (lev < 0.2))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9632 -2.6323  0.1466  2.2529  9.6687
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.221e+01  9.319e+00   2.384   0.0218 *
## pop15       -3.403e-01  1.798e-01  -1.893   0.0655 .
## pop75       -1.124e+00  1.398e+00  -0.804   0.4258
## dpi         -4.499e-05  1.160e-03  -0.039   0.9692
## ddpi         5.273e-01  2.775e-01   1.900   0.0644 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.805 on 41 degrees of freedom
## Multiple R-squared:  0.2959, Adjusted R-squared:  0.2272
## F-statistic: 4.308 on 4 and 41 DF,  p-value: 0.005315
#summary( g )
```

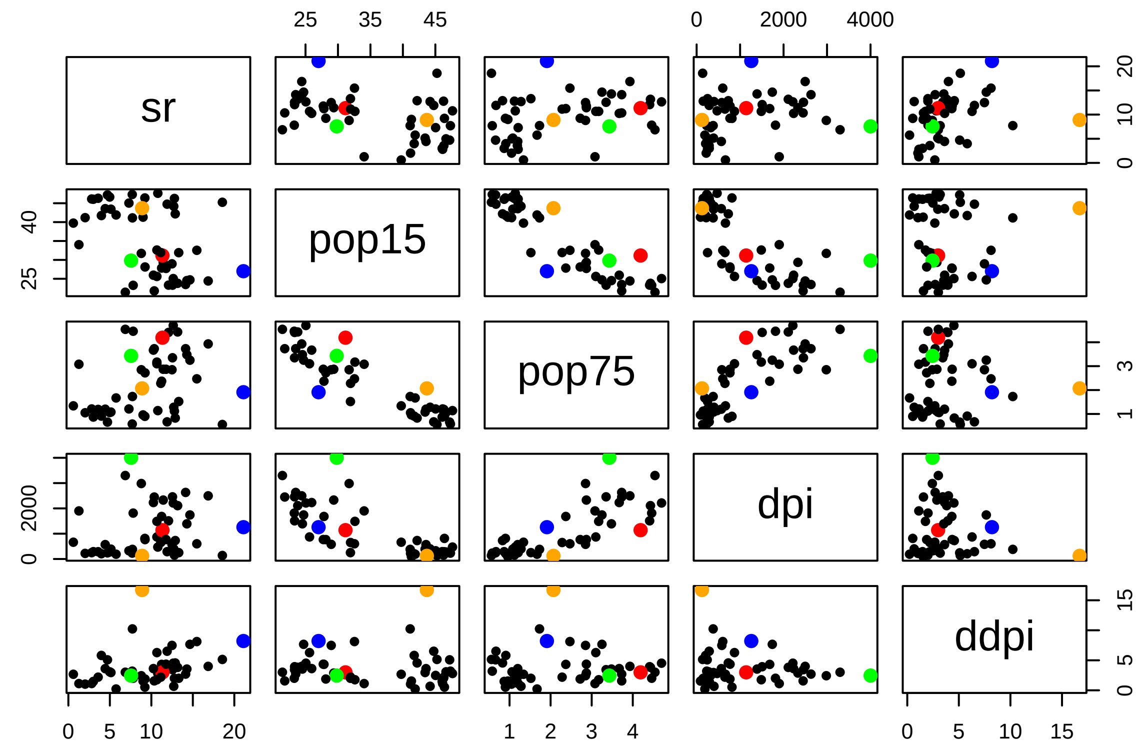
Moreover, investigate the relative variation of $\hat{\beta}$ due to these influential points.

```
abs( ( g$coefficients - gl$coefficients ) / g$coefficients )
## (Intercept)       pop15       pop75         dpi        ddpi
##   0.2223914   0.2622274   0.3353998   0.8664714   0.2871002
```

The leverages affect the estimates heavily (there is a variation of 22% at least).

We can also visualize the position of leverages for each covariate couple.

```
colors = rep( 'black', nrow( savings ) )
colors[ watchout_ids_lev ] = c('red', 'blue', 'green', 'orange')

pairs( savings[ , c( 'sr', 'pop15', 'pop75', 'dpi', 'ddpi' ) ],
       pch = 16, col = colors, cex = 1 + 0.5 * as.numeric( colors != 'black' ))
```



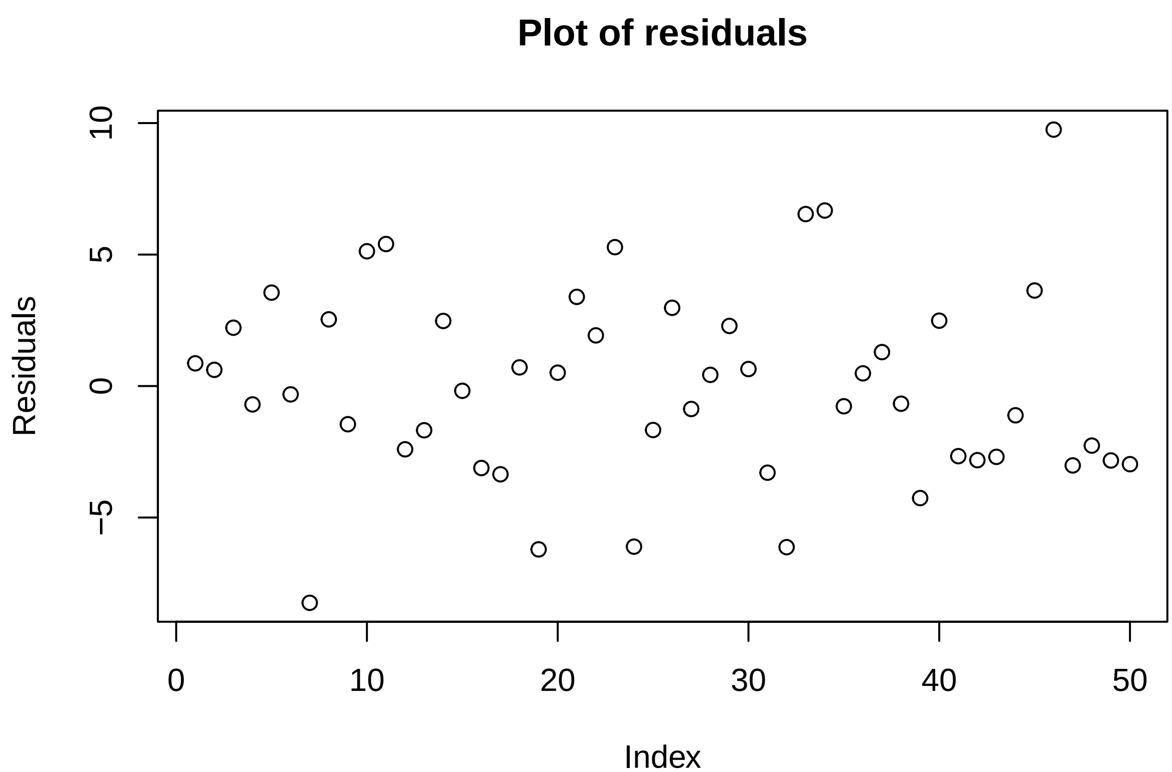
b. Standardized Residuals

Plot the residuals of the complete model.

solution

```
# Residui non standardizzati (nè studentizzati)

plot( g$res, ylab = "Residuals", main = "Plot of residuals" )

sort( g$res )
##          Chile        Iceland       Paraguay          Korea         Sweden
##     -8.2422307     -6.2105820     -6.1257589     -6.1069814     -4.2602834
##      Guatamala         Panama         Greece        Jamaica       Malaysia
##     -3.3552838     -3.2941656     -3.1161685     -3.0185314     -2.9708690
##          Libya        Tunisia United Kingdom         Turkey        Ecuador
##     -2.8295257     -2.8179200     -2.6924128     -2.6656824     -2.4056313
##        Uruguay        Finland     Luxembourg       Colombia  United States
##     -2.2638273     -1.6810857     -1.6708066     -1.4517071     -1.1115901
##         Norway       Portugal        Bolivia          Spain         Canada
##     -0.8717854     -0.7684447     -0.6983191     -0.6711565     -0.3168924
##        Germany    Netherlands   South Africa          India        Austria
##     -0.1806993      0.4255455      0.4831656      0.5086740      0.6163860
##      Nicaragua       Honduras      Australia South Rhodesia          Italy
##      0.6463966      0.7100245      0.8635798      1.2914342      1.9267549
##        Belgium    New Zealand         France    Switzerland          China
```

```
##      2.2189579      2.2855548      2.4754718      2.4868259      2.5360361
##          Malta        Ireland         Brazil      Venezuela     Costa Rica
##      2.9749098      3.3911306      3.5528094      3.6325177      5.1250782
##          Japan        Denmark           Peru    Philippines         Zambia
##      5.2814855      5.4002388      6.5394410      6.6750084      9.7509138
sort( g$res ) [ c( 1, 50 ) ]
##     Chile    Zambia
## -8.242231  9.750914

countries = row.names( savings )

identify( 1:50, g$res, countries )
```




```
## integer(0)
# click 2 times on the points you want to make a label appear
# it works only by console and plots window
```

`identify` is a useful function for detecting influent points. In input, you should call the x and y axes of the plot and the labels of data.

Usually, the residuals are represented wrt y-values or the single predictors.

This is useful also for testing the model hypotheses: homoschedasticity and normality of residuals (we are going to explain them later).

The representation with the index of the observation as x-axis is not that useful (except if we are interested in investigating the distribution of the residuals wrt the procedure used for data collection).

Plot the **Standardized Residuals** of the complete model.

Rule of thumb Given that standardized residuals are defined as:

$$r_i^{std} = \frac{y_i - \hat{y}_i}{\hat{S}} = \frac{\hat{\varepsilon}_i}{\hat{S}},$$

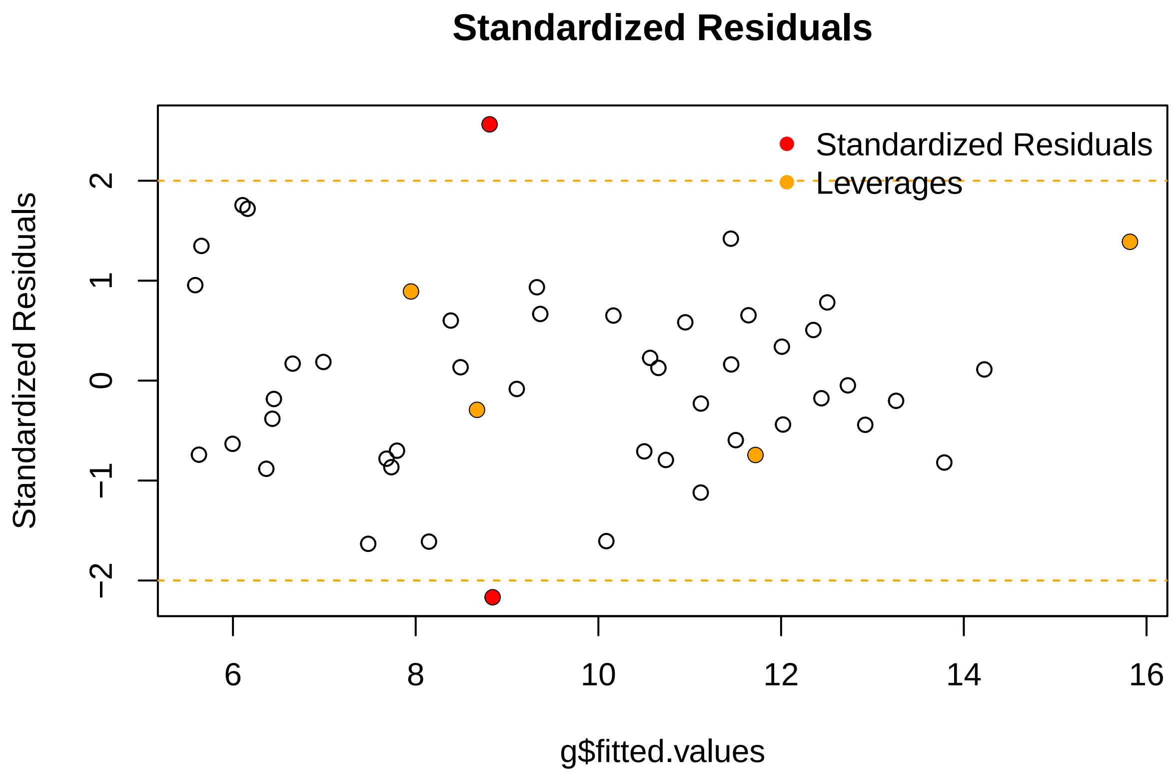
where $\hat{S}$ is the sample standard deviation of y, influential points satisfy the following inequality:

$$|r_i^{std}| > 2$$

solution

It is easy to see that influential points according to standardized residuals and to leverages are different.

```
gs = summary(g)
res_std = g$res/gs$sigma
watchout_ids_rstd = which( abs( res_std ) > 2 )
watchout_rstd = res_std[ watchout_ids_rstd ]
watchout_rstd
##     Chile    Zambia
## -2.167486  2.564229

# Residui standardizzati

plot( g$fitted.values, res_std, ylab = "Standardized Residuals", main = "Standardized Residuals" )
abline( h = c(-2,2), lty = 2, col = 'orange' )
points( g$fitted.values[watchout_ids_rstd],
        res_std[watchout_ids_rstd], col = 'red', pch = 16 )
points( g$fitted.values[watchout_ids_lev],
        res_std[watchout_ids_lev], col = 'orange', pch = 16 )
legend('topright', col = c('red','orange'),
       c('Standardized Residuals', 'Leverages'), pch = rep( 16, 2 ), bty = 'n' )
```



c. Studentized Residuals

Compute the Studentized Residuals, highlighting the influential points.

solution

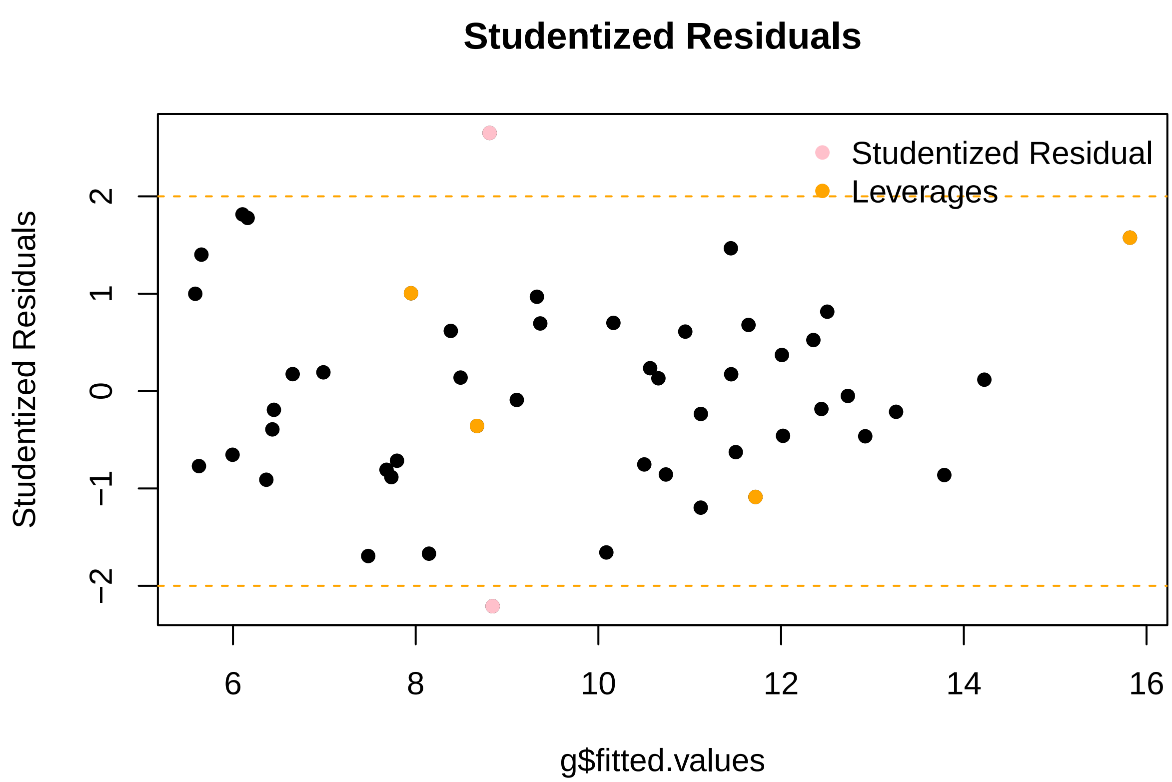
Studentized residuals, r_i, are computed as:

$$r_i = \frac{\hat{\varepsilon}_i}{\hat{S} \cdot \sqrt{(1 - h_{ii})}} \sim t_{n-p}$$

Since r_i is distributed according to a Student-t with $(n-p)$ degrees of freedom, we can calculate a p-value to test whether point $i - th$ is an outlier.

```
gs = summary( g )

gs$sigma
## [1] 3.802669

# manually
stud = g$residuals / ( gs$sigma * sqrt( 1 - lev ) )

# 'rstandard' gives studentized residuals automatically
stud = rstandard( g )

watchout_ids_stud = which( abs( stud ) > 2 )
watchout_stud = stud[ watchout_ids_stud ]
```

```r
watchout_stud
##     Chile    Zambia
## -2.209074  2.650915


plot( g$fitted.values, stud, ylab = "Studentized Residuals", main = "Studentized Residuals", pch = 16 )
points( g$fitted.values[watchout_ids_stud],
        stud[watchout_ids_stud], col = 'pink', pch = 16 )
points( g$fitted.values[watchout_ids_lev],
        stud[watchout_ids_lev], col = 'orange', pch = 16 )
abline( h = c(-2,2), lty = 2, col = 'orange' )
legend('topright', col = c('pink','orange'),
       c('Studentized Residual', 'Leverages'), pch = rep( 16, 3 ), bty = 'n' )
```



Studentized residuals and Standardized residuals identify the same influential points in this case.

d. Cook's distance

Cook's distance is a commonly used influential measure that combines the two characteristics of an influential point, that is the residual effect and the leverage, i.e. how observations are fitted by the model and how they are influential to the fitting. It can be expressed as:

$$C_i = \frac{r_i^2}{r} \cdot \left[\frac{h_{ii}}{1 - h_{ii}}\right]$$

in which r_i are the studentized residuals.

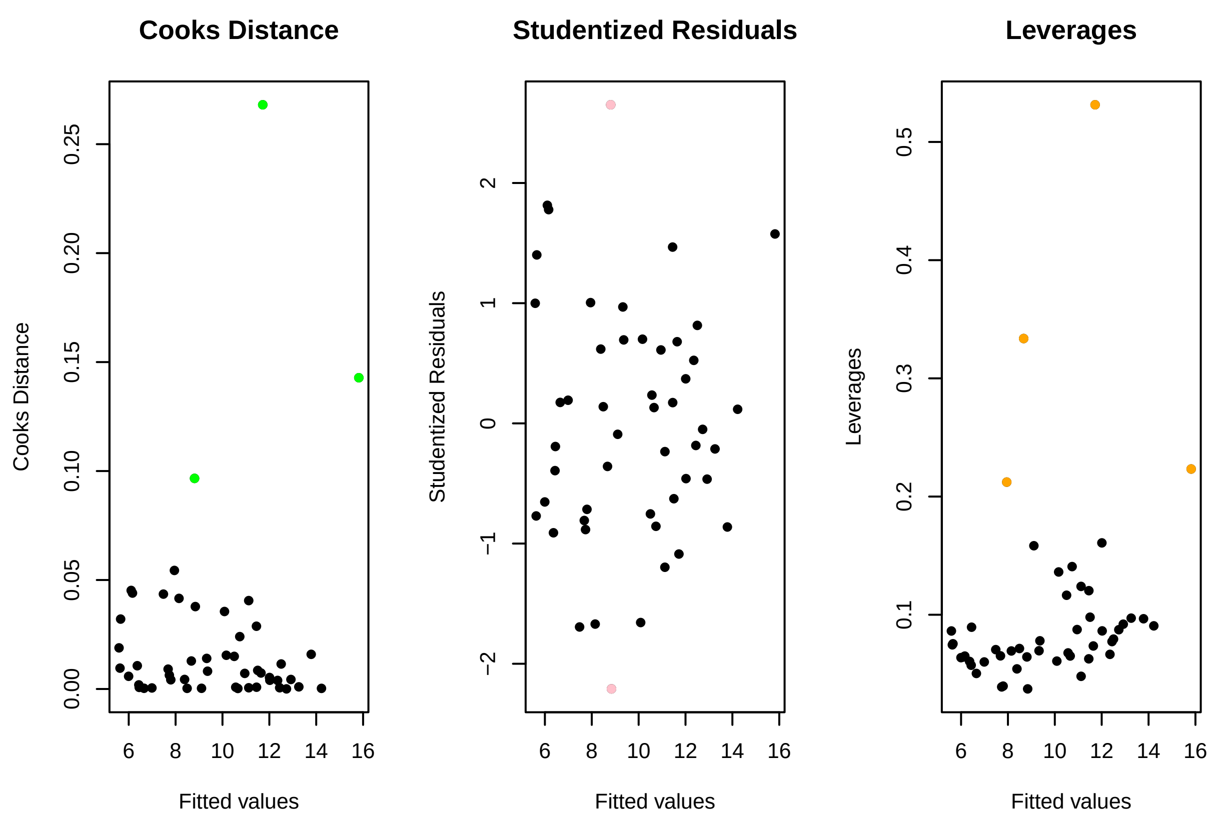
Rule of thumb A point is defined influential if:

$$C_i > \frac{4}{n-p}$$

```
Cdist = cooks.distance( g )

watchout_ids_Cdist = which( Cdist > 4/(n-p) )
watchout_Cdist = Cdist[ watchout_ids_Cdist ]
watchout_Cdist
##      Japan     Zambia      Libya
## 0.14281625 0.09663275 0.26807042
```

Three suspect points are detected.

```
par( mfrow = c( 1, 3 ) )
plot( g$fitted.values, Cdist, pch = 16, xlab = 'Fitted values',
      ylab = 'Cooks Distance', main = 'Cooks Distance' )
points( g$fitted.values[ watchout_ids_Cdist ], Cdist[ watchout_ids_Cdist ],
        col = 'green', pch = 16 )
plot( g$fitted.values, stud, pch = 16, xlab = 'Fitted values',
      ylab = 'Studentized Residuals', main = 'Studentized Residuals' )
points( g$fitted.values[ watchout_ids_stud ], stud[ watchout_ids_stud ],
        col = 'pink', pch = 16 )
plot( g$fitted.values, lev, pch = 16, xlab = 'Fitted values',
      ylab = 'Leverages', main = 'Leverages' )
points( g$fitted.values[ watchout_ids_lev ], lev[ watchout_ids_lev ],
        col = 'orange', pch = 16 )
```



Fit the model without influential points wrt Cook's distance and compare the outcome to the former model (on the complete dataset).

solution

```
#id_to_keep = (1:n)[ - watchout_ids_Cdist ]
id_to_keep = !( 1:n %in% watchout_ids_Cdist )

gl = lm( sr ~ pop15 + pop75 + dpi + ddpi, savings[ id_to_keep, ]  )

summary( gl )
##
## Call:
## lm(formula = sr ~ pop15 + pop75 + dpi + ddpi, data = savings[id_to_keep,
##     ])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.4552 -2.5129 -0.1117  1.7477  6.6646
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 19.8321854  7.8341981   2.531   0.0152 *
## pop15       -0.3047007  0.1513371  -2.013   0.0505 .
## pop75       -0.3030249  1.1135478  -0.272   0.7869
## dpi         -0.0005535  0.0008469  -0.654   0.5170
## ddpi         0.4137823  0.2526006   1.638   0.1089
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.441 on 42 degrees of freedom
## Multiple R-squared:  0.3503, Adjusted R-squared:  0.2885
## F-statistic: 5.662 on 4 and 42 DF,  p-value: 0.0009742
```

Observe that the fitting in terms of R^2 slightly improved wrt to the complete model.

```
abs( ( gl$coef - g$coef )/g$coef )
## (Intercept)       pop15       pop75         dpi        ddpi
## 0.305743704 0.339320881 0.820854095 0.642906116 0.009976742
```

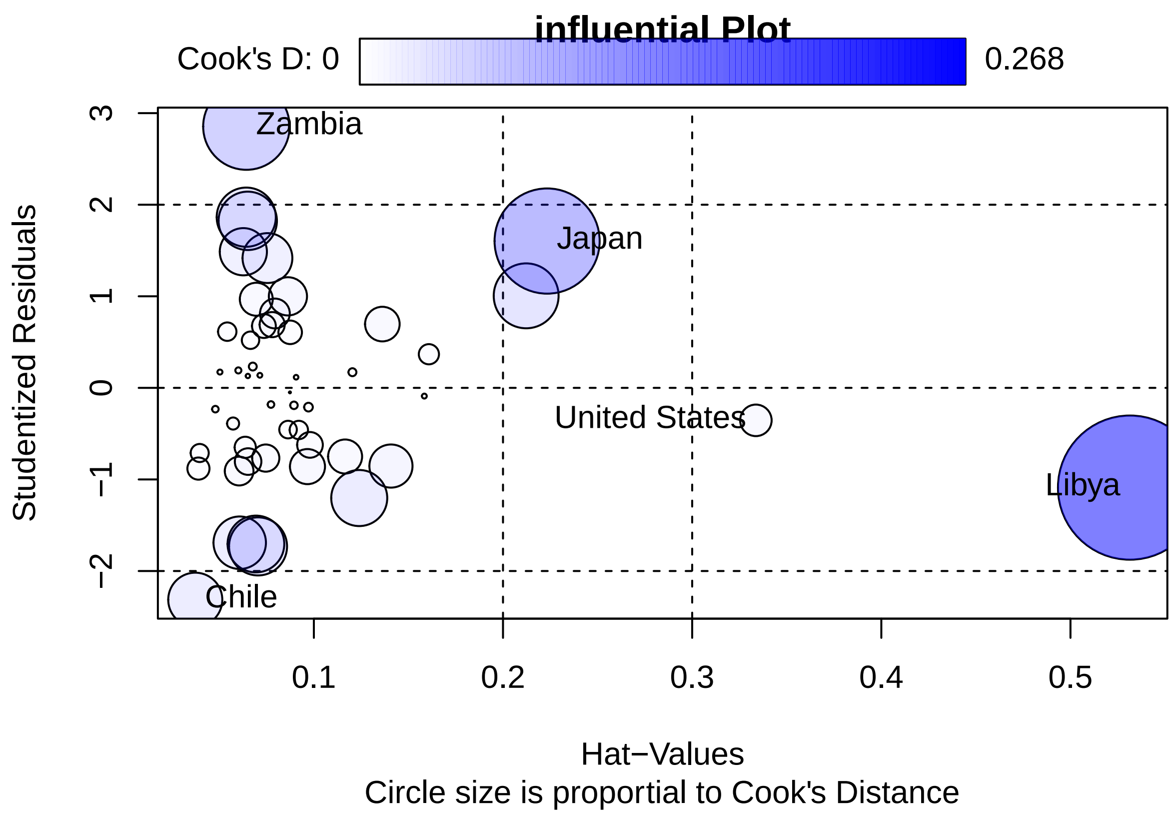
The coefficient for dpi changed by about 64%, the coefficient of pop75 by 82%.

All together: Influential Plot

The influential plot represents the studentized residuals vs leverages, and highlights them with a circle which is proportional to Cook's distance.

```
influencePlot( g, id.method = "identify", main = "influential Plot",
               sub = "Circle size is proportial to Cook's Distance" )
## Warning in plot.window(...): parametro grafico "id.method" non valido
## Warning in plot.xy(xy, type, ...): parametro grafico "id.method" non valido
## Warning in axis(side = side, at = at, labels = labels, ...): parametro grafico
## "id.method" non valido

## Warning in axis(side = side, at = at, labels = labels, ...): parametro grafico
## "id.method" non valido
## Warning in box(...): parametro grafico "id.method" non valido
```

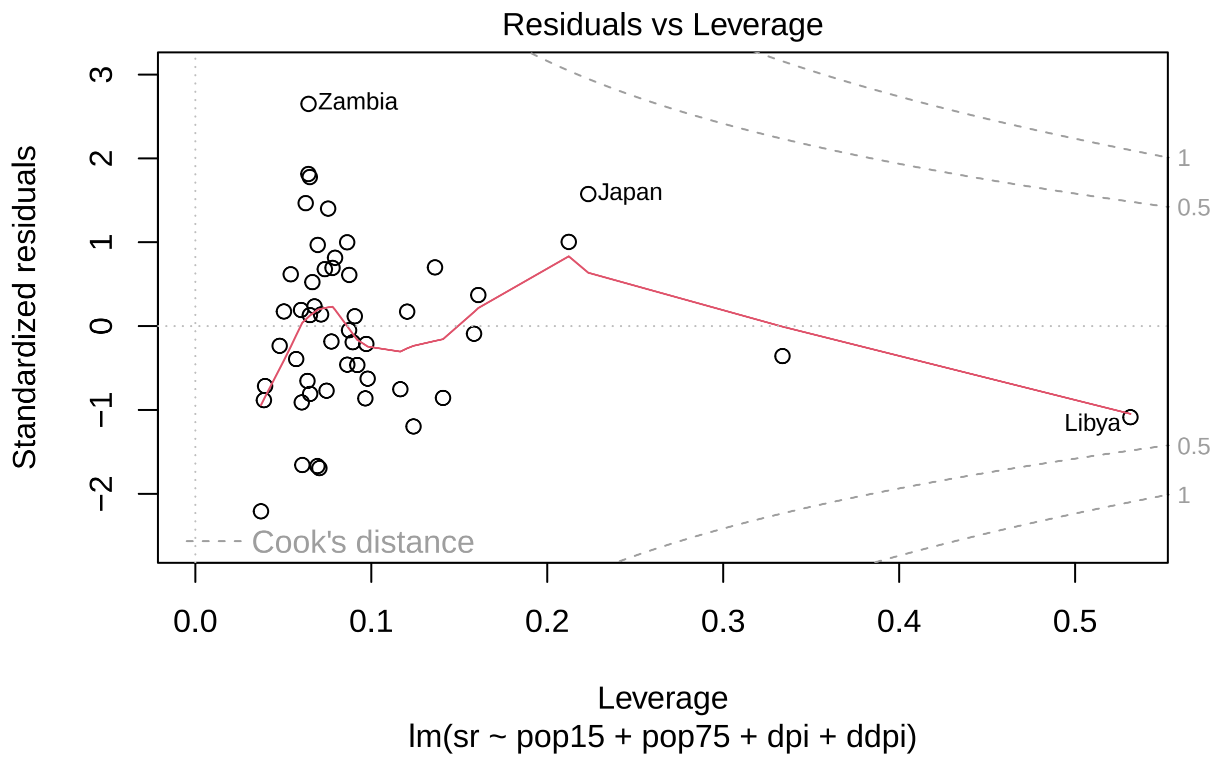
```
## Warning in title(...): parametro grafico "id.method" non valido
## Warning in plot.xy(xy.coords(x, y), type = type, ...): parametro grafico
## "id.method" non valido
```




```
##                  StudRes        Hat      CookD
## Chile         -2.3134295 0.03729796 0.03781324
## Japan          1.6032158 0.22330989 0.14281625
## United States -0.3546151 0.33368800 0.01284481
## Zambia         2.8535583 0.06433163 0.09663275
## Libya         -1.0893033 0.53145676 0.26807042
```

There is another easy way to visually detect the influential points by Cook's distance.

```
plot(g, which = 5)
```



All together: Influential measures

`influential.measures` produces a class "infl" object tabular display showing several diagnostics measures (such as h_{ii} and Cook's distance). Those cases which are influential with respect to any of these measures are marked with an asterisk.

```
influence.measures( g )
## Influence measures of
##   lm(formula = sr ~ pop15 + pop75 + dpi + ddpi, data = savings) :
##
##                  dfb.1_ dfb.pp15 dfb.pp75  dfb.dpi  dfb.ddpi   dffit cov.r
## Australia       0.01232 -0.01044 -0.02653  0.04534 -0.000159  0.0627 1.193
## Austria        -0.01005  0.00594  0.04084 -0.03672 -0.008182  0.0632 1.268
## Belgium        -0.06416  0.05150  0.12070 -0.03472 -0.007265  0.1878 1.176
## Bolivia         0.00578 -0.01270 -0.02253  0.03185  0.040642 -0.0597 1.224
## Brazil          0.08973 -0.06163 -0.17907  0.11997  0.068457  0.2646 1.082
## Canada          0.00541 -0.00675  0.01021 -0.03531 -0.002649 -0.0390 1.328
## Chile          -0.19941  0.13265  0.21979 -0.01998  0.120007 -0.4554 0.655
## China           0.02112 -0.00573 -0.08311  0.05180  0.110627  0.2008 1.150
## Colombia        0.03910 -0.05226 -0.02464  0.00168  0.009084 -0.0960 1.167
## Costa Rica     -0.23367  0.28428  0.14243  0.05638 -0.032824  0.4049 0.968
## Denmark        -0.04051  0.02093  0.04653  0.15220  0.048854  0.3845 0.934
## Ecuador         0.07176 -0.09524 -0.06067  0.01950  0.047786 -0.1695 1.139
## Finland        -0.11350  0.11133  0.11695 -0.04364 -0.017132 -0.1464 1.203
## France         -0.16600  0.14705  0.21900 -0.02942  0.023952  0.2765 1.226
## Germany        -0.00802  0.00822  0.00835 -0.00697 -0.000293 -0.0152 1.226
## Greece         -0.14820  0.16394  0.02861  0.15713 -0.059599 -0.2811 1.140
```

```
## Guatamala       0.01552 -0.05485  0.00614  0.00585  0.097217 -0.2305 1.085
## Honduras       -0.00226  0.00984 -0.01020  0.00812 -0.001887  0.0482 1.186
## Iceland         0.24789 -0.27355 -0.23265 -0.12555  0.184698 -0.4768 0.866
## India           0.02105 -0.01577 -0.01439 -0.01374 -0.018958  0.0381 1.202
## Ireland        -0.31001  0.29624  0.48156 -0.25733 -0.093317  0.5216 1.268
## Italy           0.06619 -0.07097  0.00307 -0.06999 -0.028648  0.1388 1.162
## Japan           0.63987 -0.65614 -0.67390  0.14610  0.388603  0.8597 1.085
## Korea          -0.16897  0.13509  0.21895  0.00511 -0.169492 -0.4303 0.870
## Luxembourg     -0.06827  0.06888  0.04380 -0.02797  0.049134 -0.1401 1.196
## Malta           0.03652 -0.04876  0.00791 -0.08659  0.153014  0.2386 1.128
## Norway          0.00222 -0.00035 -0.00611 -0.01594 -0.001462 -0.0522 1.168
## Netherlands     0.01395 -0.01674 -0.01186  0.00433  0.022591  0.0366 1.229
## New Zealand    -0.06002  0.06510  0.09412 -0.02638 -0.064740  0.1469 1.134
## Nicaragua      -0.01209  0.01790  0.00972 -0.00474 -0.010467  0.0397 1.174
## Panama          0.02828 -0.05334  0.01446 -0.03467 -0.007889 -0.1775 1.067
## Paraguay       -0.23227  0.16416  0.15826  0.14361  0.270478 -0.4655 0.873
## Peru           -0.07182  0.14669  0.09148 -0.08585 -0.287184  0.4811 0.831
## Philippines    -0.15707  0.22681  0.15743 -0.11140 -0.170674  0.4884 0.818
## Portugal       -0.02140  0.02551 -0.00380  0.03991 -0.028011 -0.0690 1.233
## South Africa    0.02218 -0.02030 -0.00672 -0.02049 -0.016326  0.0343 1.195
## South Rhodesia  0.14390 -0.13472 -0.09245 -0.06956 -0.057920  0.1607 1.313
## Spain          -0.03035  0.03131  0.00394  0.03512  0.005340 -0.0526 1.208
## Sweden          0.10098 -0.08162 -0.06166 -0.25528 -0.013316 -0.4526 1.086
## Switzerland     0.04323 -0.04649 -0.04364  0.09093 -0.018828  0.1903 1.147
## Turkey         -0.01092 -0.01198  0.02645  0.00161  0.025138 -0.1445 1.100
## Tunisia         0.07377 -0.10500 -0.07727  0.04439  0.103058 -0.2177 1.131
## United Kingdom  0.04671 -0.03584 -0.17129  0.12554  0.100314 -0.2722 1.189
## United States   0.06910 -0.07289  0.03745 -0.23312 -0.032729 -0.2510 1.655
## Venezuela      -0.05083  0.10080 -0.03366  0.11366 -0.124486  0.3071 1.095
## Zambia          0.16361 -0.07917 -0.33899  0.09406  0.228232  0.7482 0.512
## Jamaica         0.10958 -0.10022 -0.05722 -0.00703 -0.295461 -0.3456 1.200
## Uruguay        -0.13403  0.12880  0.02953  0.13132  0.099591 -0.2051 1.187
## Libya           0.55074 -0.48324 -0.37974 -0.01937 -1.024477 -1.1601 2.091
## Malaysia        0.03684 -0.06113  0.03235 -0.04956 -0.072294 -0.2126 1.113
##                  cook.d    hat inf
## Australia      8.04e-04 0.0677
## Austria        8.18e-04 0.1204
## Belgium        7.15e-03 0.0875
## Bolivia        7.28e-04 0.0895
## Brazil         1.40e-02 0.0696
## Canada         3.11e-04 0.1584
## Chile          3.78e-02 0.0373   *
## China          8.16e-03 0.0780
## Colombia       1.88e-03 0.0573
## Costa Rica     3.21e-02 0.0755
## Denmark        2.88e-02 0.0627
## Ecuador        5.82e-03 0.0637
## Finland        4.36e-03 0.0920
## France         1.55e-02 0.1362
## Germany        4.74e-05 0.0874
## Greece         1.59e-02 0.0966
## Guatamala      1.07e-02 0.0605
## Honduras       4.74e-04 0.0601
```

```
## Iceland        4.35e-02 0.0705
## India          2.97e-04 0.0715
## Ireland        5.44e-02 0.2122
## Italy          3.92e-03 0.0665
## Japan          1.43e-01 0.2233
## Korea          3.56e-02 0.0608
## Luxembourg     3.99e-03 0.0863
## Malta          1.15e-02 0.0794
## Norway         5.56e-04 0.0479
## Netherlands    2.74e-04 0.0906
## New Zealand    4.38e-03 0.0542
## Nicaragua      3.23e-04 0.0504
## Panama         6.33e-03 0.0390
## Paraguay       4.16e-02 0.0694
## Peru           4.40e-02 0.0650
## Philippines    4.52e-02 0.0643
## Portugal       9.73e-04 0.0971
## South Africa   2.41e-04 0.0651
## South Rhodesia 5.27e-03 0.1608
## Spain          5.66e-04 0.0773
## Sweden         4.06e-02 0.1240
## Switzerland    7.33e-03 0.0736
## Turkey         4.22e-03 0.0396
## Tunisia        9.56e-03 0.0746
## United Kingdom 1.50e-02 0.1165
## United States  1.28e-02 0.3337   *
## Venezuela      1.89e-02 0.0863
## Zambia         9.66e-02 0.0643   *
## Jamaica        2.40e-02 0.1408
## Uruguay        8.53e-03 0.0979
## Libya          2.68e-01 0.5315   *
## Malaysia       9.11e-03 0.0652
# DFBETA measures the difference in each parameter estimate with and without the influential point
```

There are other indices for detecting influential points, such as DFBETAs and DFFITs.

`https://cran.r-project.org/web/packages/olsrr/vignettes/influence_measures.html`

3. Hypotheses of the model

In Laboratory 1, we analysed the normality and homoschedasticity of the residuals. Let's now look at the collinearity and nonlinearity.

Nonlinearity/Collinearity

Partial regression plots

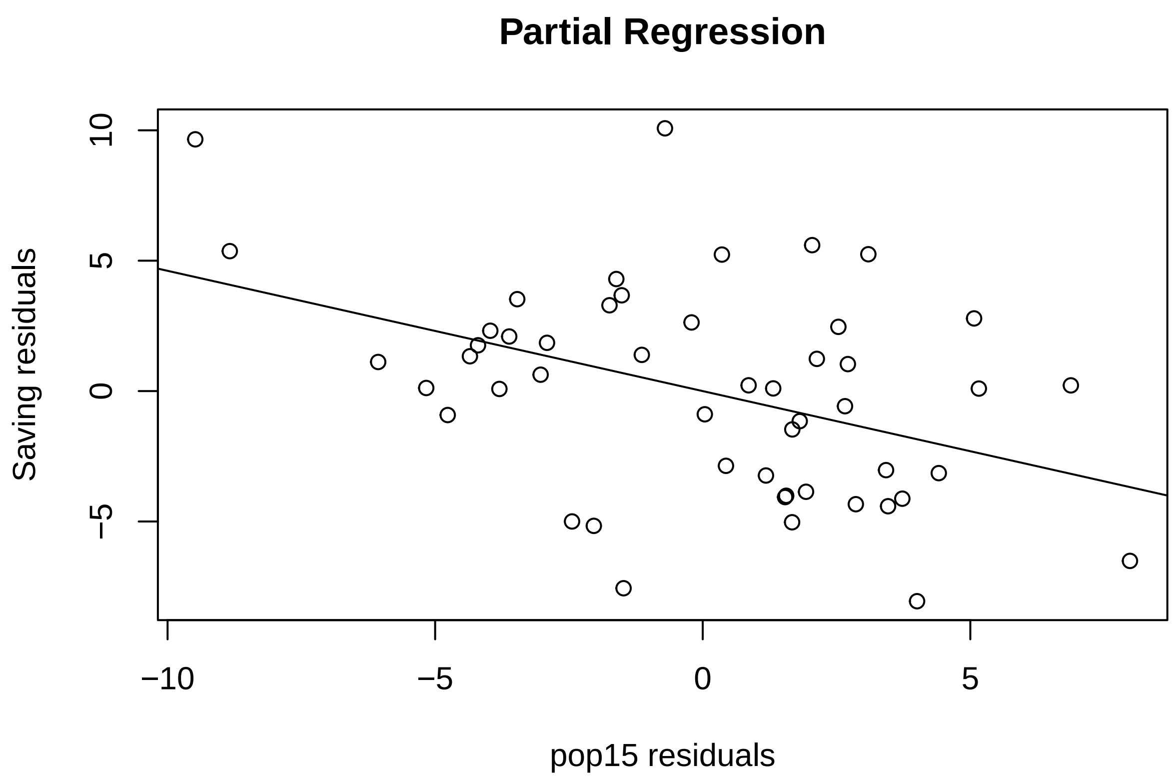
Partial Regression or Added Variable plots can help isolate the effect of x_i on y.

1. Regress y on all x except x_i, get residuals $\hat{\delta}$. This represents y with the other X-effect taken out.
2. Regress x_i on all x except x_i, get residuals $\hat{\gamma}$.This represents x_i with the other X-effect taken out.
3. Plot $\hat{\delta}$ against $\hat{\gamma}$

The slope of a line fitted to the plot adds some insight into the meaning of regression coefficients. Look for non-linearity and outliers and/or influential points.

We construct a partial regression (added variable) plot for pop15:

```
d <- lm(sr ~ pop75 + dpi + ddpi,savings)$res
m <- lm(pop15 ~ pop75 + dpi + ddpi,savings)$res
plot(m,d,xlab="pop15 residuals",ylab="Saving residuals", main="Partial Regression")
abline(0,g$coef['pop15'])
```



Compare the slope on the plot to the original regression and show the line on the plot.

```
lm(d ~ m)$coef
##   (Intercept)             m
## -1.545720e-16 -4.611931e-01

g$coef
##   (Intercept)         pop15         pop75           dpi          ddpi
## 28.5660865407 -0.4611931471 -1.6914976767 -0.0003369019  0.4096949279
```

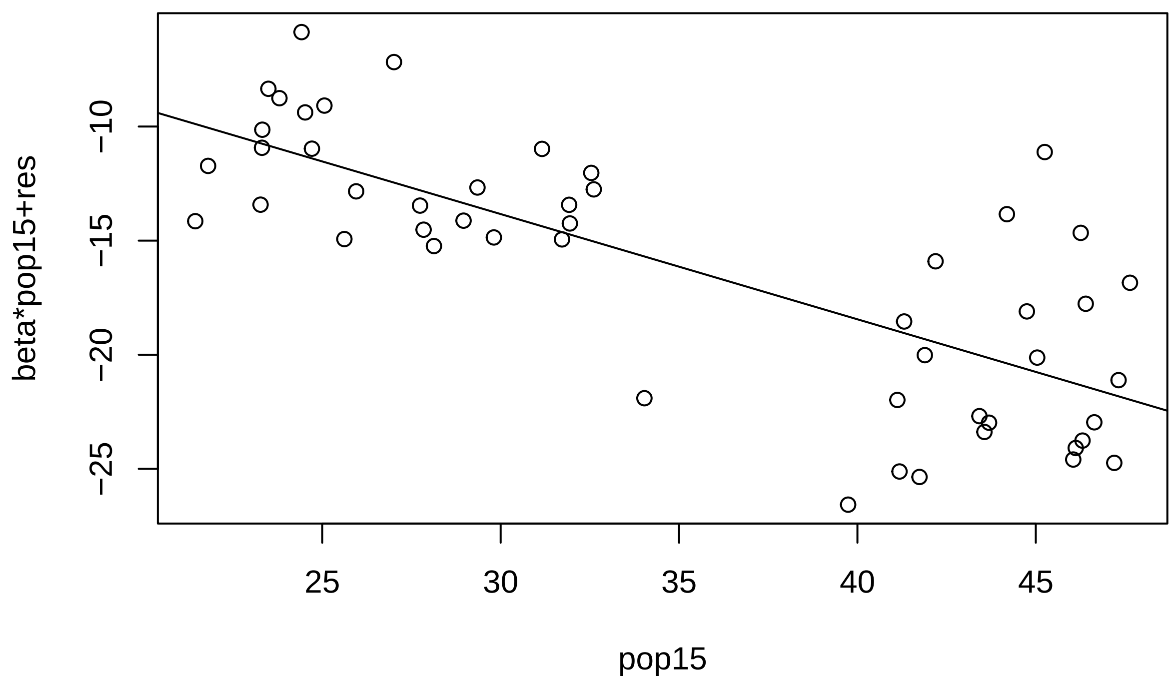
Notice how the slope in the plot and the slope for pop15 in the regression fit are the same.

The partial regression plot also provides some intuition about the meaning of regression coefficients. We are looking at the marginal relationship between the response and the predictor after the effect of the other predictors has been removed. Multiple regression is difficult because we cannot visualize the full relationship because of the high dimensionality. The partial regression plot allows us to focus on the relationship between one predictor and the response, much as in simple regression.

Partial Residual plots

Partial Residual plots are a competitor to added variable plots. These plot $\varepsilon_i + \beta_i x_i$ against x_i. The slope on the plot will have the same interpretation of Partial regression plots. Partial residual plots are reckoned to be better for non-linearity detection while added variable plots are better for outlier/influential detection. A partial residual plot is easier to do:

```r
prplot(g,1) # 1 stands for the position of the independent variable
```




```r
# plot(savings$pop15,g$res+g$coef['pop15']*savings$pop15,xlab="pop under 15", ylab="Saving(adjusted)",m
# abline(0,g$coef['pop15'])
```

VIF Variance Inflation Factor is an index of collinearity.

$$Var(\beta_j) = \frac{S^2}{(n-1)\cdot S_j^2} \times \frac{1}{1-R_j^2}$$

where S_j^2 is the variance of x_j and the $VIF_j = \frac{1}{1-R_j^2}$. R_j is the Rsquare of a lm with x_j as response variable and all other veriables in $\mathbf{x}$ as predictors. Collinearity means that two covariates share a lot of variability and thus are likely to carry the same information. (Rule of thumb: VIF > 5 or 10)

```r
vif( g )
##    pop15    pop75      dpi     ddpi
## 5.937661 6.629105 2.884369 1.074309
```

In this case, **pop75** and **pop15** show the highest collinearity.

4. Transformation: Box-Cox

In this section we would like to answer the following question: what should we do when there is a clear violation of hypotheses? The answer consists in investigating variable transformations (transformation of the outcome).

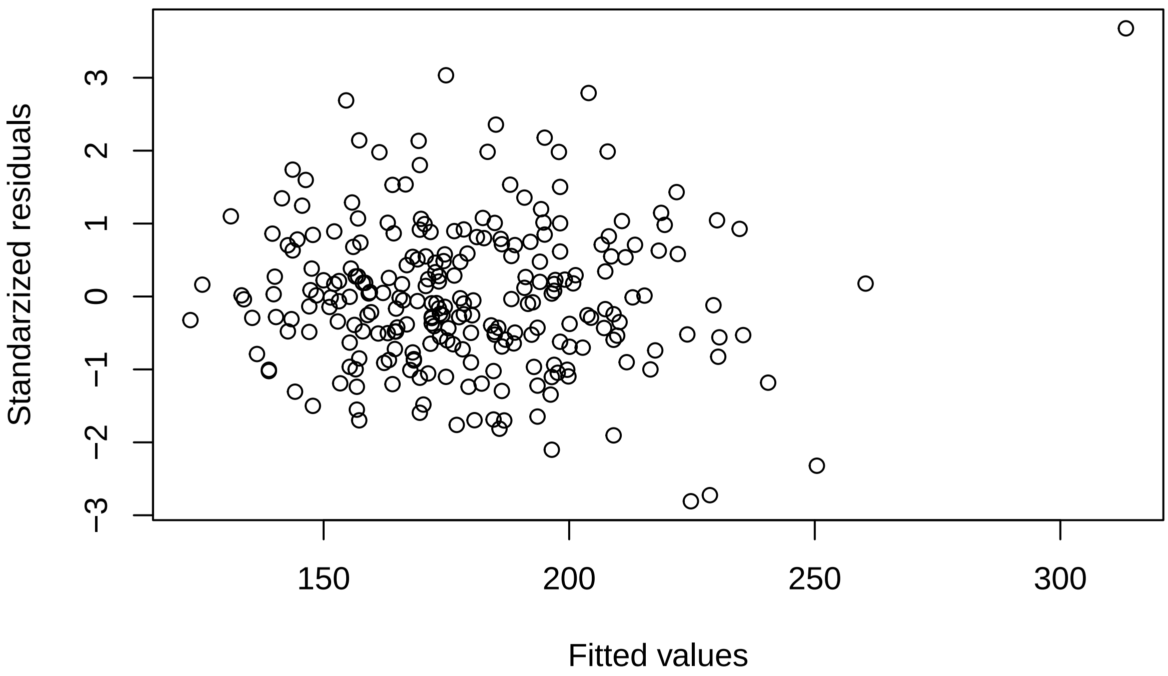
Warning Transforming a variable can lead to a more difficult interpretation of the model.

An algorithm that helps us in variable transformation is the *Box-Cox* algorithm. It detects the best λ among a family of transformations ($\frac{y^{\lambda}-1}{\lambda}$, if $\lambda \neq 0$, otherwise $log(y)$) in order to gain the Normality/homoscedasticity for **positive data**.

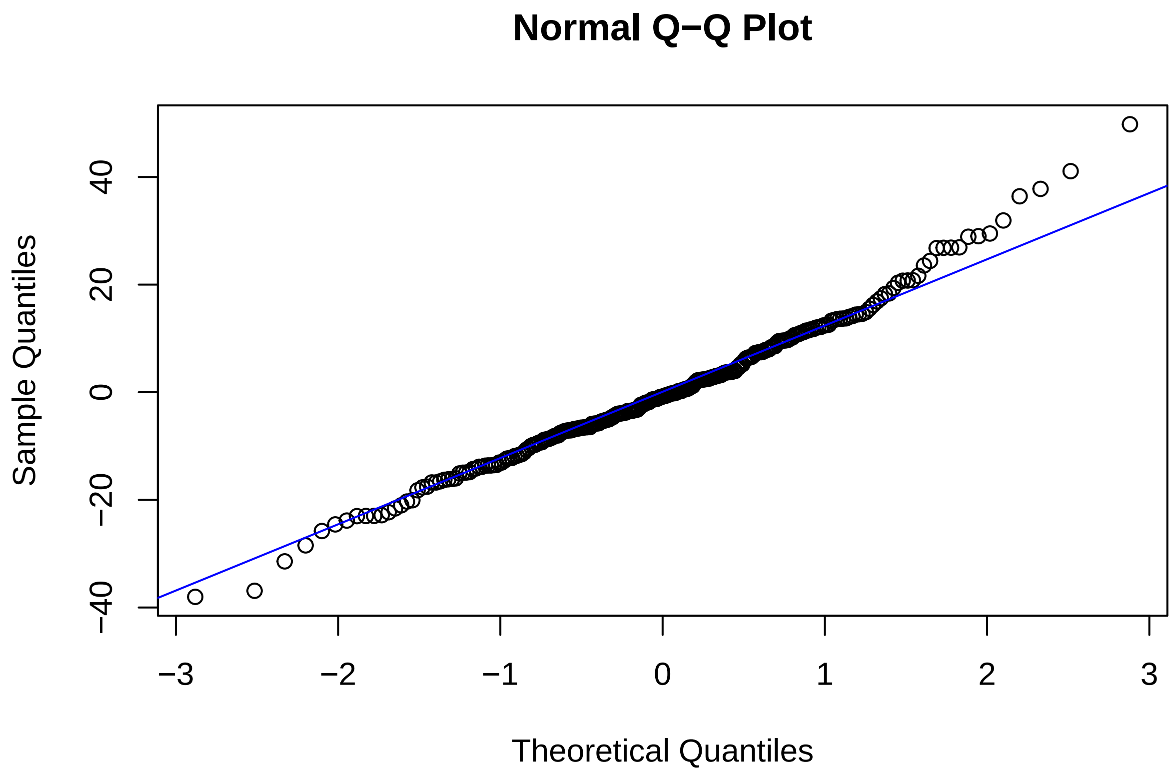
Here we report an example (linear regression with one predictor). For a group of 252 male subjects, various body measurements were obtained. An accurate measurement of the percentage of body fat is recorded for each. The goal is to use the feature 'Abdomen' (indicating abdomen circumference (cm)) for predicting the weight.

```r
library(BAS)
data(bodyfat) # help(bodyfat)
# summary(bodyfat)
mod = lm(Weight ~ Abdomen, data = bodyfat)
summary(mod)
##
## Call:
## lm(formula = Weight ~ Abdomen, data = bodyfat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -38.023  -8.219  -0.796   8.390  49.797
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -45.08142    7.38608  -6.104 3.93e-09 ***
## Abdomen       2.42022    0.07927  30.532  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.54 on 250 degrees of freedom
## Multiple R-squared:  0.7885, Adjusted R-squared:  0.7877
## F-statistic: 932.2 on 1 and 250 DF,  p-value: < 2.2e-16
```

```r
mod_res = mod$residuals/summary(mod)$sigma

plot( mod$fitted, mod_res, xlab = 'Fitted values',  ylab = 'Standarzized residuals'  )
```




```r
qqnorm( mod$residuals )
qqline( mod$residuals, col = 'blue' )
```




```r
# abline( 0, 1, col = 'red' )

shapiro.test( mod_res )
##
##  Shapiro-Wilk normality test
##
## data:  mod_res
## W = 0.98721, p-value = 0.02412
```

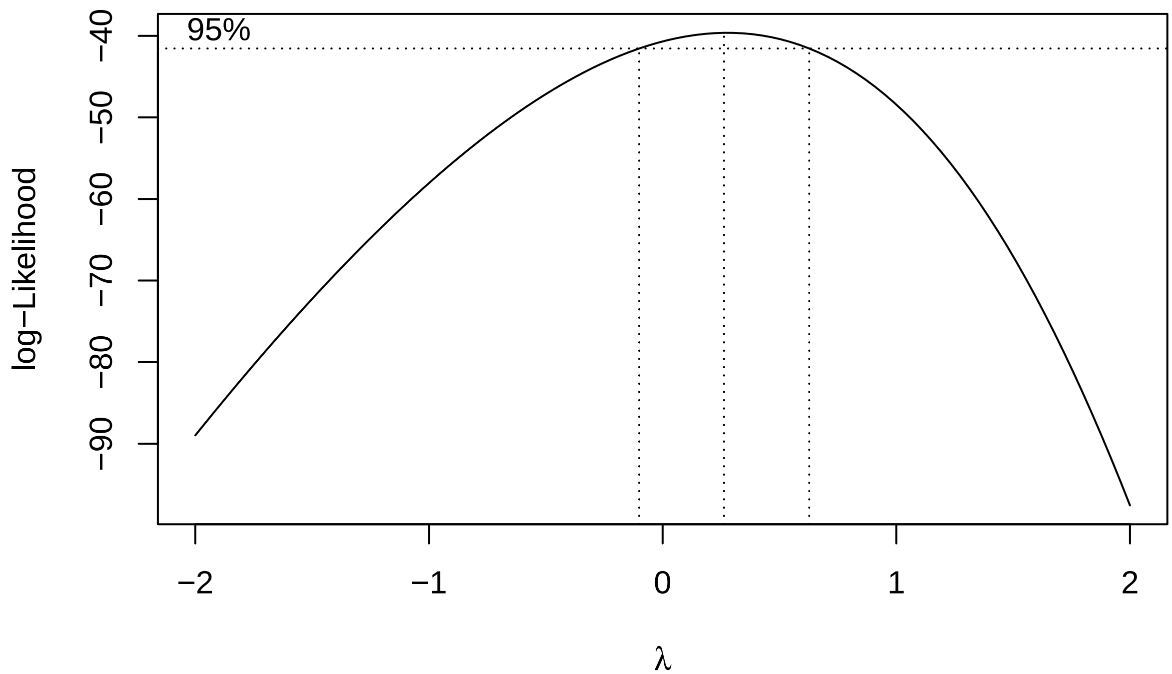
Very good fit of the model: $R^2 = 78.9\%$ and the predictor is significant with a $p-value < 2e-16$. Nonetheless, the normality assumption cannot be accepted with a lot of confidence: QQ plots show heavy tails (especially the right one) and Shapiro-Wilks test return a p-value of 0.02412.

So, we apply the Box-Cox transformation.

Remark We can apply the Box-Cox transformation, because variable is positive.

The best λ that is chosen is the one maximizing the likelihood of the transformed data of being

```r
b = boxcox(Weight ~ Abdomen, data = bodyfat)
```

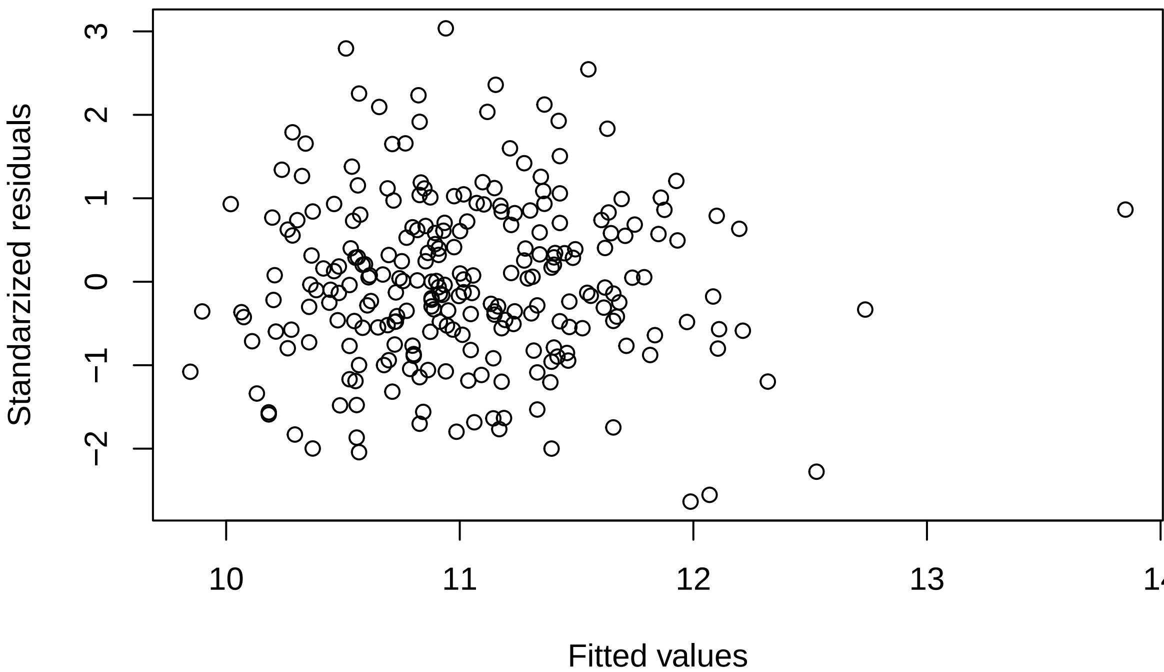



```r
names(b)
## [1] "x" "y"
#y likelihood evaluation
#x lambda evaluated
best_lambda_ind = which.max( b$y )
best_lambda = b$x[ best_lambda_ind ]
best_lambda
## [1] 0.2626263
```

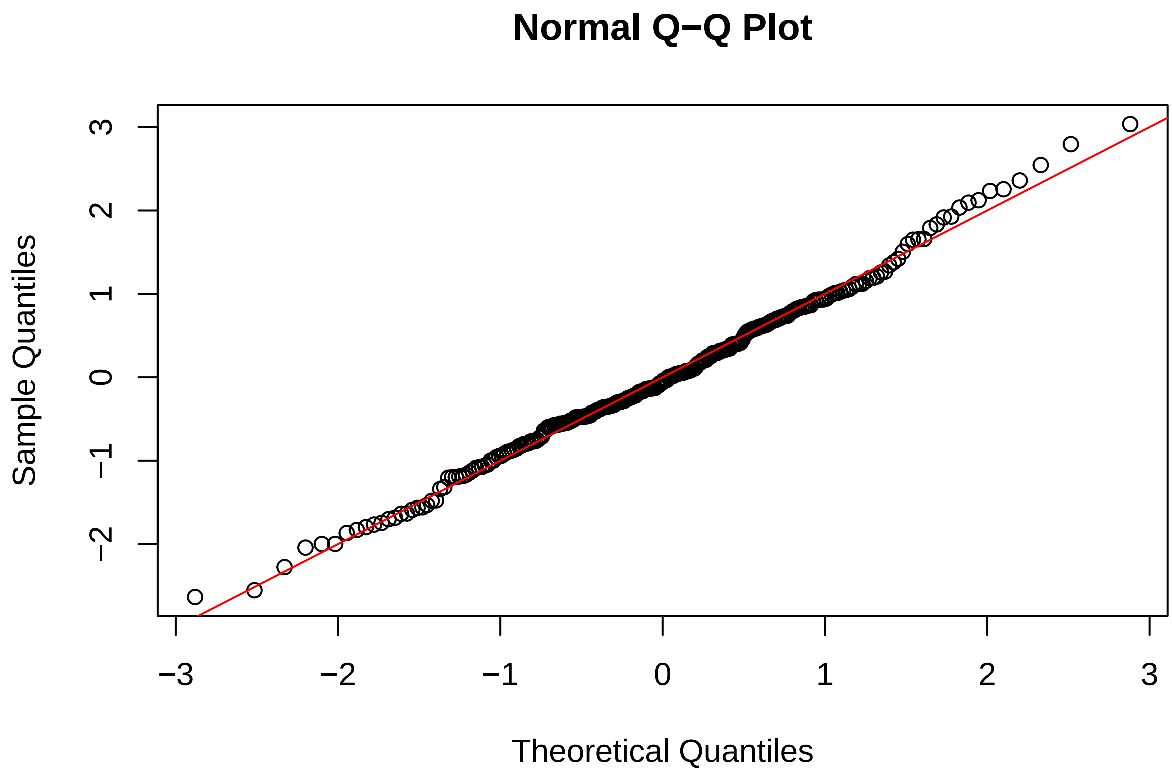
We can see that the best transformation is the one related to the *maximum* of the curve. The estimates are obtained through *Maximum Likelihood* method. According to this method, the best λ is $ 0.2626263$.

Finally, we test the new model and we investigate the standardized residuals.

```r
mod1 = lm( (Weight ^ best_lambda - 1)/best_lambda ~ Abdomen, data = bodyfat )
#summary(mod1)

mod1_res = mod1$residuals/summary( mod1 )$sigma

plot( mod1$fitted, mod1_res, xlab = 'Fitted values',  ylab = 'Standarzized residuals'  )
```




```r
qqnorm( mod1_res )
abline( 0, 1, col = 'red' )
```




```
shapiro.test( residuals( mod1 ) )
##
##  Shapiro-Wilk normality test
##
## data:  residuals(mod1)
## W = 0.99482, p-value = 0.5518
```

The normality of residuals improved after Box-Cox Transformation.