Simple Linear Regression_lab - Pre

Lídia Montero & Josep Franquet

2021

Load Data

```
# Clear plots
if(!is.null(dev.list())) dev.off()
```

```
## null device
##           1
```

```
# Clean workspace
rm(list=ls())

setwd("/home/pau/Escriptori/adei/apunts")
load("Anscombe73raw.RData")

ls()
```

```
## [1] "anscombe"     "last.warning"
```

```
anscombe
```

```
##    XA    YA XB   YB XC    YC XD    YD
## 1  10  8.04 10 9.14 10  7.46  8  6.58
## 2   8  6.95  8 8.14  8  6.77  8  5.76
## 3  13  7.58 13 8.74 13 12.74  8  7.71
## 4   9  8.81  9 8.77  9  7.11  8  8.84
## 5  11  8.33 11 9.26 11  7.81  8  8.47
## 6  14  9.96 14 8.10 14  8.84  8  7.04
## 7   6  7.24  6 6.13  6  6.08  8  5.25
## 8   4  4.26  4 3.10  4  5.39 19 12.50
## 9  12 10.84 12 9.13 12  8.15  8  5.56
## 10  7  4.82  7 7.26  7  6.42  8  7.91
## 11  5  5.68  5 4.74  5  5.73  8  6.89
```

```
attach(anscombe) #Thus, we will not have to write anscombe$var when accessing a variable
summary(anscombe) #Summary of the whole data (at a variable level)
```

```
##        XA             YA              XB             YB              XC    
##  Min.   : 4.0   Min.   : 4.260   Min.   : 4.0   Min.   :3.100   Min.   : 4.0  
##  1st Qu.: 6.5   1st Qu.: 6.315   1st Qu.: 6.5   1st Qu.:6.695   1st Qu.: 6.5  
##  Median : 9.0   Median : 7.580   Median : 9.0   Median :8.140   Median : 9.0  
##  Mean   : 9.0   Mean   : 7.501   Mean   : 9.0   Mean   :7.501   Mean   : 9.0  
##  3rd Qu.:11.5   3rd Qu.: 8.570   3rd Qu.:11.5   3rd Qu.:8.950   3rd Qu.:11.5  
##  Max.   :14.0   Max.   :10.840   Max.   :14.0   Max.   :9.260   Max.   :14.0  
##        YC              XD           YD        
##  Min.   : 5.39   Min.   : 8   Min.   : 5.250  
##  1st Qu.: 6.25   1st Qu.: 8   1st Qu.: 6.170  
##  Median : 7.11   Median : 8   Median : 7.040  
##  Mean   : 7.50   Mean   : 9   Mean   : 7.501  
##  3rd Qu.: 7.98   3rd Qu.: 8   3rd Qu.: 8.190  
##  Max.   :12.74   Max.   :19   Max.   :12.500
```

Teoria

Los modelos de regresión lineal son una técnica estadística utilizada para estudiar la relación entre una variable dependiente y una o más variables independientes. La regresión lineal se puede utilizar para predecir el valor de la variable dependiente a partir de las variables independientes.

Hay varios tipos de modelos de regresión lineal, algunos de los cuales se describen a continuación:

1. **Regresión lineal simple**: Es un modelo en el que hay una sola variable independiente y una variable dependiente. Se utiliza para predecir la relación entre dos variables continuas.

2. **Regresión lineal múltiple**: Es un modelo en el que hay dos o más variables independientes y una variable dependiente. Se utiliza para predecir la relación entre varias variables continuas.

3. **Regresión lineal polinómica**: Es un modelo en el que se ajusta una curva polinómica a los datos en lugar de una línea recta. Se utiliza cuando la relación entre las variables no es lineal.

4. **Regresión logística**: Es un modelo en el que se utiliza una función logística para predecir una variable dependiente categórica (por ejemplo, sí/no, éxito/fracaso) a partir de una o más variables independientes.

5. **Regresión de Poisson**: Es un modelo en el que se utiliza una distribución de Poisson para predecir el número de eventos raros en un período de tiempo determinado, a partir de una o más variables independientes.

Cada modelo de regresión lineal tiene sus propias suposiciones y limitaciones, y se debe elegir el modelo adecuado en función de los datos y el problema que se esté abordando.

Regresión simple

El modelo de regresión lineal simple es una técnica estadística utilizada para estudiar la relación entre dos variables continuas: una variable independiente y una variable dependiente. El objetivo es encontrar la línea recta que mejor se ajuste a los datos y que pueda utilizarse para predecir el valor de la variable dependiente a partir de la variable independiente.

Para ajustar un modelo de regresión lineal simple, se deben realizar los siguientes pasos:

1. Recolectar los datos de la variable independiente y la variable dependiente.

2. Graficar los datos en un diagrama de dispersión para visualizar la relación entre las variables.

3. Calcular la correlación entre las variables para determinar la fuerza y dirección de la relación.
4. Estimar la ecuación de la línea recta que mejor se ajusta a los datos utilizando el método de mínimos cuadrados.
5. Evaluar la calidad del ajuste utilizando medidas como el coeficiente de determinación (R^2) y el error estándar de la estimación (SEE).
6. Utilizar la ecuación de la línea recta para predecir el valor de la variable dependiente a partir de la variable independiente.

Los modelos de regresión lineal simple son útiles para analizar la relación entre dos variables continuas y para predecir el valor de la variable dependiente a partir de la variable independiente. Sin embargo, es importante tener en cuenta que este modelo se basa en ciertas suposiciones, como la linealidad y la independencia de los errores, y que su aplicación debe ser cuidadosa y crítica.

Regresión múltiple

El modelo de regresión lineal múltiple es una técnica estadística utilizada para estudiar la relación entre una variable dependiente y dos o más variables independientes. El objetivo es encontrar la ecuación de una superficie de ajuste que mejor se adapte a los datos y que pueda utilizarse para predecir el valor de la variable dependiente a partir de las variables independientes.

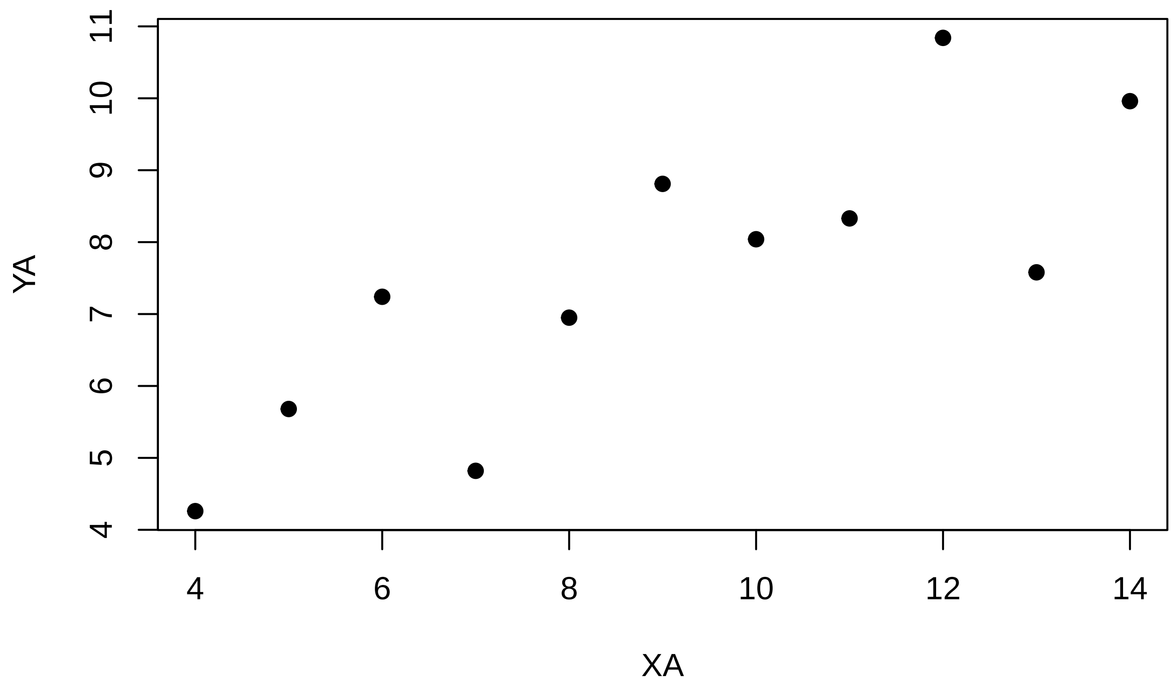
Para ajustar un modelo de regresión lineal múltiple, se deben realizar los siguientes pasos:

1. Recolectar los datos de la variable dependiente y las variables independientes.
2. Graficar los datos para visualizar la relación entre las variables.
3. Calcular la correlación entre las variables para determinar la fuerza y dirección de la relación.
4. Estimar la ecuación de la superficie de ajuste utilizando el método de mínimos cuadrados.
5. Evaluar la calidad del ajuste utilizando medidas como el coeficiente de determinación (R^2) y el error estándar de la estimación (SEE).
6. Utilizar la ecuación de la superficie de ajuste para predecir el valor de la variable dependiente a partir de las variables independientes.

Los modelos de regresión lineal múltiple son útiles para analizar la relación entre una variable dependiente y dos o más variables independientes y para predecir el valor de la variable dependiente a partir de las variables independientes. Sin embargo, al igual que con los modelos de regresión lineal simple, es importante tener en cuenta que este modelo se basa en ciertas suposiciones y que su aplicación debe ser cuidadosa y crítica.

Set A

```r
# Set A
par(mfrow=c(1,1))
plot(XA,YA,pch=19)
```



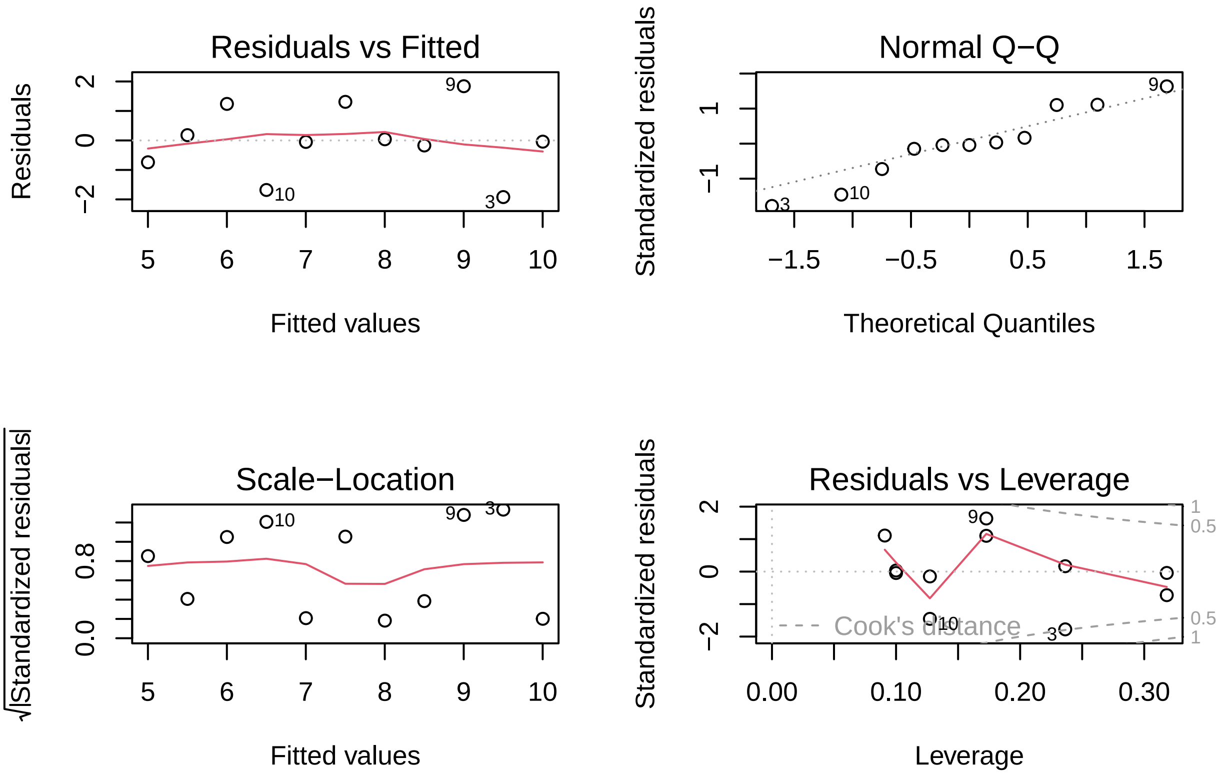

```
ma<-lm(YA~XA,data=anscombe)
ls() #New linear model: ma
```

```
## [1] "anscombe"     "last.warning" "ma"
```

```
summary(ma)
```

```
##
## Call:
## lm(formula = YA ~ XA, data = anscombe)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.92127 -0.45577 -0.04136  0.70941  1.83882
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.0001     1.1247   2.667  0.02573 *
## XA            0.5001     0.1179   4.241  0.00217 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.237 on 9 degrees of freedom
## Multiple R-squared:  0.6665, Adjusted R-squared:  0.6295
## F-statistic: 17.99 on 1 and 9 DF,  p-value: 0.00217
```

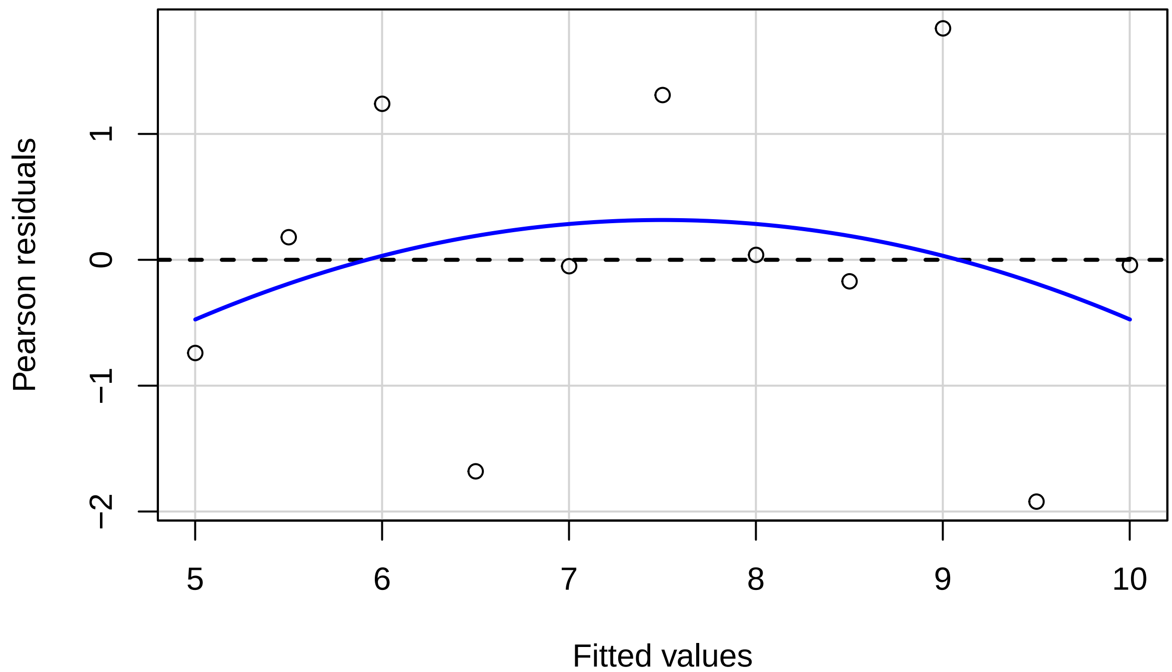
```r
# Default residual analysis:
par(mfrow=c(2,2))
plot(ma)
```




```r
# Metrics related to residuals:
library(car)
```

```
## Loading required package: carData
```

```r
par(mfrow=c(1,1))
residualPlot(ma)
```



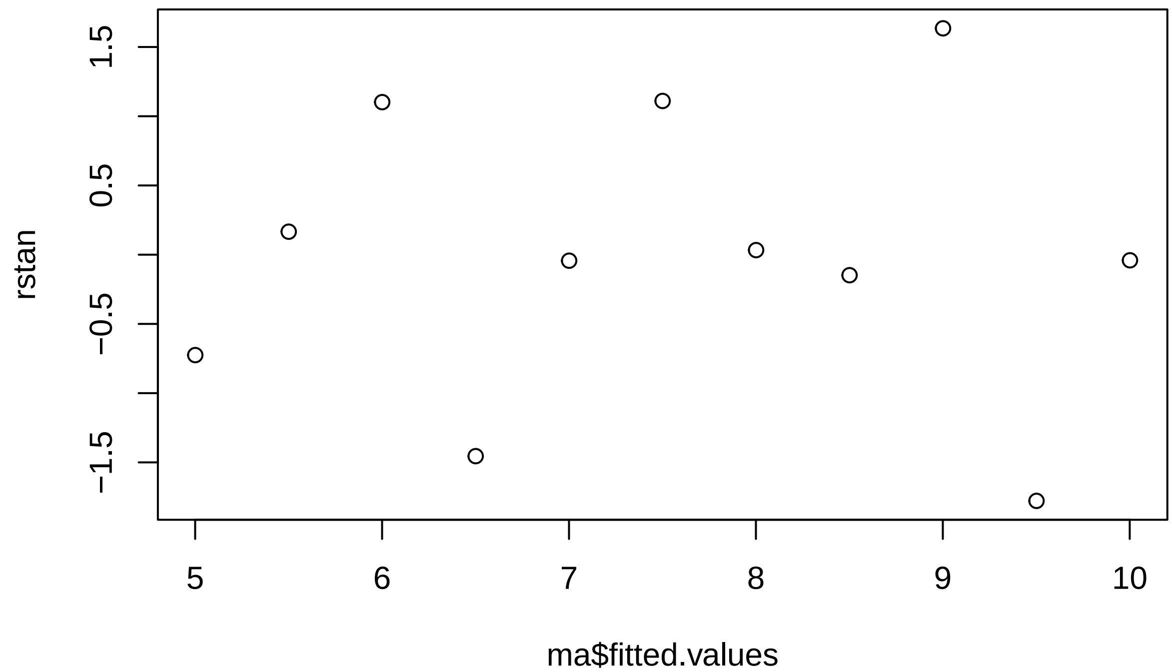

```
rstan <- rstandard(ma) #Standardized residuals
rstud <- rstudent(ma) #Studentized residuals
dcook <- cooks.distance(ma) #Cook distance for a posteriori influential observations
dcook
```

```
##            1            2            3            4            5            6
## 6.139788e-05 1.042467e-04 4.892093e-01 6.163700e-02 1.599342e-03 3.828995e-04
##            7            8            9           10           11
## 1.267565e-01 1.226999e-01 2.790296e-01 1.543412e-01 4.268011e-03
```

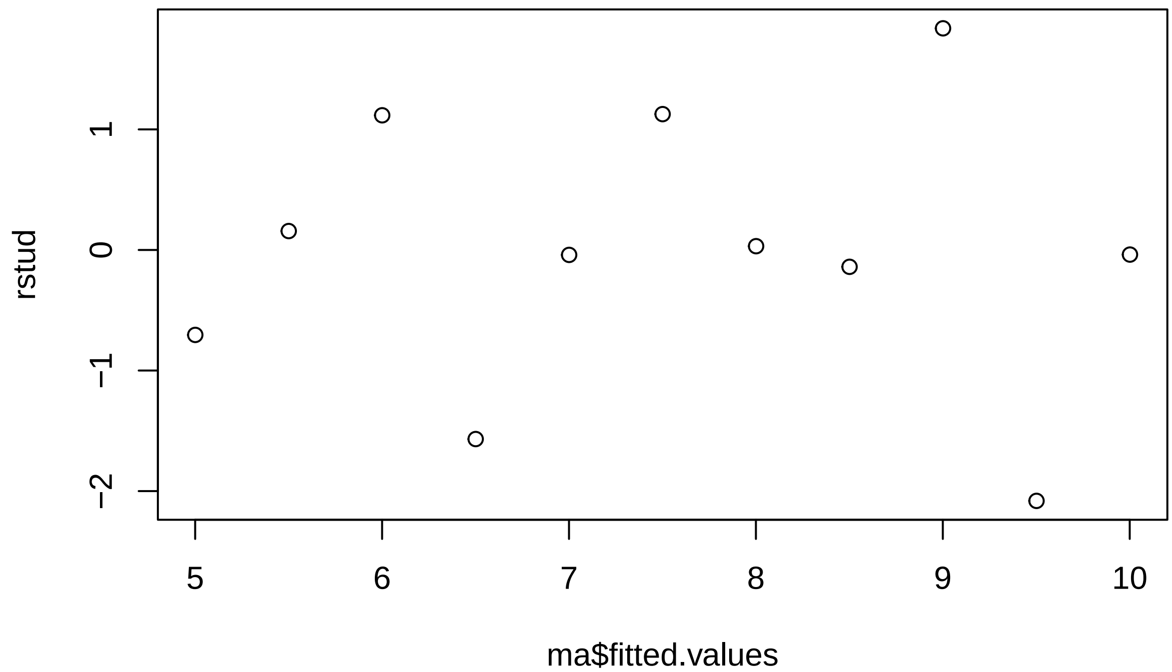
```
leverage <- hatvalues (ma) #Leverage of observations for a priori influential observations
leverage
```

```
##          1          2          3          4          5          6          7
## 0.10000000 0.10000000 0.23636364 0.09090909 0.12727273 0.31818182 0.17272727
##          8          9         10         11
## 0.31818182 0.17272727 0.12727273 0.23636364
```

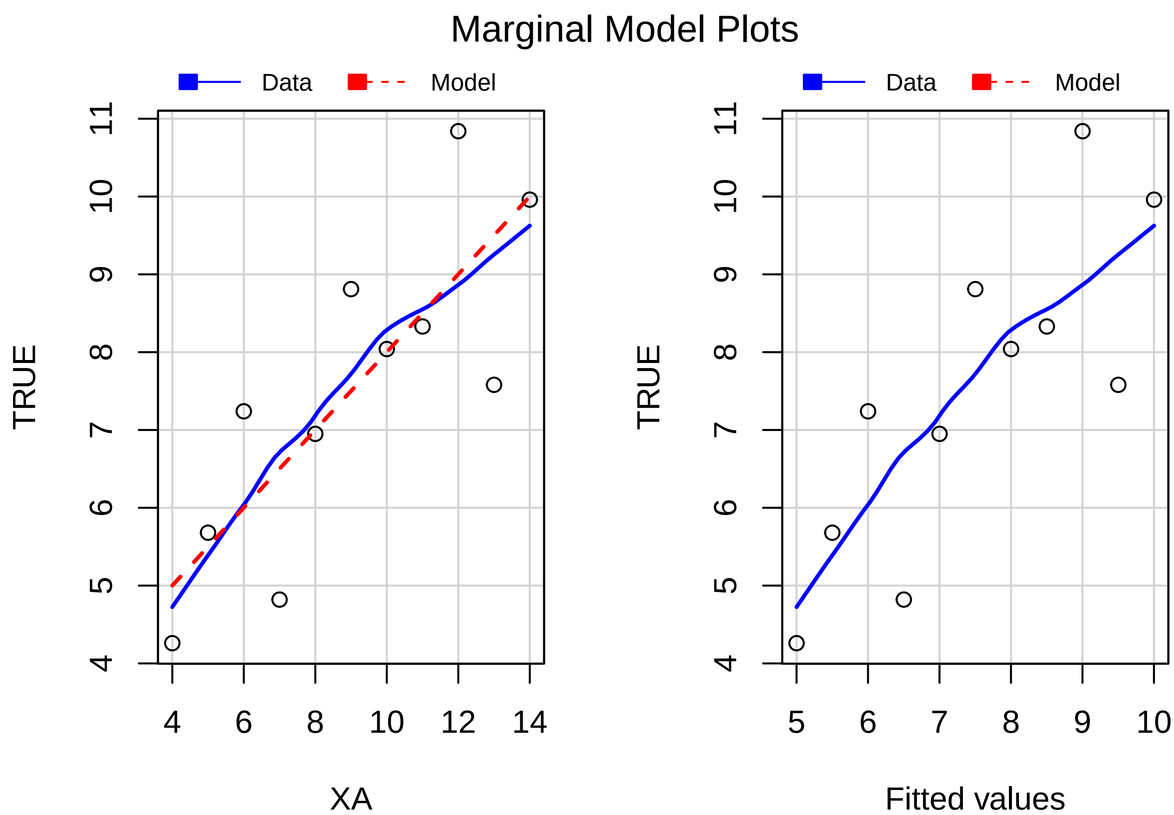
```
plot(ma$fitted.values, rstan) #Standardized residuals vs fitted values
```



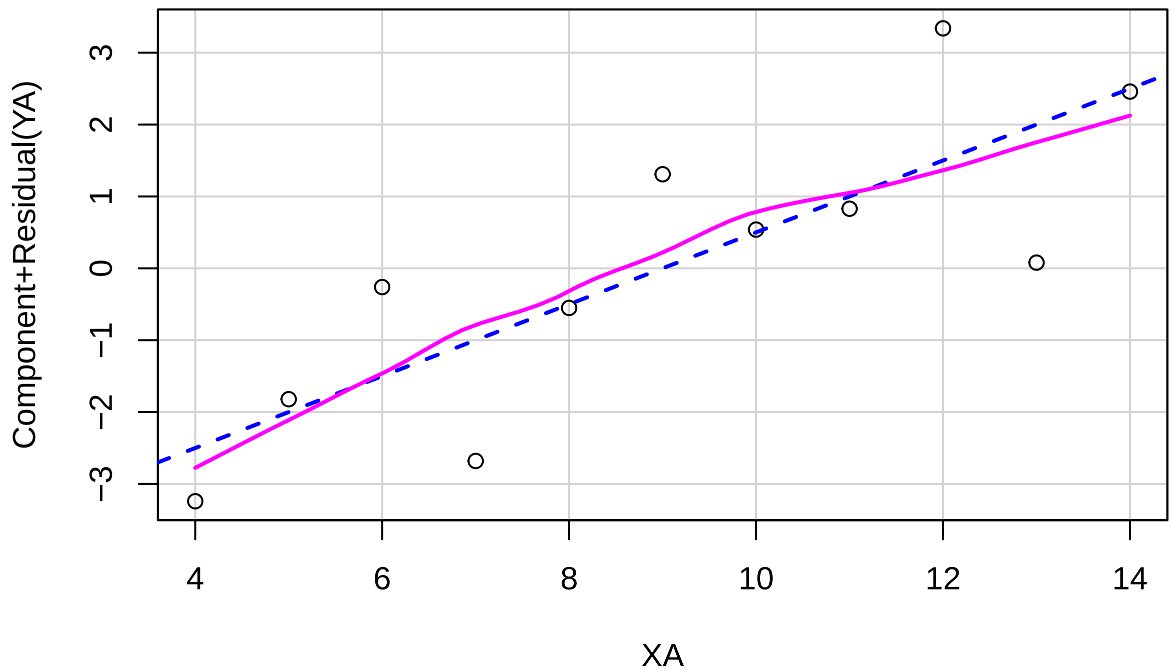

```
plot(ma$fitted.values, rstud) #Studentized residuals vs fitted values
```




```
marginalModelPlots(ma)
```



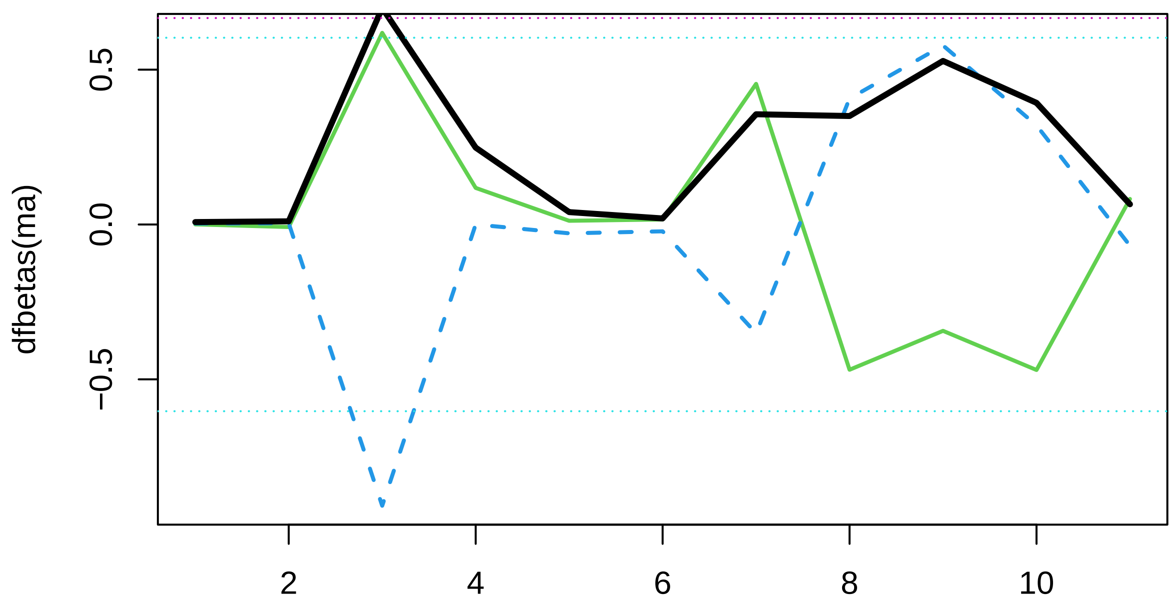

```
crPlot(ma, "XA") #Partial regression between XA and YA
```




```
# Used to check linearity between regressor and response

dfbetas(ma) #Beta coefficients without observation i
```

```
##      (Intercept)            XA
## 1   0.0003302365  3.150255e-03
## 2  -0.0081762052  4.105054e-03
## 3   0.6188789765 -9.082660e-01
## 4   0.1181207282 -4.100185e-18
## 5   0.0119658765 -2.853680e-02
## 6   0.0161820696 -2.205244e-02
## 7   0.4541481136 -3.512673e-01
## 8  -0.4690748609  4.067899e-01
## 9  -0.3434243650  5.781282e-01
## 10 -0.4698673679  3.201606e-01
## 11  0.0825027437 -6.843704e-02
```

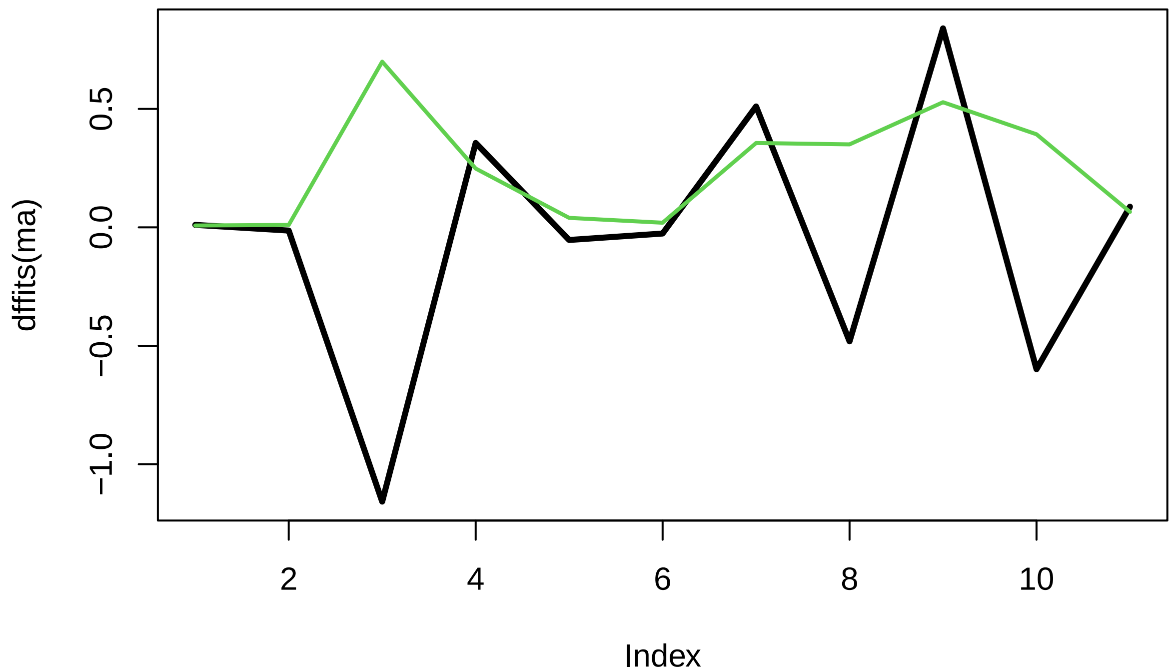
```
# Detection of influential data:
matplot(dfbetas(ma), type="l", col=3:4,lwd=2)
lines(sqrt(cooks.distance(ma)),col=1,lwd=3)
abline(h=2/sqrt(dim(anscombe)[1]), lty=3,lwd=1,col=5)
abline(h=-2/sqrt(dim(anscombe)[1]), lty=3,lwd=1,col=5)
abline(h=sqrt(4/(dim(anscombe)[1]-length(names(coef(ma))))), lty=3,lwd=1,col=6)
```




```r
llegenda<-c("Cook d", names(coef(ma)), "DFBETA Cut-off", "Ch-H Cut-off")
# legend(locator(n=1), legend=llegenda, col=1:length(llegenda), # lty=c(1,2,2,2,3,3), lwd=c(3,2,2,2,1,1

# Dffits: another metric for influential data:
par(mfrow=c(1,1))
dffits(ma)
```

```
##           1           2           3           4           5           6
##  0.01044821 -0.01361492 -1.15781655  0.35632543 -0.05338746 -0.02609280
##           7           8           9          10          11
##  0.51037958 -0.48132031  0.84000076 -0.59896572  0.08724046
```

```r
plot(dffits(ma),type="l",lwd=3)
pp=length(names(coef(ma)))
lines(sqrt(cooks.distance(ma)),col=3,lwd=2)
abline(h=2*(sqrt(pp/(nrow(ma)-pp))),lty=3,lwd=1,col=2)
abline(h=-2*(sqrt(pp/(nrow(ma)-pp))),lty=3,lwd=1,col=2)
```




```
llegenda<-c("DFFITS","DFFITS Cut-off","Cooks D")
# legend(locator(n=1),legend=llegenda,col=1:3,lty=c(1,3,1),lwd=c(3,1,2))

# AIC and BIC:
AIC(ma)
```

```
## [1] 39.68137
```

```
AIC(ma, k=log(nrow(anscombe))) #This is used to calculate BIC of a linear model
```

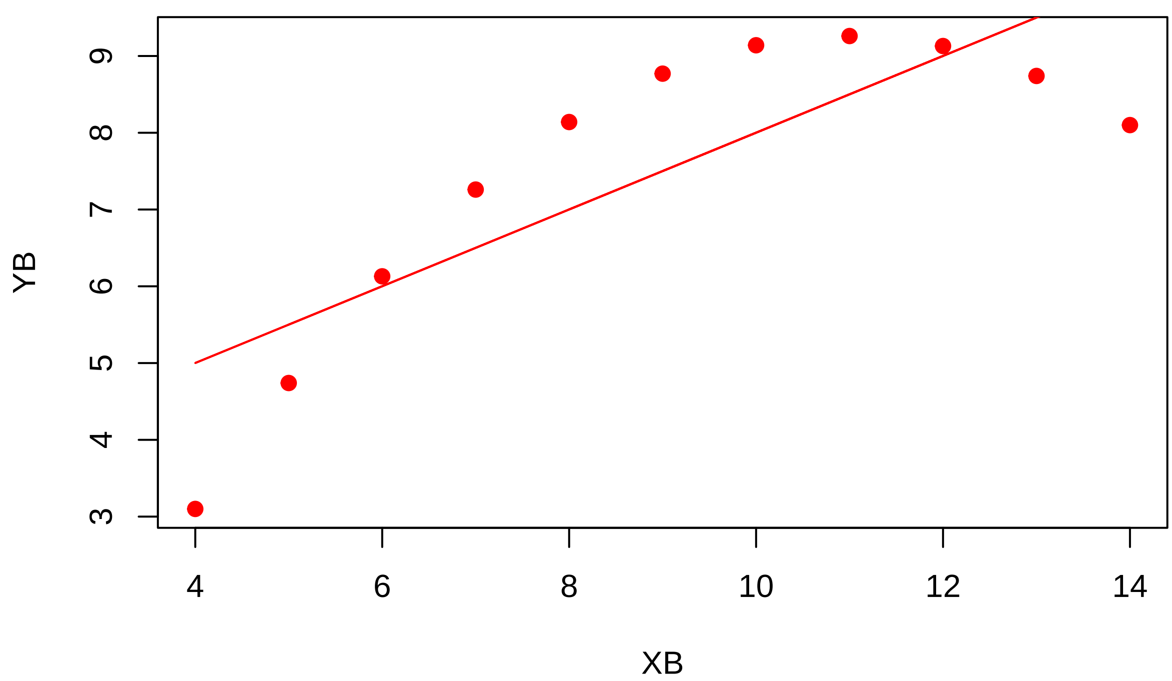
```
## [1] 40.87506
```

```
#Stepwise regression:
ma_0 <- lm(YA ~ 1, anscombe)
step(ma_0, ~XA, direction="forward",data=anscombe)
```

```
## Start:  AIC=16.55
## YA ~ 1
##
##        Df Sum of Sq    RSS     AIC
## + XA    1     27.51 13.763  6.4647
## <none>              41.273 16.5454
##
## Step:  AIC=6.46
## YA ~ XA
```

```
##
## Call:
## lm(formula = YA ~ XA, data = anscombe)
##
## Coefficients:
## (Intercept)           XA
##      3.0001       0.5001
```

```r
# Set B
par(mfrow=c(1,1))
plot(XB,YB,pch=19,col="red")
mb<-lm(YB~XB,data=anscombe)
lines(XB,fitted(mb),col="red")
```




```r
ls()
```

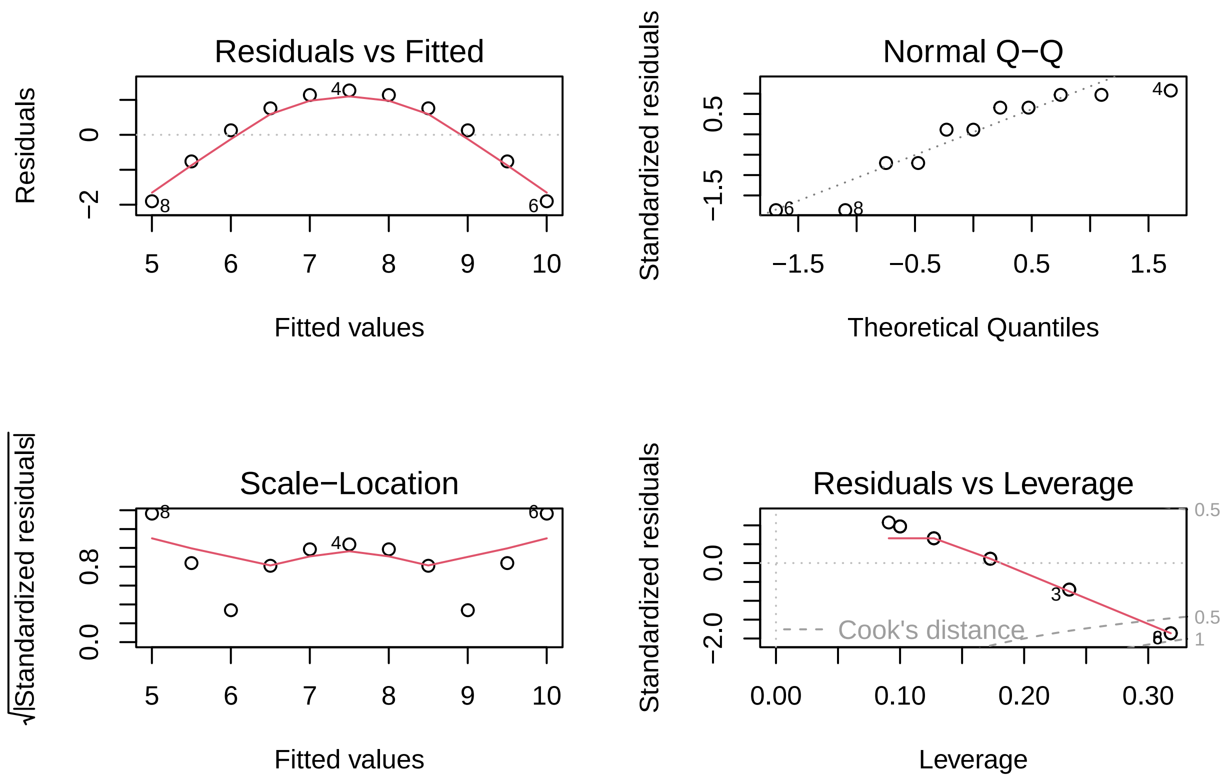
```
##  [1] "anscombe"     "dcook"        "last.warning" "leverage"     "llegenda"
##  [6] "ma"           "ma_0"         "mb"           "pp"           "rstan"
## [11] "rstud"
```

```r
summary(mb)
```

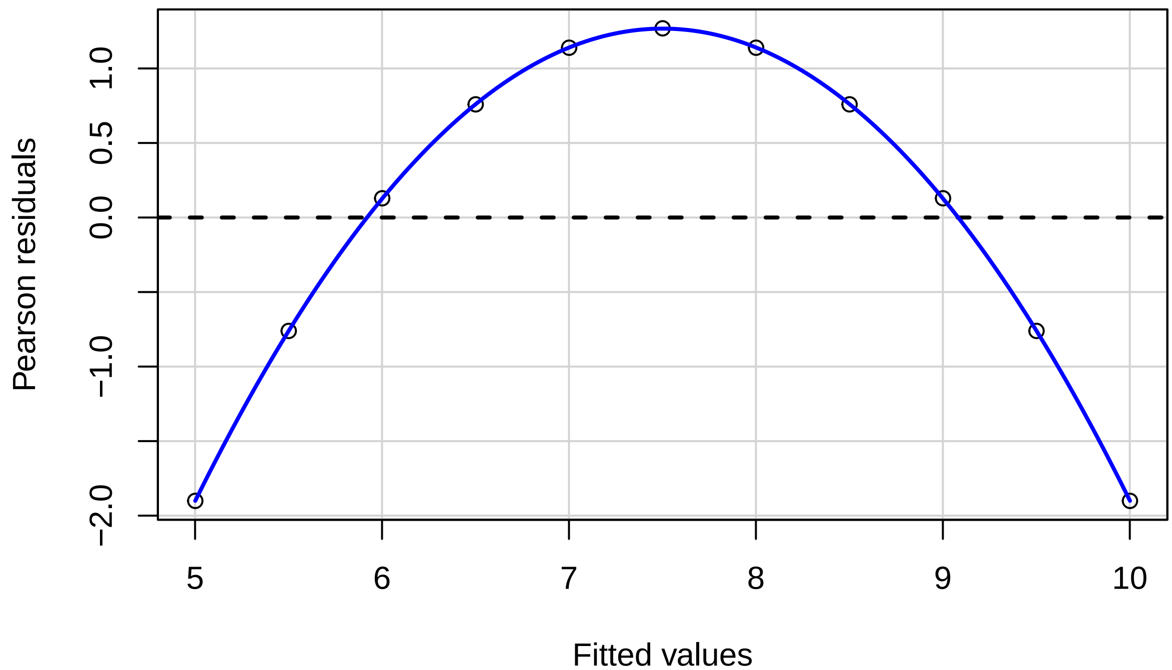
```
##
## Call:
```

```
## lm(formula = YB ~ XB, data = anscombe)
##
## Residuals:
##      Min       1Q  Median       3Q      Max
## -1.9009 -0.7609  0.1291  0.9491  1.2691
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    3.001      1.125   2.667  0.02576 *
## XB             0.500      0.118   4.239  0.00218 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.237 on 9 degrees of freedom
## Multiple R-squared:  0.6662, Adjusted R-squared:  0.6292
## F-statistic: 17.97 on 1 and 9 DF,  p-value: 0.002179
```

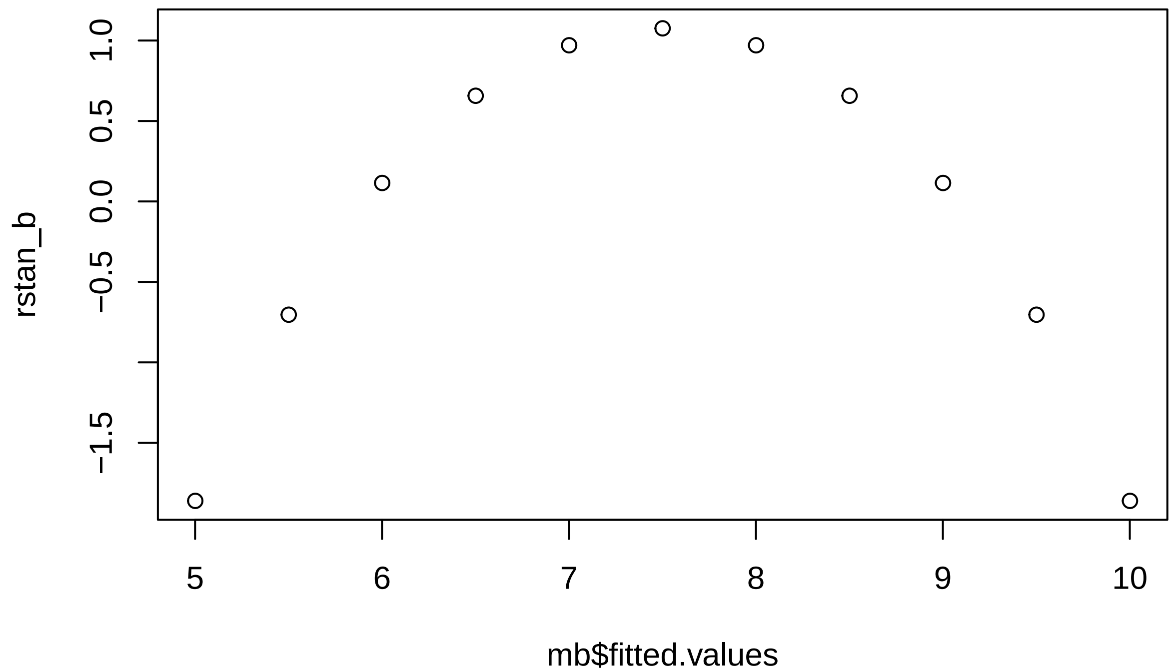
```
par(mfrow=c(2,2))
plot(mb)
```



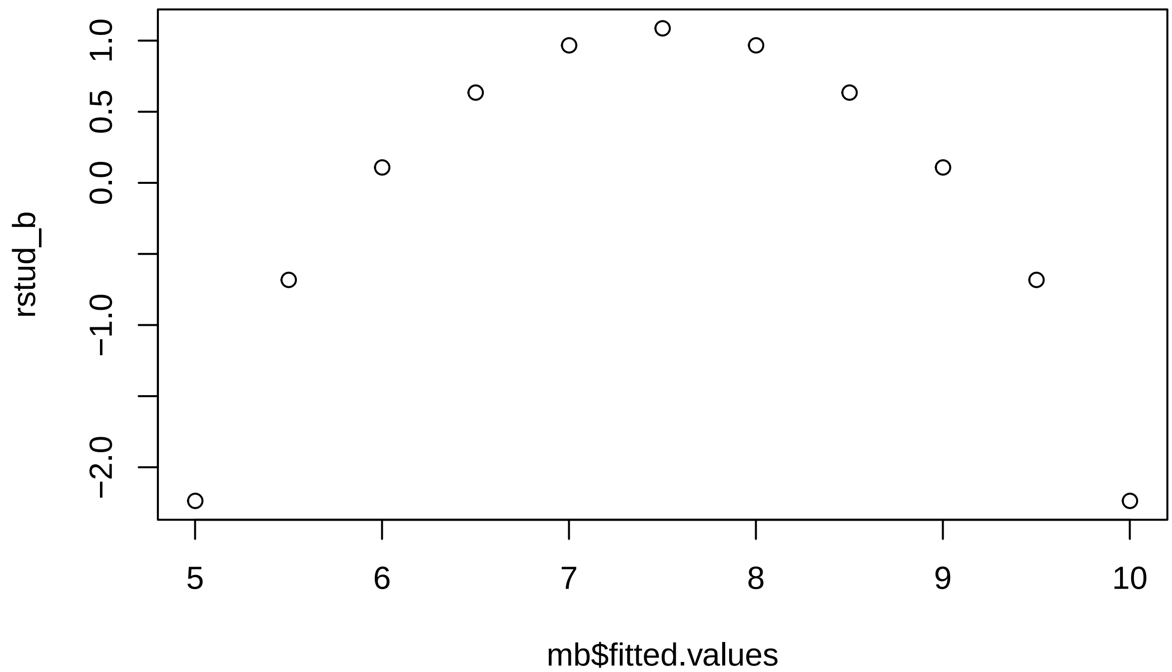

```
par(mfrow=c(1,1))
residualPlot(mb)
```



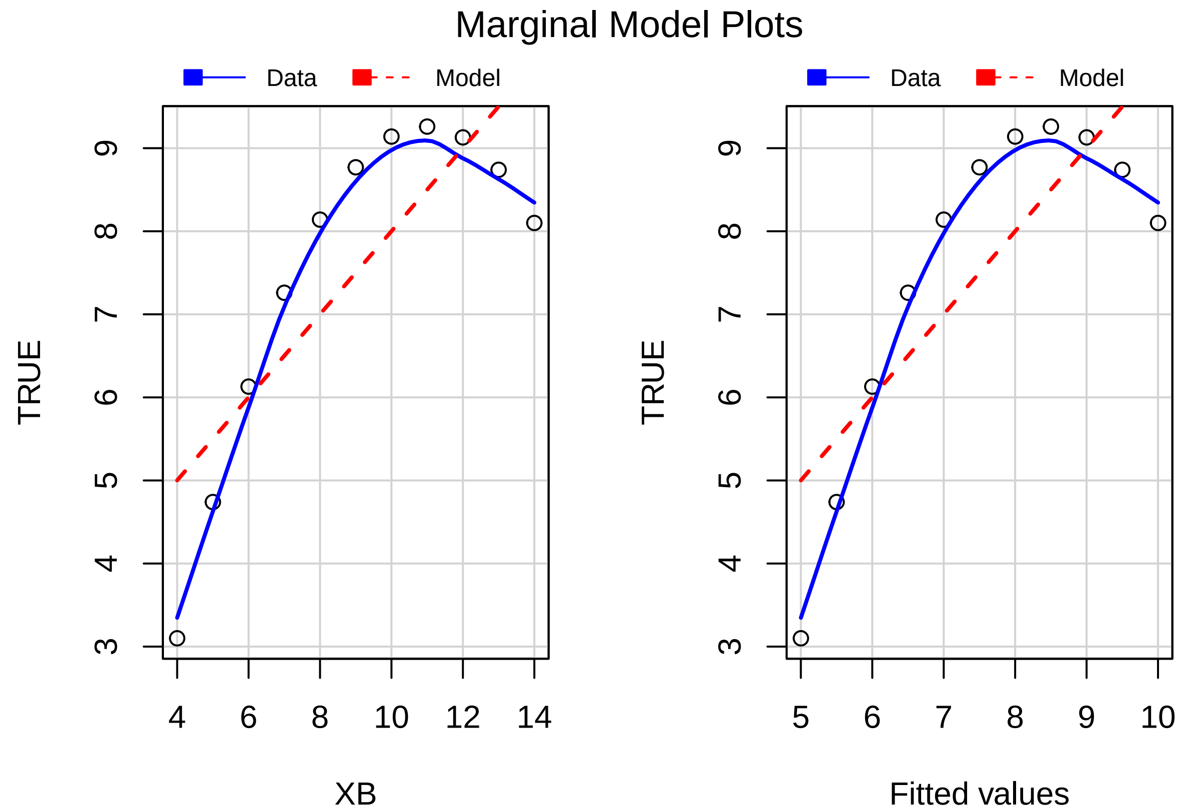

```
rstan_b <- rstandard(mb) #Standardized residuals
rstud_b <- rstudent(mb) #Studentized residuals
plot(mb$fitted.values, rstan_b) #Standardized residuals vs fitted values
```



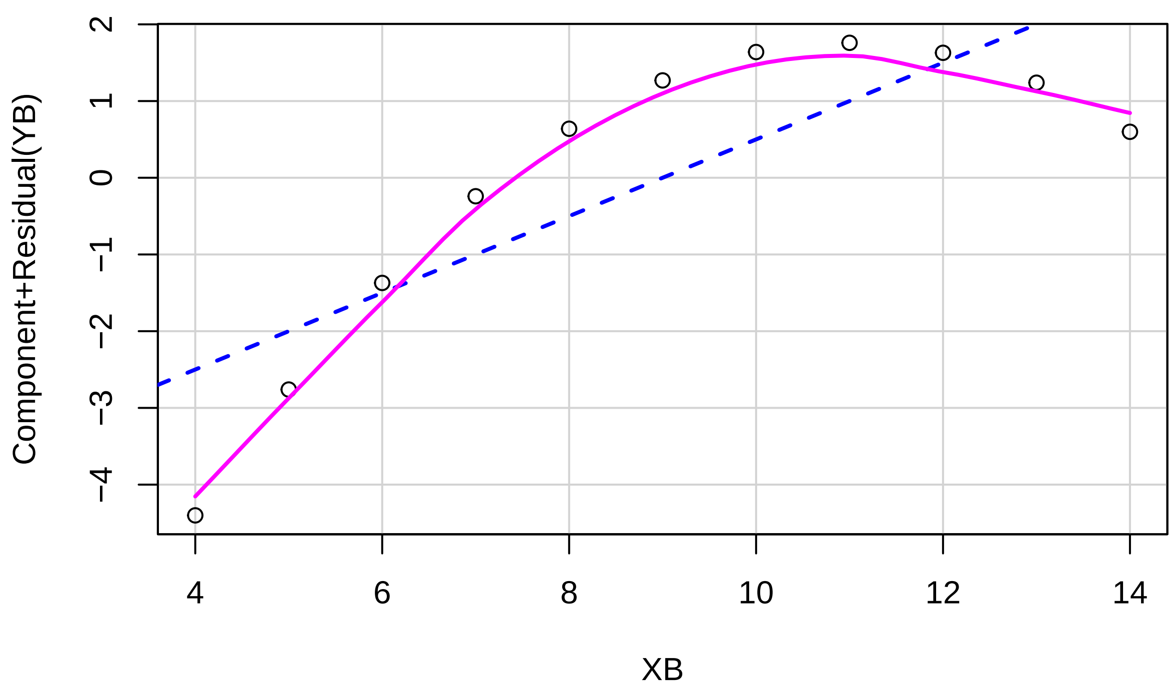

```
plot(mb$fitted.values, rstud_b) #Studentized residuals vs fitted values
```




```
marginalModelPlots(mb)
```



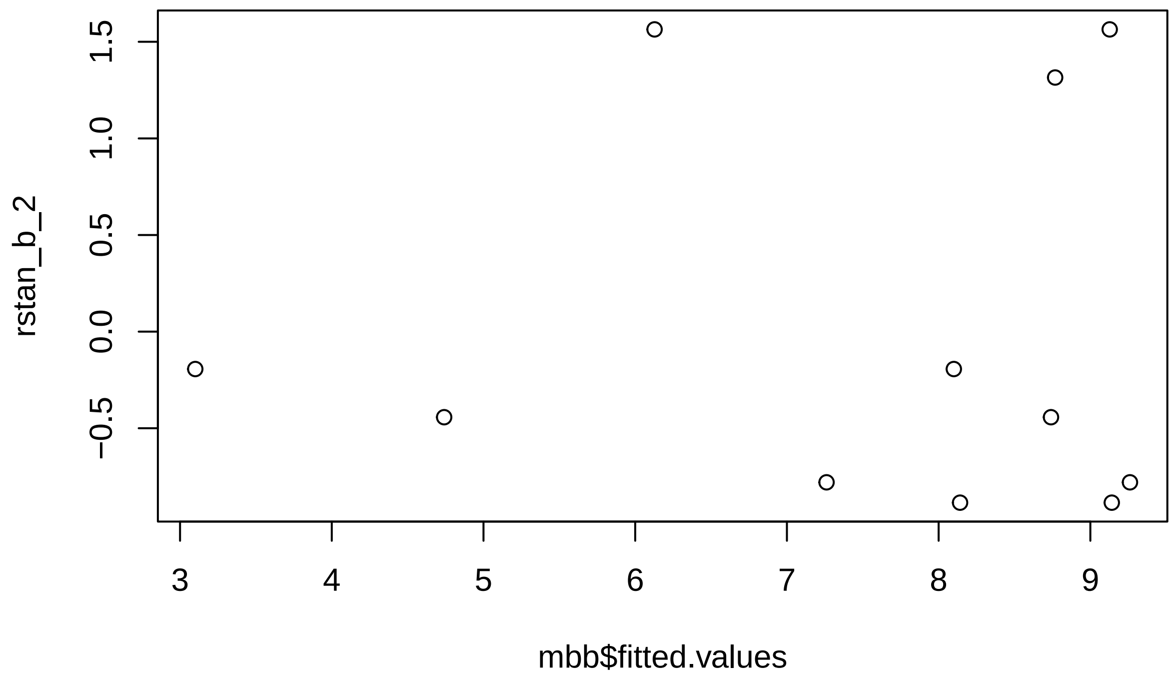

```
crPlot(mb, "XB") # Partial regression between XB and YB
```



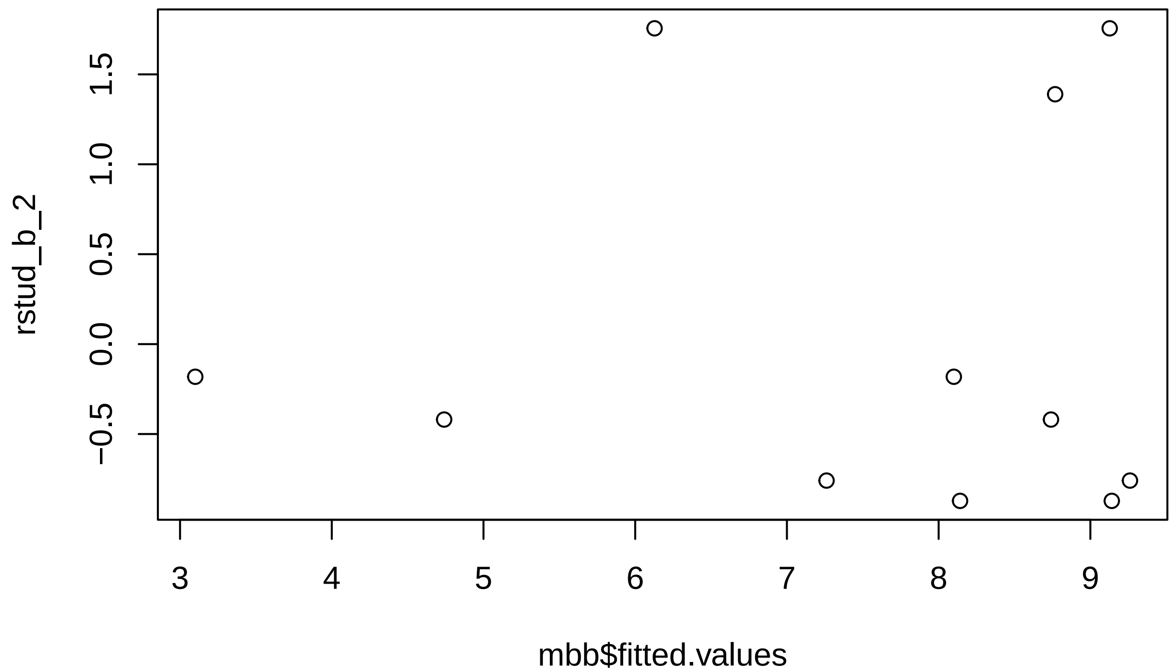

```
# Used to check linearity between regressor and response

mbb<-lm(YB~XB+I(XB^2),data=anscombe)
summary(mbb)
```

```
##
## Call:
## lm(formula = YB ~ XB + I(XB^2), data = anscombe)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0013287 -0.0011888 -0.0006294  0.0008741  0.0023776
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.9957343  0.0043299   -1385   <2e-16 ***
## XB           2.7808392  0.0010401    2674   <2e-16 ***
## I(XB^2)     -0.1267133  0.0000571   -2219   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.001672 on 8 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 7.378e+06 on 2 and 8 DF,  p-value: < 2.2e-16
```

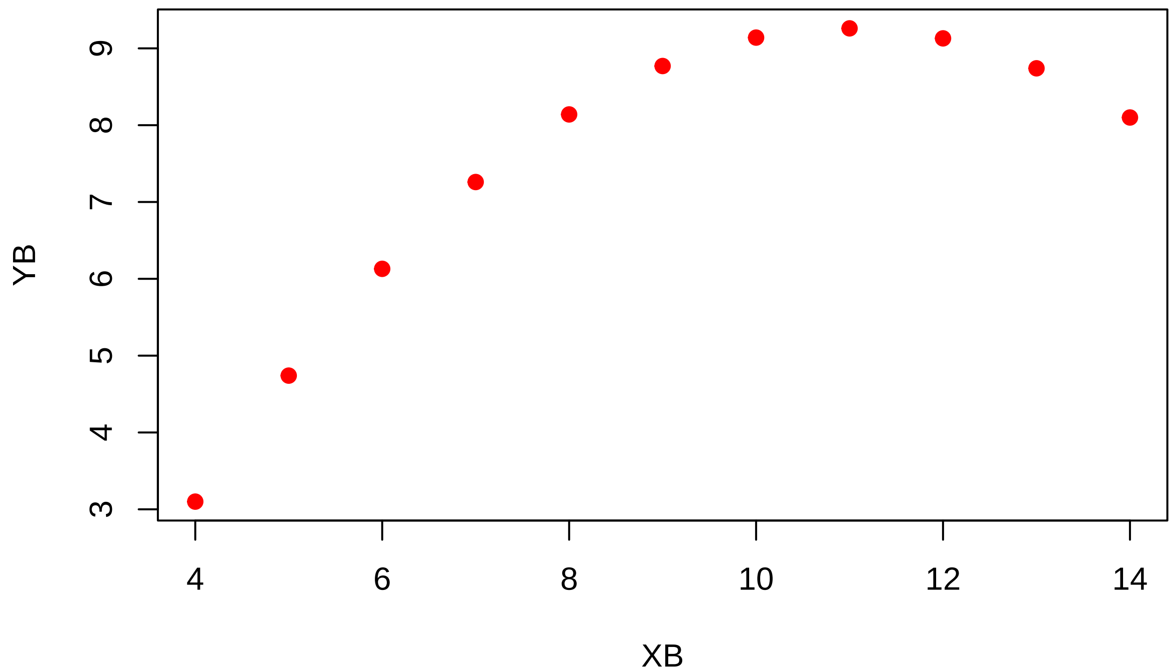
```r
rstan_b_2 <- rstandard(mbb) #Standardized residuals
rstud_b_2 <- rstudent(mbb) #Studentized residuals
plot(mbb$fitted.values, rstan_b_2) #Standardized residuals vs fitted values
```



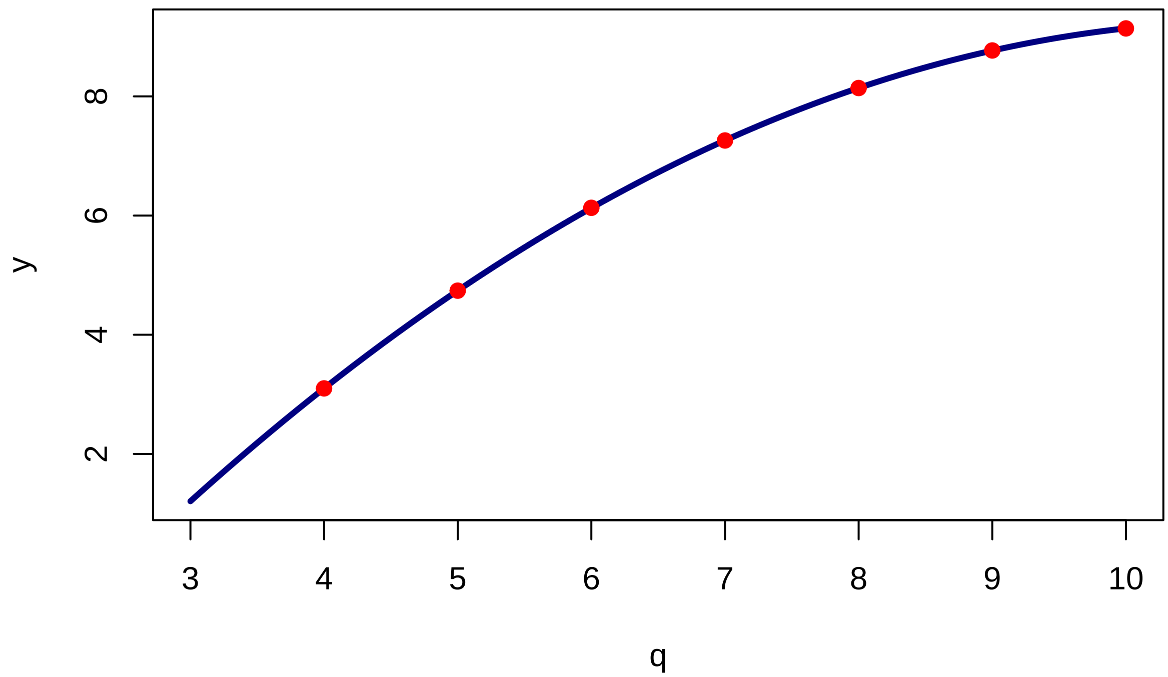

```r
plot(mbb$fitted.values, rstud_b_2) #Studentized residuals vs fitted values
```



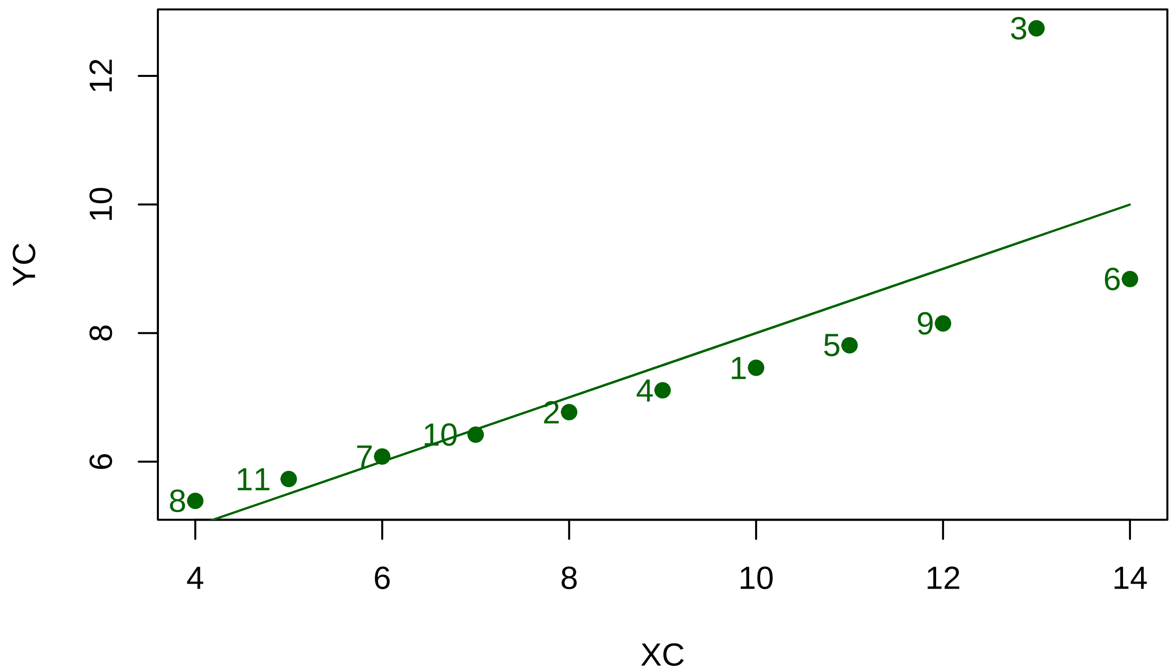

```
plot(YB~XB,pch=19,col="red")
```




```
q <- seq(3, 10, 0.01)
y <- -5.9957343 + 2.7808392*q -0.1267133*q^2
plot(q,y,type='l',col='navy', lwd=3)
points(YB~XB, col = "red", pch = 19)
```




```
# Set C
par(mfrow=c(1,1))
plot(XC,YC,pch=19,col="darkgreen")
text(XC,YC,label=row.names(anscombe),col="darkgreen",adj=1.5)
mc<-lm(YC~XC,data=anscombe)
lines(XC,fitted(mc),col="darkgreen")
```




```
ls()
```

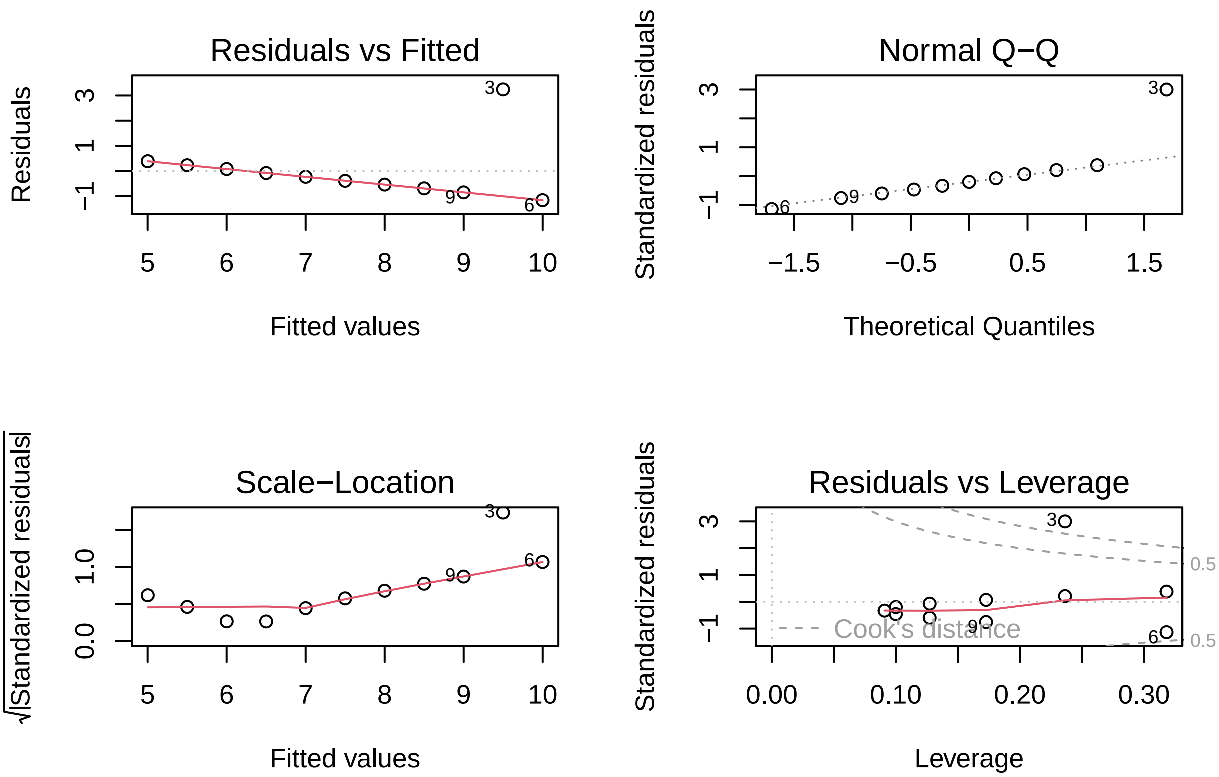
```
##  [1] "anscombe"     "dcook"        "last.warning" "leverage"     "llegenda"
##  [6] "ma"           "ma_0"         "mb"           "mbb"          "mc"
## [11] "pp"           "q"            "rstan"        "rstan_b"      "rstan_b_2"
## [16] "rstud"        "rstud_b"      "rstud_b_2"    "y"
```

```
summary(mc)
```

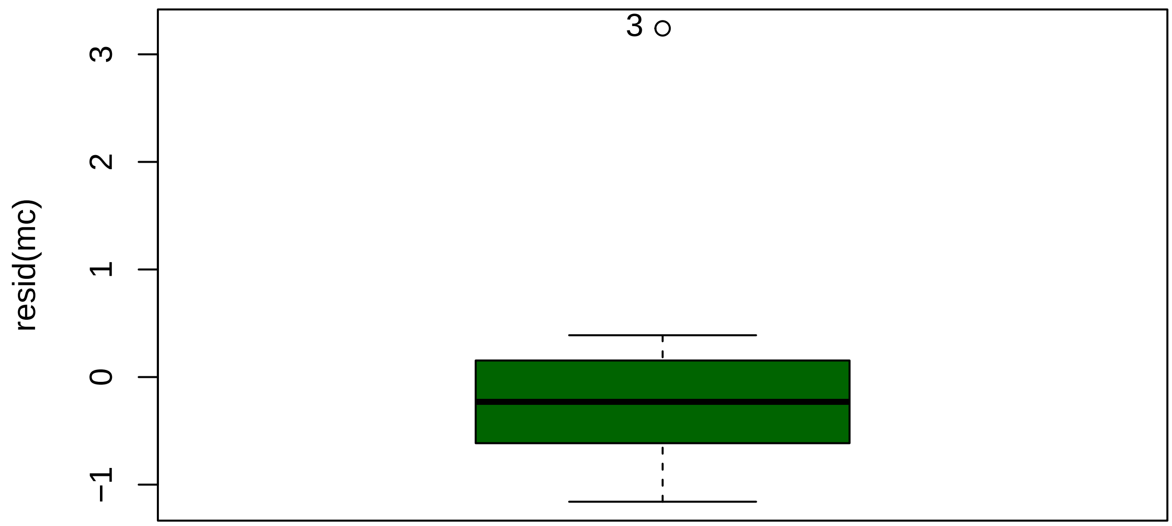
```
##
## Call:
## lm(formula = YC ~ XC, data = anscombe)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.1586 -0.6146 -0.2303  0.1540  3.2411
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.0025     1.1245   2.670  0.02562 *
## XC            0.4997     0.1179   4.239  0.00218 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.236 on 9 degrees of freedom
```

```
## Multiple R-squared:  0.6663, Adjusted R-squared:  0.6292
## F-statistic: 17.97 on 1 and 9 DF,  p-value: 0.002176
```

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




```
par(mfrow=c(1,1))
Boxplot(resid(mc),col="darkgreen")
```

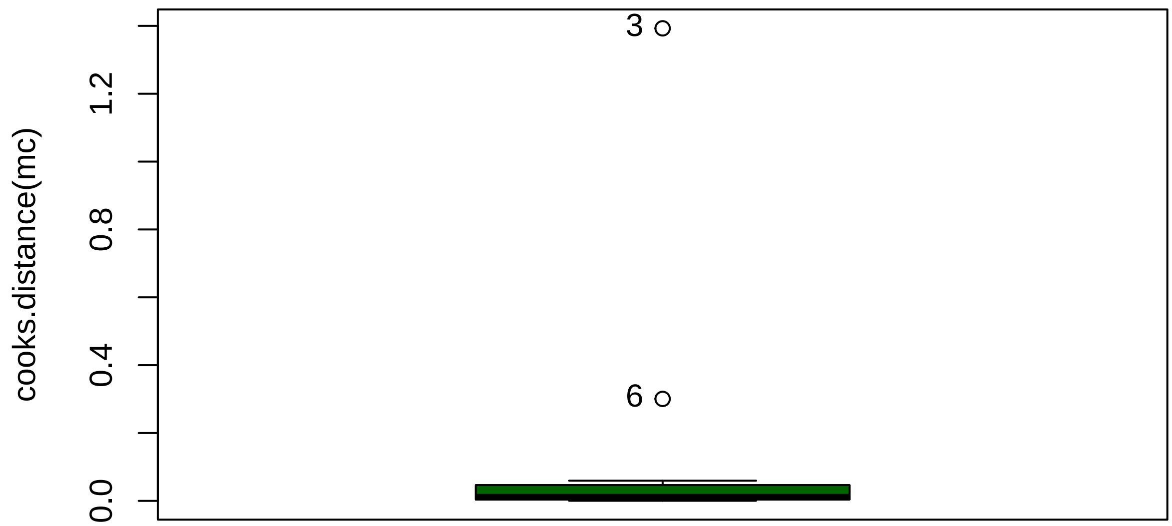



```
## [1] 3
```

```r
cooks.distance(mc)
```

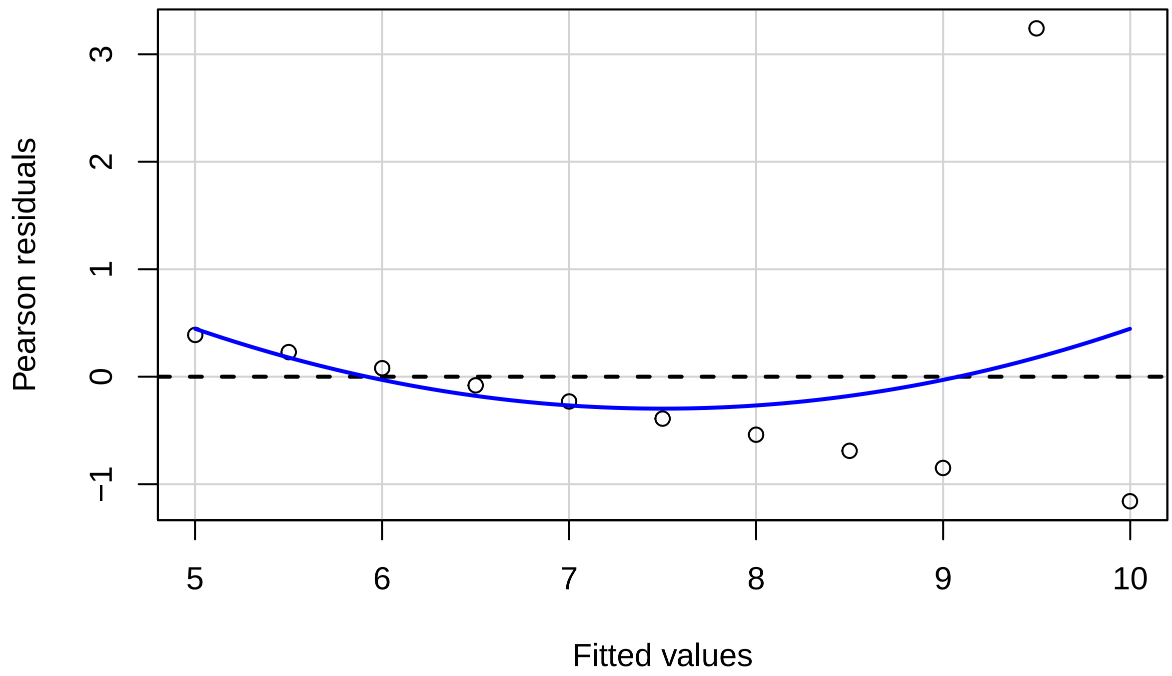
```
##            1            2            3            4            5            6 
## 0.0117646226 0.0021414813 1.3928494503 0.0054731354 0.0259838693 0.3005708107 
##            7            8            9           10           11 
## 0.0005176411 0.0338173336 0.0595359333 0.0003546293 0.0069478084
```

```r
Boxplot(cooks.distance(mc),col="darkgreen")
```




```
## [1] 3 6
```

```
residualPlot(mc)
```



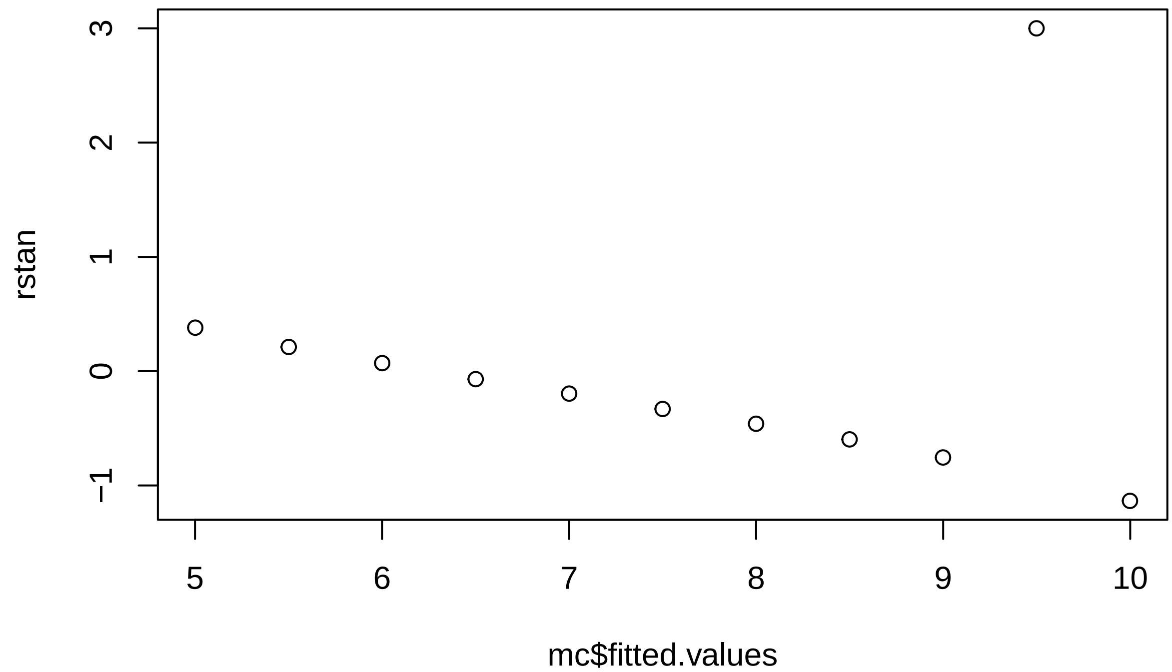

```
rstan <- rstandard(mc) #Standardized residuals
rstud <- rstudent(mc) #Studentized residuals
dcook <- cooks.distance(mc) #Cook distance
dcook
```

```
##            1            2            3            4            5            6
## 0.0117646226 0.0021414813 1.3928494503 0.0054731354 0.0259838693 0.3005708107
##            7            8            9           10           11
## 0.0005176411 0.0338173336 0.0595359333 0.0003546293 0.0069478084
```

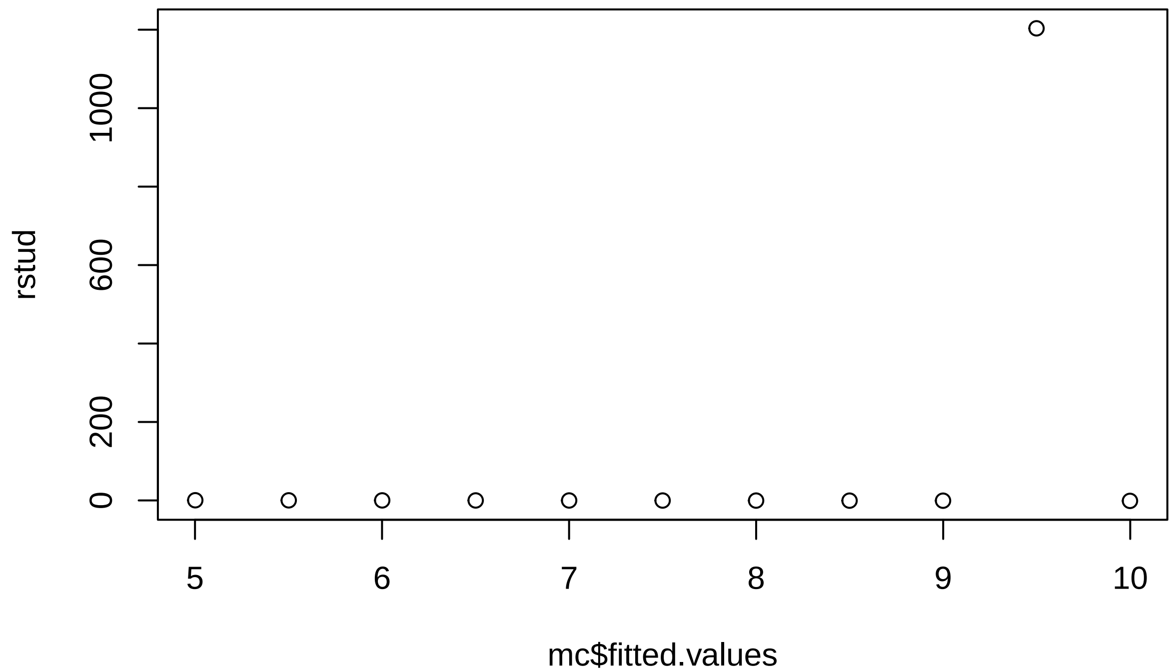
```
leverage <- hatvalues (mc) #Leverage of observations: a priori influential observations
leverage
```

```
##          1          2          3          4          5          6          7
## 0.10000000 0.10000000 0.23636364 0.09090909 0.12727273 0.31818182 0.17272727
##          8          9         10         11
## 0.31818182 0.17272727 0.12727273 0.23636364
```

```
plot(mc$fitted.values, rstan) #Standardized residuals vs fitted values
```




```r
plot(mc$fitted.values, rstud) #Studentized residuals vs fitted values
```




```r
dfbetas(mc) #Beta coefficients without observation i
```

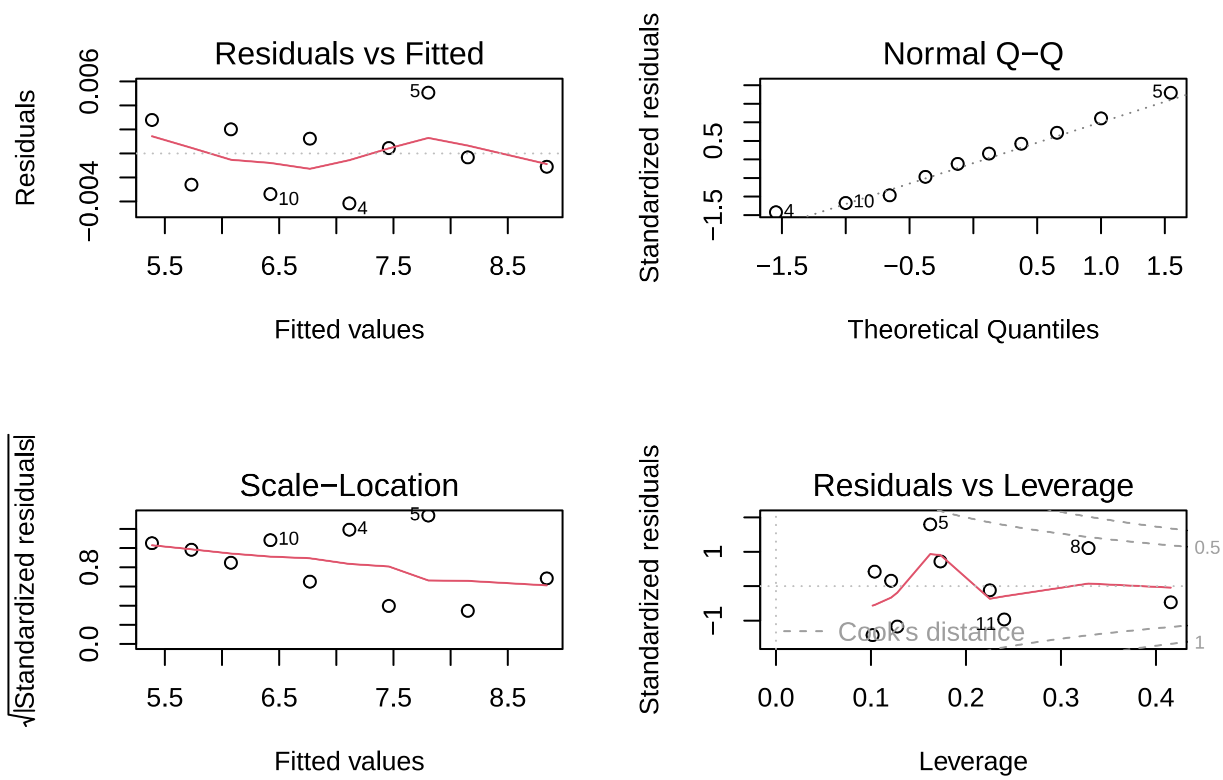
```
##      (Intercept)            XC
## 1  -4.625738e-03 -4.412673e-02
## 2  -3.713338e-02  1.864368e-02
## 3  -3.579096e+02  5.252677e+02
## 4  -3.289981e-02  1.142012e-18
## 5   4.915510e-02 -1.172274e-01
## 6   4.897424e-01 -6.674064e-01
## 7   2.700082e-02 -2.088417e-02
## 8   2.409027e-01 -2.089150e-01
## 9   1.374342e-01 -2.313597e-01
## 10 -1.970229e-02  1.342485e-02
## 11  1.053656e-01 -8.740210e-02
```

```r
mcc<-lm(YC~XC,data=anscombe[-3,])
summary(mcc)
```

```
##
## Call:
## lm(formula = YC ~ XC, data = anscombe[-3, ])
##
## Residuals:
##        Min          1Q      Median          3Q         Max
```

```
## -0.0041558 -0.0022240  0.0000649  0.0018182  0.0050649
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.0056494  0.0029242    1370   <2e-16 ***
## XC          0.3453896  0.0003206    1077   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.003082 on 8 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 1.161e+06 on 1 and 8 DF,  p-value: < 2.2e-16
```

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

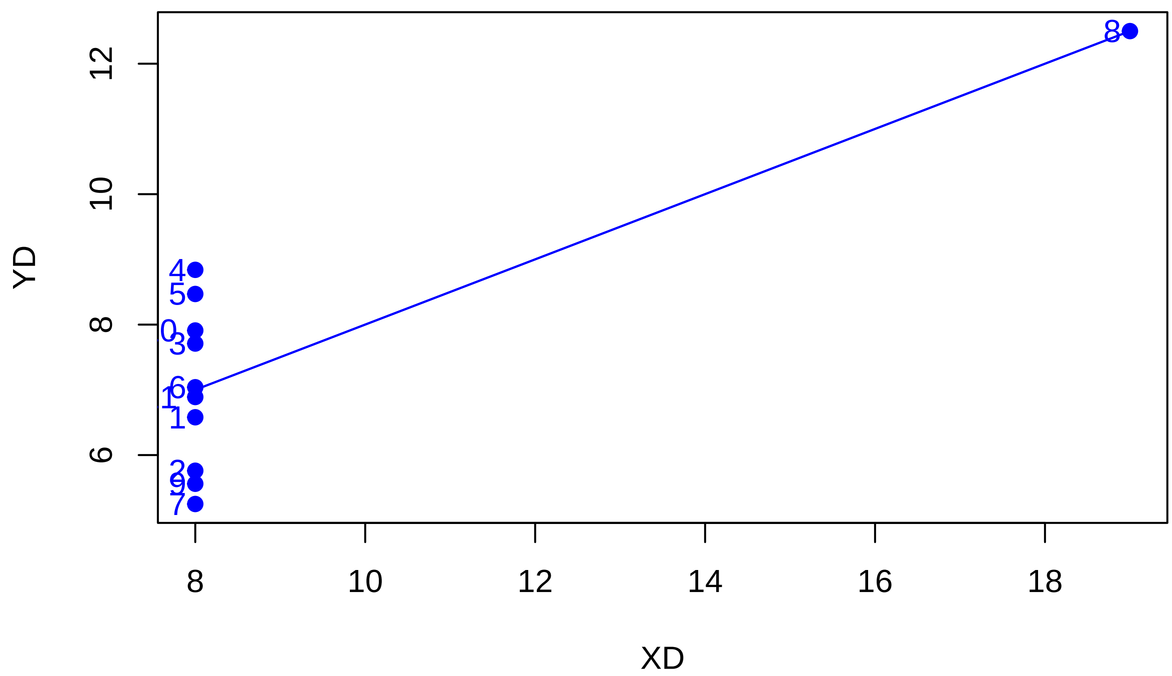



```
# Set D
par(mfrow=c(1,1))
plot(XD,YD,pch=19,col="blue")
text(XD,YD,label=row.names(anscombe),col="blue",adj=1.5)
md<-lm(YD~XD,data=anscombe)
summary(md)
```

```
##
## Call:
```

```
## lm(formula = YD ~ XD, data = anscombe)
##
## Residuals:
##     Min      1Q Median      3Q     Max
## -1.751 -0.831  0.000  0.809  1.839
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.0017     1.1239   2.671  0.02559 *
## XD            0.4999     0.1178   4.243  0.00216 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.236 on 9 degrees of freedom
## Multiple R-squared:  0.6667, Adjusted R-squared:  0.6297
## F-statistic:    18 on 1 and 9 DF,  p-value: 0.002165
```

```
lines(XD,fitted(md),col="blue")
```




```
ls()
```

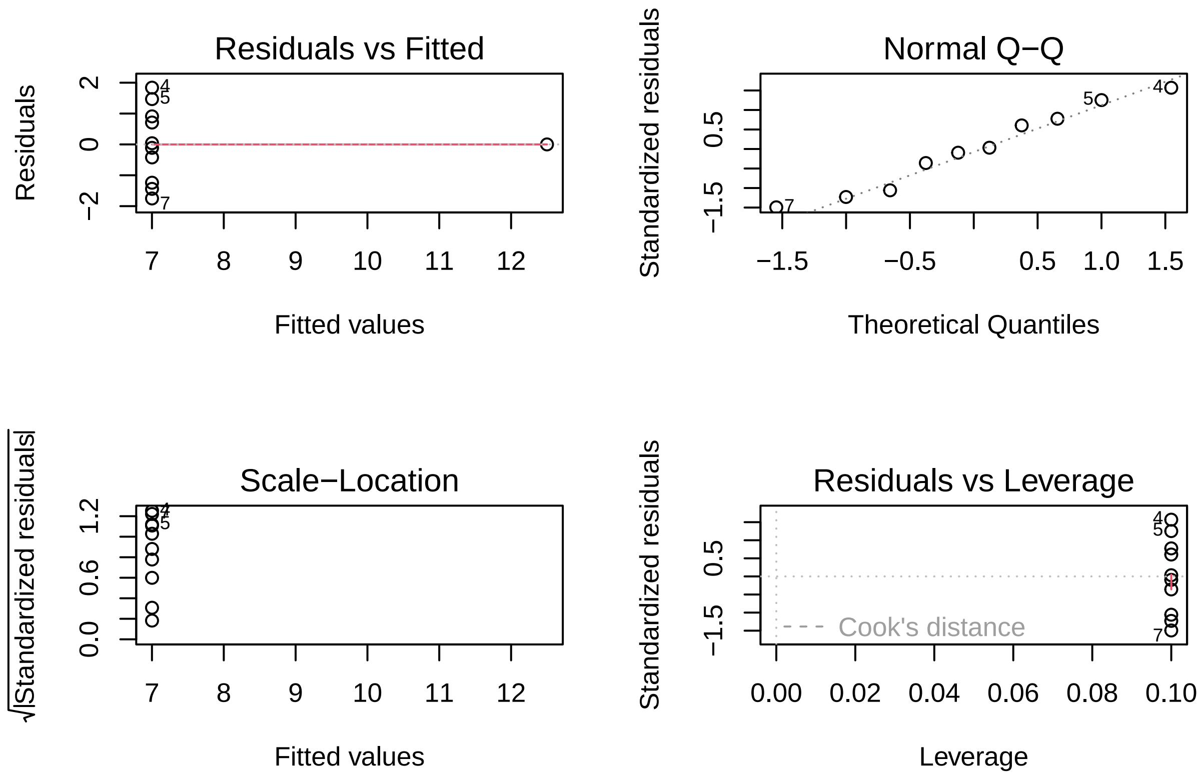
```
##  [1] "anscombe"     "dcook"        "last.warning" "leverage"     "llegenda"
##  [6] "ma"           "ma_0"         "mb"           "mbb"          "mc"
## [11] "mcc"          "md"           "pp"           "q"            "rstan"
## [16] "rstan_b"      "rstan_b_2"    "rstud"        "rstud_b"      "rstud_b_2"
## [21] "y"
```

```r
summary(md)
```

```
##
## Call:
## lm(formula = YD ~ XD, data = anscombe)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -1.751 -0.831  0.000  0.809  1.839
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.0017     1.1239   2.671  0.02559 *
## XD            0.4999     0.1178   4.243  0.00216 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.236 on 9 degrees of freedom
## Multiple R-squared:  0.6667, Adjusted R-squared:  0.6297
## F-statistic:    18 on 1 and 9 DF,  p-value: 0.002165
```

```r
par(mfrow=c(2,2))
plot(md)
```

```
## Warning: not plotting observations with leverage one:
##   8
```




```
resid(md)
```

```
##             1             2             3             4             5
## -4.210000e-01 -1.241000e+00  7.090000e-01  1.839000e+00  1.469000e+00
##             6             7             8             9            10
##  3.900000e-02 -1.751000e+00  1.110223e-16 -1.441000e+00  9.090000e-01
##            11
## -1.110000e-01
```

```
cooks.distance(md)
```

```
##            1            2            3            4            5            6
## 7.165166e-03 6.225950e-02 2.032144e-02 1.367179e-01 8.723799e-02 6.148813e-05
##            7            8            9           10           11
## 1.239465e-01          NaN 8.394407e-02 3.340334e-02 4.980902e-04
```

```
hatvalues(md)
```

```
##   1   2   3   4   5   6   7   8   9  10  11
## 0.1 0.1 0.1 0.1 0.1 0.1 0.1 1.0 0.1 0.1 0.1
```

```
# Better approximation:
mdd<-lm(YD~XD,data=anscombe[-8,])
summary(mdd)
```

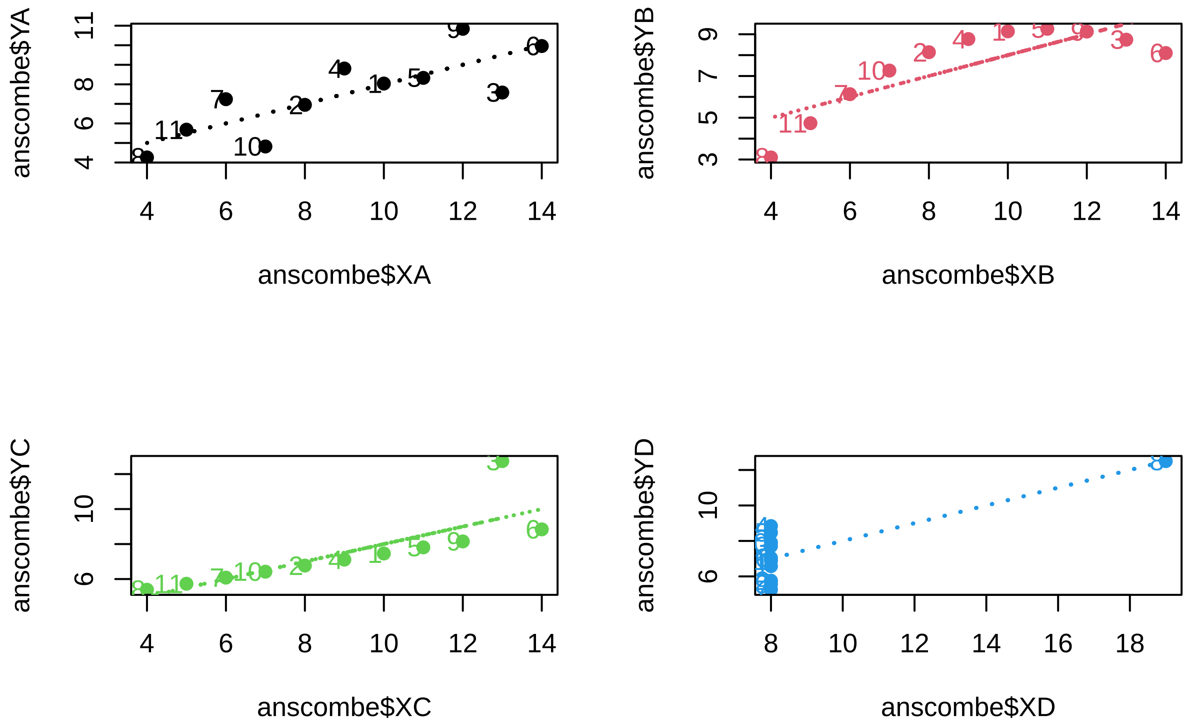
```
##
## Call:
## lm(formula = YD ~ XD, data = anscombe[-8, ])
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -1.751 -1.036 -0.036  0.859  1.839
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   7.0010     0.3908   17.92 2.39e-08 ***
## XD                NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.236 on 9 degrees of freedom
```

```
mean(YD[-8])
```

```
## [1] 7.001
```

All in all:

```
### Prospecció dels jocs de dades A,B,C,D
par(mfrow=c(2,2))
anscombe.lmA <- lm(anscombe$YA ~ anscombe$XA, data=anscombe)
plot(anscombe$XA, anscombe$YA,pch=19,col=1)
lines(anscombe$XA,anscombe.lmA$fitted.values, col=1, lty=3,lwd=2)
text(x=anscombe$XA,y=anscombe$YA,labels=row.names(anscombe),adj=1.1, col=1)

anscombe.lmB <- lm(anscombe$YB ~ anscombe$XB, data=anscombe)
plot(anscombe$XB, anscombe$YB,pch=19,col=2)
lines(anscombe$XB,anscombe.lmB$fitted.values, col=2, lty=3,lwd=2)
text(x=anscombe$XB,y=anscombe$YB,labels=row.names(anscombe),adj=1.1, col=2)

anscombe.lmC <- lm(anscombe$YC ~ anscombe$XC, data=anscombe)
plot(anscombe$XC, anscombe$YC,pch=19,col=3)
lines(anscombe$XC,anscombe.lmC$fitted.values, col=3, lty=3,lwd=2)
text(x=anscombe$XC,y=anscombe$YC,labels=row.names(anscombe),adj=1.1, col=3)

anscombe.lmD <- lm(anscombe$YD ~ anscombe$XD, data=anscombe)
plot(anscombe$XD, anscombe$YD,pch=19,col=4)
lines(anscombe$XD,anscombe.lmD$fitted.values, col=4, lty=3,lwd=2)
text(x=anscombe$XD,y=anscombe$YD,labels=row.names(anscombe),adj=1.1, col=4)
```


anscombe$YA
anscombe$XA
anscombe$YB
anscombe$XB
anscombe$YC
anscombe$XC
anscombe$YD
anscombe$XD