General Linear Model

F.A. BarriosInstituto de Neurobiología UNAM

2020-12-04

```
library(tidyverse)
library(multcomp)
library(car)
library(emmeans)
library(HSAUR2)
library(wordcloud)

setwd("~/Dropbox/Fdo/ClaseStats/RegressionClass/RegressionR_code")
# To set the working directory at the user dir
hers <- read_csv("DataRegressBook/Chap3/hersdata.csv")
```

General Linear Model GLM (Modelo lineal General)

Linear Regression

The term "regression" was introduced by Francis Galton (Darwin's nephew) during the XIX century to describe a biological phenomenon. The heights of the descendants of tall ancestors have the tendency to "return", come back, to the normal average high in the population, known as the regression to the media. (Mr. Galton was an Eugenics supporter)

Examples for "simple" linear regression

The general equation for the straight line is $y = mx + b_0$, this form is the "slope, intersection form". The slope is the rate of change the gives the change in y for a unit change in x. Remember that the slope formula for two pair of points (x_1, y_1) and (x_2, y_2) is:

$$m = \frac{(y_2 - y_1)}{(x_2 - x_1)}$$

```
setwd("~/Dropbox/Fdo/ClaseStats/RegressionClass/RegressionR_code")
# Changing wd to load the data file
Exa9.3 = read.csv(file="DataOther/EXA_C09_S03_01.csv", header=TRUE)
names(Exa9.3)
```

```
[1] "SUBJ" "X"    "Y"
```

```
plot(Exa9.3$Y ~ Exa9.3$X, pch = 20)
Ybar=mean(Exa9.3$Y)
Xbar=mean(Exa9.3$X)
abline(h=Ybar, col = 2, lty = 2)
abline(v=Xbar, col = 2, lty = 2)
Lin9.3 = lm(Y ~ X, data=Exa9.3)
summary(Lin9.3)
```

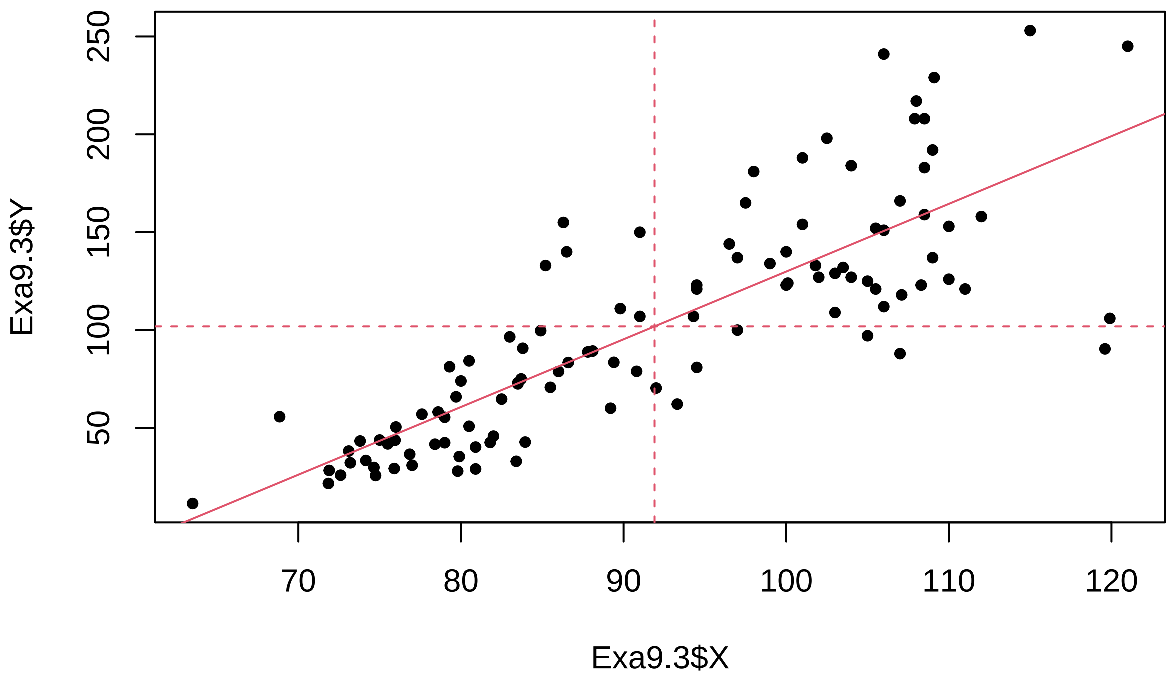
```
Call:
lm(formula = Y ~ X, data = Exa9.3)

Residuals:
     Min       1Q   Median       3Q      Max
-107.288  -19.143   -2.939   16.376   90.342

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -215.9815    21.7963  -9.909   <2e-16 ***
X              3.4589     0.2347  14.740   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 33.06 on 107 degrees of freedom
Multiple R-squared:   0.67, Adjusted R-squared:  0.667
F-statistic: 217.3 on 1 and 107 DF,  p-value: < 2.2e-16
```

```
abline(Lin9.3, col=2)
```



Following with more examples from Daniel's book.

```
setwd("~/Dropbox/Fdo/ClaseStats/RegressionClass/RegressionR_code")
# Changing wd to load the data file
# Problem 9.3.3 Methadone dose and the QTc Ventricular
# Tachycardia
Exr3.3=read.csv(file="DataOther/EXR_C09_S03_03.csv", header=TRUE)
names(Exr3.3)
```

```
[1] "DOSE" "QTC"
```

```
plot(Exr3.3$QTC ~ Exr3.3$DOSE, pch=20)
LinExr3.3 = lm(QTC ~ DOSE, data=Exr3.3)
summary(LinExr3.3)
```

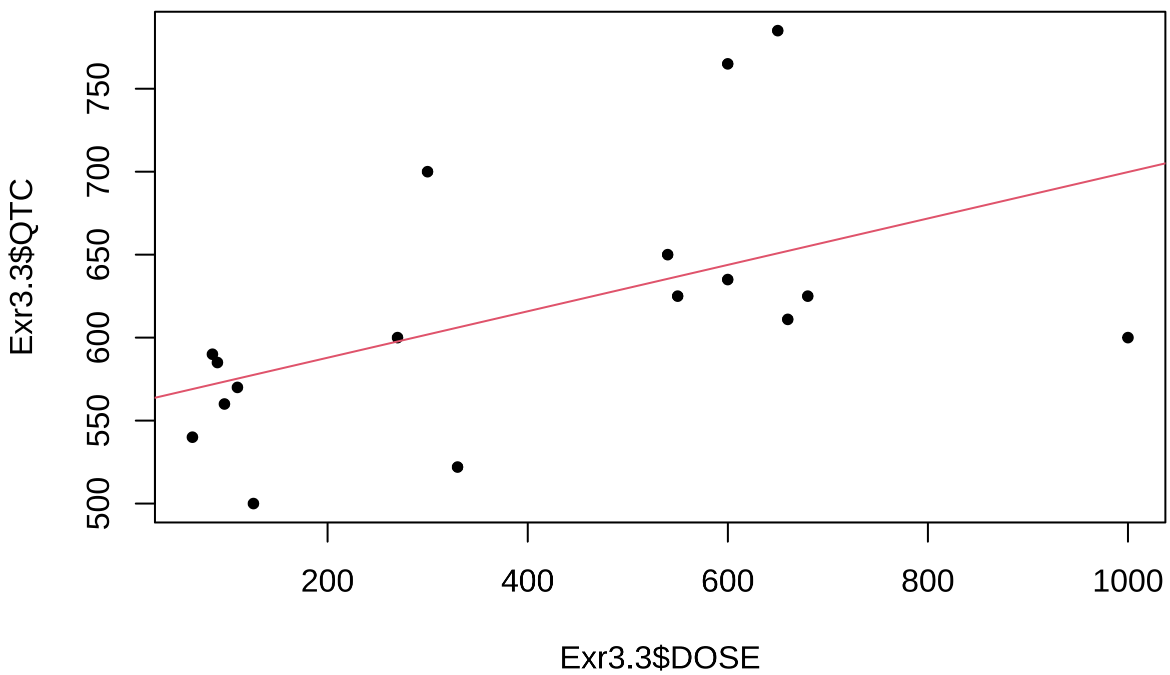
```
Call:
lm(formula = QTC ~ DOSE, data = Exr3.3)

Residuals:
    Min      1Q  Median      3Q     Max 
-99.789 -30.026  -8.835  14.559 134.171 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 559.90280   29.12926  19.221 5.61e-12 ***
DOSE          0.13989    0.06033   2.319   0.0349 *  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 68.28 on 15 degrees of freedom
Multiple R-squared:  0.2639,	Adjusted R-squared:  0.2148 
F-statistic: 5.377 on 1 and 15 DF,  p-value: 0.03493
```

```r
abline(LinExr3.3, col=2)
```




```r
# Res y = 559.9 + 0.139 x
```

GLM

Response variable Y is a random variable that is measured and has a distribution with expected value $E(Y|x)$ given a set of independent variables x.

$$Y_j(j = 1, ..., J)$$

for a set of x_jl predictor variables (or independent variables) defined as vectors for each j

$$x_jl(l = 1, ..., L)$$

with $L(L < J$, a general linear model with an error function ϵ_j can be expressed:

$$Y_j = x_{j1}\beta_1 + x_{j2}\beta_2 + x_{j3}\beta_3 + ... + x_{jL}\beta_L + \epsilon_j$$

with ϵ_j an independent variable identically distributed to the Normal with mean equal to zero.

$$\epsilon_j \approx N(0, \sigma^2)_{iid}$$

Linear Regresion (Chap 4, Vittinghoff et all.)

Example of simple linear regression: exercise and glucose Glucose levels above 125 mg/dL are diagnostic of diabetes, while 100-125 mg/dL signal increased risk. Data from HERS (public data) has baseline of glucose levels among 2,032 participants in a clinical trial of Hormone Therapy (HT). Women with diabetes are excluded, to study if the exercise might help prevent progression to diabetes.

```r
# hers data structure
hers_nodi <- filter(hers, diabetes == "no")
hers_nodi_Fit <- lm(glucose ~ exercise, data = hers_nodi)
# the linear model results can be printed using summary
summary(hers_nodi_Fit)
```

```
Call:
lm(formula = glucose ~ exercise, data = hers_nodi)

Residuals:
    Min      1Q  Median      3Q     Max 
-48.668  -6.668  -0.668   5.639  29.332 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  97.3610     0.2815 345.848  < 2e-16 ***
exerciseyes  -1.6928     0.4376  -3.868 0.000113 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 9.715 on 2030 degrees of freedom
Multiple R-squared:  0.007318,  Adjusted R-squared:  0.006829 
F-statistic: 14.97 on 1 and 2030 DF,  p-value: 0.000113
```

Simple linear regression model shows coefficient estimate (β_1) for exercise shows that average baseline glucose levels were about 1.7mg/dL lower among women who exercised at least three times a week than among women who exercised less.

For a multiple linear model

There are models to regress several predictor variables to relate several random independent variables.

$$y_i = E[y_i|x_i] + \epsilon_i$$

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p$$

Multiple linear regression model coefficients, the betas, give the change in $E[Y|x]$ for an increase of one unit on the predictor x_j , holding other factors in the model constant; each of the estimates is adjusted for the effects of all the other predictors. As in the simple linear model the intercept β_0 (beta zero) gives the value $E[Y|x]$ when all the predictors are equal to zero. Example of multiple linear model estimate is done with: glucose ~ exercise + age + drinkany + BMI.

In general in R we can write:$Y = \beta_1 variable_1 + \beta_2 variable_2 + \beta_3 variable_3 + \beta_4 variable_4$ for a multiple linear model, in this case four regressors.

```r
hers_nodi_multFit <- lm(glucose ~ exercise + age + drinkany + BMI, data = hers_nodi)
# the linear model results can be printed using summary
summary(hers_nodi_multFit)
```

```
Call:
lm(formula = glucose ~ exercise + age + drinkany + BMI, data = hers_nodi)

Residuals:
    Min      1Q  Median      3Q     Max 
-47.560  -6.400  -0.886   5.496  32.060 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 78.96239    2.59284  30.454   <2e-16 ***
exerciseyes -0.95044    0.42873  -2.217   0.0267 *  
age          0.06355    0.03139   2.024   0.0431 *  
drinkanyyes  0.68026    0.42196   1.612   0.1071    
BMI          0.48924    0.04155  11.774   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 9.389 on 2023 degrees of freedom
  (4 observations deleted due to missingness)
Multiple R-squared:  0.07197,	Adjusted R-squared:  0.07013 
F-statistic: 39.22 on 4 and 2023 DF,  p-value: < 2.2e-16
```

Multiple linear model, with interactions

In general in R we can write the interaction term as the product of the regressors that we are studying the interaction:$variable_1 : varible_2$ for a multiple linear model with two regressors and interaction the equation looks like:

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2$$

(The following is a very good link: http://www.sthda.com/english/articles/40-regression-analysis/)

Multiple Linear Model

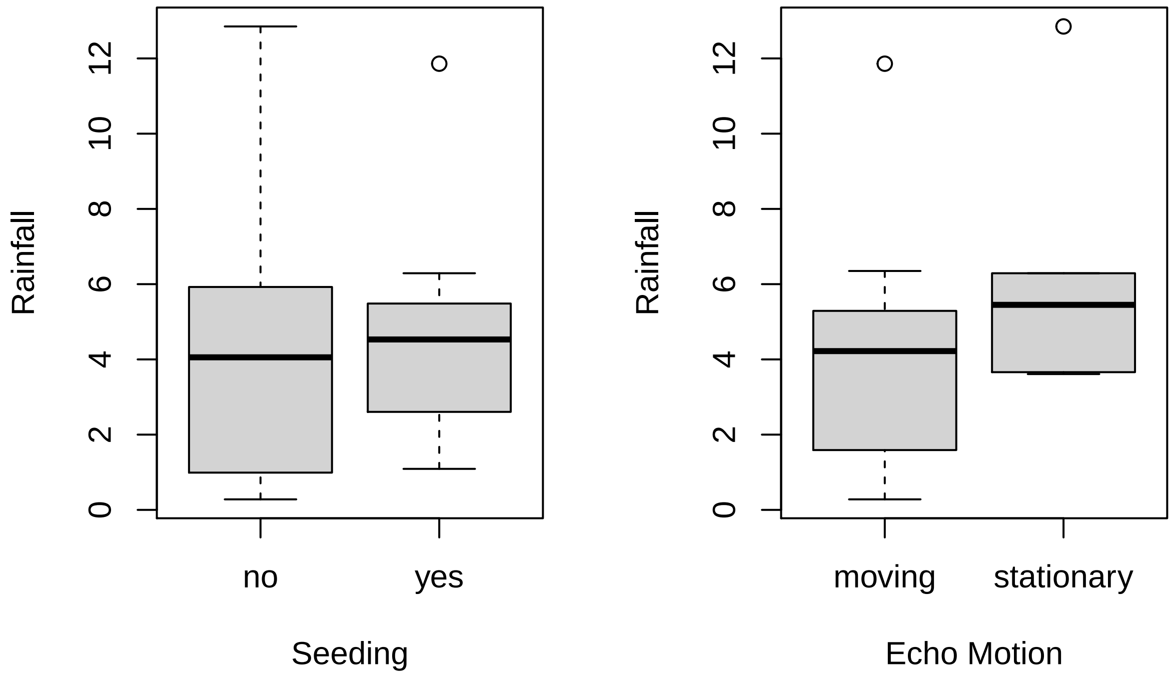
This are only additive linear terms to explain a random response variable Y and the adjusted parameters are the β_i of the independent variables or predictors. These variables are random too. They can be numbers and factors. Example of multiple linear regression using the clouds data clouds from HSAUR

```r
data(clouds)
head(clouds)
```

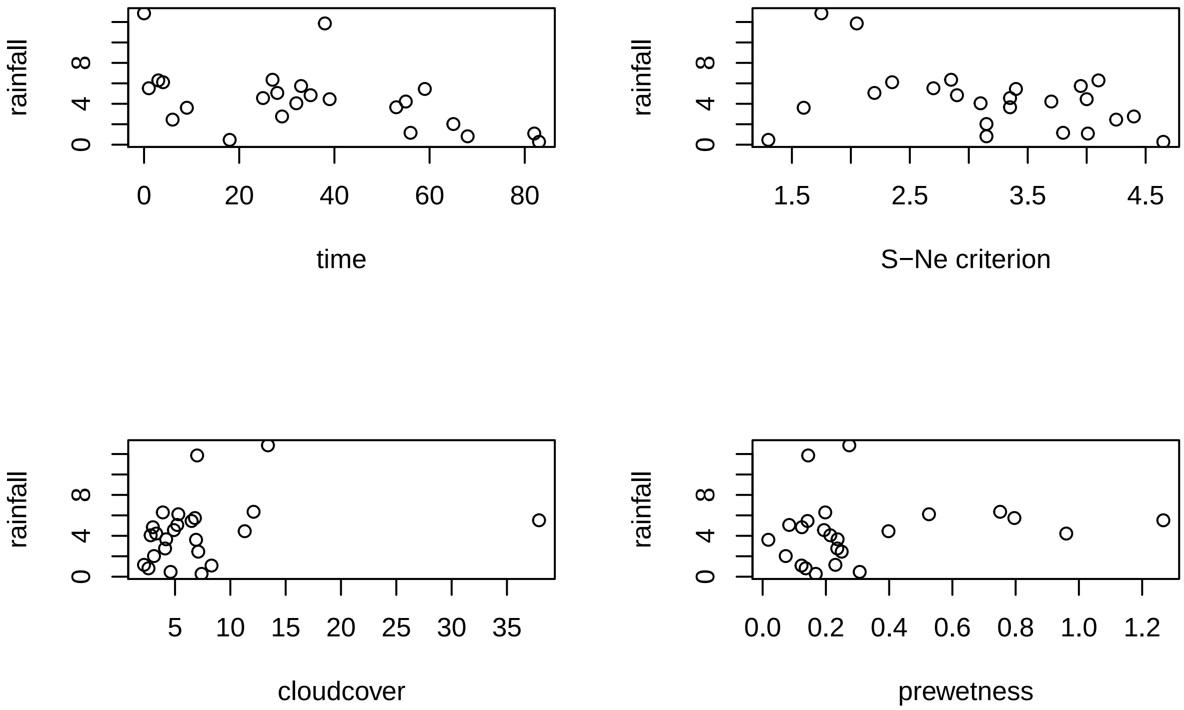
```
  seeding time  sne cloudcover prewetness echomotion rainfall
1      no    0 1.75       13.4      0.274 stationary    12.85
2     yes    1 2.70       37.9      1.267     moving     5.52
3     yes    3 4.10        3.9      0.198 stationary     6.29
4      no    4 2.35        5.3      0.526     moving     6.11
5     yes    6 4.25        7.1      0.250     moving     2.45
```

```
6       no     9 1.60         6.9      0.018 stationary     3.61
```

```
# looking the datafor rainfall
# boxplot(rainfall~seeding, data=clouds)
# boxplot(rainfall~echomotion, data=clouds)
layout(matrix(1:2, ncol = 2))
boxplot(rainfall ~ seeding, data = clouds, ylab = "Rainfall", xlab = "Seeding")
boxplot(rainfall ~ echomotion, data = clouds, ylab = "Rainfall", xlab = "Echo Motion")
```




```
#
layout(matrix(1:4, nrow = 2))
plot(rainfall ~ time, data = clouds)
plot(rainfall ~ cloudcover, data = clouds)
plot(rainfall ~ sne, data = clouds, xlab="S-Ne criterion")
plot(rainfall ~ prewetness, data = clouds)
```




```
#
clouds_formula <- rainfall ~ seeding + seeding:(sne+cloudcover+prewetness+echomotion) + time
Xstar <- model.matrix(clouds_formula, data = clouds)
attr(Xstar, "contrasts")
```

```
$seeding
[1] "contr.treatment"

$echomotion
[1] "contr.treatment"
```

```
clouds_lm <- lm(clouds_formula, data = clouds)
summary(clouds_lm)
```

```

Call:
lm(formula = clouds_formula, data = clouds)

Residuals:
    Min      1Q  Median      3Q     Max 
-2.5259 -1.1486 -0.2704  1.0401  4.3913 

Coefficients:
                                Estimate Std. Error t value Pr(>|t|)   
(Intercept)                     -0.34624    2.78773  -0.124  0.90306   
seedingyes                      15.68293    4.44627   3.527  0.00372 **
time                            -0.04497    0.02505  -1.795  0.09590 . 
seedingno:sne                    0.41981    0.84453   0.497  0.62742   
seedingyes:sne                  -2.77738    0.92837  -2.992  0.01040 * 
seedingno:cloudcover             0.38786    0.21786   1.780  0.09839 . 
seedingyes:cloudcover           -0.09839    0.11029  -0.892  0.38854   
seedingno:prewetness             4.10834    3.60101   1.141  0.27450   
```

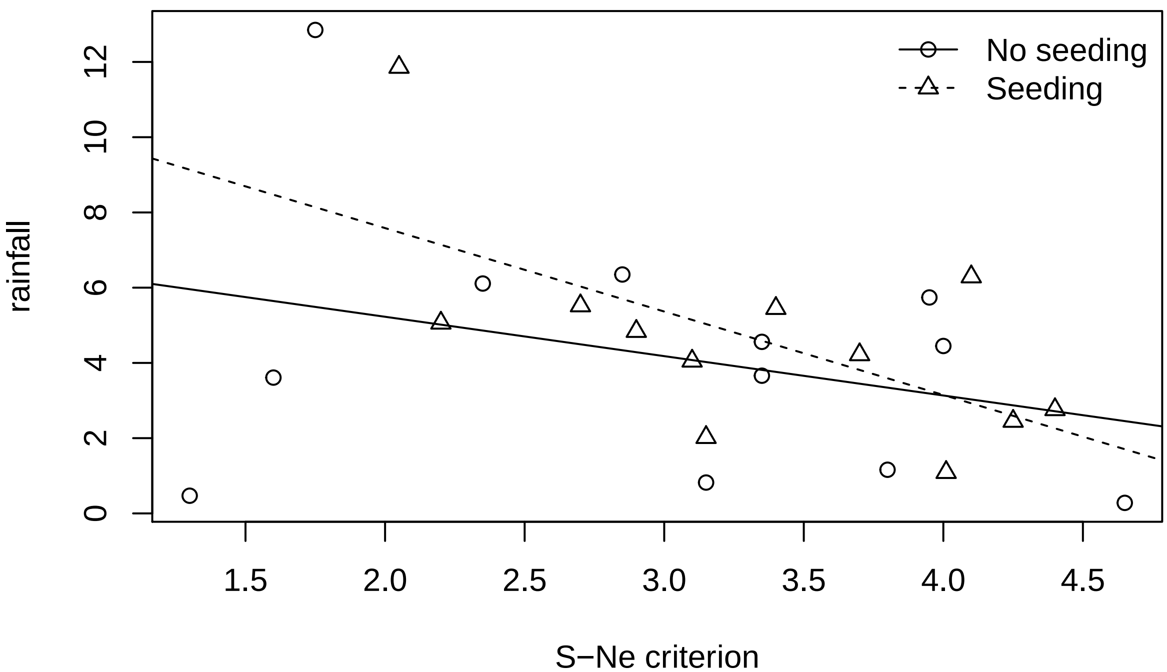
```
seedingyes:prewetness             1.55127    2.69287   0.576  0.57441
seedingno:echomotionstationary    3.15281    1.93253   1.631  0.12677
seedingyes:echomotionstationary   2.59060    1.81726   1.426  0.17757
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.205 on 13 degrees of freedom
Multiple R-squared:  0.7158,    Adjusted R-squared:  0.4972
F-statistic: 3.274 on 10 and 13 DF,  p-value: 0.02431
```

```r
layout(matrix(1:1, nrow = 1))
# to list the betas* with the:
betaStar <- coef(clouds_lm)
betaStar
```

```
                    (Intercept)                      seedingyes
                    -0.34624093                     15.68293481
                           time                   seedingno:sne
                    -0.04497427                      0.41981393
                 seedingyes:sne           seedingno:cloudcover
                    -2.77737613                      0.38786207
          seedingyes:cloudcover           seedingno:prewetness
                    -0.09839285                      4.10834188
          seedingyes:prewetness  seedingno:echomotionstationary
                     1.55127493                      3.15281358
seedingyes:echomotionstationary
                     2.59059513
```

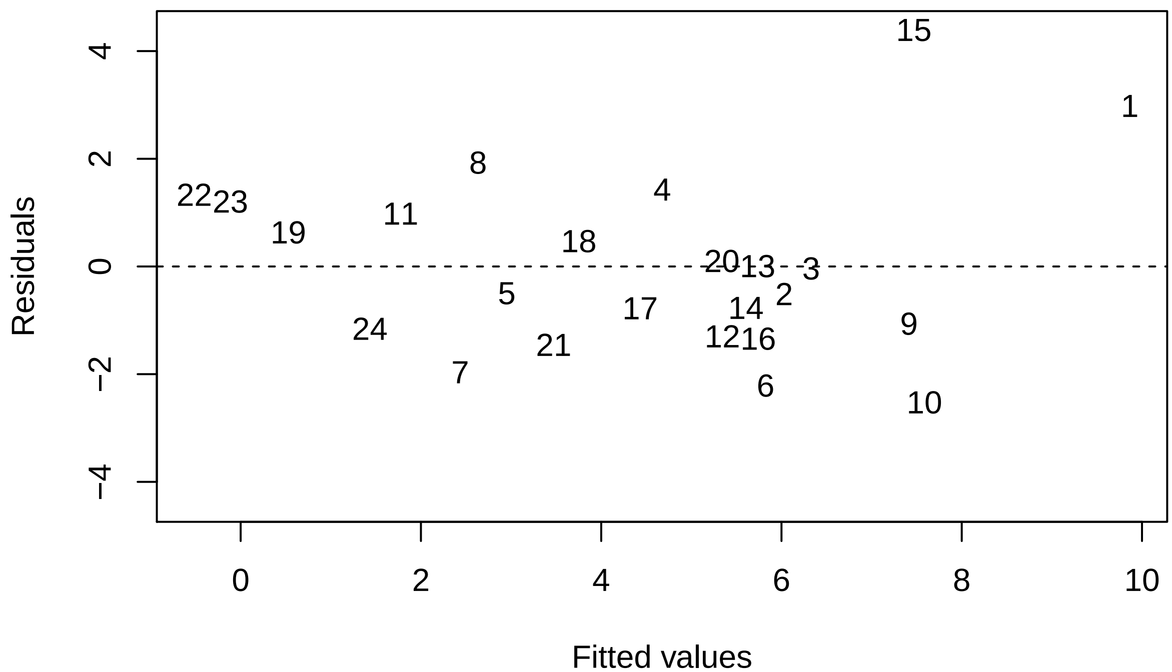
```r
# to understand the relation of seeding and sne
psymb <- as.numeric(clouds$seeding)
plot(rainfall ~ sne, data = clouds, pch = psymb, xlab = "S-Ne criterion")
abline(lm(rainfall ~ sne, data = clouds, subset = seeding == "no"))
abline(lm(rainfall ~ sne, data = clouds, subset = seeding == "yes"), lty = 2)
legend("topright", legend = c("No seeding", "Seeding"), pch = 1:2, lty = 1:2, bty = "n")
```



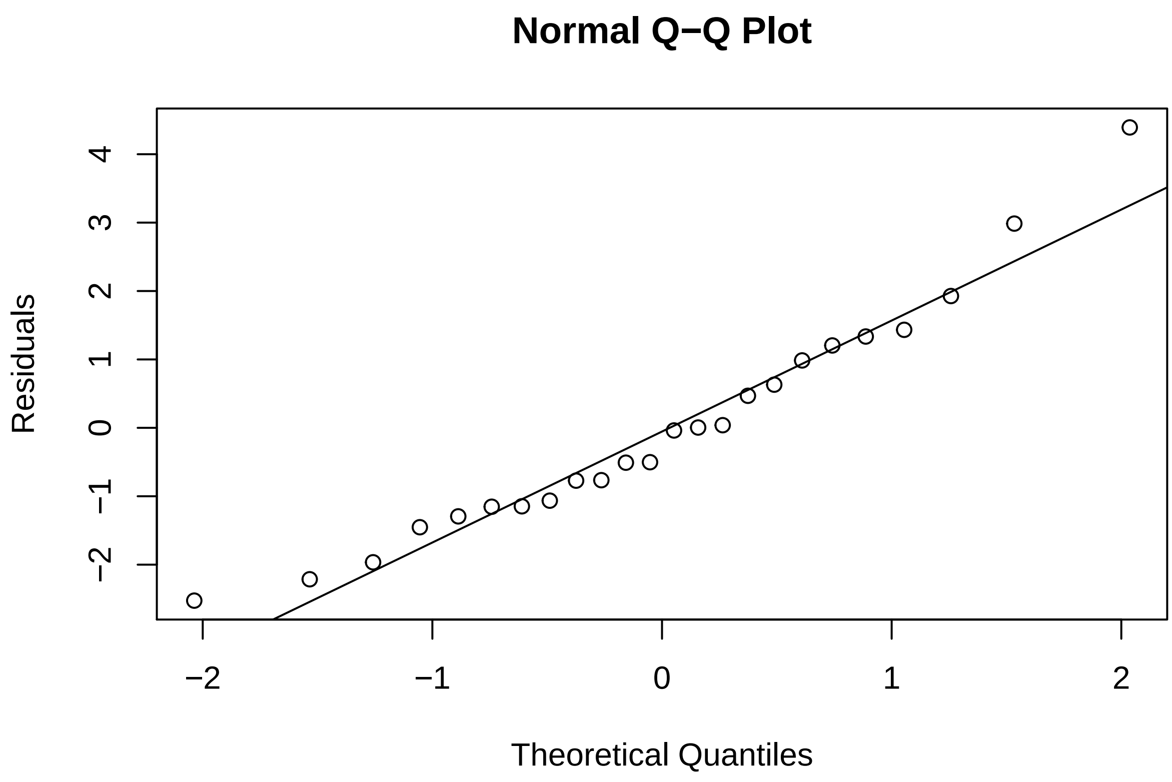
```r
#
# and the Covariant matrix Cov(beta*) with:
VbetaStar <- vcov(clouds_lm)
# Where the square roots of the diagonal elements are the standart errors
sqrt(diag(VbetaStar))
```

```
                   (Intercept)                      seedingyes
                    2.78773403                      4.44626606
                          time                  seedingno:sne
                    0.02505286                      0.84452994
                seedingyes:sne           seedingno:cloudcover
                    0.92837010                      0.21785501
         seedingyes:cloudcover           seedingno:prewetness
                    0.11028981                      3.60100694
         seedingyes:prewetness  seedingno:echomotionstationary
                    2.69287308                      1.93252592
seedingyes:echomotionstationary
                    1.81725973
```

```r
clouds_resid <- residuals(clouds_lm)
clouds_fitted <- fitted(clouds_lm)
# residuals and the fitted values can be used to construct diagnostic plot
plot(clouds_fitted, clouds_resid, xlab = "Fitted values", ylab = "Residuals", type = "n", ylim = max(abs
abline(h = 0, lty = 2)
textplot(clouds_fitted, clouds_resid, words = rownames(clouds), new = FALSE)
```




```r
qqnorm(clouds_resid, ylab = "Residuals")
qqline(clouds_resid)
```



Normal Q−Q Plot
Residuals
4
3
2
1
0
−1
−2
−2
−1
0
1
2
Theoretical Quantiles