R Companion - Clewer & Scarisbrick (2001)

To be used in AGR5266C Field Plot Techniques class (Dr. Rios)

Rodrigo R Amadeu (rramadeu at ufl dot edu)

September 8th 2020

Contents

R codes to perform most of the statistical analysis of the *Clewer, Alan G., and David H. Scarisbrick. Practical statistics and experimental design for plant and crop science. John Wiley & Sons, 2001.* from Chapters 1 to 16. For interpretation and theoretical explanation you should go to the book. If you find any issue, comment, or error, please send me an e-mail.

Chapter 2: Basic Statistical Calculations

Data input in R

Data in R are stored in vectors, to build a vector `x` with the elements from the Example 2.1 in the book (i.e., 14.8, 15.2, 17.4, 11.6, and 12.5), you can use the function `c()`, where `c` stands for combine values into a vector or list. An example:

```
x = c(14.8, 15.2, 17.4, 11.6, 12.5)
```

To visualize the built data you can look at the `Enviroment` tab (upper right) which shows you all the loaded data into your `R` environment. Another option is to explicitly call it or print it:

```
x
```

```
## [1] 14.8 15.2 17.4 11.6 12.5
```

```
print(x)
```

```
## [1] 14.8 15.2 17.4 11.6 12.5
```

Another way to input the data in R is to import from a file. In the RStudio, you can go to `Import Dataset` option under the `Environment` tab and select you data type. Each data type requires a specific command with specific arguments.

Basic Statistical Calculations

Mean

There are several ways to compute the same value in 'R', it follows some of the ways to compute the sample mean:

You can use R as a calculator and explicitly compute the mean, the signals follow the standard of other statistical software:

```
(14.8 + 15.2 + 17.4 + 11.6 + 12.5)/5
```

```
## [1] 14.3
```

Or you can use built-in function on your vector `x`, as:

```
sum(x)/length(x) #sum all the elements of 'x' and divide such value for its length (n)
```

```
## [1] 14.3
```

```
mean(x) #compute the mean of x
```

```
## [1] 14.3
```

Hereafter, we show just compact ways to compute different statistics:

```
median(x) #median of x
```

```
## [1] 14.8
```

```
var(x) #sample variance of x
```

```
## [1] 5.3
```

```
var(x)*(length(x)-1) #corrected sum of squares of x
```

```
## [1] 21.2
```

```
sd(x) #sample standard deviation of x
```

```
## [1] 2.302173
```

```
sd(x)/mean(x) #coefficient of variation of x
```

```
## [1] 0.1609911
```

Weighted mean

```
x = c(3.5, 4.8, 5.2) #sample means
n = c(4, 5, 10) #sample sizes
?weighted.mean
weighted.mean(x,n)
```

```
## [1] 4.736842
```

Data structure in R

In this course, we will use two data structures in R: i) a vector and ii) a collection of vectors of same length (a.k.a. data frame). A vector can have different classes as `numeric`, `character`, and `factor`. A vector can only has one type of class. To create a vector of character you need to specify each element within quotation marks as follow:

```
x.numeric = c(1, 2, 3, 4)
class(x.numeric)
```

```
## [1] "numeric"
```

```
y = c("1","2","3","4")
class(y)
```

```
## [1] "character"
```

Note that the vector `y` is a character vector, so, you can not do calculations with it as before:

```
sum(y)
```

To create a vector of factors, you need to first create a vector of numeric or character, and then convert it to factor.

```
y = as.factor(y)
class(y)
```

```
## [1] "factor"
```

```
y
```

```
## [1] 1 2 3 4
## Levels: 1 2 3 4
```

Now, `y` is a factor vector with 4 levels. The factor class is often used in Statistics to assign different levels for a treatment, different blocks, plots, etc. Usually in this course we will have a response variable as a `numeric` variable (e.g., yield, mortality, color index, etc) and your explanatory variables as `factor` (block, treatment, plot etc)

A data frame is a rectangular table organized by columns with same length. Here, we show how to input data frames in R.

You can create a data frame object explicitly with the function `data.frame()`. Inside of it has a collection of vectors of same length separated by comma. **Always** check the data with `str()` function, attention for data type, number of observations, variables, and number of levels for the factors. In this example you should have `variety` and `block` as factor and `yield` as numeric.

```r
MyData= data.frame(variety = c("V1", "V1", "V1",
                               "V2", "V2", "V2",
                               "V3", "V3", "V3",
                               "V4", "V4", "V4"),
                   block = c("B1", "B2", "B3",
                             "B1", "B2", "B3",
                             "B1", "B2", "B3",
                             "B1", "B2", "B3"),
                   yield = c(7.4, 6.5, 5.6,
                             9.8, 6.8, 6.2,
                             7.3, 6.1, 6.4,
                             9.5, 8.0, 7.4))

str(MyData)
```

```
## 'data.frame':    12 obs. of  3 variables:
##  $ variety: Factor w/ 4 levels "V1","V2","V3",..: 1 1 1 2 2 2 3 3 3 4 ...
##  $ block  : Factor w/ 3 levels "B1","B2","B3": 1 2 3 1 2 3 1 2 3 1 ...
##  $ yield  : num  7.4 6.5 5.6 9.8 6.8 6.2 7.3 6.1 6.4 9.5 ...
```

As a file: by graphical interface

Click into the *Environment* tab at the top-right side of RStudio. Then, click in *Import Dataset*, select the correct format. It will open a window to search for your file. At the bottom, it has your future data name and some other specifications, double-check everything and click in *Import*.

Importing data in this way is easier, however you will need to do it every time and the step-by-step is not saved in your R script. We strongly encourage you to import your data from command line. You can see the command line the R used to import your data in the console, a tip is to copy that line and let it in your script. In that way you will not have problems in the future.

As a file: by command line

Alternatively you can load the data as a file. Here it is shown how to load a csv (comma-separated-value) file which is a common output for different software. Double check if your current working directory is the same as the data is located. If not, change the working directory or put the path of the file. To write the path, I tip is to open quotation marks and press `tab` in the keyboard within RStudio. This shows all the options available in the current working directory.

```r
getwd() #check which directory I'm working
```

```
## [1] "/home/rramadeu/Dropbox (UFL)/PhD/Courses/AGR5266C_FieldPlotTechnique_TA_2020/RCompanion"
```

```r
MyData= read.csv("MyData.csv")
str(MyData)
```

```
## 'data.frame':    12 obs. of  3 variables:
##  $ variety: Factor w/ 4 levels "V1","V2","V3",..: 1 1 1 2 2 2 3 3 3 4 ...
##  $ block  : Factor w/ 3 levels "B1","B2","B3": 1 2 3 1 2 3 1 2 3 1 ...
##  $ yield  : num  7.4 6.5 5.6 9.8 6.8 6.2 7.3 6.1 6.4 9.5 ...
```

If something goes wrong, you can open your ‘csv’ file in RStudio (File -> Open) and check if the format is correct. The `read.csv()` is expecting values separated by comma. If your data have values separated by semi-colon, you can use instead the `sep` argument. For more options, see the help `?read.csv`.

Often we type the data in a Excel spreadsheet (`.xlsx` format). To load this format you need to install an `R` package as the `openxlsx`.

```r
install.packages("openxlsx")
```

```r
library(openxlsx)

MyData = read.xlsx("MyData.xlsx")
str(MyData)
```

```
## 'data.frame':    12 obs. of  3 variables:
##  $ variety: num  1 1 1 2 2 2 3 3 3 4 ...
##  $ block  : chr  "B1" "B2" "B3" "B1" ...
##  $ yield  : num  7.4 6.5 5.6 9.8 6.8 6.2 7.3 6.1 6.4 9.5 ...
```

Notice now that variety in this example is a numeric variable and block is a character, both should be factors in our analysis. So, let's convert them to factors and check the data frame structure again. For sake of the example, let's also convert yield to numeric (nothing happens here since yield is already numeric).

```r
MyData$variety = as.factor(MyData$variety)
MyData$block = as.factor(MyData$block)
MyData$yield = as.numeric(MyData$yield)
str(MyData)
```

```
## 'data.frame':    12 obs. of  3 variables:
##  $ variety: Factor w/ 4 levels "1","2","3","4": 1 1 1 2 2 2 3 3 3 4 ...
##  $ block  : Factor w/ 3 levels "B1","B2","B3": 1 2 3 1 2 3 1 2 3 1 ...
##  $ yield  : num  7.4 6.5 5.6 9.8 6.8 6.2 7.3 6.1 6.4 9.5 ...
```

To access individual columns, as saw before, it is used the `$` symbol plus the name of the column. Alternatively, you can use a numeric indicator of the column within `[[]]`

```r
MyData[[1]]
```

```
##  [1] 1 1 1 2 2 2 3 3 3 4 4 4
## Levels: 1 2 3 4
```

```r
MyData[[2]]
```

```
##  [1] B1 B2 B3 B1 B2 B3 B1 B2 B3 B1 B2 B3
## Levels: B1 B2 B3
```

```r
MyData[[3]]
```

```
##  [1] 7.4 6.5 5.6 9.8 6.8 6.2 7.3 6.1 6.4 9.5 8.0 7.4
```

Chapter 3: Basic Data Summary

3.2: Frequency distributions (discrete data)

Example 3.1: Data input number of tillers

```
x = c(1, 1, 1, 2, 2, 2, 2,
      3, 3, 3, 3, 3, 3, 3,
      3, 4, 4, 4, 4, 4, 4,
      4, 4, 4, 4, 4, 4, 4,
      4, 4, 4, 5, 5, 5, 5,
      5, 5, 5, 5, 5, 5, 5,
      5, 5, 6, 6, 6, 6, 6,
      6, 6, 6, 6, 7, 7, 7,
      7, 7, 8, 8)
```

Basic data summary

```
sum(x)
```

```
## [1] 269
```

```
length(x)
```

```
## [1] 60
```

```
mean(x)
```

```
## [1] 4.483333
```

```
var(x)
```

```
## [1] 2.762429
```

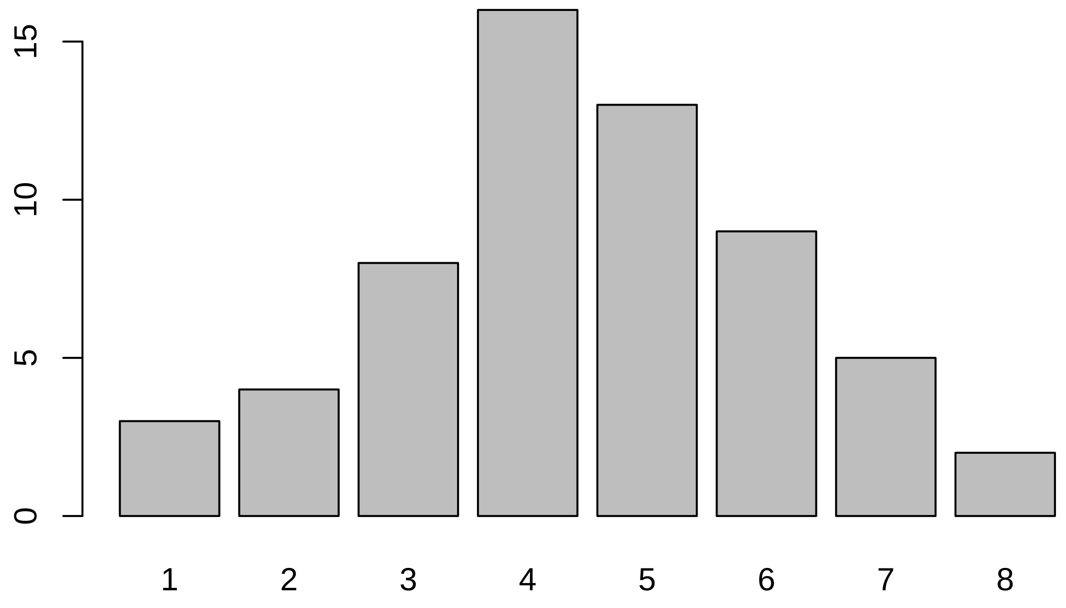
Frequency table

```
table(x)
```

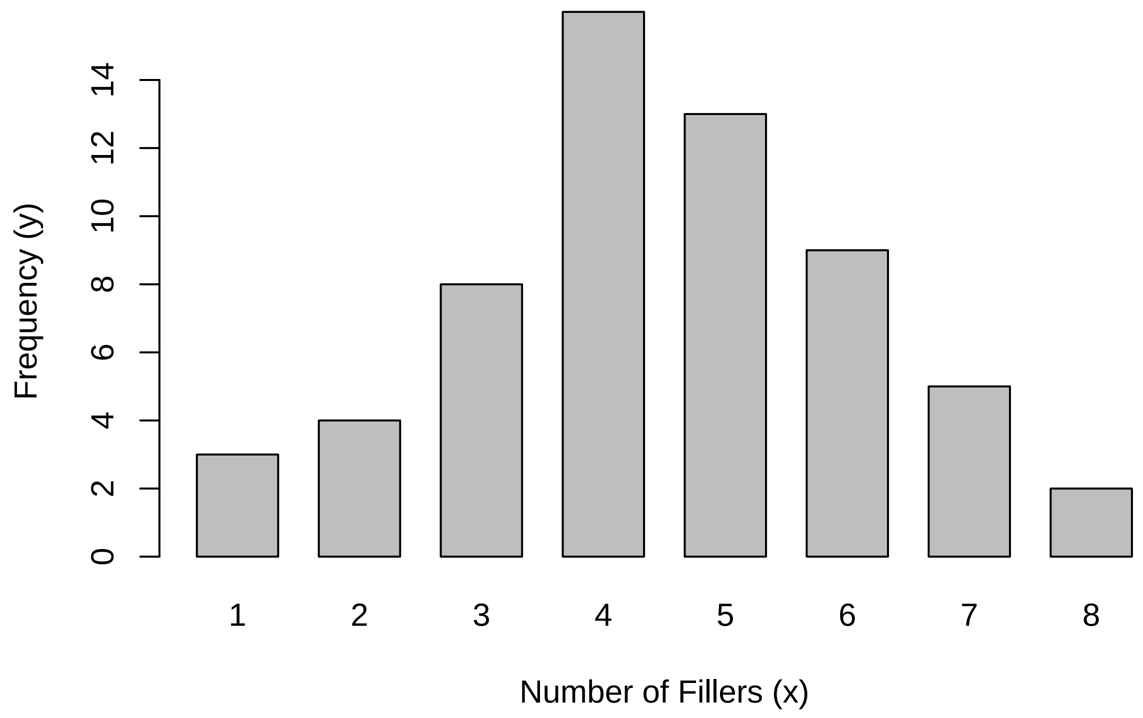
```
## x
##  1  2  3  4  5  6  7  8
##  3  4  8 16 13  9  5  2
```

Bar plot

```
counts <- table(x)
barplot(counts)
```



```
# Customizing it a little bit
barplot(counts, xlab = "Number of Fillers (x)", ylab = "Frequency (y)", space = 0.5, ylim=c(0,15))
```



The mode

```
sort(counts) #the mode is 4 with 16 counts
```

```
## x
##  8  1  2  7  3  6  5  4
##  2  3  4  5  8  9 13 16
```

```
median(x)
```

```
## [1] 4
```

3.3 Frequency distributions (continuous data)

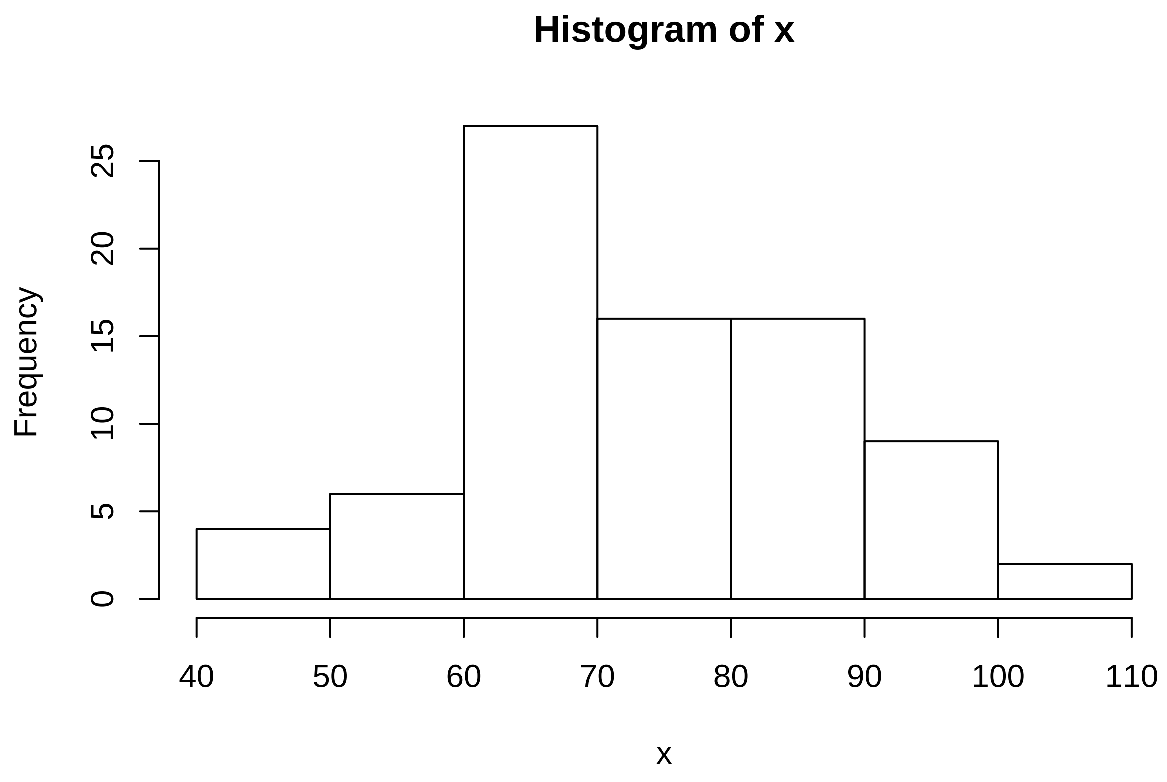
Example 3.2: Data input barley yield

```
x = c(95, 70, 68, 88, 79, 92, 64, 83, 67, 63,
      56, 70, 53, 78, 71, 62, 42, 80, 50, 68,
      78, 104, 62, 66, 90, 86, 66, 82, 83, 56,
      82, 90, 71, 77, 93, 68, 91, 98, 79, 75,
      92, 93, 73, 79, 95, 78, 77, 108, 86, 87,
      68, 68, 49, 75, 82, 61, 68, 65, 56, 96,
      52, 61, 87, 79, 64, 64, 84, 63, 64, 44,
      87, 63, 65, 64, 81, 72, 62, 58, 84, 67)

length(x) #checking dimension
```

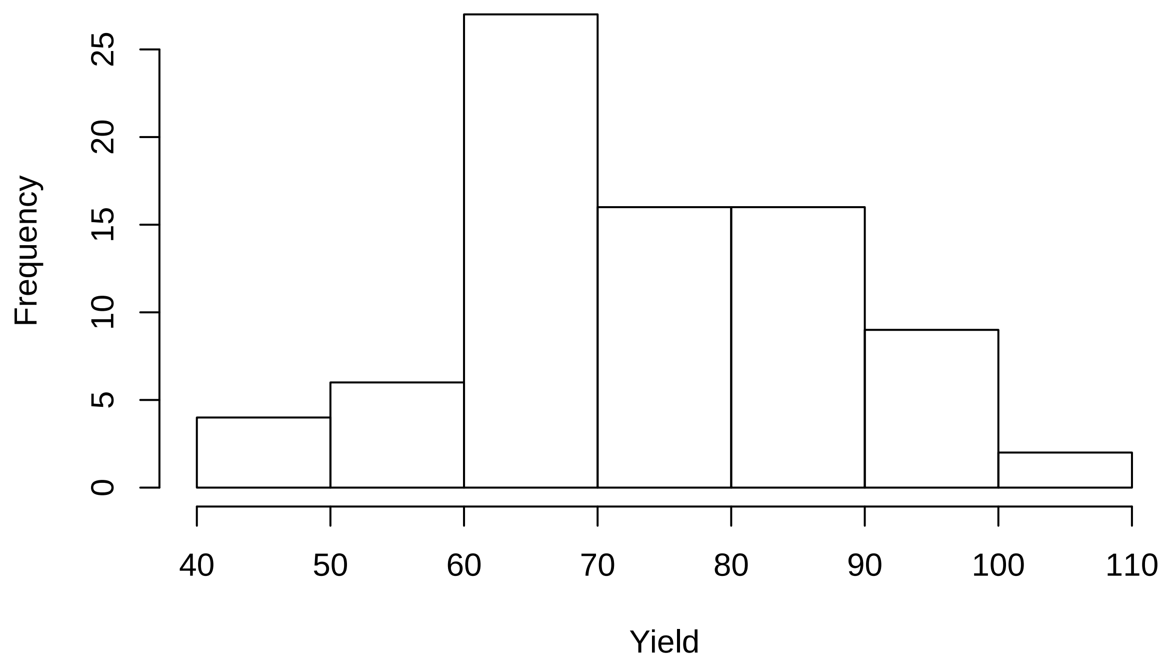
```
## [1] 80
```

The histogram

```r
hist(x)
```



```r
# Customizing it a little bit
hist(x, xlab="Yield", main=NULL)
```



Quartiles and Ranges

```r
quantile(x,0.25) #Q1
```

```
## 25%
##  64
```

```r
quantile(x,0.75) #Q3
```

```
## 75%
##  84
```

```r
IQR(x)  #IQR (i.e., Q3-Q1)
```

```
## [1] 20
```

```r
max(x)
```

```
## [1] 108
```

```r
min(x)
```

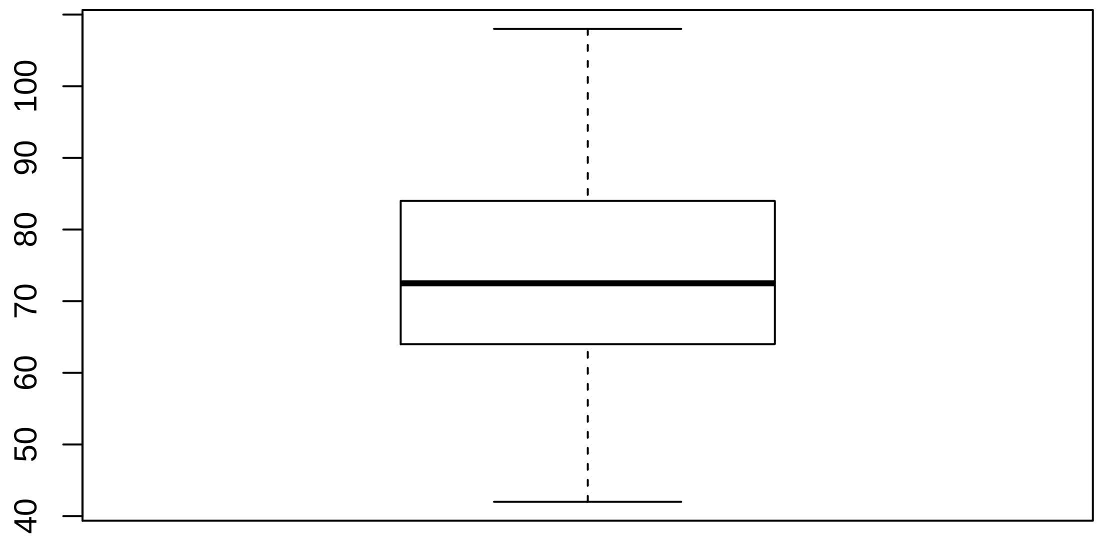
```
## [1] 42
```

```r
range(x)
```

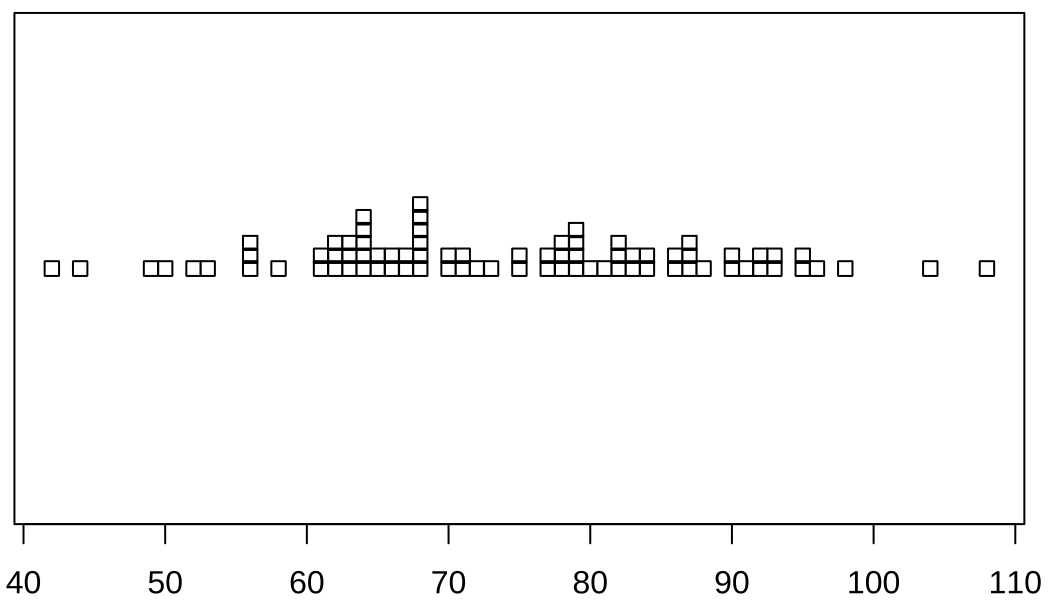
```
## [1]  42 108
```

Other graphical methods

```
boxplot(x) #boxplot
```



```
stripchart(x,method = "stack") #dotplot
```



```r
stem(x) #stem and leaf plot
```

```
##
##   The decimal point is 1 digit(s) to the right of the |
##
##    4 | 249
##    5 | 0236668
##    6 | 11222333444445566677888888
##    7 | 00112355778889999
##    8 | 012223344667778
##    9 | 00122335568
##   10 | 48
```

Descriptive statistics

The R default summary:

```r
## R Default:
length(x) #N
```

```
## [1] 80
```

```r
summary(x)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   42.00   64.00   72.50   73.96   84.00  108.00
```

```r
sd(x) #standard deviation
```

```
## [1] 14.02389
```

```r
var(x) #variance
```

```
## [1] 196.6695
```

```r
mean(x) #mean
```

```
## [1] 73.9625
```

```r
mean(x, trim=0.05) #trimmed mean by 0.05
```

```
## [1] 73.97222
```

There are several packages that provide different types of summary statistics, to see some of them click **here**.

Chapter 4: The Normal Distribution, the t-Distribution and Confidence Intervals

Simulating data in R

To simulate data in R, you need to choose a distribution (probability density function),its parameters, and how many data points (`n`) to sample from this distribution.

Here you can find several probability density functions that `R` can handle. The core idea on simulating data from such distributions is to know what each one of the simulation functions does (functions `r...`, `p...`, `q...`,and `d...`). Knowing this, you can simulate or extract data from different probability density function. Here we present the four functions of the normal (Gaussian) distribution:

##Distribution Family functions

`rnorm`

`r`andomly generated `n` numbers based on a Normal Distribution with mean `mean` and standard deviation `sd`:

```
rnorm(n = 10, mean = 0, sd = 1) #"r": random, randomly generated numbers f
```

```
##  [1] -0.3314601 -1.6784322  1.5015933  0.5811605  1.0688588 -0.3182134
##  [7]  0.2392734 -1.3967444  0.8106202 -3.1043312
```

`pnorm`

What is the `p`robability to have values lower than `q` based on a Normal Distribution with mean `mean` and standard deviation `sd`:

```
pnorm(q = 0, mean = 0, sd = 1) #"p": probability, cumulative density function
```

```
## [1] 0.5
```

`qnorm` (inverse of `pnorm`)

What is the value (`q`) correspondent to the probability `p` to have values lower based on a Normal Distribution with mean `mean` and standard deviation `sd`:

```
qnorm(p = 0.5, mean = 0, sd = 1) #"q": quantiles, cumulative density function (quantiles)
```

```
## [1] 0
```

`dnorm`

What is the density of correspondent to the point `x` based on a Normal Distribution with mean `mean` and standard deviation `sd`.

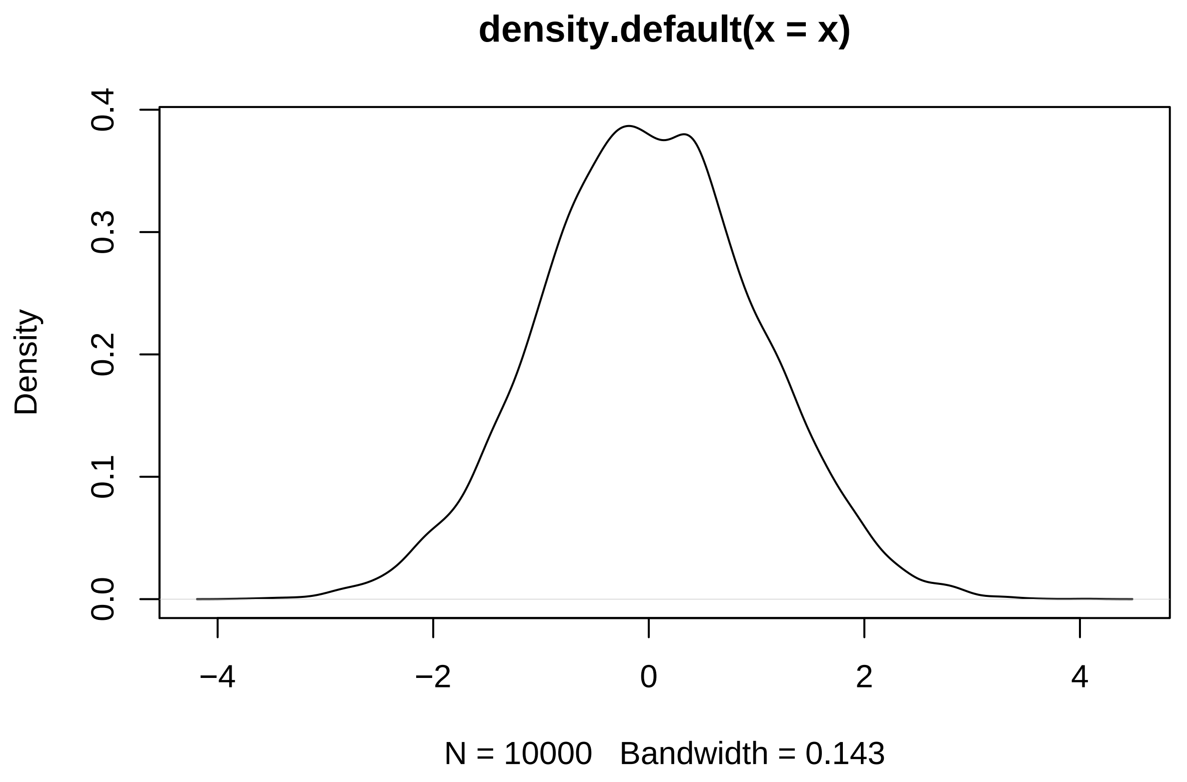
```
dnorm(x = 1, mean = 0, sd = 1)
```

```
## [1] 0.2419707
```

Visualizing the distribution

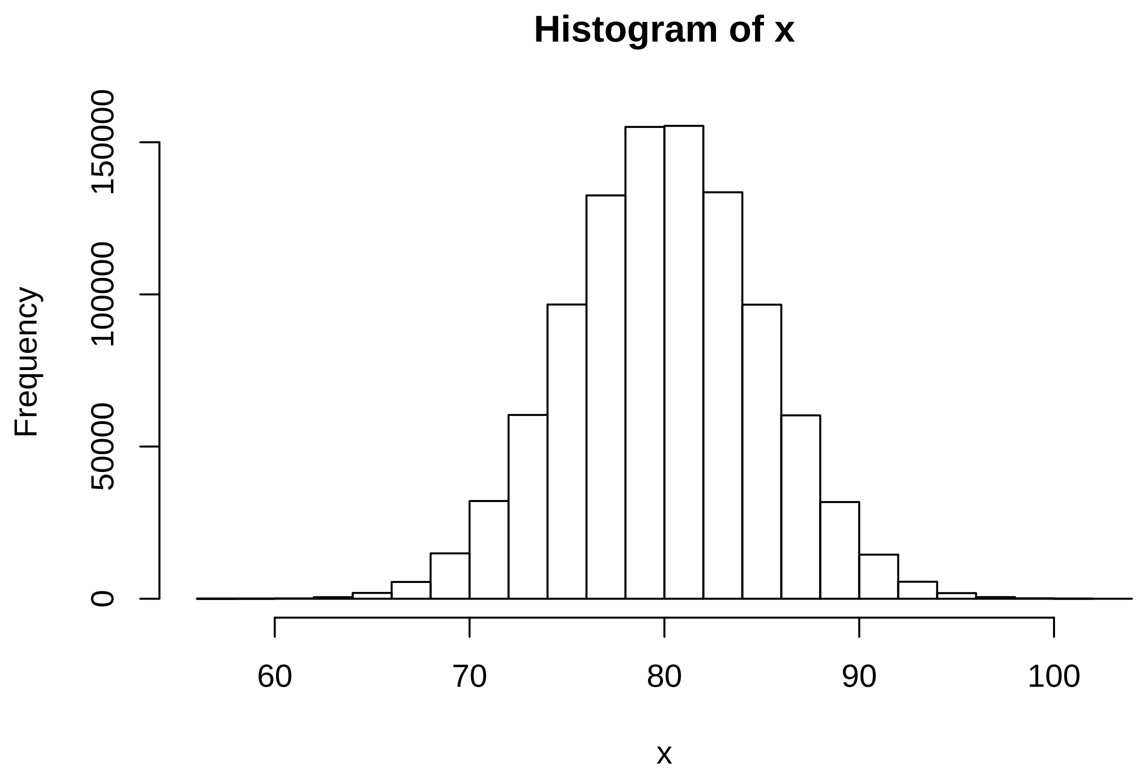
An way to visualize a distribution is to randomly sample a large number of data points from it, then, to estimate its density estimates, and to plot it in a density plot.

```
x = rnorm(n = 10000, mean = 0, sd = 1)
dens_x <- density(x)
plot(dens_x)
```

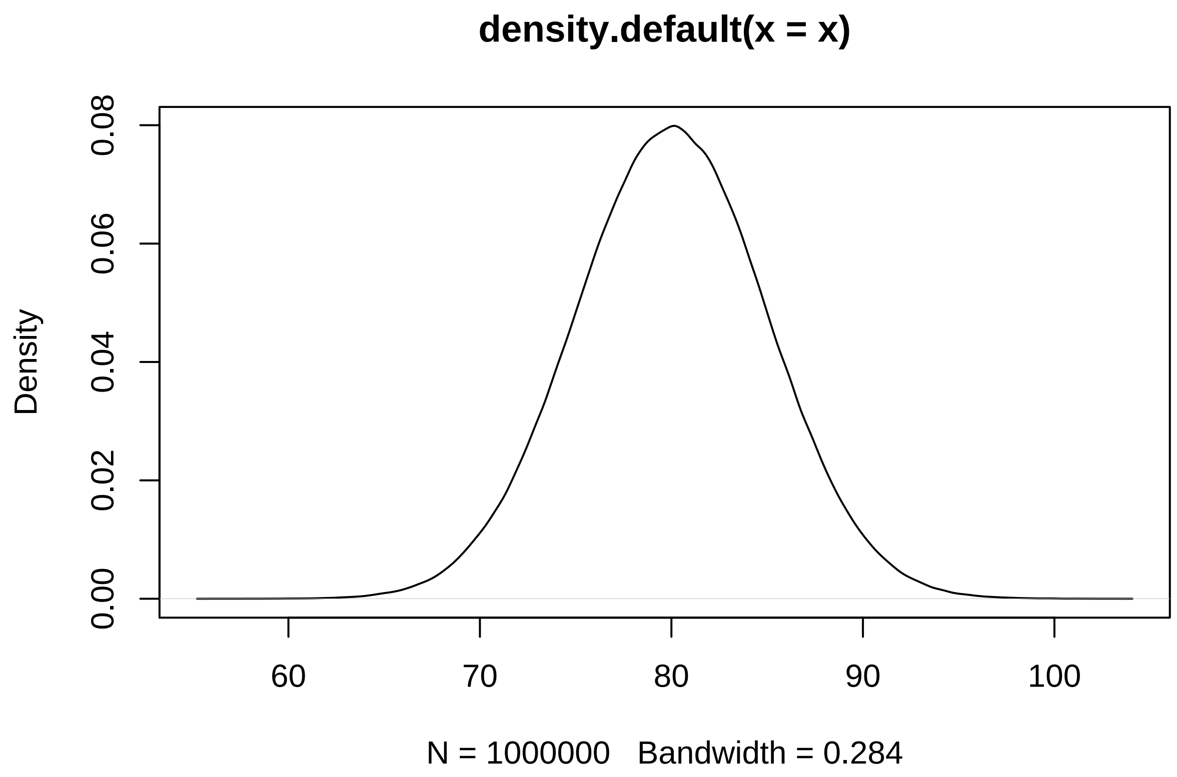


Example 4.1

```
x <- rnorm(n = 1000000, mean = 80, sd = 5)
hist(x)
```



```r
dens_x <- density(x)
plot(dens_x)
```



We can estimate some of the properties of the Normal Distribution with simulations:

(2) About 67% of the population is within one standard deviation of the mean:

```
## How many elements are within 1 sd from the mean (i.e., 75 < x < 85)
interval <- which(x > 75 & x < 85)
length(interval)/length(x)
```

```
## [1] 0.68296
```

(3) 90% of the population is within 1.645 standard deviations of the mean

```
interval <- which(x > (80 - 1.645*5) & x < 80 + (1.645*5))
length(interval)/length(x)
```

```
## [1] 0.900032
```

(4) 95% of the population is within 1.96 standard deviations of the mean, and hence only 2.5% of the population have values which are greater than 1.96 standard deviations above the mean.

```
interval <- which(x > (80 - 1.96*5) & x < 80 + (1.96*5))
length(interval)/length(x)
```

```
## [1] 0.94988
```

```
interval <- which(x > 80 + (1.96*5))
length(interval)/length(x)
```

```
## [1] 0.024805
```

(5) 99% of the population is within 2.576 standard deviations of the mean

```r
interval <- which(x > (80 - 2.576*5) & x < 80 + (2.576*5))
length(interval)/length(x)
```

```
## [1] 0.990002
```

(6) 99.9% of the population is within 3.29 standard deviations of the mean

```r
interval <- which(x > (80 - 3.29*5) & x < 80 + (3.29*5))
length(interval)/length(x)
```

```
## [1] 0.998954
```

Example 4.2

To find the $\theta(z)$ in the table of the normal distribution function in R, you can use:

```r
pnorm(q = -1.60, mean = 0, sd = 1) #find the theta(-1.60) value
```

```
## [1] 0.05479929
```

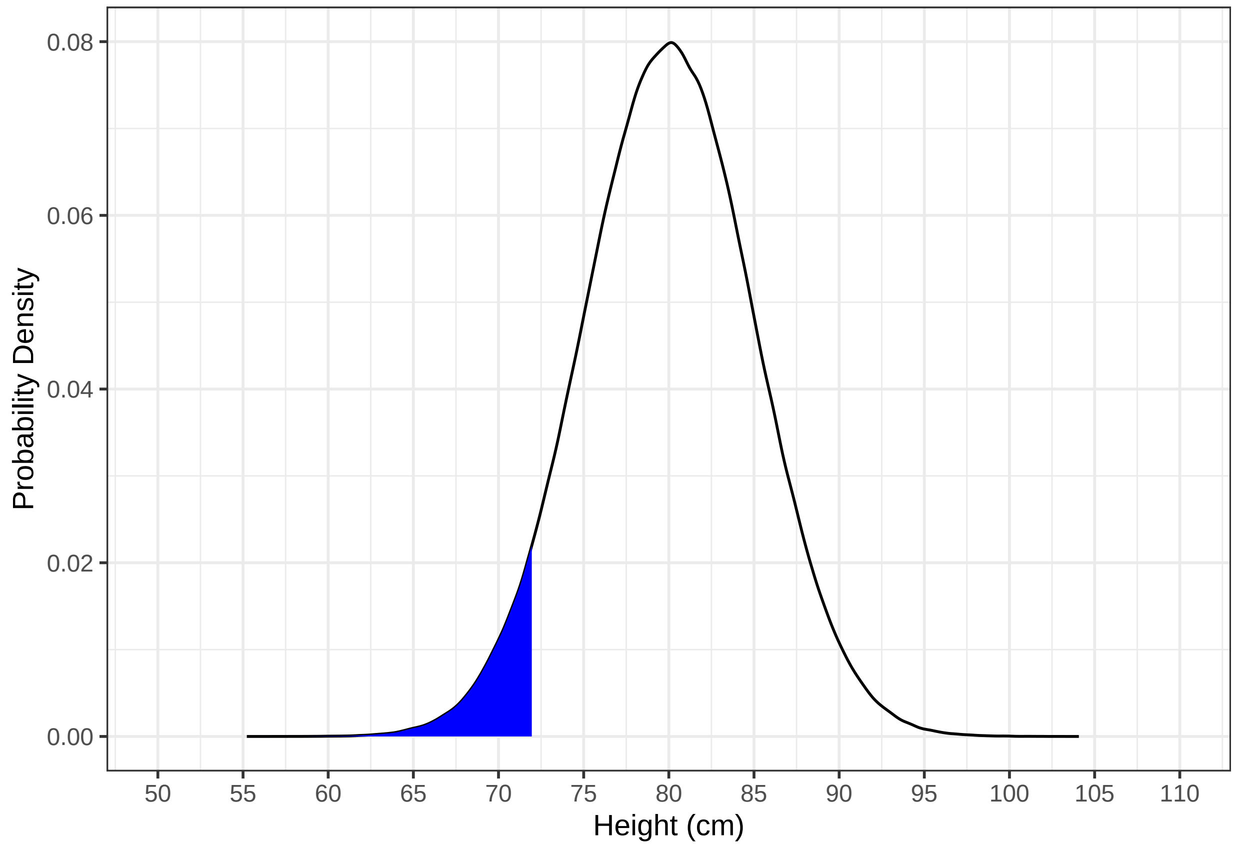
Otherwise, you can just plug the parameters distribution and avoid the use of the table. Find the proportion of plants having heights using `pnorm` function: (1) less than 72 cm

```r
pnorm(q = 72, mean = 80, sd = 5)
```

```
## [1] 0.05479929
```

This value corresponds to the area of the following colored area in probability density plot:

```
## Warning: Removed 336 rows containing missing values (position_stack).
```



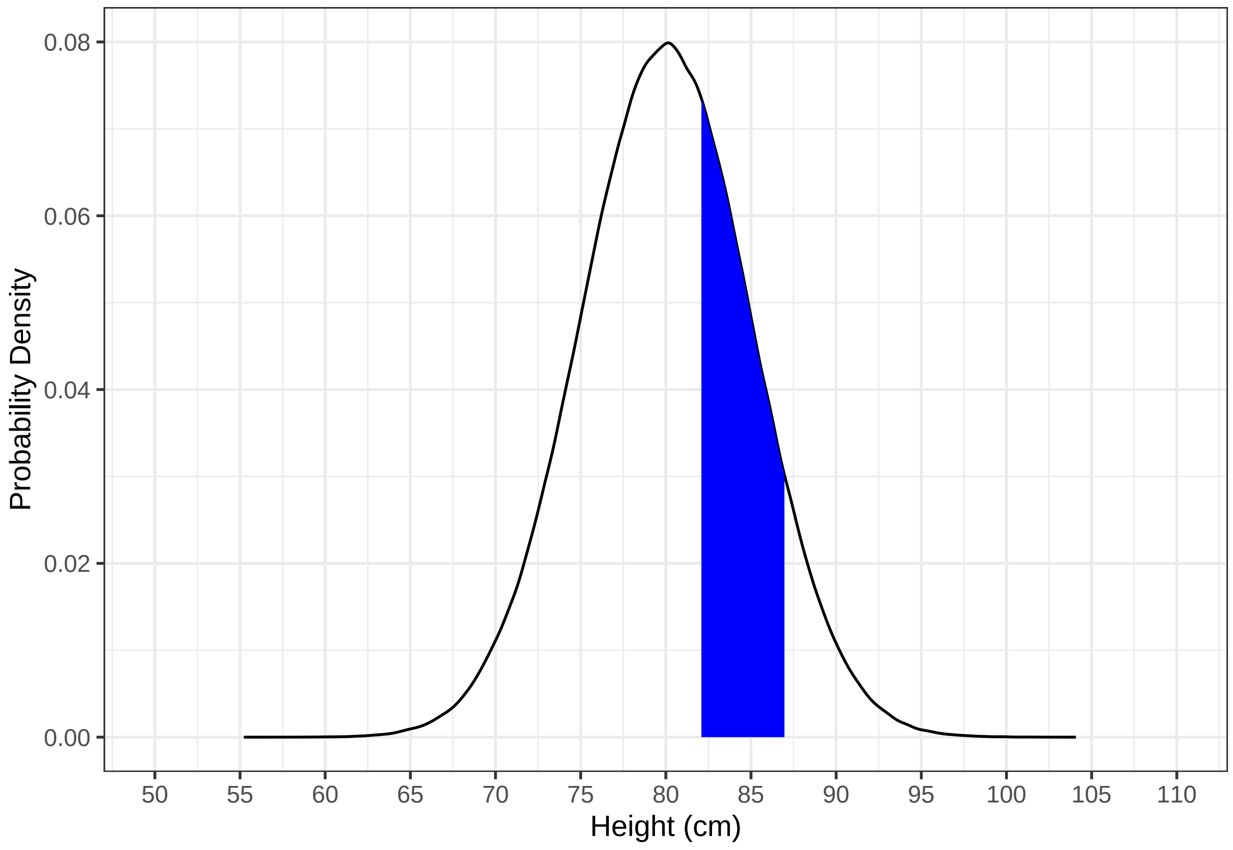
(2) between 82 and 87 cm

```r
pnorm(q = 87, mean = 80, sd = 5) - pnorm(q = 82, mean = 80, sd = 5) #attention for the minus signal
```

```
## [1] 0.2638216
```

This value corresponds to the area of the following colored area in probability density plot:

```
## Warning: Removed 460 rows containing missing values (position_stack).
```



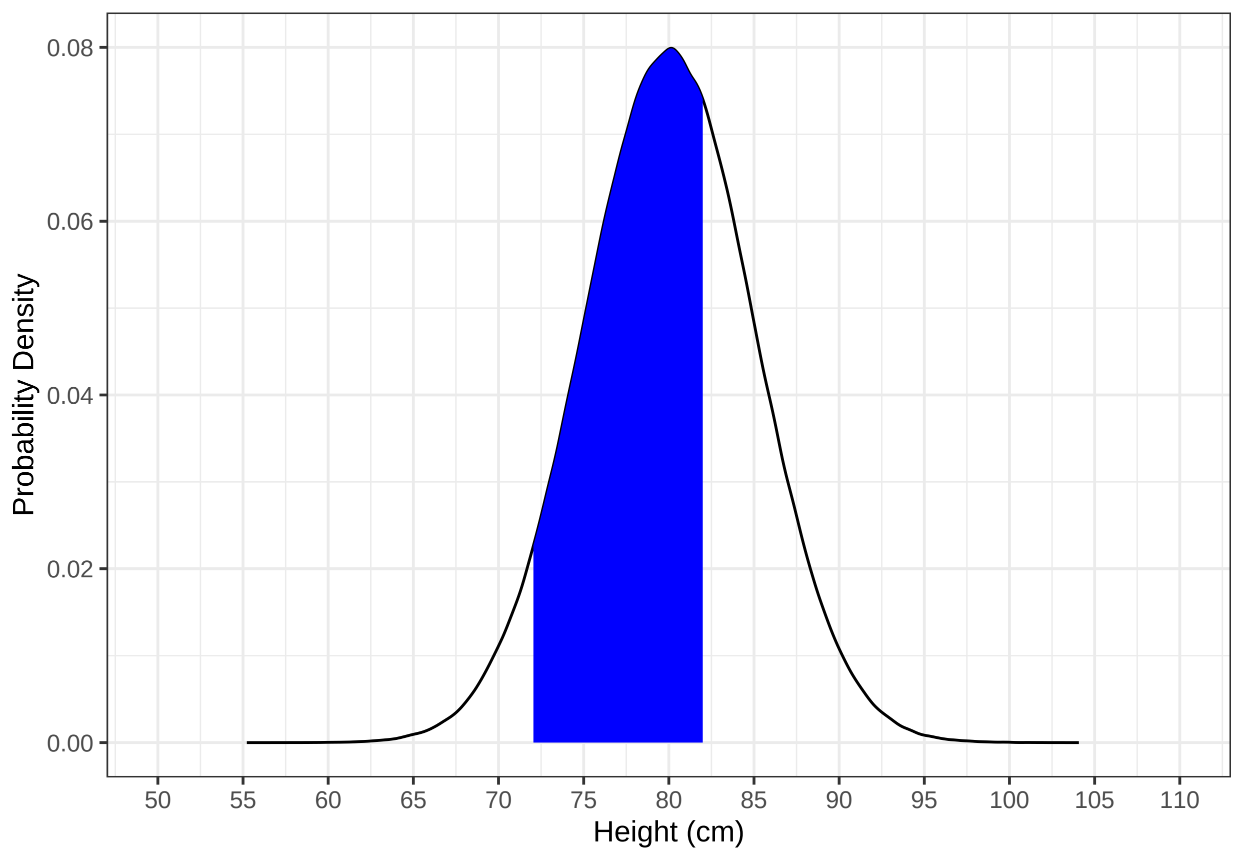
(3) between 72 and 82 cm

```r
pnorm(q = 82, mean = 80, sd = 5) - pnorm(q = 72, mean = 80, sd = 5) #attention for the minus signal
```

```
## [1] 0.6006224
```

This value corresponds to area of the following colored area in probability density plot:

```
## Warning: Removed 407 rows containing missing values (position_stack).
```



Example 4.3

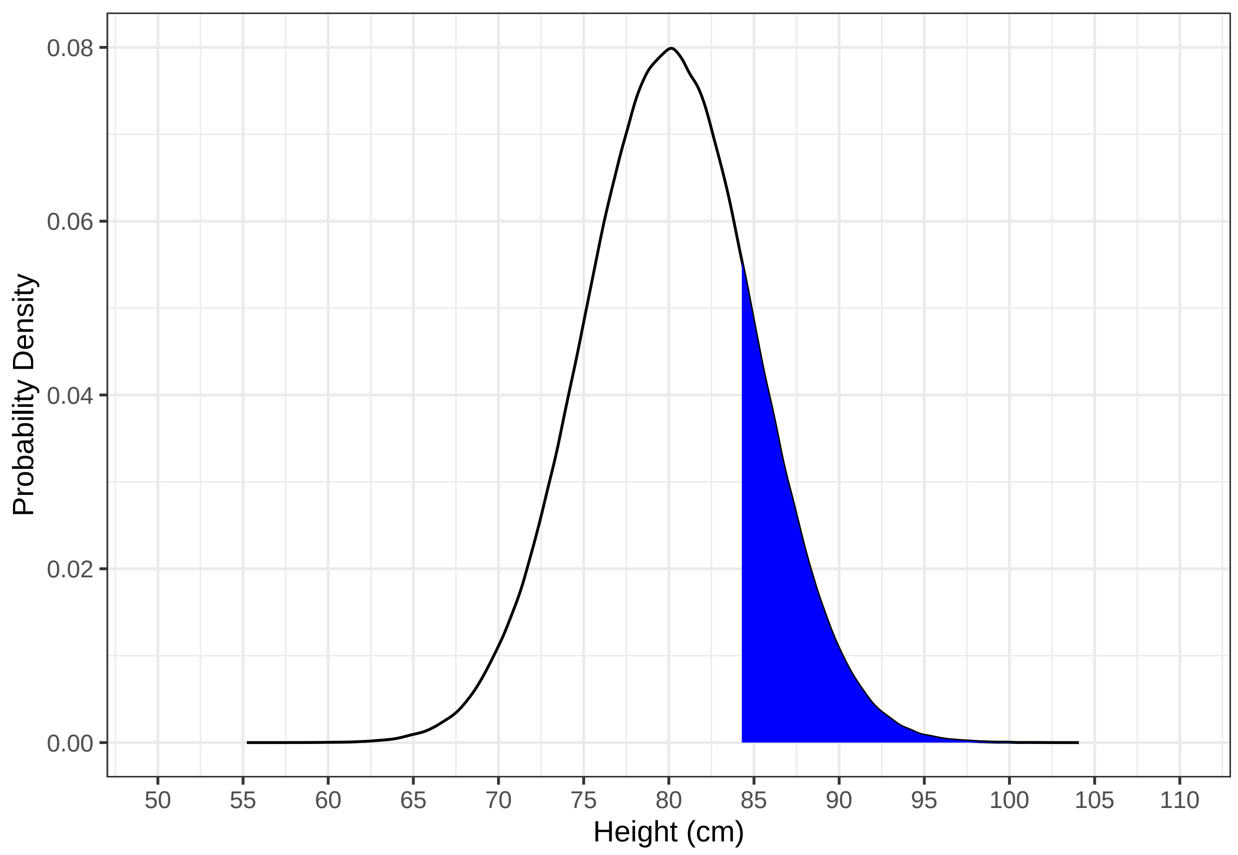
(a) Above what height are the top 20% of plants?

```
qnorm(p = 0.8, mean = 80, sd = 5) #80% of the points are below, then, 20% are above.
```

```
## [1] 84.20811
```

Top 20% of the data is the following colored area in probability density plot:

```
## Warning: Removed 304 rows containing missing values (position_stack).
```



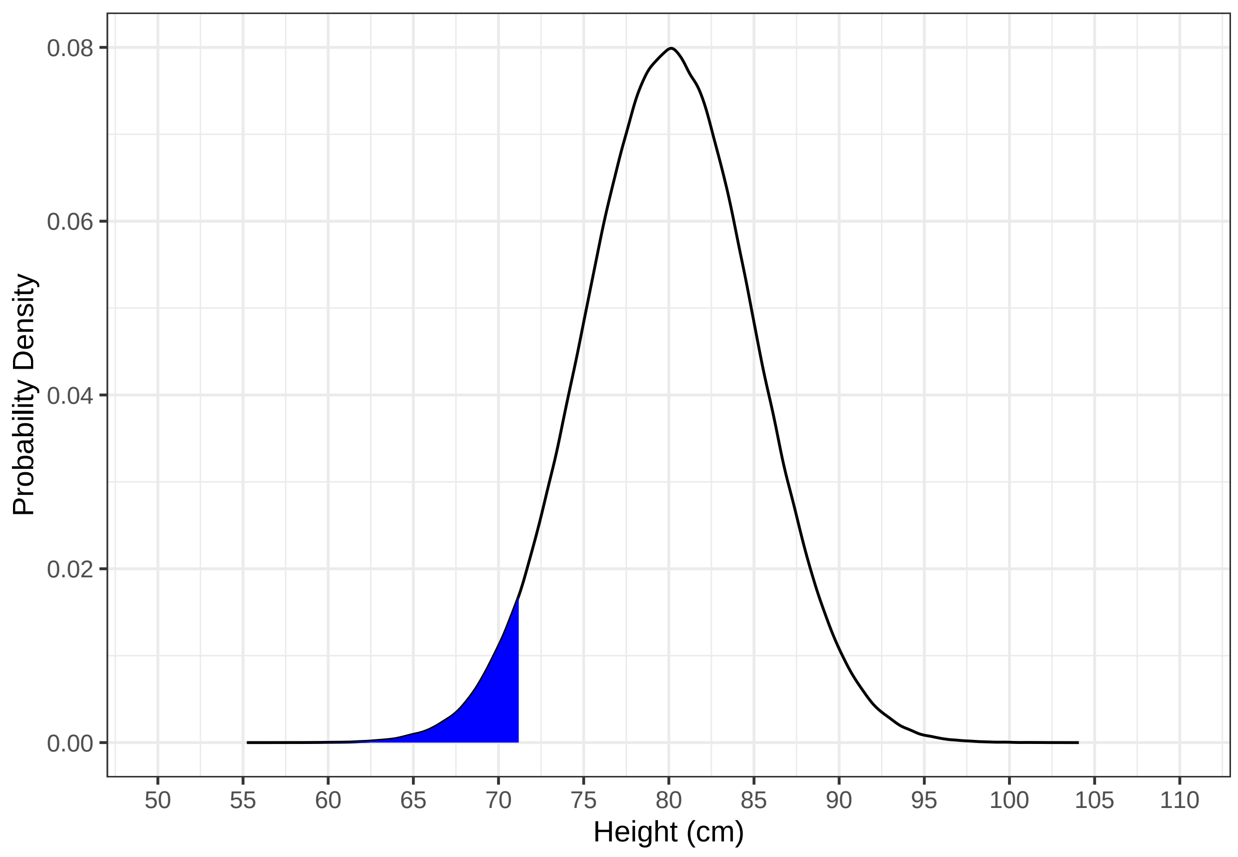
(b) Below what height are the bottom 4% of plants?

```
qnorm(p = 0.04, mean = 80, sd = 5) #80% of the points are below, then, 20% are above.
```

```
## [1] 71.24657
```

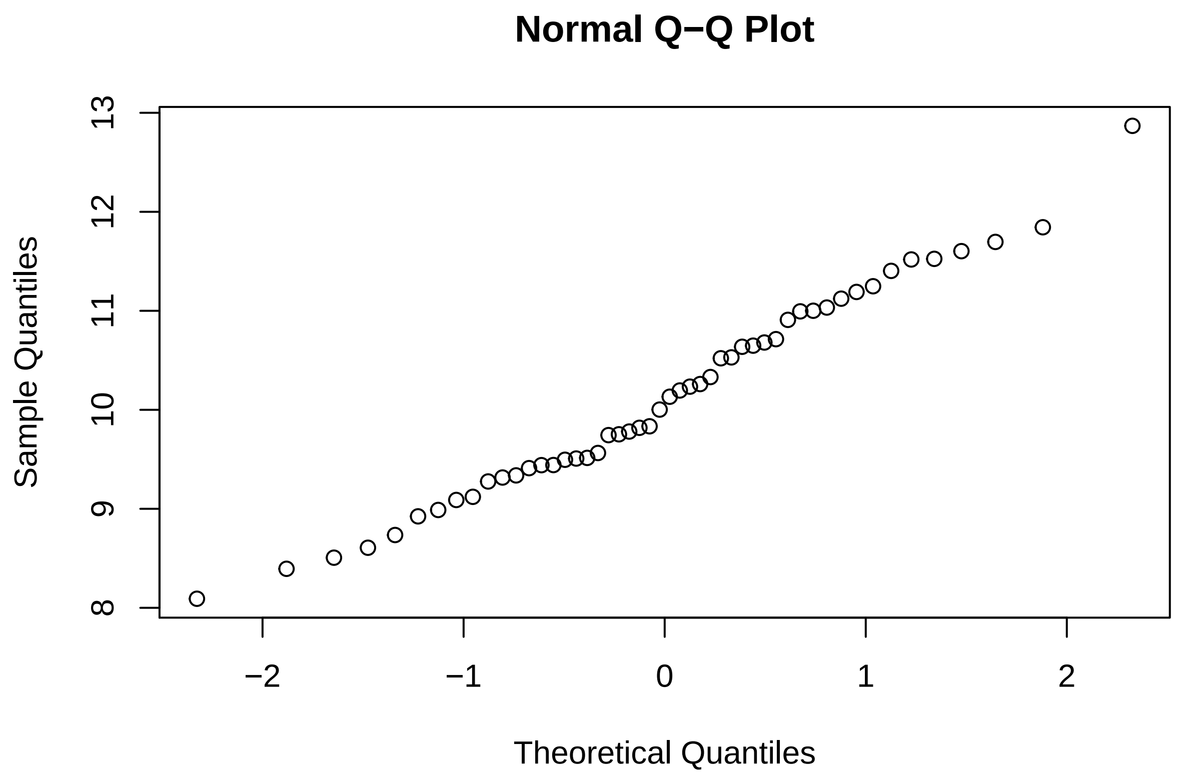
Bottom 4% of the data is the following colored area in probability density plot:

```
## Warning: Removed 344 rows containing missing values (position_stack).
```



Normal probability plot

```
## Generating some data
x <- rnorm(n = 50, mean = 10, sd = 1)
qqnorm(x)
```



4.5 Example

```
x = c(72.3, 78.9, 82.6, 71.8, 86.1, 80.5, 72.0, 91.8, 77.3, 88.2)
qt(p = 0.975, df = 9) #take the t-table value
```

```
## [1] 2.262157
```

```
## To find the interval you can carry out a t-test
t.test(x, conf.level = 0.95) #CI 95%
```

```
##
##  One Sample t-test
##
## data:  x
## t = 35.864, df = 9, p-value = 5.042e-11
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  75.09445 85.20555
## sample estimates:
## mean of x
##     80.15
```

```
t.test(x, conf.level = 0.99) #CI 95%
```

```
##
##  One Sample t-test
##
```

```
## data:  x
## t = 35.864, df = 9, p-value = 5.042e-11
## alternative hypothesis: true mean is not equal to 0
## 99 percent confidence interval:
##  72.88714 87.41286
## sample estimates:
## mean of x
##     80.15
```

4.6 Example

```
x = c(171.8, 267.7, 274.7, 203.2, 208.6, 267.2, 184.1, 234.5)
qt(p = 0.975, df = 7) #take the t-table value
```

```
## [1] 2.364624
```

```
## To find the interval you can carry out a t-test
t.test(x, conf.level = 0.95) #CI 95%
```

```
##
##  One Sample t-test
##
## data:  x
## t = 15.877, df = 7, p-value = 9.536e-07
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  192.7453 260.2047
## sample estimates:
## mean of x
##   226.475
```

Chapter 5: Introduction to Hypothesis Testing

5.1 Example

```
x = c(2.6, 2.1, 2.5, 2.4, 1.9, 2.3)
t.test(x, mu = 2.0, conf.level = 0.95) #CI 95%
```

```
##
##  One Sample t-test
##
## data:  x
## t = 2.818, df = 5, p-value = 0.0372
## alternative hypothesis: true mean is not equal to 2
## 95 percent confidence interval:
##  2.026341 2.573659
## sample estimates:
## mean of x
##       2.3
```

```
## p-value = 0.0373 which is lower than 5%, therefore, we reject H0 at the 5% level.
```

5.1 Example

```
x = c(8.1, 8.7, 9.2, 7.8, 8.4, 9.4)
t.test(x, mu = 8, conf.level = 0.95)
```

```
##
##  One Sample t-test
##
## data:  x
## t = 2.3595, df = 5, p-value = 0.0648
## alternative hypothesis: true mean is not equal to 8
## 95 percent confidence interval:
##  7.94631 9.25369
## sample estimates:
## mean of x
##       8.6
```

```
## computed p-value = 0.0648 which is higher than 5%, therefore, we don't reject H0 at the 5% level.
```

5.2 Example

```
A <- c(17.8, 18.5, 12.2, 19.7, 10.8, 11.9, 15.6, 12.5)
B <- c(14.7, 15.2, 12.9, 18.3, 10.1, 12.2, 13.5, 9.9)
diff <- A-B #compute the difference
t.test(diff)
```

```
##
##  One Sample t-test
##
## data:  diff
## t = 2.8446, df = 7, p-value = 0.02488
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.2573084 2.7926916
## sample estimates:
## mean of x
##     1.525
```

```
## computed p-value = 0.02488 which is lower than 5%, therefore, we reject H0 at the 5% level.
```

Chapter 6: Comparison of Two Independent Sample Means

6.1 Example

```r
Control = c(2.65, 3.28, 2.62, 2.84, 2.61, 2.80, 2.23, 2.45, 2.95, 3.12)
Growth = c(3.02, 2.21, 2.29, 3.21, 3.30, 3.13, 2.86, 3.35, 2.72, 3.16)
mean(Control)
```

```
## [1] 2.755
```

```r
mean(Growth)
```

```
## [1] 2.925
```

Changing the two values

```r
Control = c(2.65, 3.28, 2.62, 2.84, 2.61, 2.80, 3.23, 2.45, 2.95, 3.12)
Growth = c(3.02, 2.21, 2.29, 3.21, 3.30, 3.13, 2.86, 2.35, 2.72, 3.16)
mean(Control)
```

```
## [1] 2.855
```

```r
mean(Growth)
```

```
## [1] 2.825
```

6.2 Example

```r
New = c(2.6, 2.1, 2.5, 2.4, 1.9, 2.3)
Standard = c(1.7, 2.1, 2.0, 1.8, 2.3, 1.6, 2.0, 2.1, 2.2, 1.9)
t.test(New, Standard, var.equal = TRUE)
```

```
##
##  Two Sample t-test
##
## data:  New and Standard
## t = 2.7056, df = 14, p-value = 0.01707
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.06840468 0.59159532
## sample estimates:
## mean of x mean of y
##      2.30      1.97
```

```r
## computed p-value = 0.01707 which is lower than 5%, therefore, we reject H0 at the 5% level.
```

6.3 Example

```r
var.test(New,Standard)
```

```
##
##  F test to compare two variances
##
## data:  New and Standard
## F = 1.3878, num df = 5, denom df = 9, p-value = 0.6296
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
```

```
##  0.309462 9.271667
## sample estimates:
## ratio of variances
##           1.387755
```

```
## computed p-value = 0.6296 which is higer than 5%, therefore, we don't reject H0 at the 5% level.
```

Chapter 7: Linear Regression and Correlation

7.1 Example

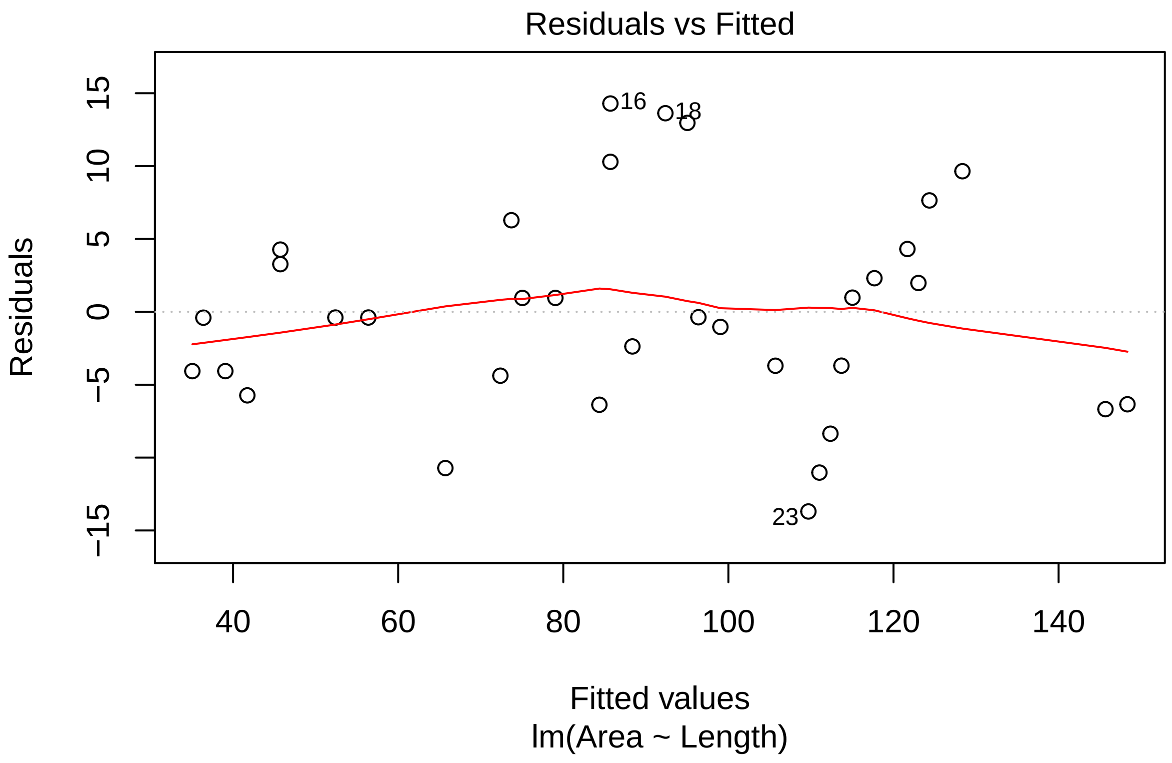
```r
Length = c(22, 23, 25, 27, 30, 30, 35, 38, 45,
           50, 51, 52, 55, 59, 60, 60, 62, 65,
           67, 68, 70, 75, 78, 79, 80, 81,
           82, 84, 87, 88, 89, 92, 105, 107)

Area = c(31, 36, 35, 36, 50, 49, 52, 56, 55,
         68, 80, 76, 80, 78, 96, 100, 86, 106,
         108, 96, 98, 102, 96, 100, 104, 110,
         116, 120, 126, 125, 132, 138, 139, 142)

## The Linear Model (regression)
model = lm(Area ~ Length)
summary(model)
```

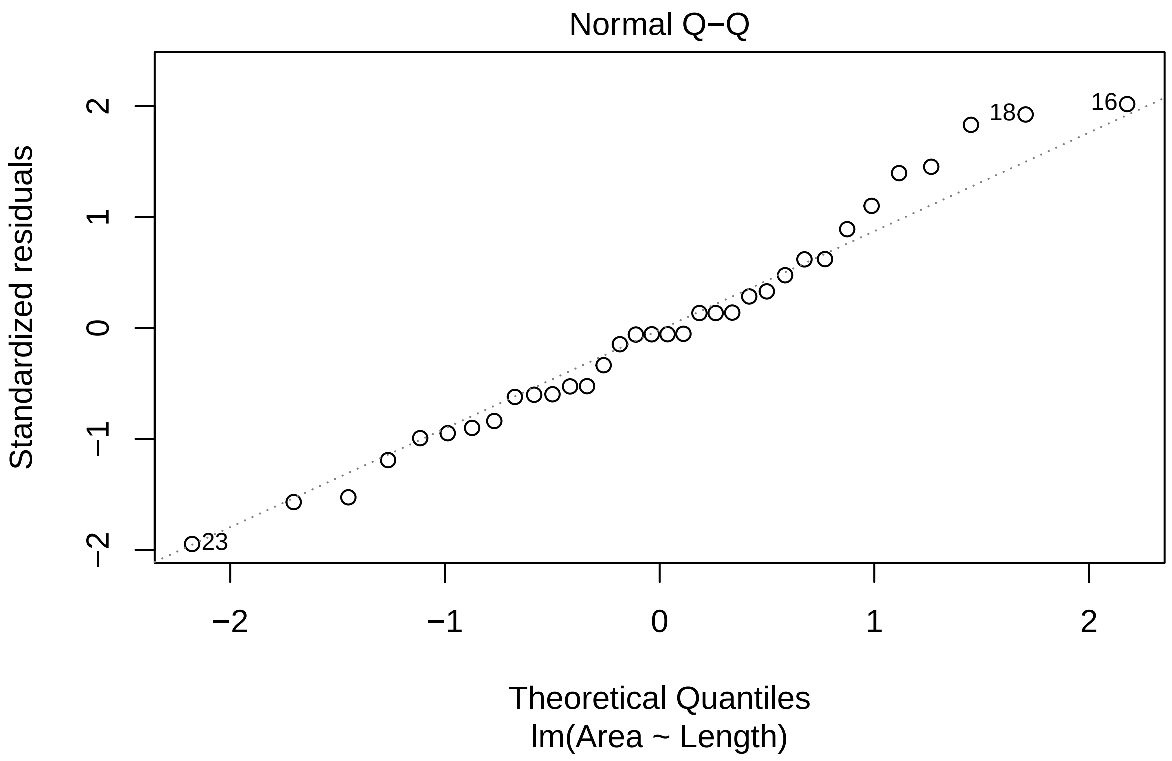
```
##
## Call:
## lm(formula = Area ~ Length)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.6951  -4.3027  -0.3905   4.0217  14.2925
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.74914    3.47886   1.653    0.108
## Length       1.33264    0.05215  25.555   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.189 on 32 degrees of freedom
## Multiple R-squared:  0.9533, Adjusted R-squared:  0.9518
## F-statistic: 653.1 on 1 and 32 DF,  p-value: < 2.2e-16
```

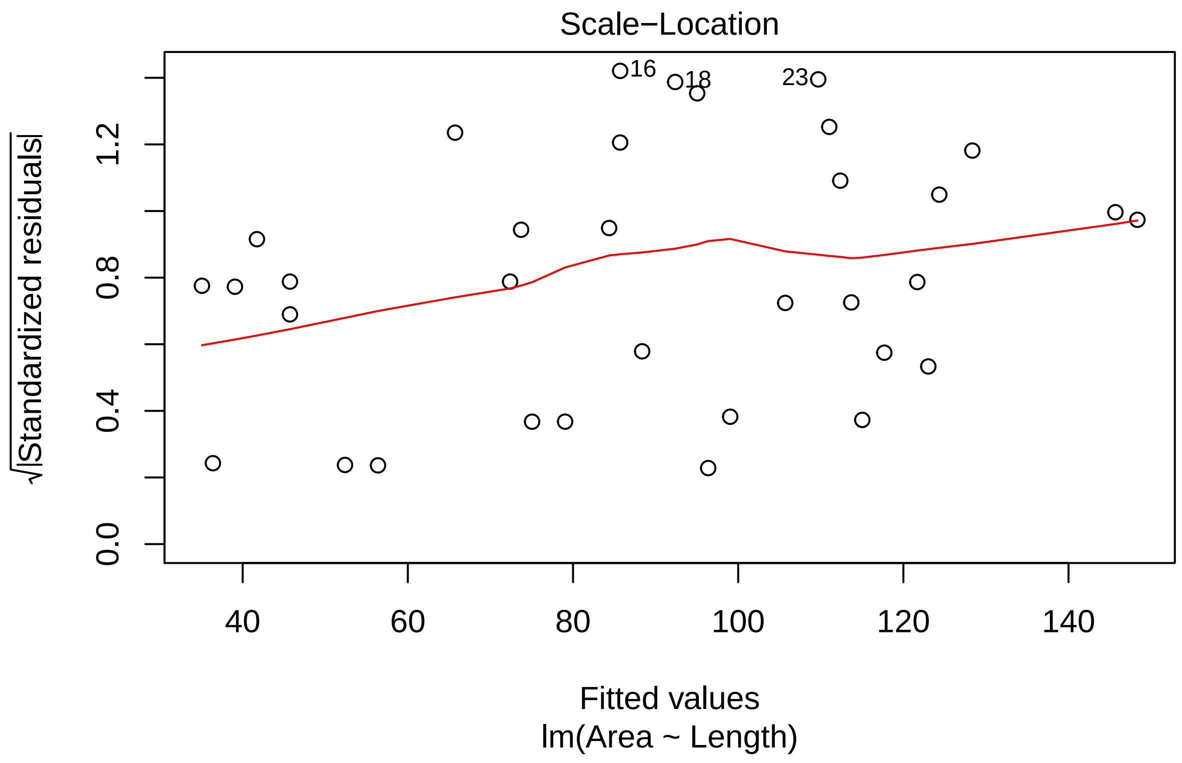
```r
## Analysis of Variance
anova(model)
```

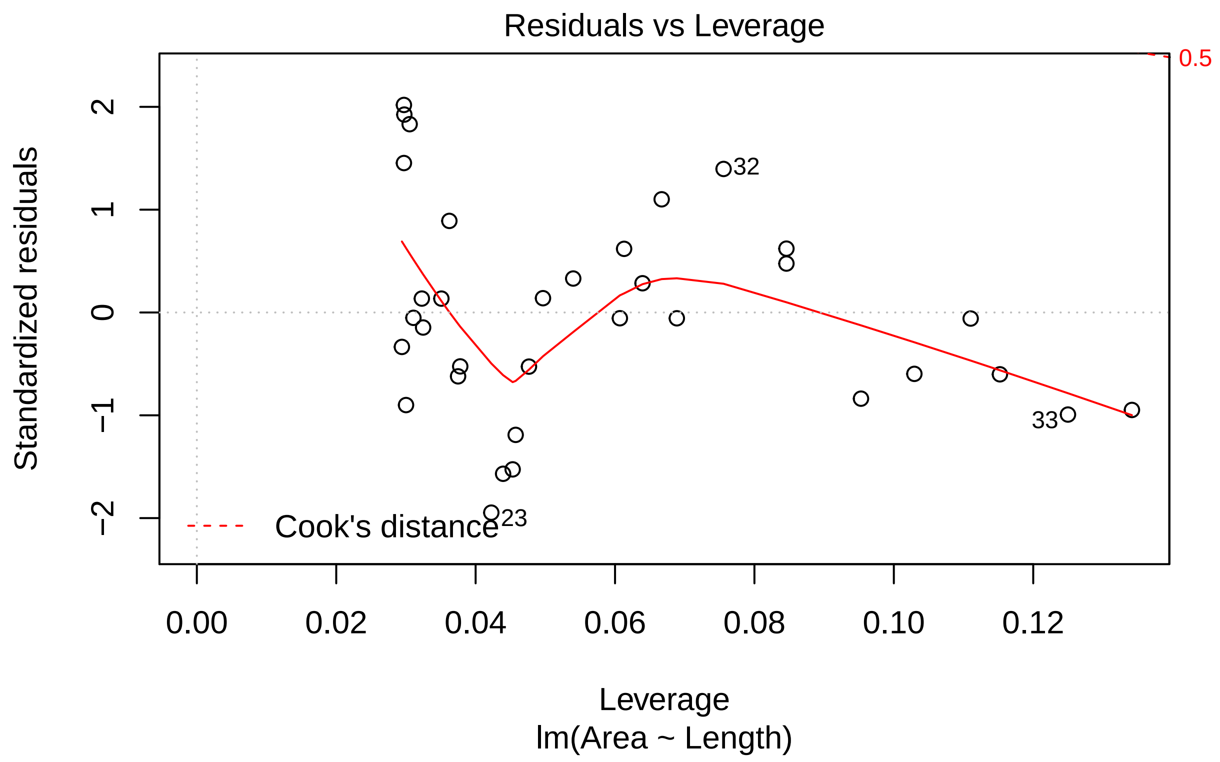
```
## Analysis of Variance Table
##
## Response: Area
##           Df Sum Sq Mean Sq F value    Pr(>F)
## Length     1  33750   33750  653.07 < 2.2e-16 ***
## Residuals 32   1654      52
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
## Residual Diagnostics
plot(model)
```

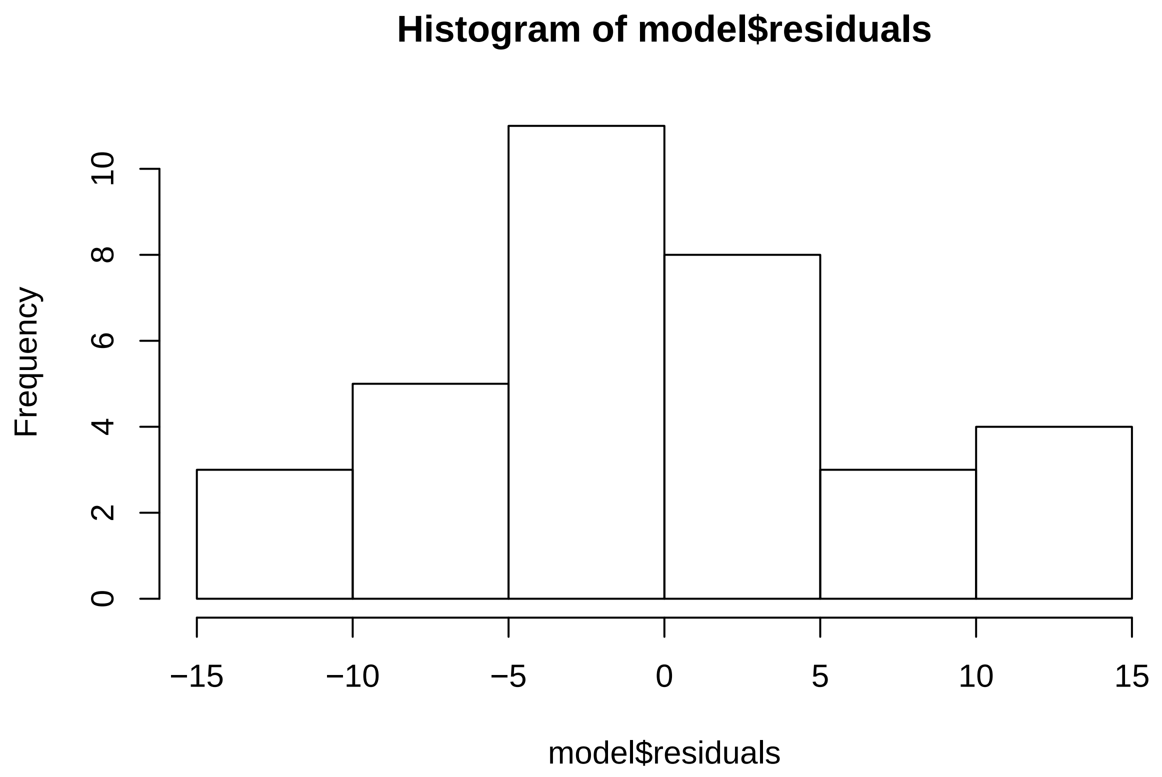

Residuals vs Fitted
Residuals
Fitted values
lm(Area ~ Length)
16
18
23
15
10
5
0
−5
−15
40
60
80
100
120
140


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(Area ~ Length)
23
18
16


Scale−Location
√|Standardized residuals|
0.0
0.4
0.8
1.2
40
60
80
100
120
140
Fitted values
lm(Area ~ Length)
16
18
23



```r
hist(model$residuals)
```



7.2 Example

```r
x = c(0, 25, 50, 75, 100, 125)
y = c(3.70, 4.45, 4.75, 5.20, 5.15, 4.95)
model = lm(y ~ x)
summary(model)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##        1        2        3        4        5        6
## -0.37143  0.12714  0.17571  0.37429  0.07286 -0.37857
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.071429   0.249066  16.347  8.2e-05 ***
## x           0.010057   0.003291   3.056   0.0378 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3441 on 4 degrees of freedom
## Multiple R-squared:  0.7002, Adjusted R-squared:  0.6252
## F-statistic: 9.341 on 1 and 4 DF,  p-value: 0.03779
```

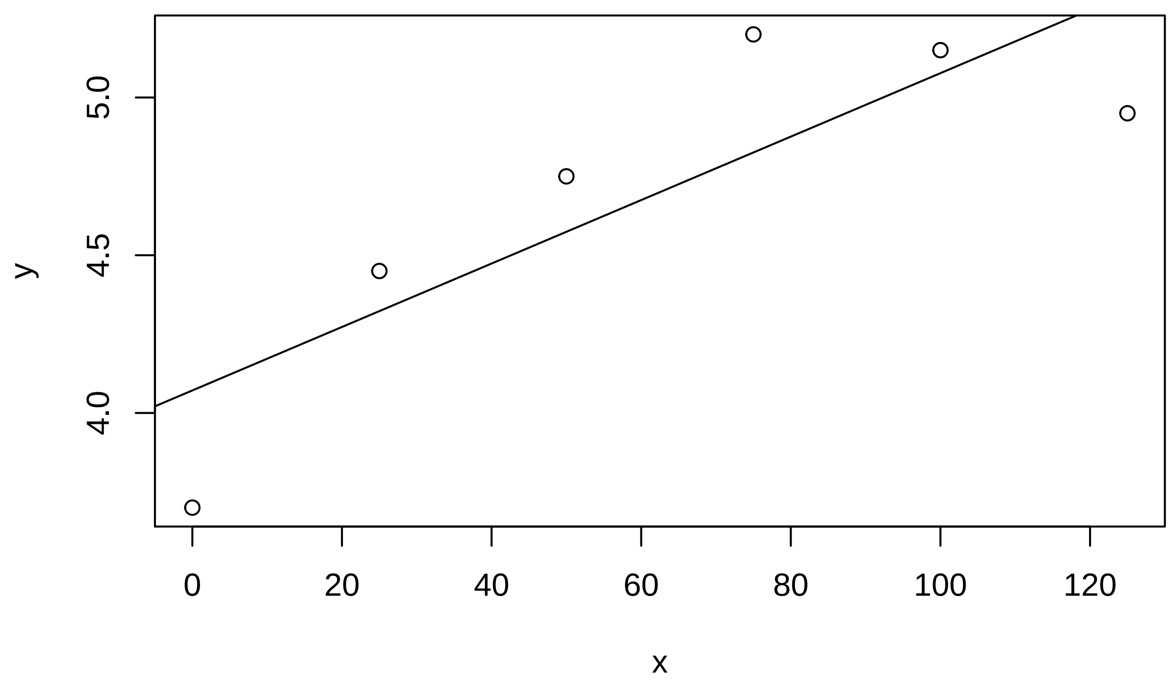
7.3 Example

```
anova(model)
```

```
## Analysis of Variance Table
##
## Response: y
##           Df  Sum Sq Mean Sq F value  Pr(>F)
## x          1 1.10629 1.10629  9.3414 0.03779 *
## Residuals  4 0.47371 0.11843
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Graphics

```
plot(x,y)
abline(model)
```



Chapter 8: Curve Fitting

8.1 Example

```
y = c(3.70, 4.45, 4.75, 5.20, 5.15, 4.95)
x = c(0, 25, 50, 75, 100, 125)
x2 = x^2
model = lm(y ~ x + x2)
summary(model)
```

```
##
## Call:
## lm(formula = y ~ x + x2)
##
## Residuals:
##          1          2          3          4          5          6
## -0.0053571  0.0539286 -0.1171429  0.0814286 -0.0003571 -0.0125000
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.705e+00  8.012e-02   46.24 2.23e-05 ***
## x            3.202e-02  3.015e-03   10.62  0.00178 **
## x2          -1.757e-04  2.315e-05   -7.59  0.00474 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08841 on 3 degrees of freedom
## Multiple R-squared:  0.9852, Adjusted R-squared:  0.9753
## F-statistic: 99.58 on 2 and 3 DF,  p-value: 0.001808
```

```
anova(model)
```

```
## Analysis of Variance Table
##
## Response: y
##           Df  Sum Sq Mean Sq F value   Pr(>F)
## x          1 1.10629 1.10629 141.551 0.001277 **
## x2         1 0.45027 0.45027  57.612 0.004745 **
## Residuals  3 0.02345 0.00782
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

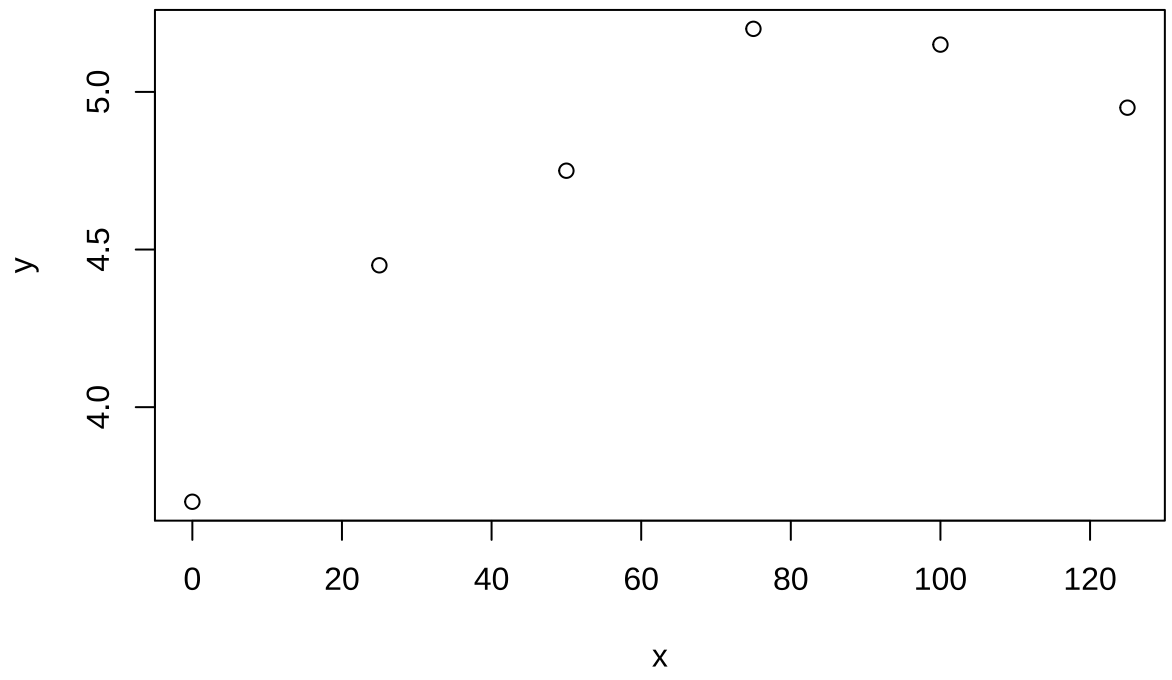
```
model$fitted.values
```

```
##        1        2        3        4        5        6
## 3.705357 4.396071 4.867143 5.118571 5.150357 4.962500
```

```
model$residuals
```

```
##             1             2             3             4             5
## -0.0053571429  0.0539285714 -0.1171428571  0.0814285714 -0.0003571429
##             6
## -0.0125000000
```

```
plot(x,y)
```



8.2 Example

```r
x = c(2.5, 3, 7.2, 7.8, 8.3, 9.8,
      10.8, 15.5, 24., 31.5, 40.2, 64.4)

y = c(5.5, 7.9, 9.8, 11, 13.6, 10.9,
      12.3, 17.5, 20.5, 25.6, 20.4, 26.8)
lnx = log(x)
lny = log(y)


## Model 1: y = a + bx
model1 = lm(y~x)
summary(model1)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.2375 -1.9219 -0.8859  2.3061  6.2033
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   8.9048     1.3901   6.406 7.78e-05 ***
## x              0.3331     0.0539   6.179 0.000104 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.306 on 10 degrees of freedom
## Multiple R-squared:  0.7925, Adjusted R-squared:  0.7717
## F-statistic: 38.18 on 1 and 10 DF,  p-value: 0.0001042
```

```
model1$fitted.values
```

```
##         1         2         3         4         5         6         7         8
##  9.737508  9.904046 11.302967 11.502813 11.669351 12.168966 12.502042 14.067502
##         9        10        11        12
## 16.898651 19.396724 22.294489 30.354939
```

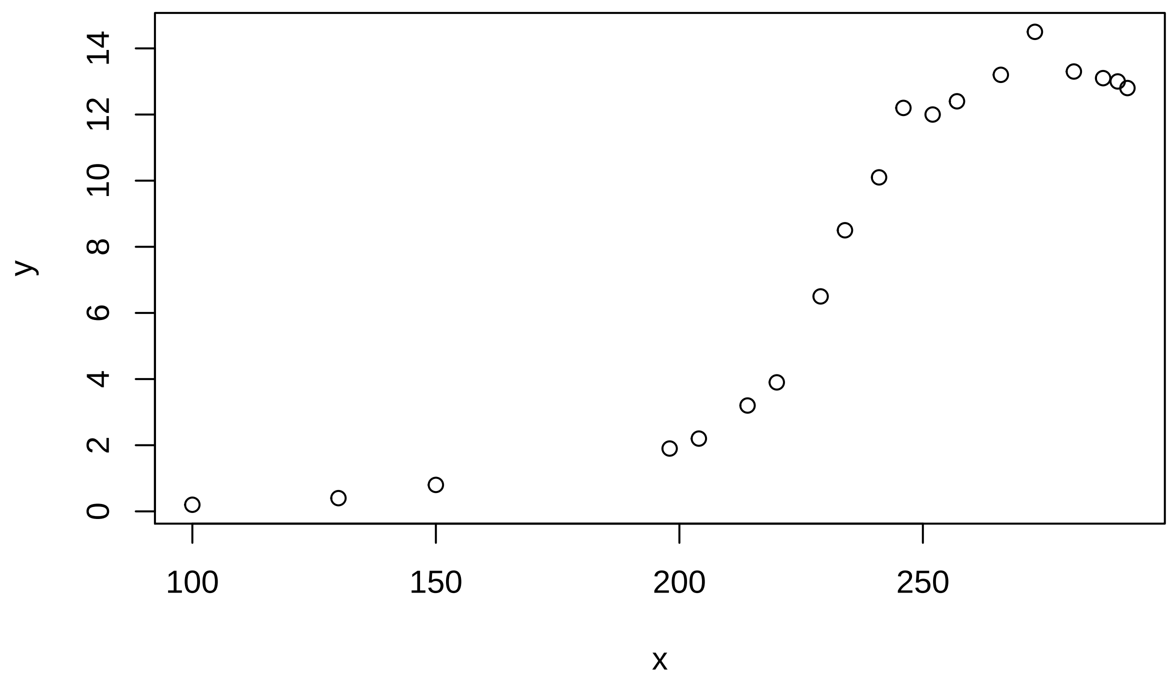
```
model1$residuals
```

```
##          1          2          3          4          5          6          7
## -4.2375081 -2.0040464 -1.5029673 -0.5028132  1.9306486 -1.2689660 -0.2020424
##          8          9         10         11         12
##  3.4324984  3.6013488  6.2032756 -1.8944893 -3.5549387
```

```
## Model 2: ln(y) = a + ln(x)
model2 = lm(lny~lnx)
```

8.3 Example

```
x = c(100, 130, 150, 198, 204, 214, 220, 229, 234,
      241, 246, 252, 257, 266, 273, 281, 287, 290, 292)

y = c(0.2, 0.4, 0.8, 1.9, 2.2, 3.2, 3.9, 6.5, 8.5,
      10.1, 12.2, 12, 12.4, 13.2, 14.5, 13.3, 13.1, 13, 12.8)

plot(x,y)
```



```r
## Logistic growth curve
model = nls(y~alpha/(1+exp(h+c*x)), start=list(alpha=13.54, h=19.70, c=-0.0864))
```

Chapter 9: The Completely Randomized Design

9.1 Example

From Chapter 9 and beyond we should have our data settled in a data frame (spreadsheet) which is a combination of vectors of same size. You can create it internally in R or in a externally with the aid of a spreadsheet software (MS Excel, LibreOffice Calc etc) and import in R with `Import Dataset` from RStudio.

Below, we build the data frame of Example 9.1. You can read the data in different ways. Here, we read the data row after row based on Table 9.3.

```
data = data.frame(variety = c("A", "D", "B", "D", "C",
                              "C", "D", "D", "A", "D",
                              "A", "B", "C", "C", "B",
                              "A", "B", "A", "C", "B"),
                  yield = c(22.2, 23.9, 24.1, 21.7, 25.9,
                            18.4, 24.8, 28.2, 17.3, 26.4,
                            21.2, 30.3, 23.2, 21.9, 27.4,
                            25.2, 26.4, 16.1, 22.6, 34.8))

## To access the vectors use $ sign
print(data)
```

```
##    variety yield
## 1        A  22.2
## 2        D  23.9
## 3        B  24.1
## 4        D  21.7
## 5        C  25.9
## 6        C  18.4
## 7        D  24.8
## 8        D  28.2
## 9        A  17.3
## 10       D  26.4
## 11       A  21.2
## 12       B  30.3
## 13       C  23.2
## 14       C  21.9
## 15       B  27.4
## 16       A  25.2
## 17       B  26.4
## 18       A  16.1
## 19       C  22.6
## 20       B  34.8
```

```
data$variety
```

```
##  [1] A D B D C C D D A D A B C C B A B A C B
## Levels: A B C D
```

```
data$yield
```

```
##  [1] 22.2 23.9 24.1 21.7 25.9 18.4 24.8 28.2 17.3 26.4 21.2 30.3 23.2 21.9 27.4
## [16] 25.2 26.4 16.1 22.6 34.8
```

```
## Always check the structure of your data:
str(data)
```

```
## 'data.frame':    20 obs. of  2 variables:
##  $ variety: Factor w/ 4 levels "A","B","C","D": 1 4 2 4 3 3 4 4 1 4 ...
##  $ yield  : num  22.2 23.9 24.1 21.7 25.9 18.4 24.8 28.2 17.3 26.4 ...
```

```
#num = quantitative variable
#factor = qualitative variable
```

Be careful, always check your data structure, if the variable types are correct, if the number of factor levels are correct etc.

```
tapply(data$yield, data$variety, length) #N
```

```
## A B C D
## 5 5 5 5
```

```
tapply(data$yield, data$variety, mean) #Mean
```

```
##    A    B    C    D
## 20.4 28.6 22.4 25.0
```

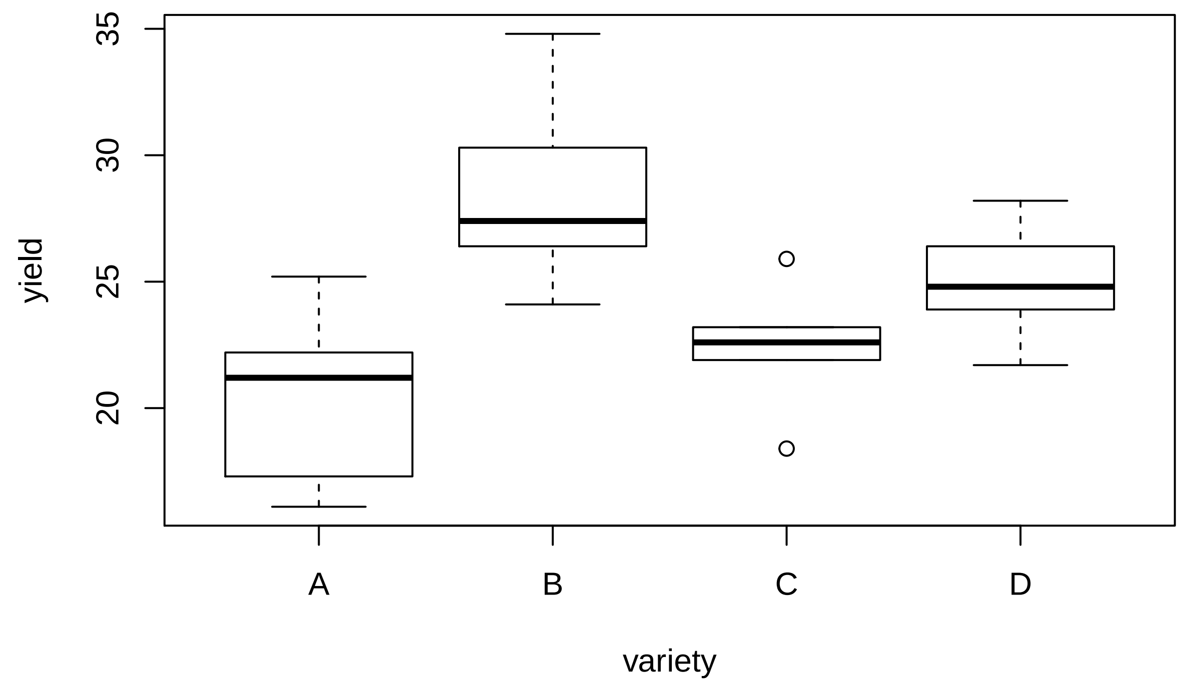
```
tapply(data$yield, data$variety, sd) #StDev
```

```
##        A        B        C        D
## 3.708773 4.118859 2.700926 2.466779
```

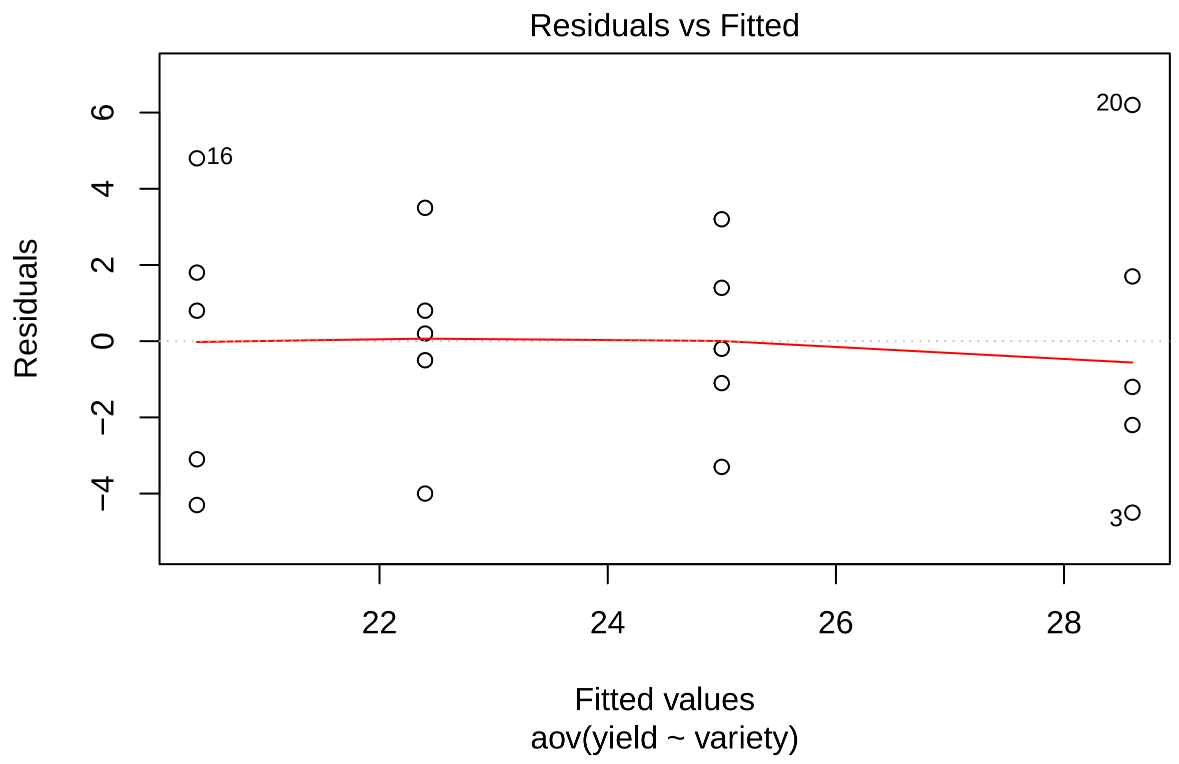
```
model = aov(yield ~ variety, data = data)
anova(model)
```

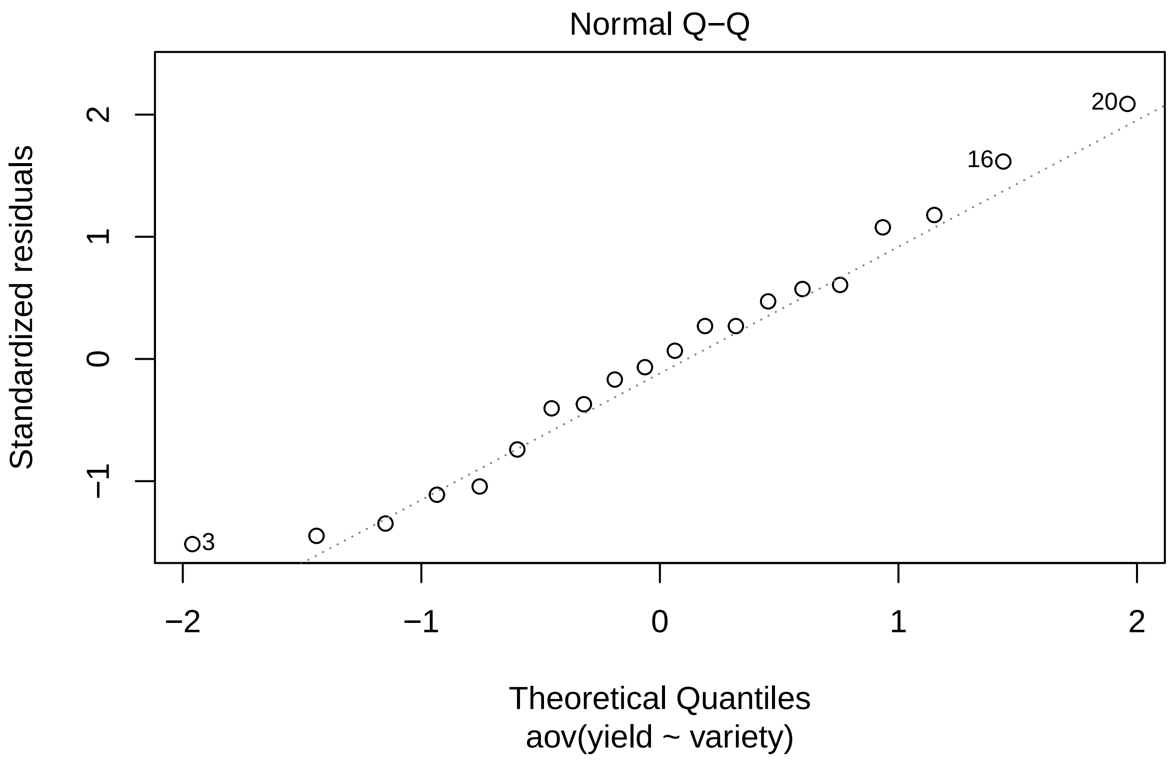
```
## Analysis of Variance Table
##
## Response: yield
##           Df Sum Sq Mean Sq F value  Pr(>F)
## variety    3  188.2  62.733  5.6901 0.00756 **
## Residuals 16  176.4  11.025
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

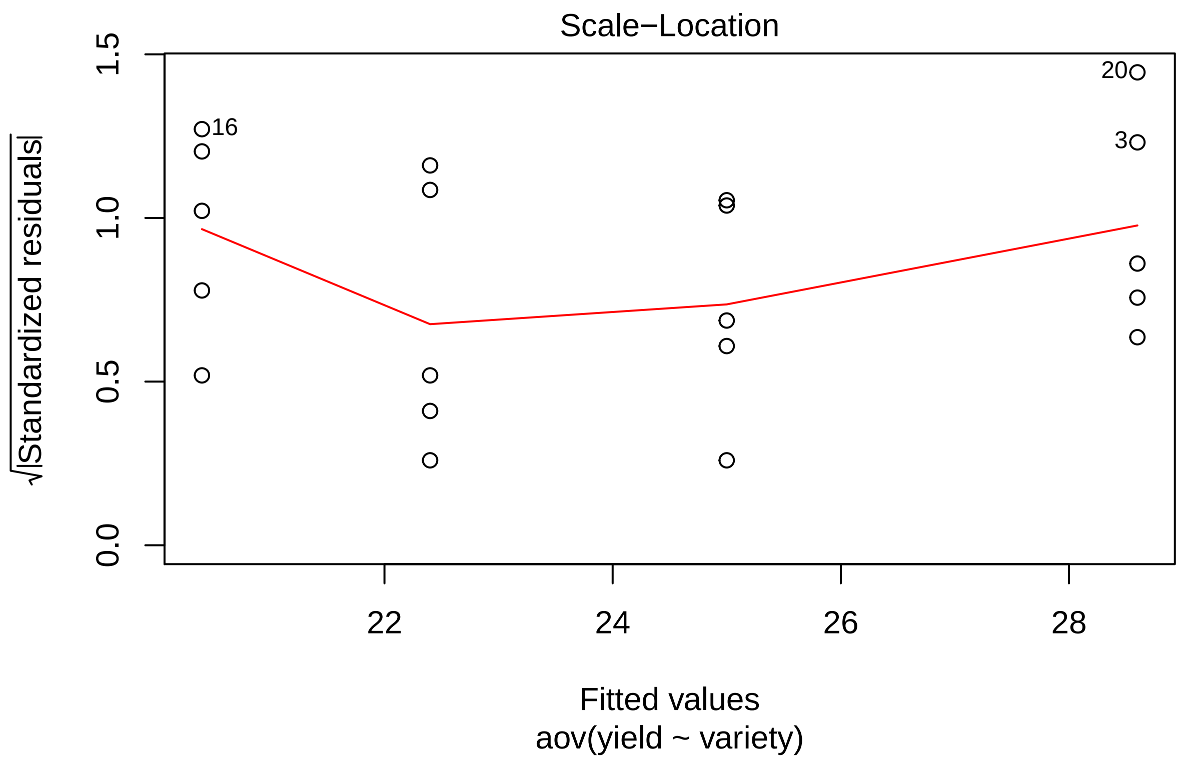
```
boxplot(yield ~ variety, data = data)
```

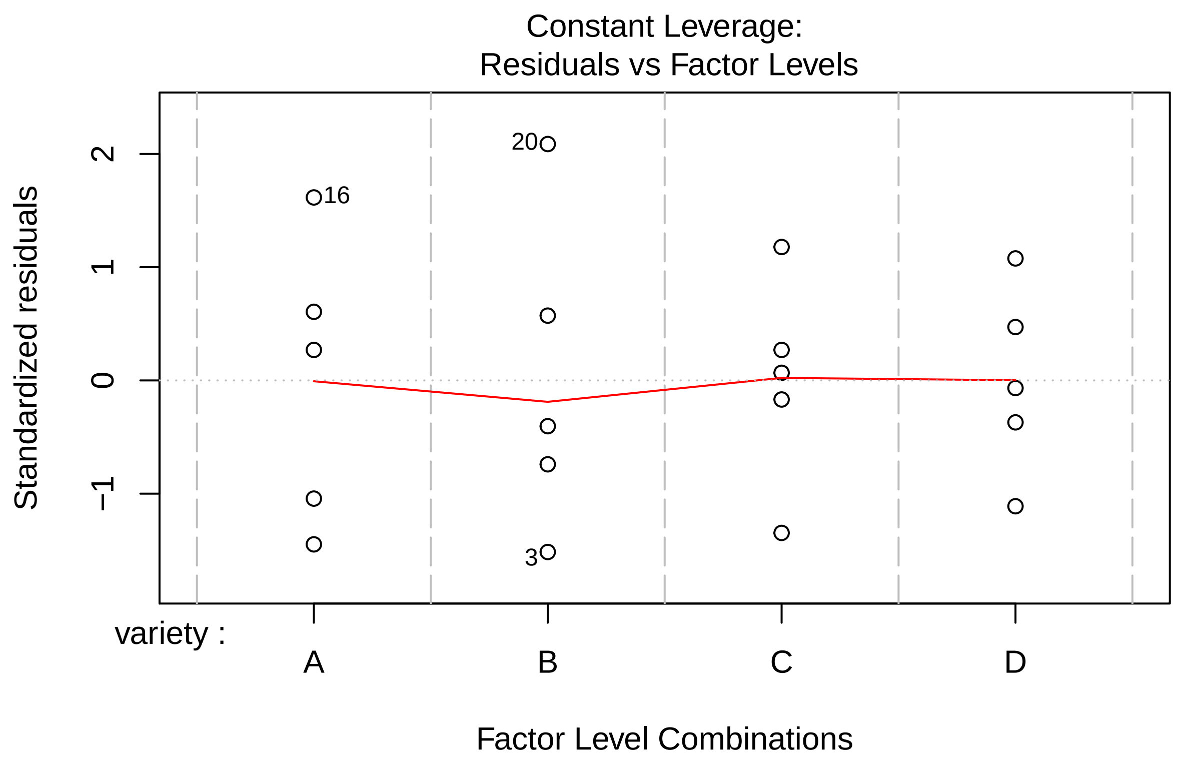


```r
plot(model)
```


Residuals vs Fitted
Residuals
6
4
2
0
−2
−4
16
20
3
22
24
26
28
Fitted values
aov(yield ~ variety)


Normal Q−Q
Standardized residuals
Theoretical Quantiles
aov(yield ~ variety)
3
16
20
−2
−1
0
1
2


Scale−Location
√|Standardized residuals|
1.5
1.0
0.5
0.0
16
20
3
22
24
26
28
Fitted values
aov(yield ~ variety)



9.2 Example

```
data = data.frame(variety = c("A","A","A",
                              "B","B","B","B",
                              "C","C","C","C",
                              "D","D","D","D","D"),
                  yield = c(17.3, 21.2, 16.1,
                            24.1, 30.3, 26.4, 34.8,
                            25.9, 18.4, 21.9, 22.6,
                            23.9, 21.7, 24.8, 28.2, 26.4))

tapply(data$yield, data$variety, length) #N
```

```
## A B C D
## 3 4 4 5
```

```
tapply(data$yield, data$variety, mean) #Mean
```

```
##    A    B    C    D
## 18.2 28.9 22.2 25.0
```

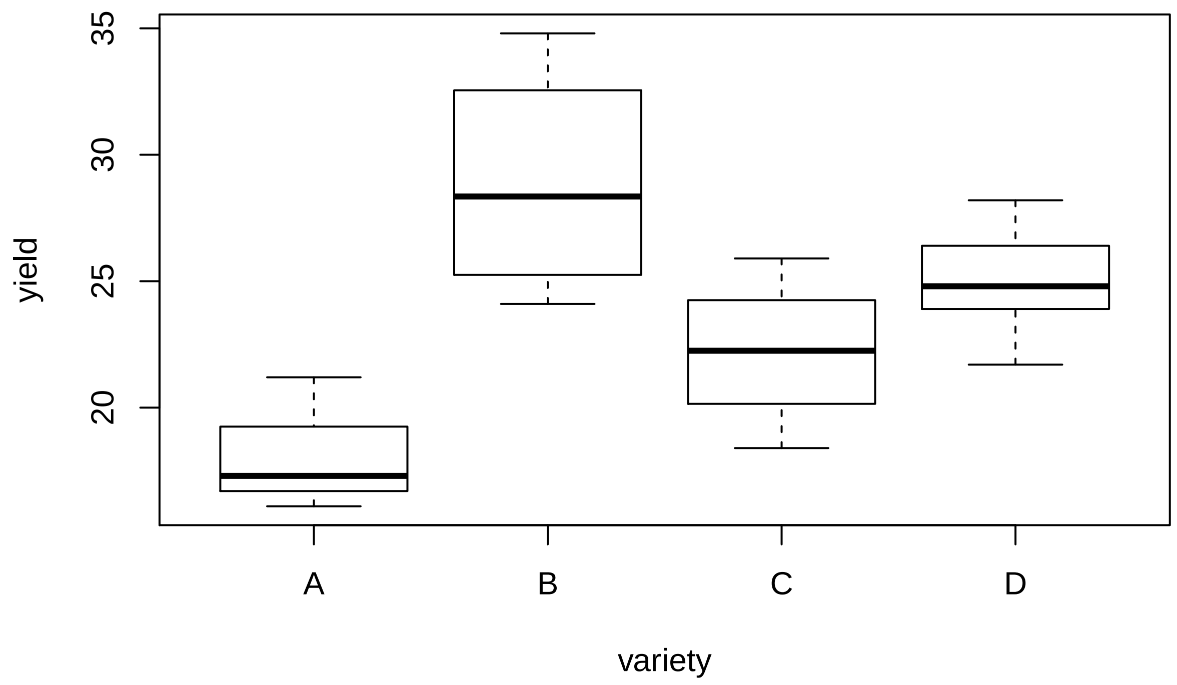
```
tapply(data$yield, data$variety, sd) #StDev
```

```
##        A        B        C        D
## 2.666458 4.692547 3.075711 2.466779
```

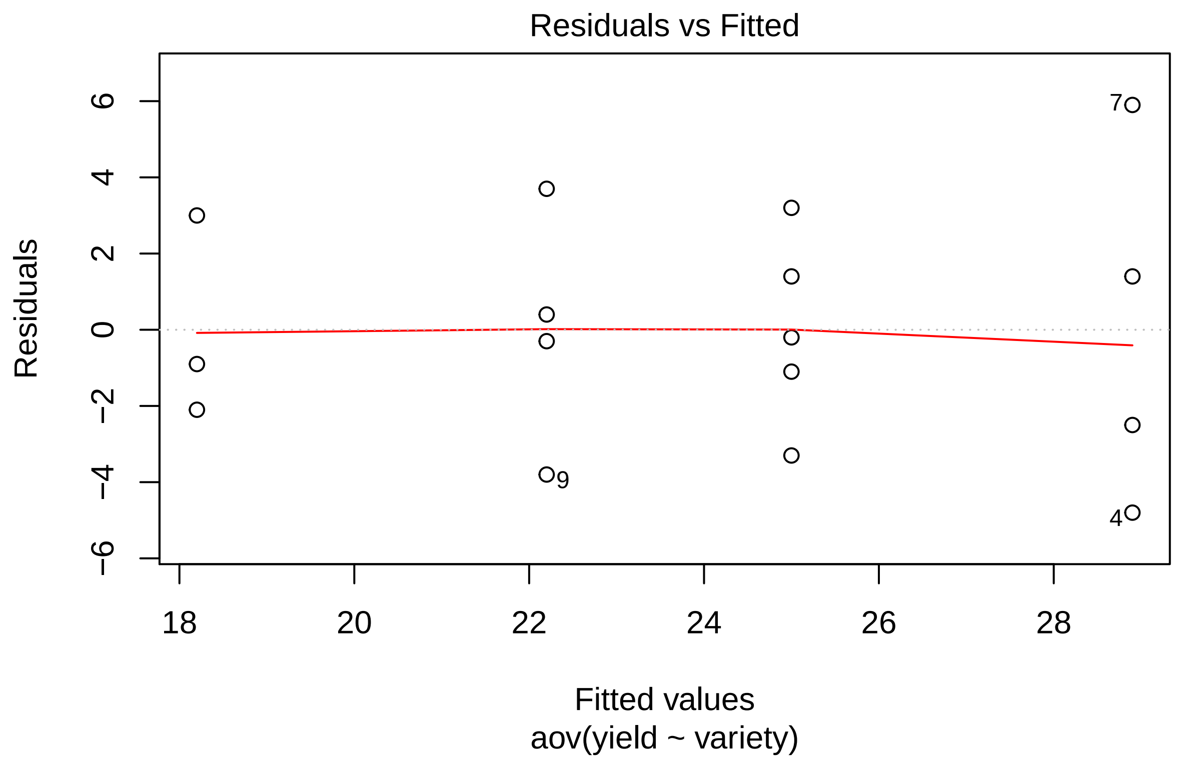
```r
model = aov(yield ~ variety, data = data)
anova(model)
```

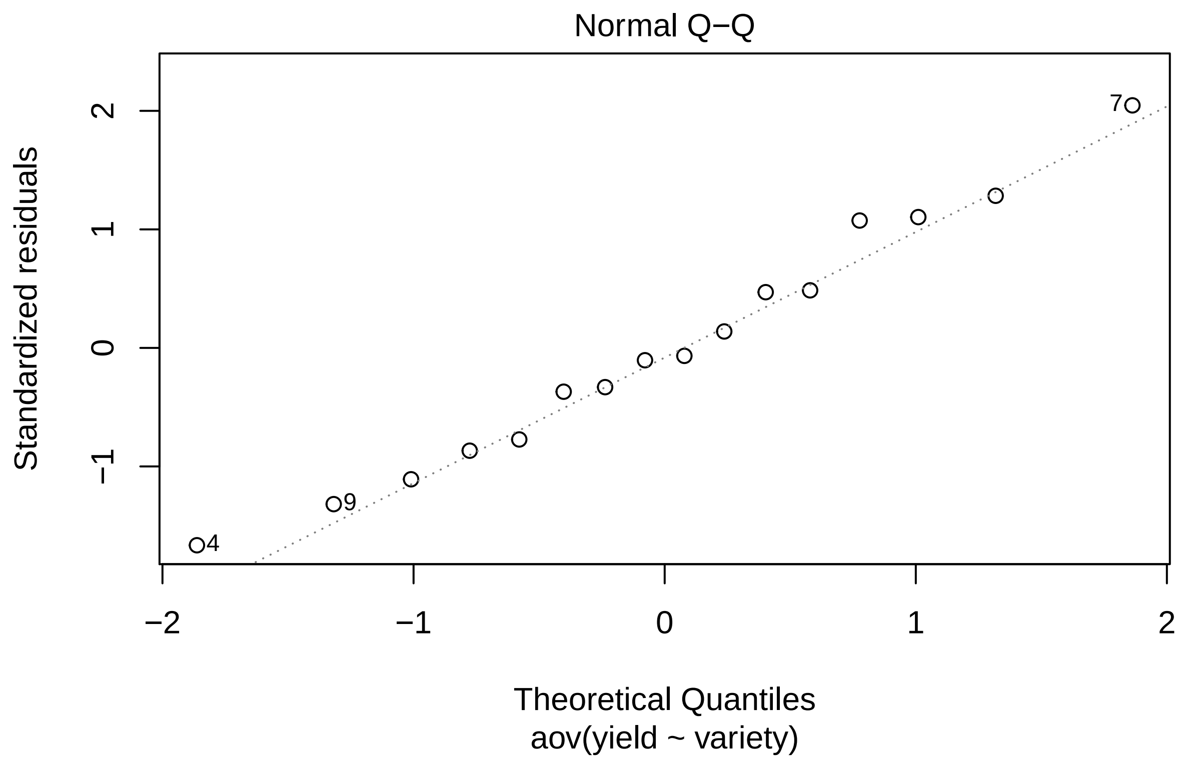
```
## Analysis of Variance Table
##
## Response: yield
##           Df Sum Sq Mean Sq F value   Pr(>F)
## variety    3 214.92  71.640  6.4638 0.007499 **
## Residuals 12 133.00  11.083
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

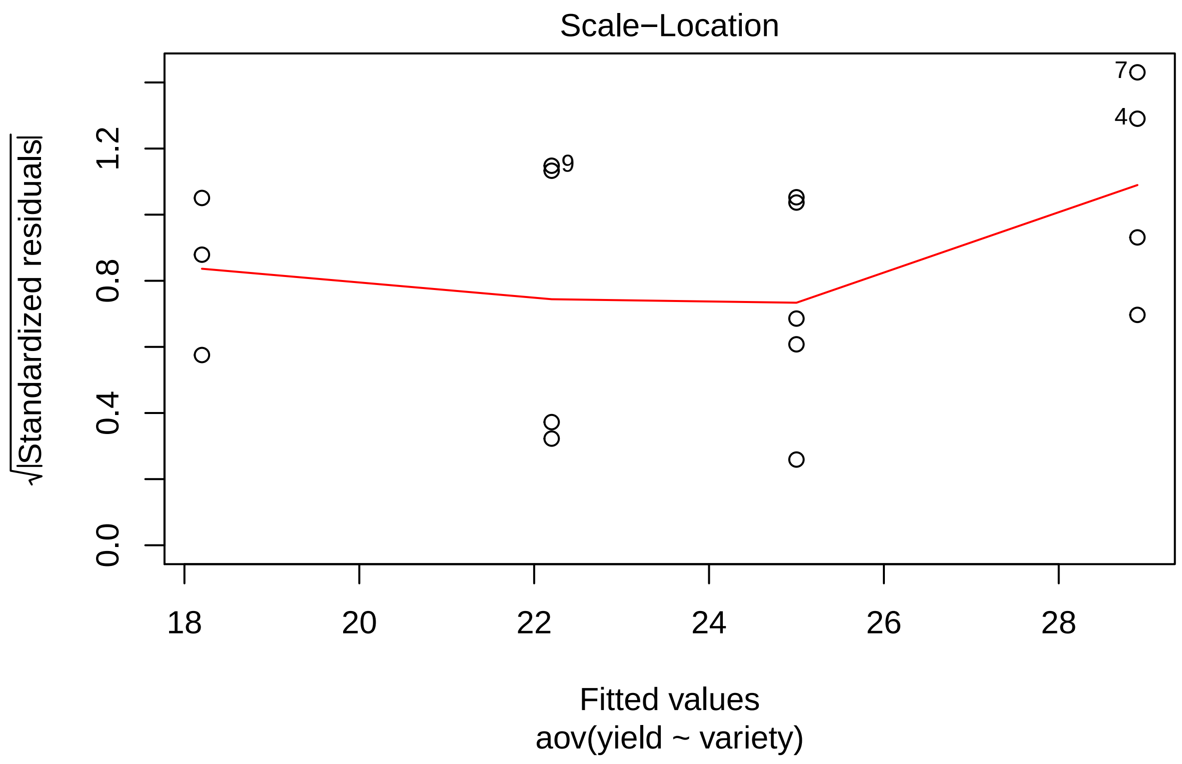
```r
boxplot(yield ~ variety, data = data)
```

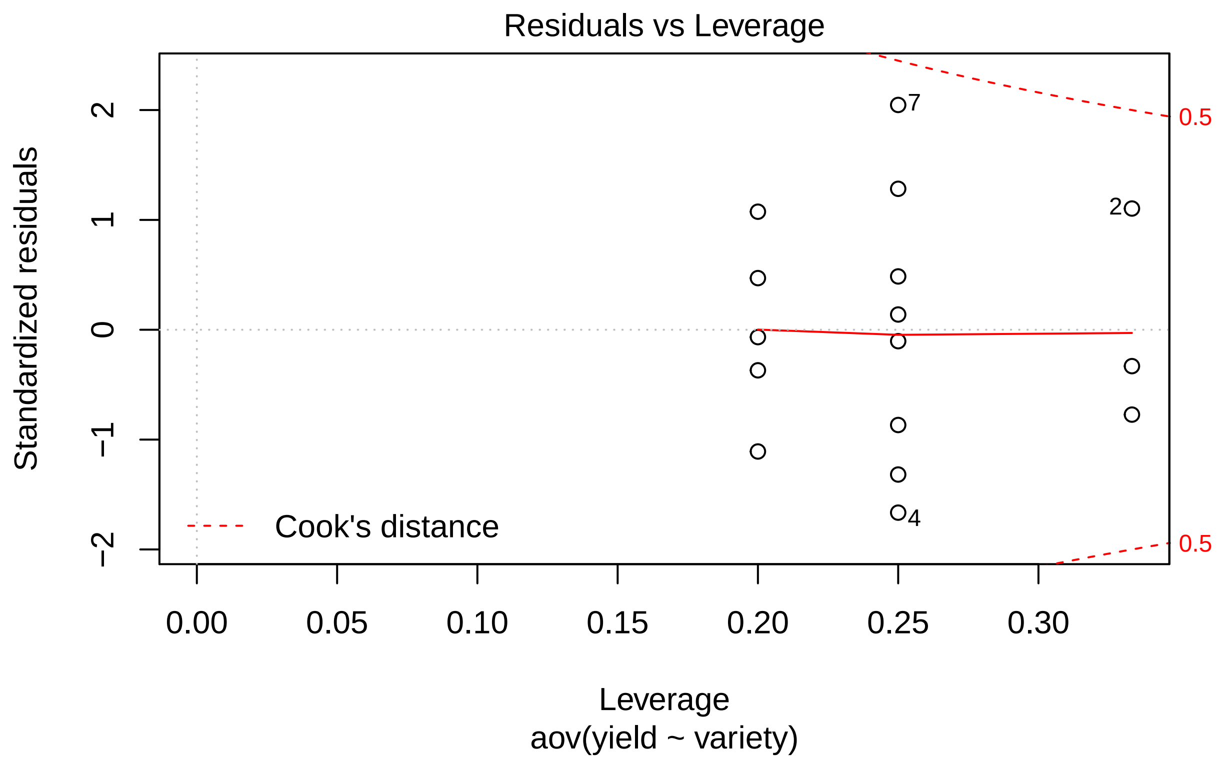


```r
plot(model)
```


Residuals vs Fitted
Residuals
Fitted values
aov(yield ~ variety)
6
4
2
0
−2
−4
−6
18
20
22
24
26
28
7
9
4


Normal Q−Q
Standardized residuals
Theoretical Quantiles
aov(yield ~ variety)
−2
−1
0
1
2
4
9
7


Scale−Location
√|Standardized residuals|
0.0
0.4
0.8
1.2
18
20
22
24
26
28
Fitted values
aov(yield ~ variety)
7
4
9



agricolae package

There are several R package for Experimental Design Analysis for Agricultural and Plant Breeding experiments (more *here*). `agricolae` *is by far the most-used package from this task view (status: October 2017).*

Here, we show how to do the RCD analysis using `agricolae`

To install and load `agricolae`

```
#install.packages("agricolae")
library(agricolae)
```

Sampling the treatments

```
variety = c("A","B","C","D")

field_design = design.crd(variety, r=5)
field_design$book
```

```
##    plots r variety
## 1    101 1       B
## 2    102 2       B
## 3    103 1       C
## 4    104 1       D
## 5    105 1       A
## 6    106 2       C
## 7    107 2       D
## 8    108 3       D
## 9    109 2       A
## 10   110 3       A
## 11   111 4       A
## 12   112 3       B
## 13   113 3       C
## 14   114 4       C
## 15   115 4       B
## 16   116 5       B
## 17   117 5       C
## 18   118 5       A
## 19   119 4       D
## 20   120 5       D
```

Agricolae data analysis for 9.1 example

```
data = data.frame(variety = c("A", "D", "B", "D", "C",
                              "C", "D", "D", "A", "D",
                              "A", "B", "C", "C", "B",
                              "A", "B", "A", "C", "B"),
                  yield = c(22.2, 23.9, 24.1, 21.7, 25.9,
                            18.4, 24.8, 28.2, 17.3, 26.4,
                            21.2, 30.3, 23.2, 21.9, 27.4,
                            25.2, 26.4, 16.1, 22.6, 34.8))

model = aov(yield ~ variety, data = data)
anova(model)
```

```
## Analysis of Variance Table
##
## Response: yield
##           Df Sum Sq Mean Sq F value   Pr(>F)
## variety    3  188.2  62.733  5.6901 0.00756 **
## Residuals 16  176.4  11.025
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

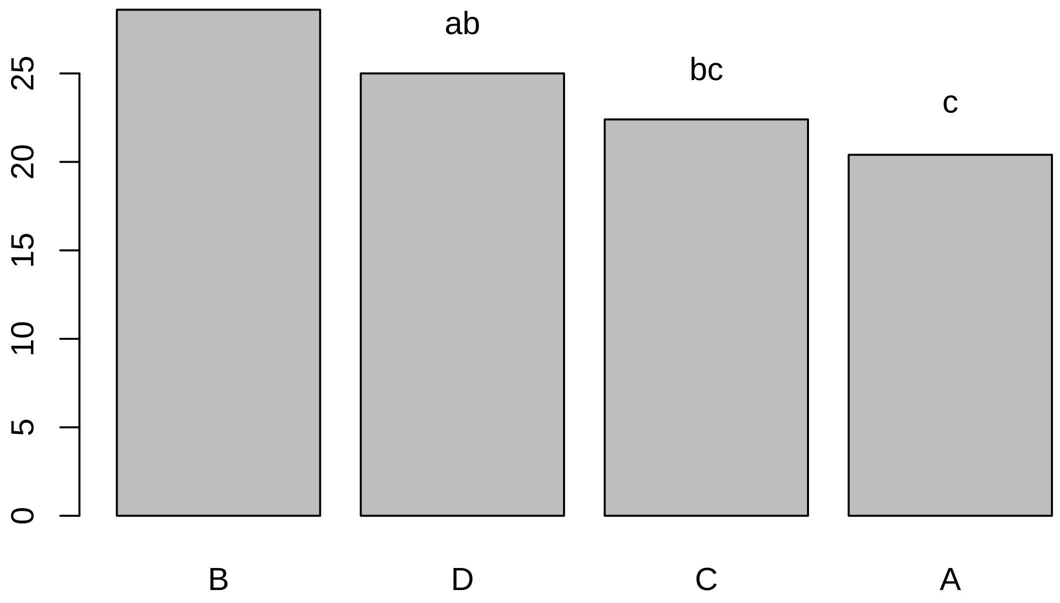
```
## Coefficient of Variation
cv.model(model)
```

```
## [1] 13.77756
```

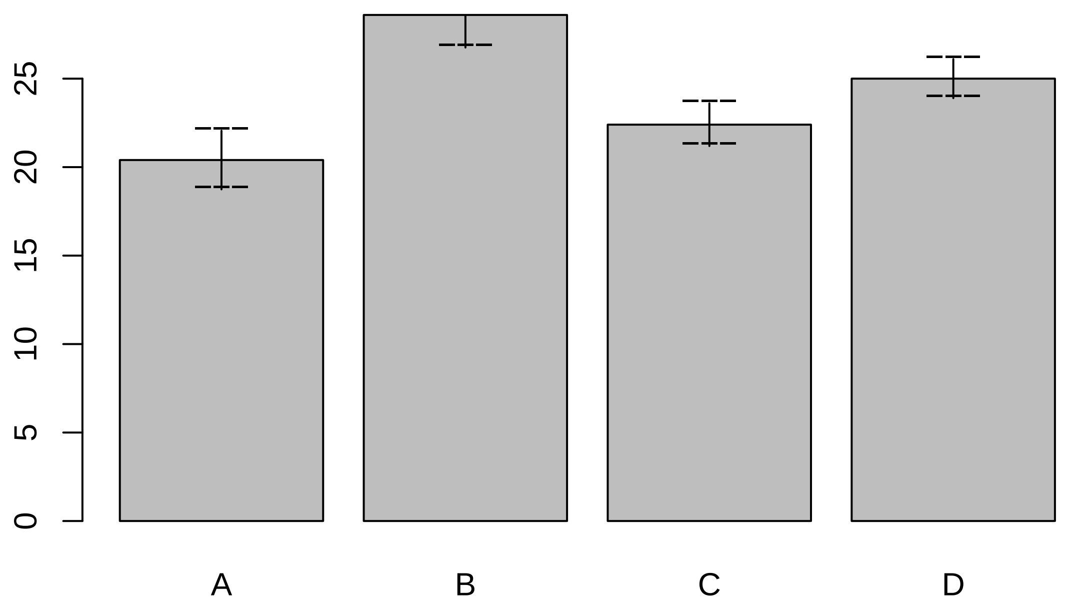
```
## Least Significant Difference Analysis
LSD = LSD.test(model, "variety", console = TRUE)
```

```
##
## Study: model ~ "variety"
##
## LSD t Test for yield
##
## Mean Square Error:  11.025
##
## variety,  means and individual ( 95 %) CI
##
##   yield      std r     LCL     UCL  Min  Max
## A  20.4 3.708773 5 17.2521 23.5479 16.1 25.2
## B  28.6 4.118859 5 25.4521 31.7479 24.1 34.8
## C  22.4 2.700926 5 19.2521 25.5479 18.4 25.9
## D  25.0 2.466779 5 21.8521 28.1479 21.7 28.2
##
## Alpha: 0.05 ; DF Error: 16
## Critical Value of t: 2.119905
##
## least Significant Difference: 4.451801
##
## Treatments with the same letter are not significantly different.
##
##   yield groups
## B  28.6      a
## D  25.0     ab
## C  22.4     bc
## A  20.4      c
```

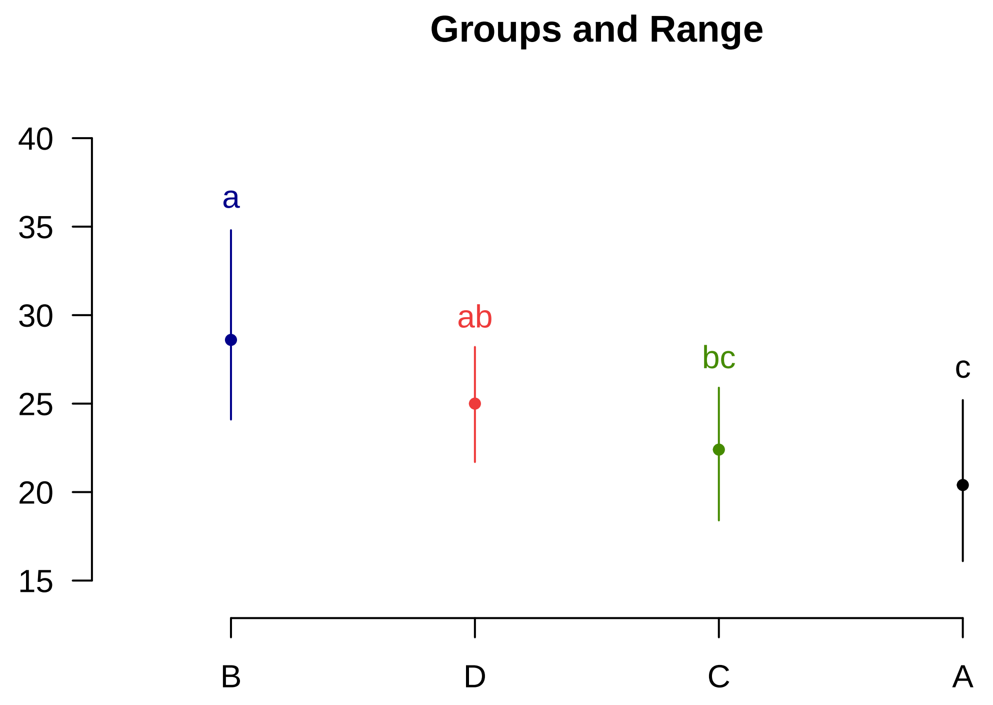
```r
bar.group(LSD$group)
```



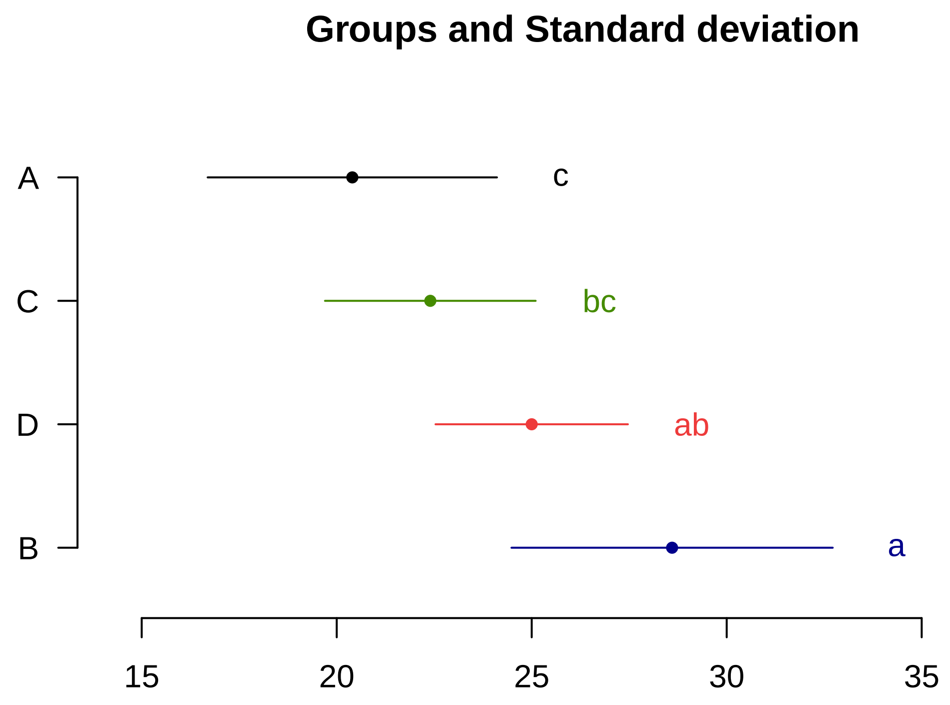
```r
bar.err(LSD$means)
```



```r
plot(LSD, variation="range",las=1)
```



```
plot(LSD, horiz=TRUE, variation="SD",las=1)
```



```r
## Tukey's (HSD) Test
HSD.test(model, "variety", console=TRUE)
```

```
##
## Study: model ~ "variety"
##
## HSD Test for yield
##
## Mean Square Error:  11.025
##
## variety,  means
##
##   yield      std r  Min  Max
## A  20.4 3.708773 5 16.1 25.2
## B  28.6 4.118859 5 24.1 34.8
## C  22.4 2.700926 5 18.4 25.9
## D  25.0 2.466779 5 21.7 28.2
##
## Alpha: 0.05 ; DF Error: 16
## Critical Value of Studentized Range: 4.046093
##
## Minimun Significant Difference: 6.008142
##
## Treatments with the same letter are not significantly different.
##
##   yield groups
## B  28.6      a
## D  25.0     ab
## C  22.4      b
## A  20.4      b
```

Agricolae data analysis for 9.2 example

```
data = data.frame(variety = c("A","A","A",
                              "B","B","B","B",
                              "C","C","C","C",
                              "D","D","D","D","D"),
                  yield = c(17.3, 21.2, 16.1,
                            24.1, 30.3, 26.4, 34.8,
                            25.9, 18.4, 21.9, 22.6,
                            23.9, 21.7, 24.8, 28.2, 26.4))

tapply(data$yield, data$variety, length) #N
```

```
## A B C D
## 3 4 4 5
```

```
tapply(data$yield, data$variety, mean) #Mean
```

```
##    A    B    C    D
## 18.2 28.9 22.2 25.0
```

```
tapply(data$yield, data$variety, sd) #StDev
```

```
##        A        B        C        D
## 2.666458 4.692547 3.075711 2.466779
```

```
model = aov(yield ~ variety, data = data)
anova(model)
```

```
## Analysis of Variance Table
##
## Response: yield
##           Df Sum Sq Mean Sq F value   Pr(>F)
## variety    3 214.92  71.640  6.4638 0.007499 **
## Residuals 12 133.00  11.083
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Coefficient of Variation
cv.model(model)
```

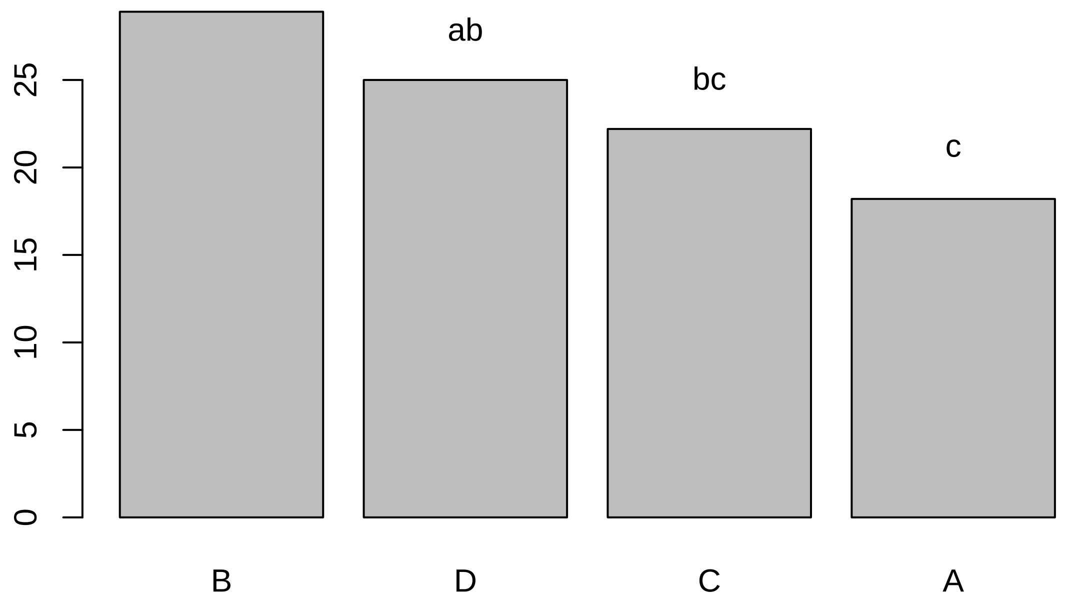
```
## [1] 13.87152
```

```
## Least Significant Difference Analysis
LSD = LSD.test(model, "variety", console = TRUE)
```

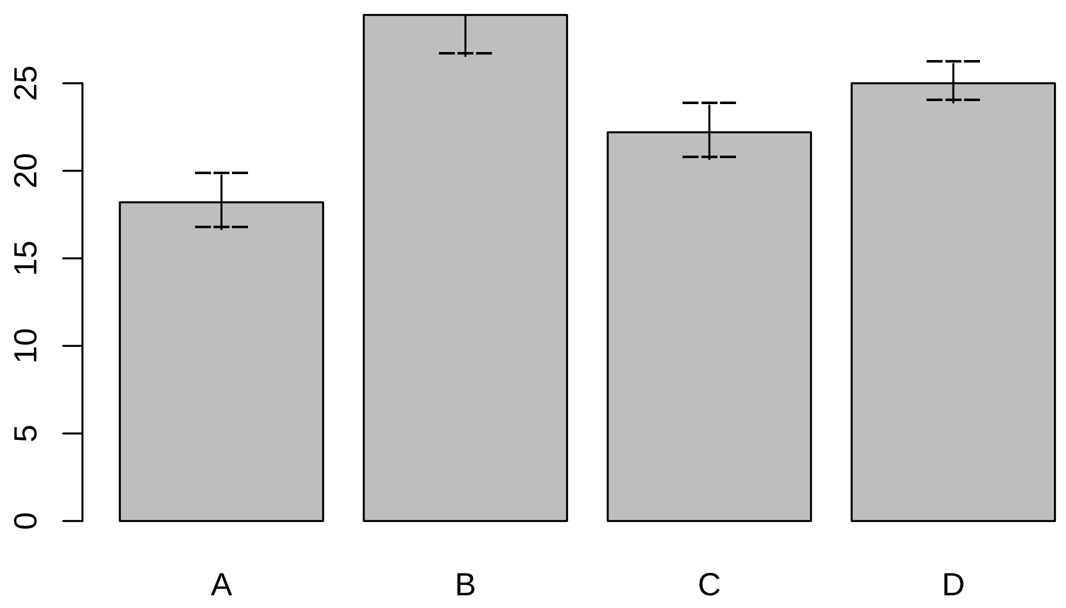
```
##
## Study: model ~ "variety"
##
## LSD t Test for yield
##
## Mean Square Error:  11.08333
##
```

```
## variety,  means and individual ( 95 %) CI
##
##   yield      std r      LCL      UCL  Min  Max
## A  18.2 2.666458 3 14.01212 22.38788 16.1 21.2
## B  28.9 4.692547 4 25.27319 32.52681 24.1 34.8
## C  22.2 3.075711 4 18.57319 25.82681 18.4 25.9
## D  25.0 2.466779 5 21.75608 28.24392 21.7 28.2
##
## Alpha: 0.05 ; DF Error: 12
## Critical Value of t: 2.178813
##
## Groups according to probability of means differences and alpha level( 0.05 )
##
## Treatments with the same letter are not significantly different.
##
##   yield groups
## B  28.9      a
## D  25.0     ab
## C  22.2     bc
## A  18.2      c
```

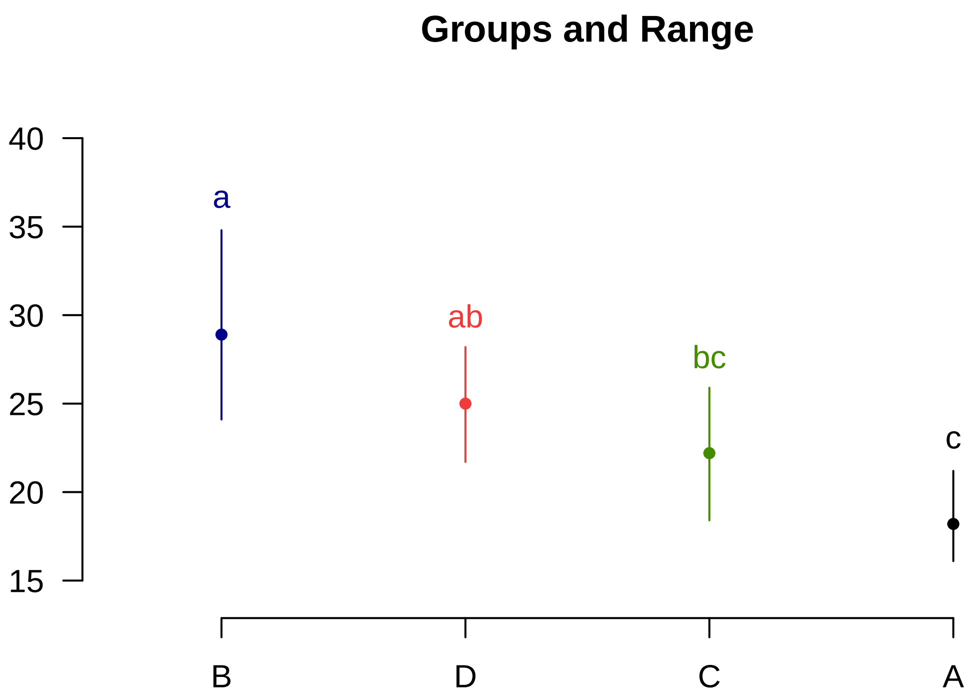
```
bar.group(LSD$group)
```



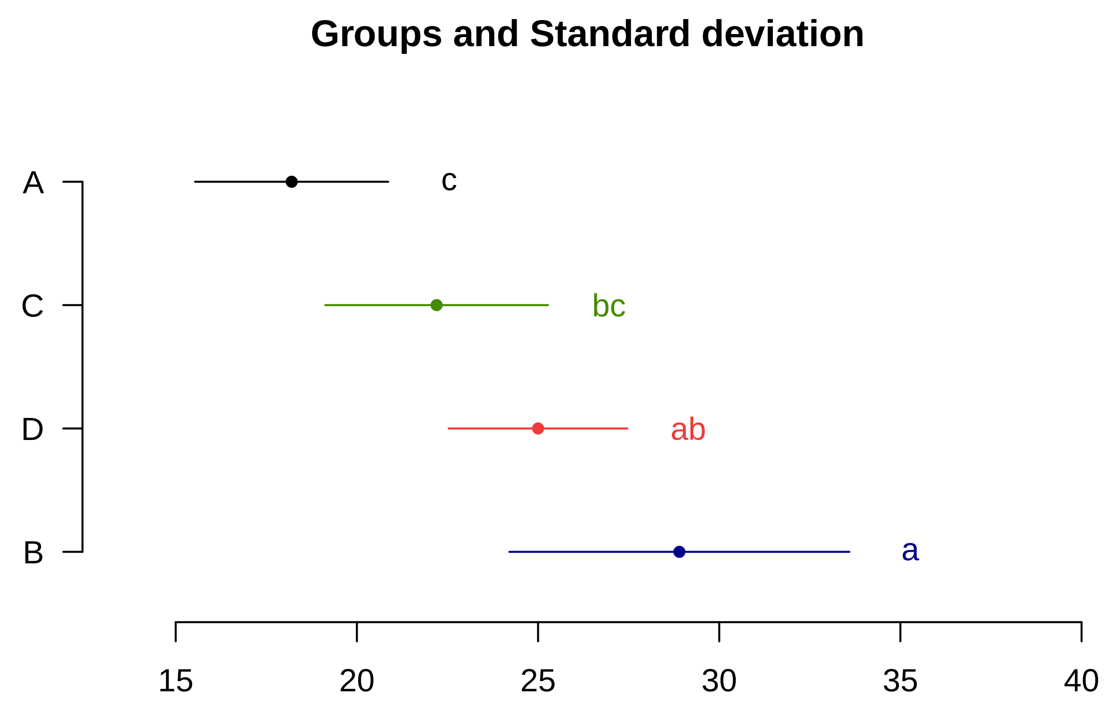
```
bar.err(LSD$means)
```



```r
plot(LSD, variation="range",las=1)
```



```
plot(LSD, horiz=TRUE, variation="SD",las=1)
```



```r
## Tukey's (HSD) Test
HSD.test(model, "variety", console=TRUE)
```

```
##
## Study: model ~ "variety"
##
## HSD Test for yield
##
## Mean Square Error:  11.08333
##
## variety,  means
##
##   yield      std r  Min  Max
## A  18.2 2.666458 3 16.1 21.2
## B  28.9 4.692547 4 24.1 34.8
## C  22.2 3.075711 4 18.4 25.9
## D  25.0 2.466779 5 21.7 28.2
##
## Alpha: 0.05 ; DF Error: 12
## Critical Value of Studentized Range: 4.19866
##
## Groups according to probability of means differences and alpha level( 0.05 )
##
## Treatments with the same letter are not significantly different.
##
##   yield groups
## B  28.9      a
## D  25.0     ab
## C  22.2     ab
## A  18.2      b
```

Chapter 10: The Randomized Block Design

Randomization

4 treatments and 3 blocks (repetitions)

```r
library(agricolae)
variety = c("A","B","C","D")

field_design = design.rcbd(variety,r = 3)
field_design$book
```

```
##    plots block variety
## 1    101     1       A
## 2    102     1       C
## 3    103     1       D
## 4    104     1       B
## 5    201     2       C
## 6    202     2       B
## 7    203     2       A
## 8    204     2       D
## 9    301     3       B
## 10   302     3       D
## 11   303     3       C
## 12   304     3       A
```

10.1 Example

Below, we build the data frame of Example 10.1. Here we are reading the data by columns.

```r
example10.1 = data.frame(variety = c("V1", "V1", "V1",
                                     "V2", "V2", "V2",
                                     "V3", "V3", "V3",
                                     "V4", "V4", "V4"),
                         block = c("B1", "B2", "B3",
                                   "B1", "B2", "B3",
                                   "B1", "B2", "B3",
                                   "B1", "B2", "B3"),
                         yield = c(7.4, 6.5, 5.6,
                                   9.8, 6.8, 6.2,
                                   7.3, 6.1, 6.4,
                                   9.5, 8.0, 7.4))
## Visualing the data frame
print(example10.1)
```

```
##    variety block yield
## 1       V1    B1   7.4
## 2       V1    B2   6.5
## 3       V1    B3   5.6
## 4       V2    B1   9.8
## 5       V2    B2   6.8
## 6       V2    B3   6.2
## 7       V3    B1   7.3
## 8       V3    B2   6.1
## 9       V3    B3   6.4
## 10      V4    B1   9.5
## 11      V4    B2   8.0
## 12      V4    B3   7.4
```

Checking the data frame structure, sums, block and variety totals

```
str(example10.1)
```

```
## 'data.frame':    12 obs. of  3 variables:
##  $ variety: Factor w/ 4 levels "V1","V2","V3",..: 1 1 1 2 2 2 3 3 3 4 ...
##  $ block  : Factor w/ 3 levels "B1","B2","B3": 1 2 3 1 2 3 1 2 3 1 ...
##  $ yield  : num  7.4 6.5 5.6 9.8 6.8 6.2 7.3 6.1 6.4 9.5 ...
```

```
#apply the function sum in yield by block
tapply(example10.1$yield, example10.1$block, sum)
```

```
##   B1   B2   B3
## 34.0 27.4 25.6
```

```
#apply the function mean in yield by block
tapply(example10.1$yield, example10.1$block, mean)
```

```
##   B1   B2   B3
## 8.50 6.85 6.40
```

```
#apply the function sum in yield by variety
tapply(example10.1$yield, example10.1$variety, sum)
```

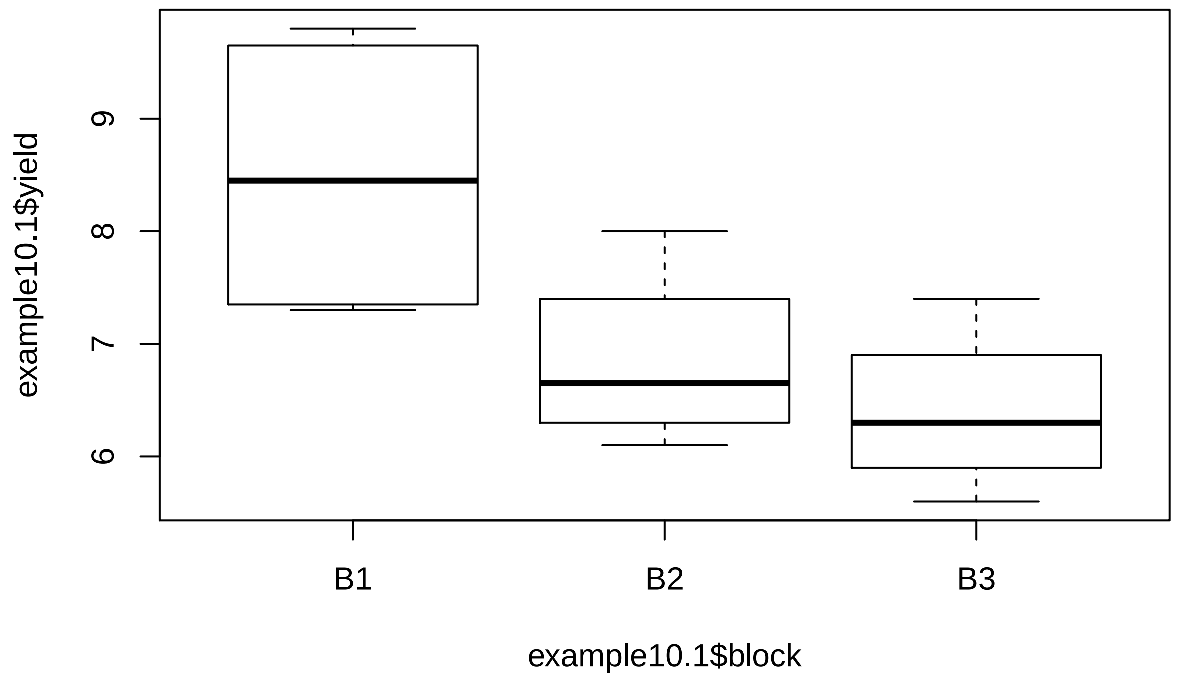
```
##   V1   V2   V3   V4
## 19.5 22.8 19.8 24.9
```

```
#apply the function mean in yield by variety
tapply(example10.1$yield, example10.1$variety, mean)
```

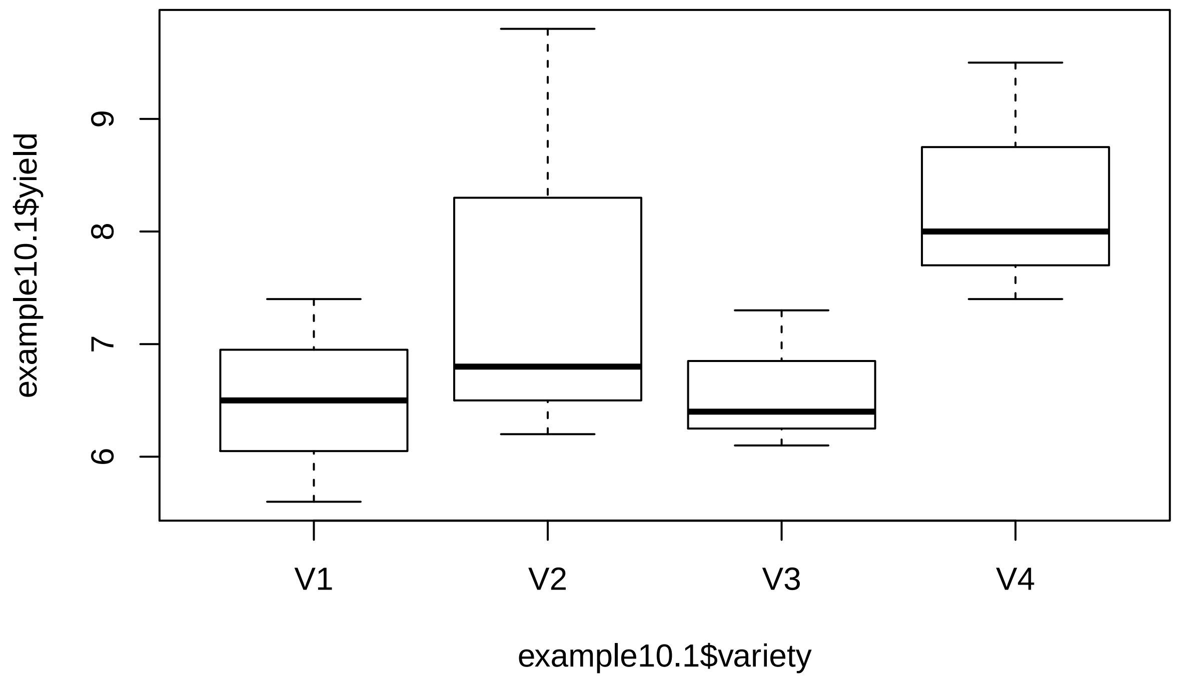
```
##  V1  V2  V3  V4
## 6.5 7.6 6.6 8.3
```

Descriptive plots

```
boxplot( example10.1$yield ~ example10.1$block)
```



```
boxplot( example10.1$yield ~ example10.1$variety)
```



10.2 Analysis ignoring blocks

```
model10.2 = aov(yield ~ variety, data = example10.1)
anova(model10.2)
```

```
## Analysis of Variance Table
##
## Response: yield
##           Df Sum Sq Mean Sq F value Pr(>F)
## variety    3   6.63  2.2100  1.4516 0.2987
## Residuals  8  12.18  1.5225
```

10.3 The analysis including blocks

```
model10.3 = aov(yield ~ block + variety, data = example10.1)
anova(model10.3)
```

```
## Analysis of Variance Table
##
## Response: yield
##           Df Sum Sq Mean Sq F value   Pr(>F)
## block      2   9.78    4.89  12.225 0.007651 **
## variety    3   6.63    2.21   5.525 0.036730 *
## Residuals  6   2.40    0.40
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Residuals and Fitted values

```
model10.3$residuals
```

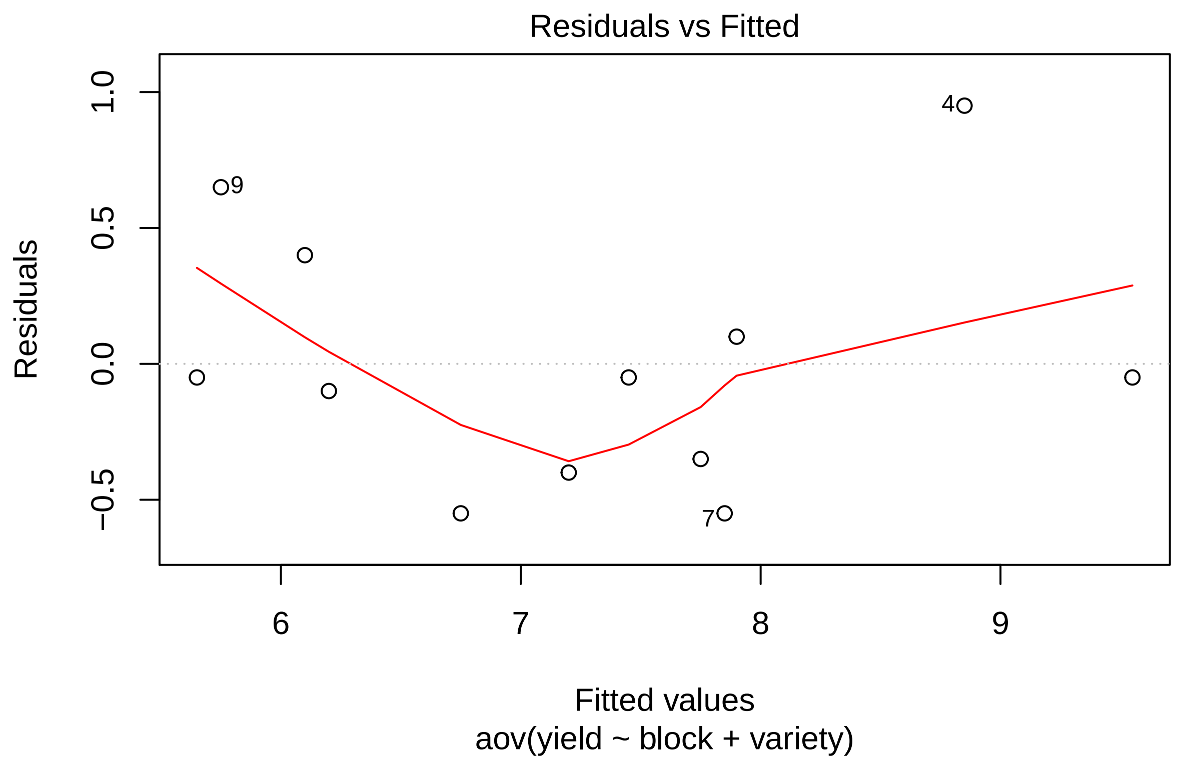
```
##     1     2     3     4     5     6     7     8     9    10    11    12
## -0.35  0.40 -0.05  0.95 -0.40 -0.55 -0.55 -0.10  0.65 -0.05  0.10 -0.05
```

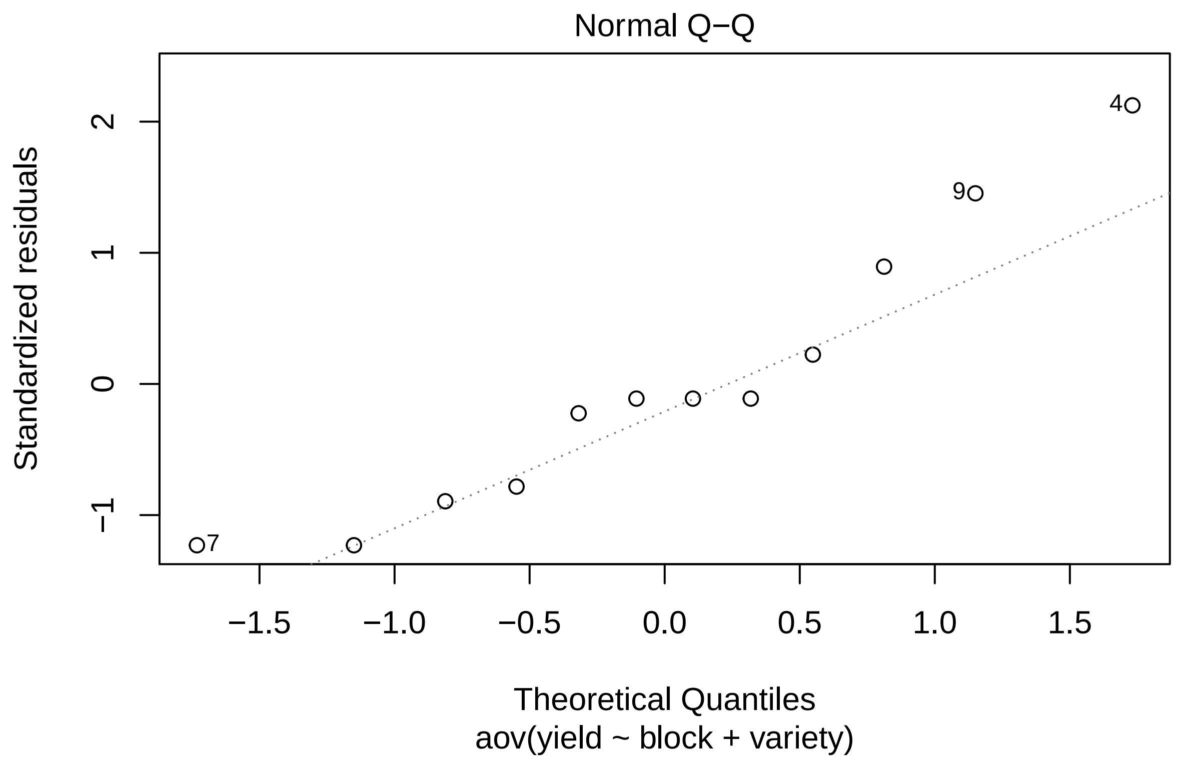
```
model10.3$fitted.values
```

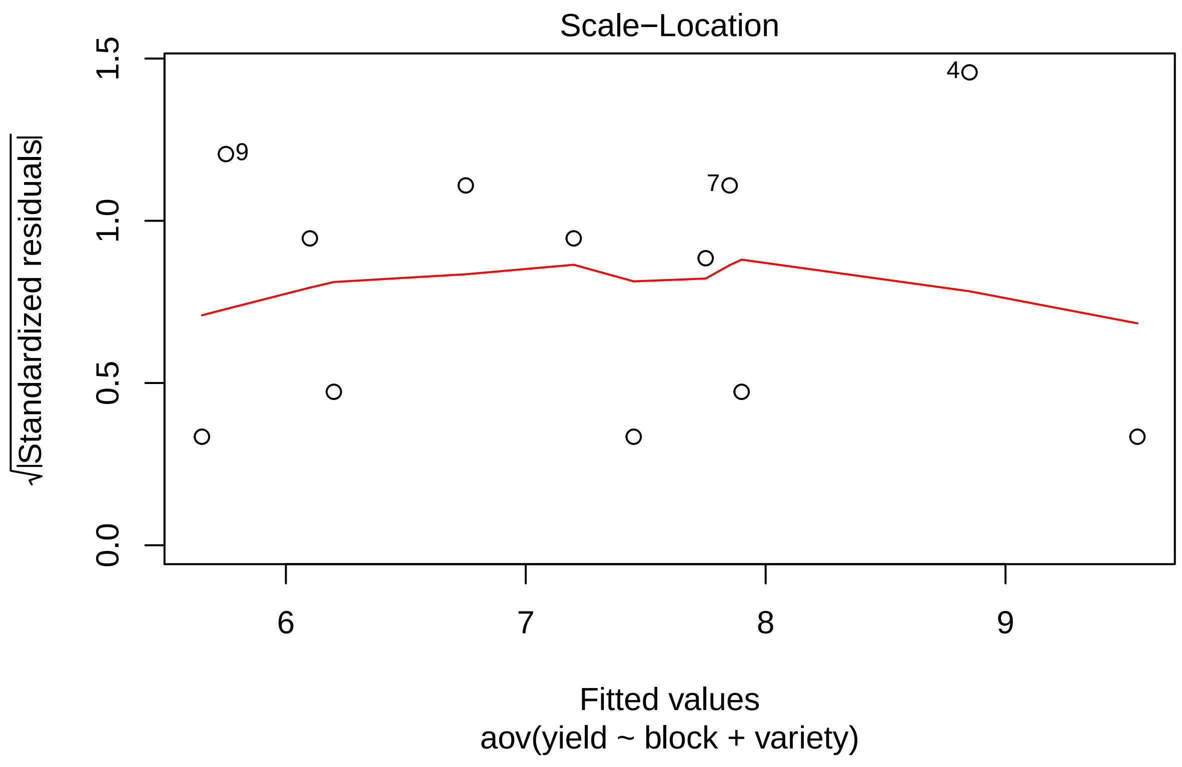
```
##    1    2    3    4    5    6    7    8    9   10   11   12
## 7.75 6.10 5.65 8.85 7.20 6.75 7.85 6.20 5.75 9.55 7.90 7.45
```

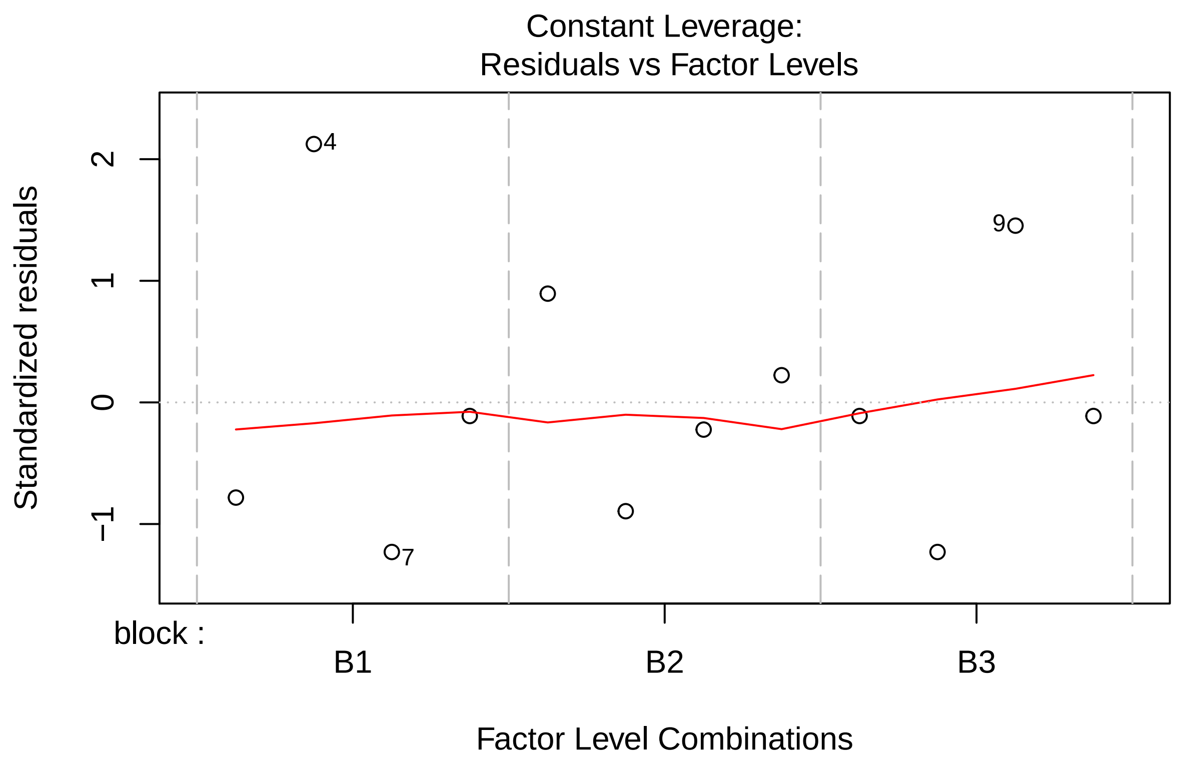
Diagnostics Plots

```
plot(model10.3)
```

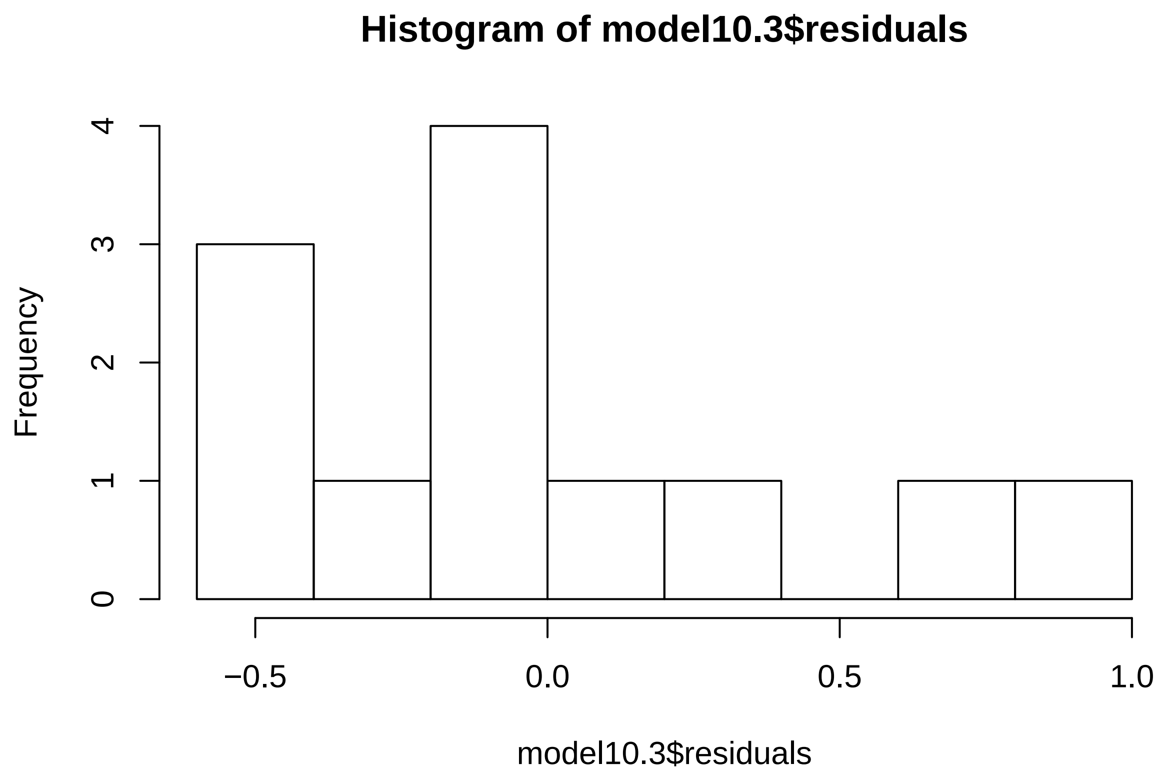



Normal Q−Q
Standardized residuals
Theoretical Quantiles
aov(yield ~ block + variety)
−1
0
1
2
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
7
9
4


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
6
7
8
9
4
9
7
Fitted values
aov(yield ~ block + variety)



```
hist(model10.3$residuals,breaks = 10)
```



10.8 Comparison of treatment means

10.8.1 LSD Analysis and Confidence Intervals

With `agricolae`

```
library(agricolae)
LSD.test(model10.3, "variety", console=TRUE)
```

```
##
## Study: model10.3 ~ "variety"
##
## LSD t Test for yield
##
## Mean Square Error:  0.4
##
## variety,  means and individual ( 95 %) CI
##
##    yield       std r      LCL      UCL Min Max
## V1   6.5 0.9000000 3 5.606514 7.393486 5.6 7.4
## V2   7.6 1.9287302 3 6.706514 8.493486 6.2 9.8
## V3   6.6 0.6244998 3 5.706514 7.493486 6.1 7.3
## V4   8.3 1.0816654 3 7.406514 9.193486 7.4 9.5
##
## Alpha: 0.05 ; DF Error: 6
## Critical Value of t: 2.446912
##
```

```
## least Significant Difference: 1.26358
##
## Treatments with the same letter are not significantly different.
##
##    yield groups
## V4   8.3      a
## V2   7.6     ab
## V3   6.6      b
## V1   6.5      b
```

Standard Errors and Confidence Intervals (by “hand””)

```
## Extracting RMS
anova(model10.3)
```

```
## Analysis of Variance Table
##
## Response: yield
##           Df Sum Sq Mean Sq F value   Pr(>F)
## block      2   9.78    4.89  12.225 0.007651 **
## variety    3   6.63    2.21   5.525 0.036730 *
## Residuals  6   2.40    0.40
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## The RMS is the Residual Mean Square error,
## In our ANOVA table, RMS is located in the 3rd row and 3rd column
RMS = anova(model10.3)[3,3]
print(RMS)
```

```
## [1] 0.4
```

```
r = 3 #number of reps (blocks)

SEM = sqrt(RMS/r)
print(SEM)
```

```
## [1] 0.3651484
```

```
lower.t = qt(0.025, 6)
upper.t = qt(0.975, 6)

variety.mean = tapply(example10.1$yield, example10.1$variety, mean)

lower.CI = variety.mean + lower.t*SEM
upper.CI = variety.mean + upper.t*SEM

CI.variety = data.frame(mean = variety.mean,
                        lower = lower.CI,
                        upper = upper.CI)
```

Chapter 11: The Latin Square Design

Randomization

```
variety = c("A","B","C","D")

field_design = design.lsd(variety)
field_design$book
```

```
##    plots row col variety
## 1    101   1   1       A
## 2    102   1   2       D
## 3    103   1   3       B
## 4    104   1   4       C
## 5    201   2   1       C
## 6    202   2   2       B
## 7    203   2   3       D
## 8    204   2   4       A
## 9    301   3   1       D
## 10   302   3   2       C
## 11   303   3   3       A
## 12   304   3   4       B
## 13   401   4   1       B
## 14   402   4   2       A
## 15   403   4   3       C
## 16   404   4   4       D
```

11.1 Example

Below, we build the data frame of Example 11.1. Here we are reading the data by columns.

```
example11.1 = data.frame(
  row = c("1","1","1","1",
          "2","2","2","2",
          "3","3","3","3",
          "4","4","4","4"),
  column = c("1","2","3","4",
             "1","2","3","4",
             "1","2","3","4",
             "1","2","3","4"),
  variety = c("C","D","B","A",
              "B","A","C","D",
              "D","C","A","B",
              "A","B","D","C"),
  yield = c(16.6,13.9,18.0,21.9,
            12.6,17.4,17.9,16.5,
            9.3,16.2,18.4,15.5,
            15.7,11.3,10.5,13.1))

## Visualing the data frame
print(example11.1)
```

```
##    row column variety yield
## 1    1      1       C  16.6
## 2    1      2       D  13.9
## 3    1      3       B  18.0
## 4    1      4       A  21.9
## 5    2      1       B  12.6
## 6    2      2       A  17.4
## 7    2      3       C  17.9
## 8    2      4       D  16.5
## 9    3      1       D   9.3
## 10   3      2       C  16.2
## 11   3      3       A  18.4
## 12   3      4       B  15.5
## 13   4      1       A  15.7
## 14   4      2       B  11.3
## 15   4      3       D  10.5
## 16   4      4       C  13.1
```

```
#apply the function sum in yield by row
tapply(example11.1$yield, example11.1$row, sum)
```

```
##    1    2    3    4
## 70.4 64.4 59.4 50.6
```

```
#apply the function sum in yield by col
tapply(example11.1$yield, example11.1$col, sum)
```

```
##    1    2    3    4
## 54.2 58.8 64.8 67.0
```

```
#apply the function sum in yield by variety
tapply(example11.1$yield, example11.1$variety, sum)
```

```
##    A    B    C    D
## 73.4 57.4 63.8 50.2
```

```
#apply the function mean in yield by variety
tapply(example11.1$yield, example11.1$variety, mean)
```

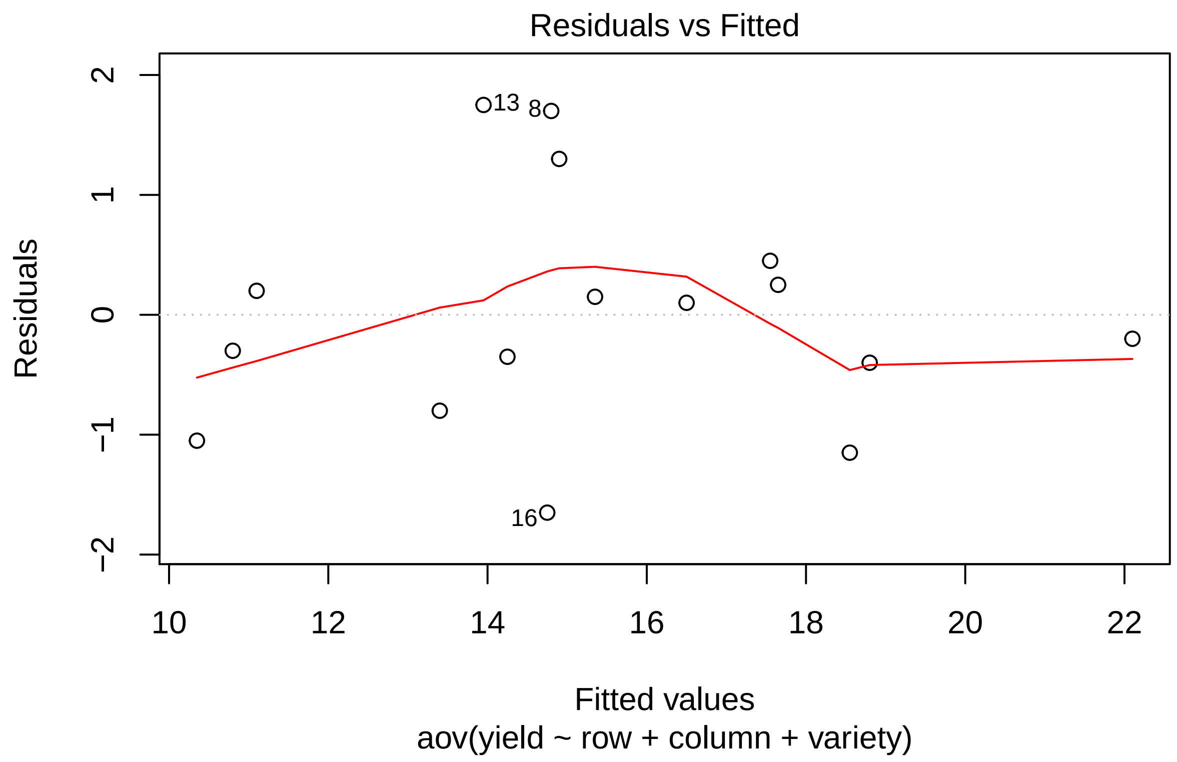
```
##     A     B     C     D
## 18.35 14.35 15.95 12.55
```

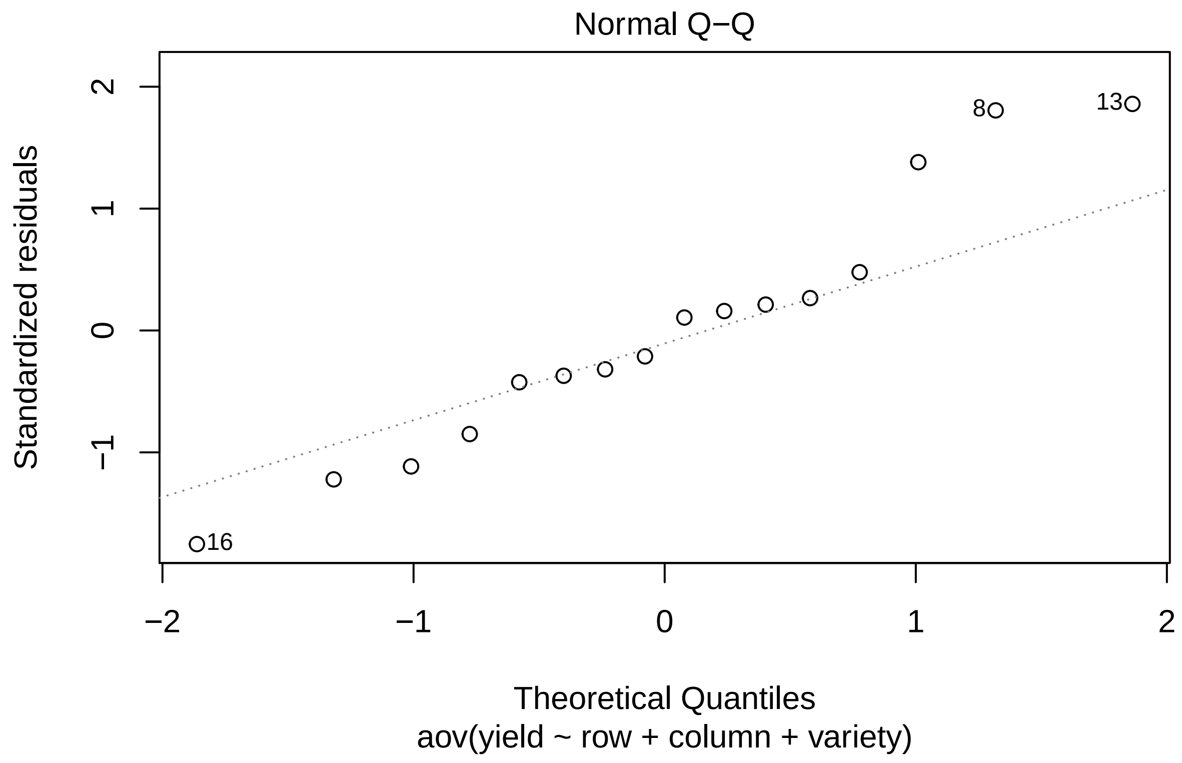
Model

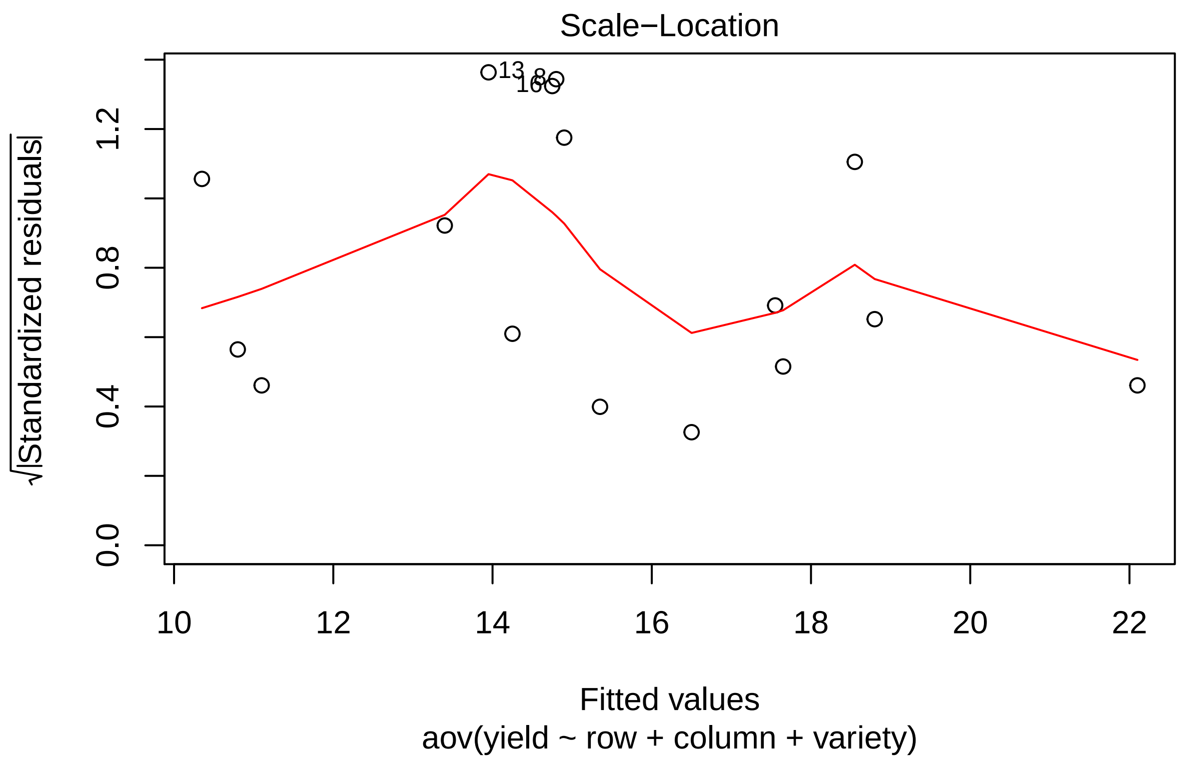
```
model11.1 = aov(yield ~ row + column + variety, data = example11.1)
```

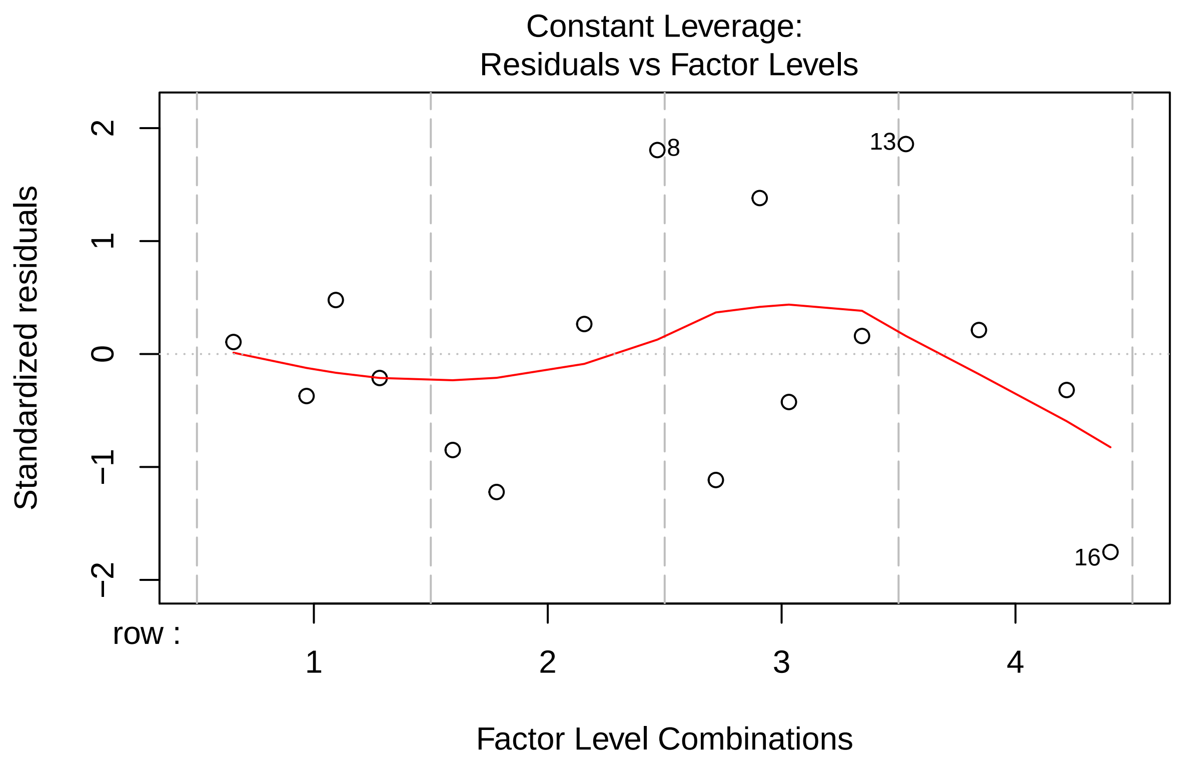
Residual analysis

```
plot(model11.1)
```


Residuals vs Fitted
Residuals
Fitted values
aov(yield ~ row + column + variety)
13
8
16
10
12
14
16
18
20
22
−2
−1
0
1
2


Normal Q−Q
Standardized residuals
Theoretical Quantiles
aov(yield ~ row + column + variety)
16
8
13
−2
−1
0
1
2


Scale−Location
√|Standardized residuals|
0.0
0.4
0.8
1.2
10
12
14
16
18
20
22
13
8
16
Fitted values
aov(yield ~ row + column + variety)



Analysis of Variance

```r
anova(model11.1)
```

```
## Analysis of Variance Table
##
## Response: yield
##           Df Sum Sq Mean Sq F value  Pr(>F)
## row        3  52.62 17.5400  7.4217 0.01918 *
## column     3  25.34  8.4467  3.5740 0.08629 .
## variety    3  72.76 24.2533 10.2623 0.00889 **
## Residuals  6  14.18  2.3633
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Comparison between the treatments

```r
LSD.test(model11.1, "variety", console=TRUE)
```

```
##
## Study: model11.1 ~ "variety"
##
## LSD t Test for yield
##
## Mean Square Error:  2.363333
##
```

```
## variety,  means and individual ( 95 %) CI
##
##   yield      std r      LCL      UCL  Min  Max
## A 18.35 2.615977 4 16.46916 20.23084 15.7 21.9
## B 14.35 3.000556 4 12.46916 16.23084 11.3 18.0
## C 15.95 2.033880 4 14.06916 17.83084 13.1 17.9
## D 12.55 3.275668 4 10.66916 14.43084  9.3 16.5
##
## Alpha: 0.05 ; DF Error: 6
## Critical Value of t: 2.446912
##
## least Significant Difference: 2.659903
##
## Treatments with the same letter are not significantly different.
##
##   yield groups
## A 18.35      a
## C 15.95     ab
## B 14.35     bc
## D 12.55      c
```

11.1 Exercise

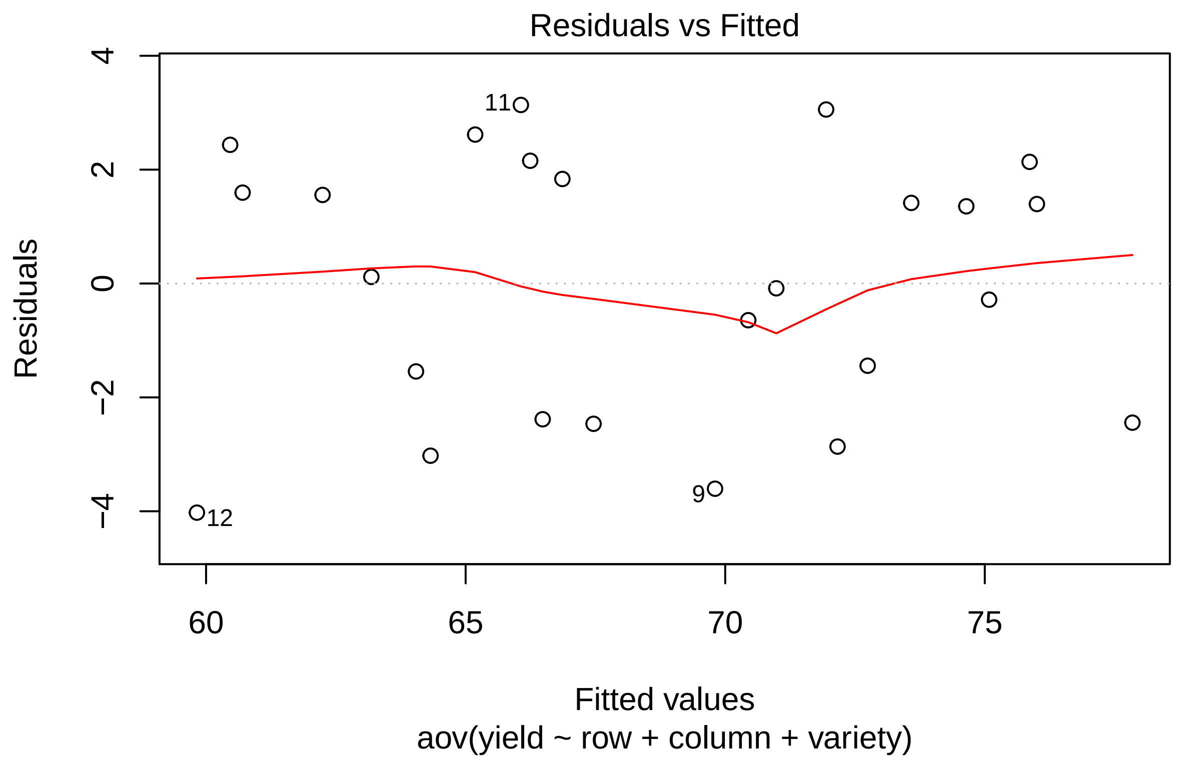
```
exercise11.1 = data.frame(
  row = c("1","1","1","1","1",
          "2","2","2","2","2",
          "3","3","3","3","3",
          "4","4","4","4","4",
          "5","5","5","5","5"),
  column = c("1","2","3","4","5",
             "1","2","3","4","5",
             "1","2","3","4","5",
             "1","2","3","4","5",
             "1","2","3","4","5"),
  variety = c("P2","P3","P5","P1","P4",
              "P4","P5","P1","P2","P3",
              "P3","P1","P2","P4","P5",
              "P5","P2","P4","P3","P1",
              "P1","P4","P3","P5","P2"),
  yield = c(62.3,61.3,62.5,63.8,75.0,
            64.1,68.4,62.9,66.2,77.4,
            69.2,55.8,67.8,71.3,74.8,
            65.0,68.7,69.8,76.0,70.9,
            63.3,75.0,69.3,78.0,75.4))
```

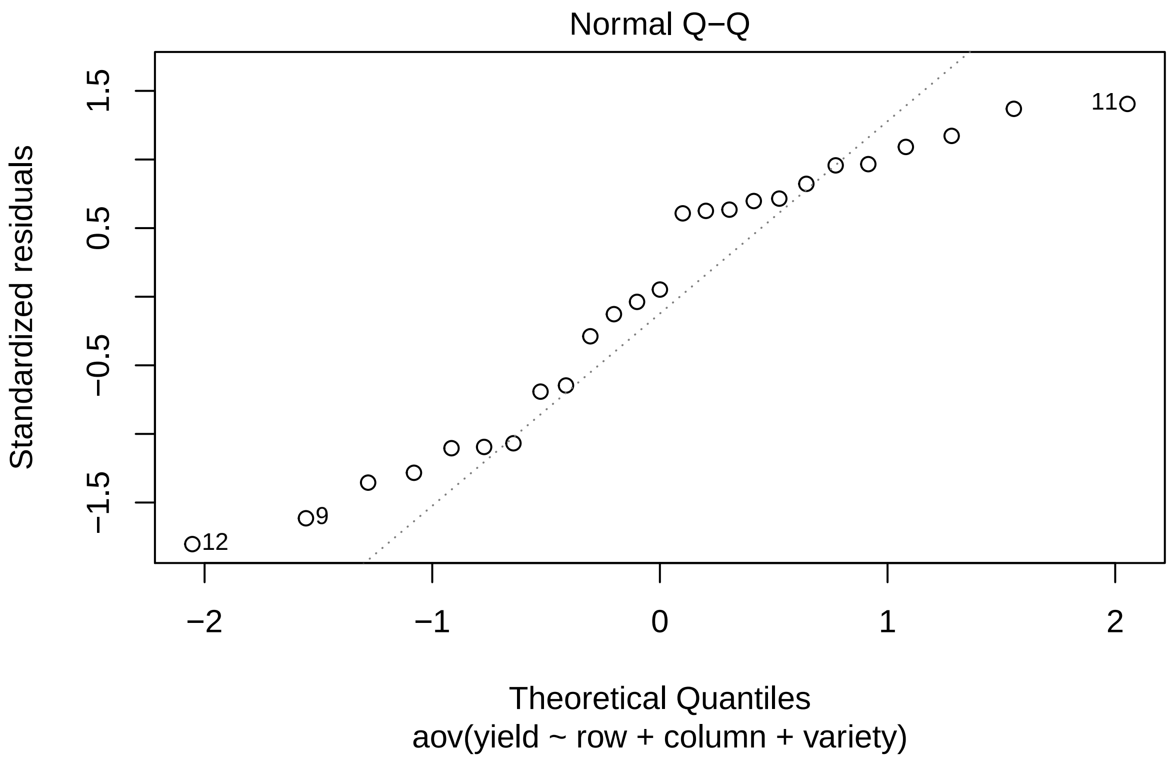
Model

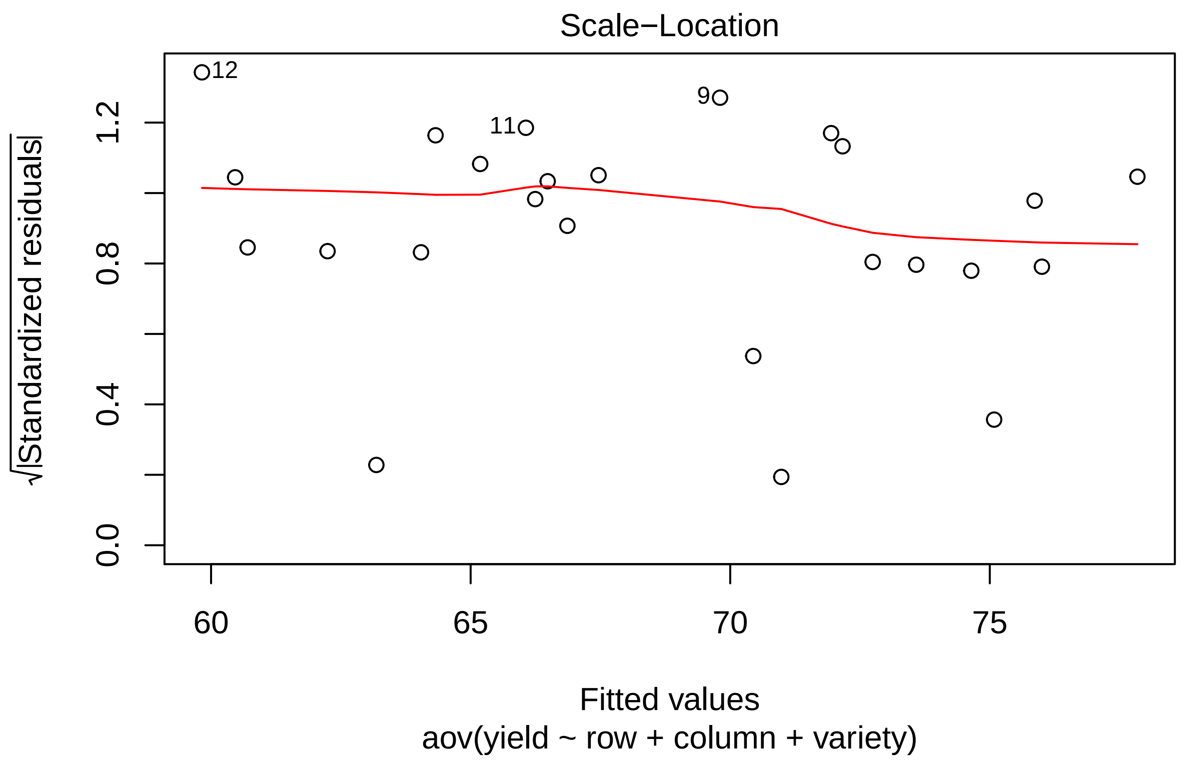
```
modelexercise11.1 = aov(yield ~ row + column + variety, data = exercise11.1)
```

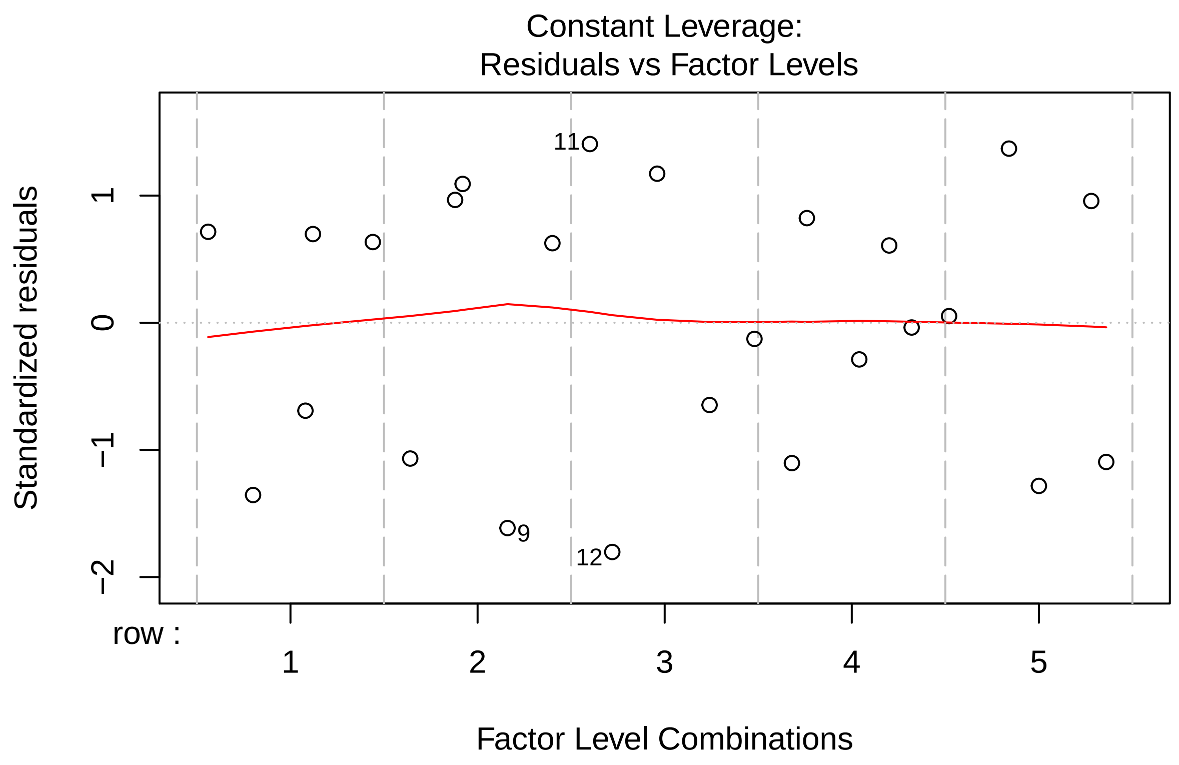
Residual analysis

```
plot(modelexercise11.1)
```


Residuals vs Fitted
Residuals
Fitted values
aov(yield ~ row + column + variety)
4
2
0
−2
−4
60
65
70
75
11
9
12


Normal Q−Q
Standardized residuals
1.5
0.5
−0.5
−1.5
−2
−1
0
1
2
11
9
12
Theoretical Quantiles
aov(yield ~ row + column + variety)


Scale−Location
√|Standardized residuals|
0.0
0.4
0.8
1.2
60
65
70
75
Fitted values
aov(yield ~ row + column + variety)
12
11
9



Analysis of Variance

```
anova(modelexercise11.1)
```

```
## Analysis of Variance Table
##
## Response: yield
##           Df Sum Sq Mean Sq F value   Pr(>F)
## row        4 147.81  36.953  3.5607 0.038911 *
## column     4 350.23  87.558  8.4369 0.001768 **
## variety    4 196.74  49.185  4.7393 0.015857 *
## Residuals 12 124.54  10.378
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Comparison between the treatments

```
LSD.test(modelexercise11.1, "variety", console=TRUE)
```

```
##
## Study: modelexercise11.1 ~ "variety"
##
## LSD t Test for yield
##
## Mean Square Error:  10.37793
##
```

```
## variety,  means and individual ( 95 %) CI
##
##    yield      std r      LCL      UCL  Min  Max
## P1 63.34 5.348177 5 60.20101 66.47899 55.8 70.9
## P2 68.08 4.769382 5 64.94101 71.21899 62.3 75.4
## P3 70.64 6.432962 5 67.50101 73.77899 61.3 77.4
## P4 71.04 4.503665 5 67.90101 74.17899 64.1 75.0
## P5 69.74 6.529012 5 66.60101 72.87899 62.5 78.0
##
## Alpha: 0.05 ; DF Error: 12
## Critical Value of t: 2.178813
##
## least Significant Difference: 4.439207
##
## Treatments with the same letter are not significantly different.
##
##    yield groups
## P4 71.04      a
## P3 70.64      a
## P5 69.74      a
## P2 68.08      a
## P1 63.34      b
```

Chapter 12: Factorial Experiments

Randomization

Let's assume that we would like to use the following factors: Factor 1 with 2 levels Factor 2 with 3 levels

We can use the same previous functions with the 6 possible treatments as follows

```
factor1 = c("A1","B1")
factor2 = c("A2","B2","C2")
treatments = paste(rep(factor1,each=3),rep(factor2,2),sep="_")

## Randomization in CRD with 3 reps
field_design = design.crd(treatments,r = 3)

## Randomization in RCBD with 3 blocks
field_design = design.rcbd(treatments,r = 3)

## Randomization in Latin Square 6x6
field_design = design.lsd(treatments)
```

Example 12.3

Hereafter, we will use the function `rep` to build the data frames

```
example12.3 = data.frame(
  block = rep(c("B1","B2","B3"),each=15),
  P = rep(rep(c("P1","P2","P3"),each=5),3),
  N = rep(c("N1","N2","N3","N4","N5"),9),
  yield = c(0.9,1.2,1.3,1.8,1.1,0.9,1.1,1.3,1.6,1.9,0.9,1.4,1.3,1.4,1.2,
            0.9,1.3,1.5,1.9,1.4,0.8,0.9,1.5,1.3,1.6,1.0,1.2,1.4,1.5,1.1,
            1.0,1.2,1.4,2.1,1.2,0.8,0.9,1.1,1.1,1.5,0.7,1.0,1.4,1.4,1.3)
  )

## Visualing the data frame
print(example12.3)
```

```
##    block  P  N yield
## 1     B1 P1 N1   0.9
## 2     B1 P1 N2   1.2
## 3     B1 P1 N3   1.3
## 4     B1 P1 N4   1.8
## 5     B1 P1 N5   1.1
## 6     B1 P2 N1   0.9
## 7     B1 P2 N2   1.1
## 8     B1 P2 N3   1.3
## 9     B1 P2 N4   1.6
## 10    B1 P2 N5   1.9
## 11    B1 P3 N1   0.9
## 12    B1 P3 N2   1.4
## 13    B1 P3 N3   1.3
## 14    B1 P3 N4   1.4
## 15    B1 P3 N5   1.2
## 16    B2 P1 N1   0.9
## 17    B2 P1 N2   1.3
## 18    B2 P1 N3   1.5
## 19    B2 P1 N4   1.9
## 20    B2 P1 N5   1.4
## 21    B2 P2 N1   0.8
## 22    B2 P2 N2   0.9
## 23    B2 P2 N3   1.5
## 24    B2 P2 N4   1.3
## 25    B2 P2 N5   1.6
## 26    B2 P3 N1   1.0
## 27    B2 P3 N2   1.2
## 28    B2 P3 N3   1.4
## 29    B2 P3 N4   1.5
## 30    B2 P3 N5   1.1
## 31    B3 P1 N1   1.0
## 32    B3 P1 N2   1.2
## 33    B3 P1 N3   1.4
## 34    B3 P1 N4   2.1
## 35    B3 P1 N5   1.2
## 36    B3 P2 N1   0.8
## 37    B3 P2 N2   0.9
## 38    B3 P2 N3   1.1
## 39    B3 P2 N4   1.1
## 40    B3 P2 N5   1.5
## 41    B3 P3 N1   0.7
## 42    B3 P3 N2   1.0
## 43    B3 P3 N3   1.4
## 44    B3 P3 N4   1.4
## 45    B3 P3 N5   1.3
```

```
#apply the function sum in yield by block
tapply(example12.3$yield, example12.3$block, sum)
```

```
##   B1   B2   B3
## 19.3 19.3 18.1
```

```
#apply the function sum in yield by P
tapply(example12.3$yield, example12.3$P, sum)
```

```
##   P1   P2   P3
## 20.2 18.3 18.2
```

```
#apply the function sum in yield by N
tapply(example12.3$yield, example12.3$N, sum)
```

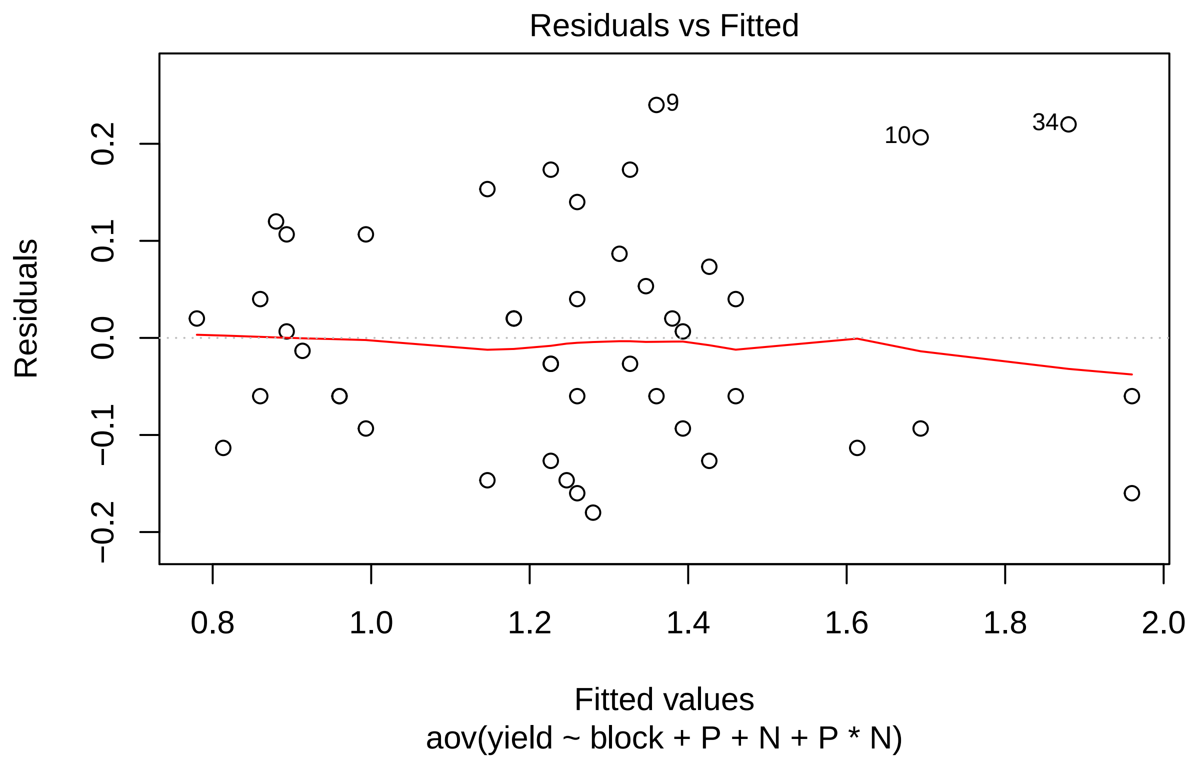
```
##   N1   N2   N3   N4   N5
##  7.9 10.2 12.2 14.1 12.3
```

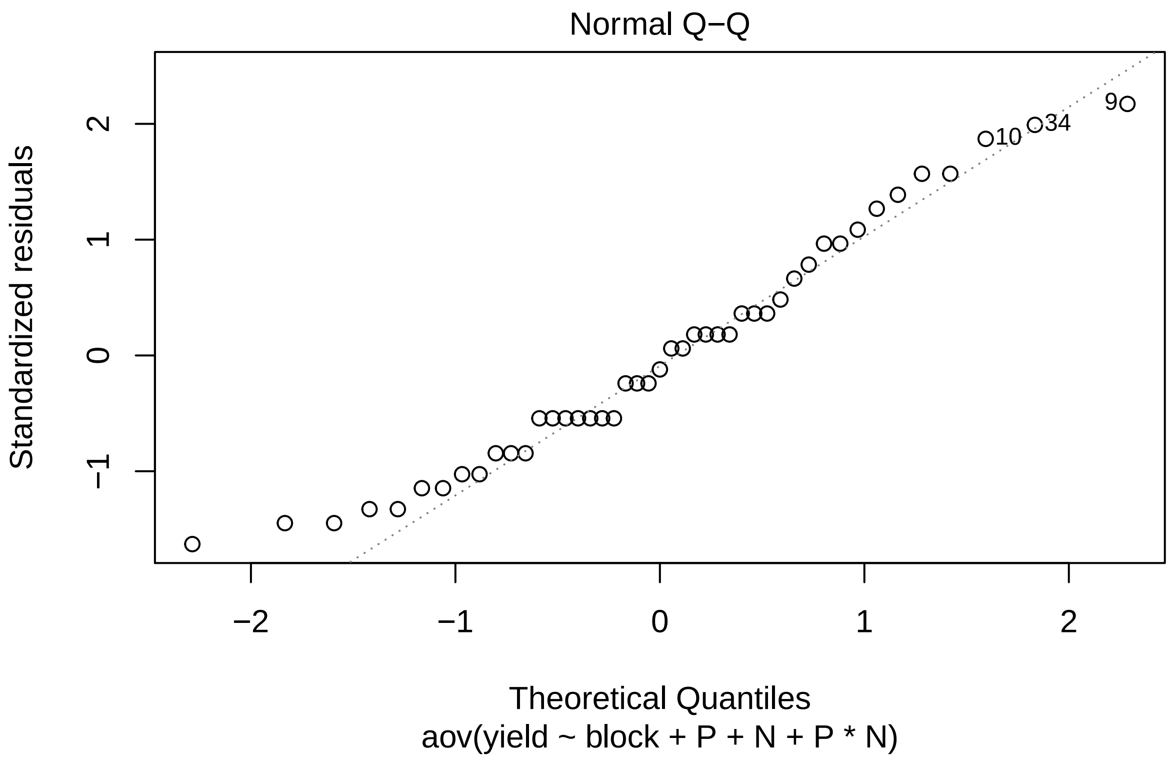
Model

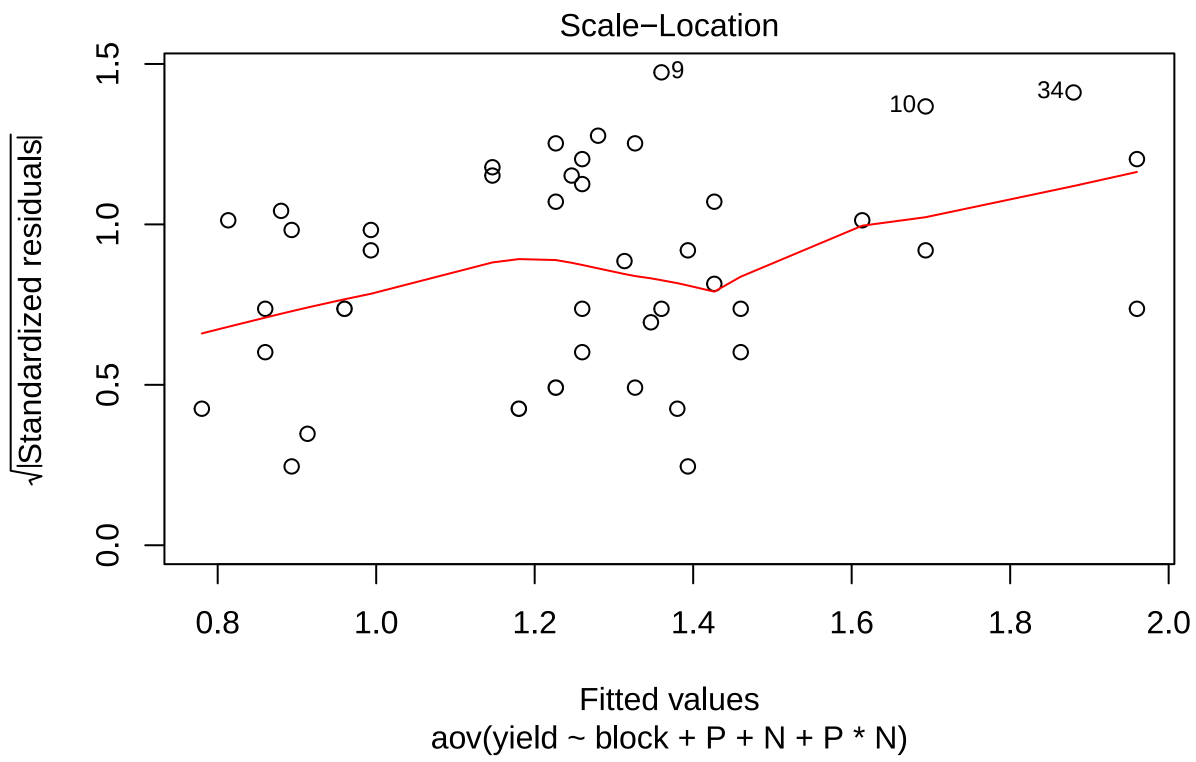
```
model12.3 <- aov(yield ~ block + P + N + P*N, data=example12.3)
```

Residual analysis

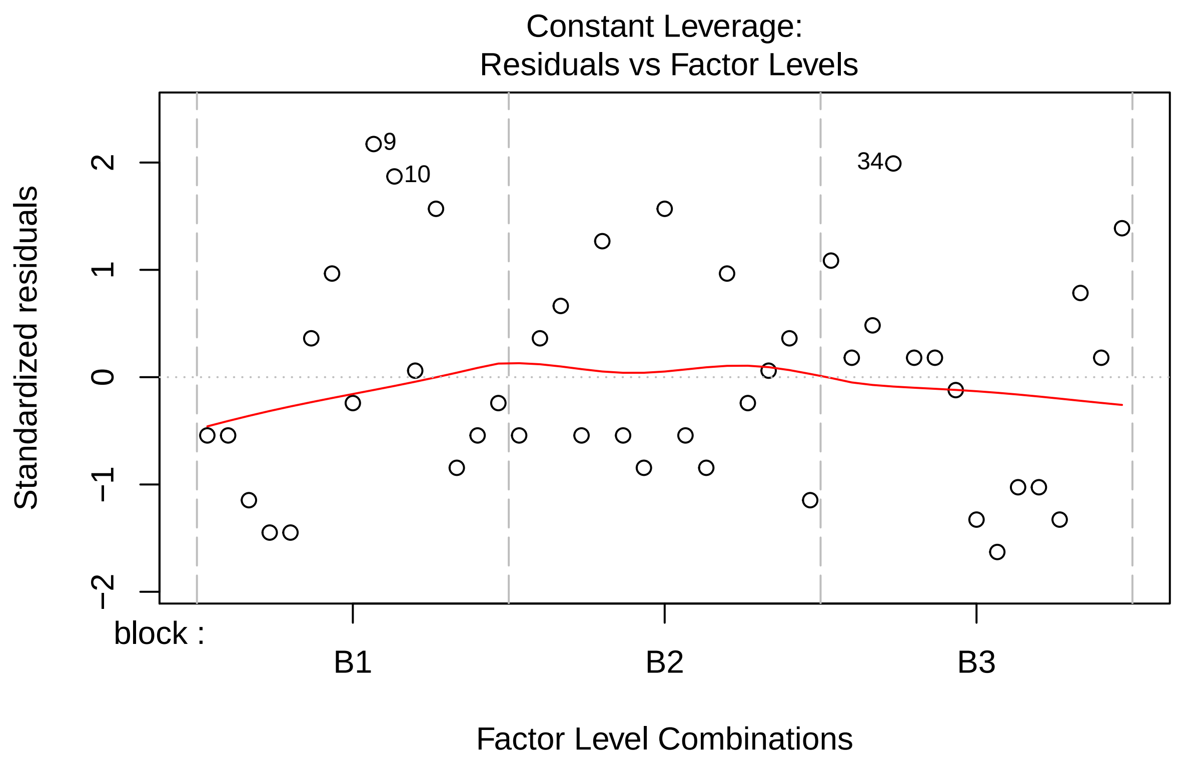
```
plot(model12.3)
```


Residuals vs Fitted
Residuals
Fitted values
aov(yield ~ block + P + N + P * N)
9
10
34
0.2
0.1
0.0
−0.1
−0.2
0.8
1.0
1.2
1.4
1.6
1.8
2.0


Normal Q−Q
Standardized residuals
Theoretical Quantiles
aov(yield ~ block + P + N + P * N)
−2
−1
0
1
2
9
10
34



Scale−Location
√|Standardized residuals|
1.5
1.0
0.5
0.0
9
10
34
0.8
1.0
1.2
1.4
1.6
1.8
2.0
Fitted values
aov(yield ~ block + P + N + P * N)



Analysis of Variance

```
anova(model12.3)
```

```
## Analysis of Variance Table
##
## Response: yield
##           Df  Sum Sq Mean Sq F value    Pr(>F)
## block      2 0.06400 0.03200  1.6311   0.21377
## P          2 0.16933 0.08467  4.3155   0.02324 *
## N          4 2.49022 0.62256 31.7322 4.946e-10 ***
## P:N        8 1.01511 0.12689  6.4676 8.979e-05 ***
## Residuals 28 0.54933 0.01962
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Comparison between the treatments

With factorial experiment we need to define for which factor we want to do the comparison

For P

```
LSD.test(model12.3, trt = "P", console=TRUE)
```

```
##
## Study: model12.3 ~ "P"
##
```

```
## LSD t Test for yield
##
## Mean Square Error:  0.01961905
##
## P,  means and individual ( 95 %) CI
##
##       yield       std  r      LCL      UCL Min Max
## P1 1.346667 0.3542934 15 1.272585 1.420748 0.9 2.1
## P2 1.220000 0.3405877 15 1.145919 1.294081 0.8 1.9
## P3 1.213333 0.2294922 15 1.139252 1.287415 0.7 1.5
##
## Alpha: 0.05 ; DF Error: 28
## Critical Value of t: 2.048407
##
## least Significant Difference: 0.104767
##
## Treatments with the same letter are not significantly different.
##
##       yield groups
## P1 1.346667      a
## P2 1.220000      b
## P3 1.213333      b
```

For N

```
LSD.test(model12.3, trt = "N", console=TRUE)
```

```
##
## Study: model12.3 ~ "N"
##
## LSD t Test for yield
##
## Mean Square Error:  0.01961905
##
## N,  means and individual ( 95 %) CI
##
##        yield        std r      LCL       UCL Min Max
## N1 0.8777778 0.09718253 9 0.782139 0.9734165 0.7 1.0
## N2 1.1333333 0.17320508 9 1.037695 1.2289721 0.9 1.4
## N3 1.3555556 0.12360331 9 1.259917 1.4511943 1.1 1.5
## N4 1.5666667 0.31622777 9 1.471028 1.6623054 1.1 2.1
## N5 1.3666667 0.26457513 9 1.271028 1.4623054 1.1 1.9
##
## Alpha: 0.05 ; DF Error: 28
## Critical Value of t: 2.048407
##
## least Significant Difference: 0.1352536
##
## Treatments with the same letter are not significantly different.
##
##        yield groups
## N4 1.5666667      a
## N5 1.3666667      b
## N3 1.3555556      b
## N2 1.1333333      c
## N1 0.8777778      d
```

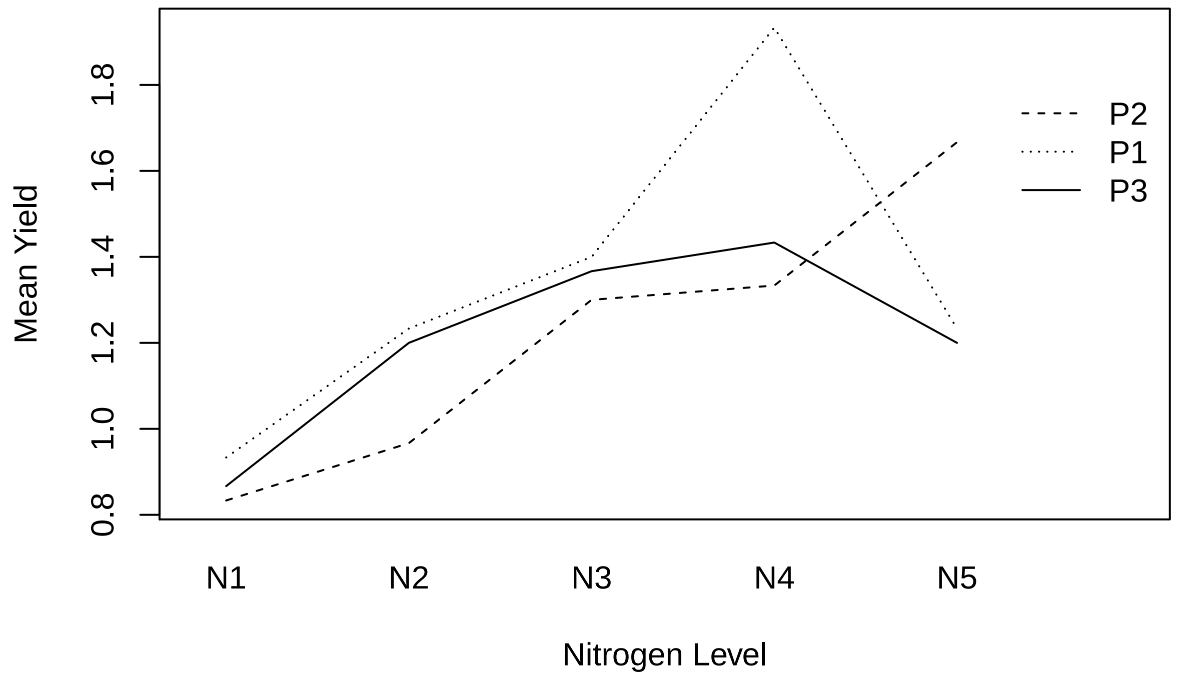
For P*N interaction, the `agricolae` syntax is `c("P","N")`

```
LSD.test(model12.3, trt = c("P","N"), console=TRUE)
```

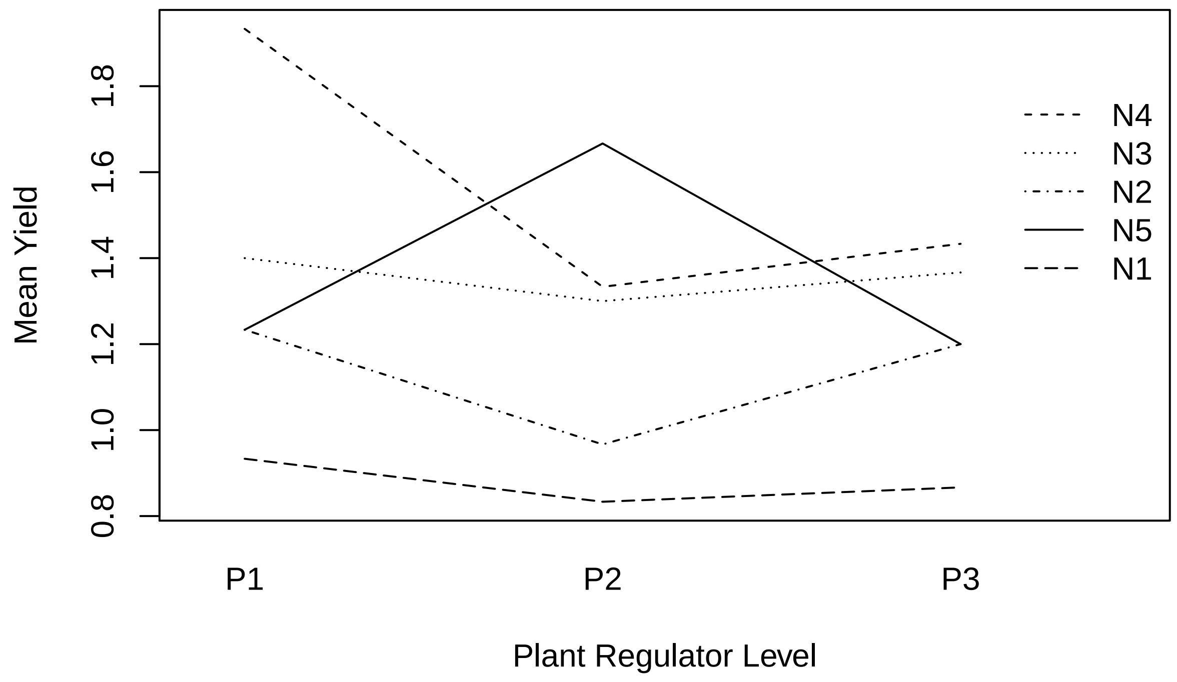
```
##
## Study: model12.3 ~ c("P", "N")
##
## LSD t Test for yield
##
## Mean Square Error:  0.01961905
##
## P:N,  means and individual ( 95 %) CI
##
##           yield        std r       LCL       UCL Min Max
## P1:N1 0.9333333 0.05773503 3 0.7676821 1.0989845 0.9 1.0
## P1:N2 1.2333333 0.05773503 3 1.0676821 1.3989845 1.2 1.3
## P1:N3 1.4000000 0.10000000 3 1.2343488 1.5656512 1.3 1.5
## P1:N4 1.9333333 0.15275252 3 1.7676821 2.0989845 1.8 2.1
## P1:N5 1.2333333 0.15275252 3 1.0676821 1.3989845 1.1 1.4
## P2:N1 0.8333333 0.05773503 3 0.6676821 0.9989845 0.8 0.9
## P2:N2 0.9666667 0.11547005 3 0.8010155 1.1323179 0.9 1.1
## P2:N3 1.3000000 0.20000000 3 1.1343488 1.4656512 1.1 1.5
## P2:N4 1.3333333 0.25166115 3 1.1676821 1.4989845 1.1 1.6
## P2:N5 1.6666667 0.20816660 3 1.5010155 1.8323179 1.5 1.9
## P3:N1 0.8666667 0.15275252 3 0.7010155 1.0323179 0.7 1.0
## P3:N2 1.2000000 0.20000000 3 1.0343488 1.3656512 1.0 1.4
## P3:N3 1.3666667 0.05773503 3 1.2010155 1.5323179 1.3 1.4
## P3:N4 1.4333333 0.05773503 3 1.2676821 1.5989845 1.4 1.5
## P3:N5 1.2000000 0.10000000 3 1.0343488 1.3656512 1.1 1.3
##
## Alpha: 0.05 ; DF Error: 28
## Critical Value of t: 2.048407
##
## least Significant Difference: 0.2342662
##
## Treatments with the same letter are not significantly different.
##
##           yield groups
## P1:N4 1.9333333      a
## P2:N5 1.6666667      b
## P3:N4 1.4333333     bc
## P1:N3 1.4000000      c
## P3:N3 1.3666667      c
## P2:N4 1.3333333      c
## P2:N3 1.3000000      c
## P1:N2 1.2333333      c
## P1:N5 1.2333333      c
## P3:N2 1.2000000     cd
## P3:N5 1.2000000     cd
## P2:N2 0.9666667     de
## P1:N1 0.9333333      e
## P3:N1 0.8666667      e
## P2:N1 0.8333333      e
```

Interaction Plots

```
interaction.plot(example12.3$N,
                 example12.3$P,
                 example12.3$yield,
                 ylab= "Mean Yield",
                 xlab = "Nitrogen Level",
                 trace.label = NULL)
```



```
interaction.plot(example12.3$P,
                 example12.3$N,
                 example12.3$yield,
                 ylab= "Mean Yield",
                 xlab = "Plant Regulator Level",
                 trace.label = NULL)
```


Mean Yield
0.8
1.0
1.2
1.4
1.6
1.8
P1
P2
P3
Plant Regulator Level
N4
N3
N2
N5
N1

Chapter 13: Comparison of Treatment Means

Since the book does not provide the Example 13.1 original data, here we show how to perform the comparison of treatment means using the example 9.1

```
example9.1 = data.frame(variety = c("A", "D", "B", "D", "C",
                                    "C", "D", "D", "A", "D",
                                    "A", "B", "C", "C", "B",
                                    "A", "B", "A", "C", "B"),
                        yield = c(22.2, 23.9, 24.1, 21.7, 25.9,
                                  18.4, 24.8, 28.2, 17.3, 26.4,
                                  21.2, 30.3, 23.2, 21.9, 27.4,
                                  25.2, 26.4, 16.1, 22.6, 34.8))
model9.1 = aov(yield~variety,data=example9.1)
anova(model9.1)
```

```
## Analysis of Variance Table
##
## Response: yield
##           Df Sum Sq Mean Sq F value   Pr(>F)
## variety    3  188.2  62.733  5.6901 0.00756 **
## Residuals 16  176.4  11.025
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

13.2.1 Least Significant Difference

```
library(agricolae)
LSD.test(model9.1, "variety", console=TRUE)
```

```
##
## Study: model9.1 ~ "variety"
##
## LSD t Test for yield
##
## Mean Square Error:  11.025
##
## variety,  means and individual ( 95 %) CI
##
##   yield      std r      LCL     UCL  Min  Max
## A  20.4 3.708773 5 17.2521 23.5479 16.1 25.2
## B  28.6 4.118859 5 25.4521 31.7479 24.1 34.8
## C  22.4 2.700926 5 19.2521 25.5479 18.4 25.9
## D  25.0 2.466779 5 21.8521 28.1479 21.7 28.2
##
## Alpha: 0.05 ; DF Error: 16
## Critical Value of t: 2.119905
##
## least Significant Difference: 4.451801
##
## Treatments with the same letter are not significantly different.
##
##   yield groups
## B  28.6      a
## D  25.0     ab
## C  22.4     bc
## A  20.4      c
```

13.2.2 Tukey's Honestly Significant Difference Test (HSD) a.k.a. Tukey's Studentised Range Test

```
library(agricolae)
HSD.test(model9.1, "variety", console=TRUE)
```

```
##
## Study: model9.1 ~ "variety"
##
## HSD Test for yield
##
## Mean Square Error:  11.025
##
## variety,  means
##
##   yield      std r  Min  Max
## A  20.4 3.708773 5 16.1 25.2
## B  28.6 4.118859 5 24.1 34.8
## C  22.4 2.700926 5 18.4 25.9
## D  25.0 2.466779 5 21.7 28.2
##
## Alpha: 0.05 ; DF Error: 16
## Critical Value of Studentized Range: 4.046093
##
## Minimun Significant Difference: 6.008142
##
## Treatments with the same letter are not significantly different.
##
##   yield groups
## B  28.6      a
## D  25.0     ab
## C  22.4      b
## A  20.4      b
```

13.2.3 Student-Newman-Keuls Test (SNK)

```
library(agricolae)
SNK.test(model9.1, "variety", console=TRUE)
```

```
##
## Study: model9.1 ~ "variety"
##
## Student Newman Keuls Test
## for yield
##
## Mean Square Error:  11.025
##
## variety,  means
##
##   yield      std r  Min  Max
## A  20.4 3.708773 5 16.1 25.2
## B  28.6 4.118859 5 24.1 34.8
## C  22.4 2.700926 5 18.4 25.9
## D  25.0 2.466779 5 21.7 28.2
##
## Alpha: 0.05 ; DF Error: 16
##
## Critical Range
##        2        3        4
## 4.451801 5.418695 6.008142
##
## Means with the same letter are not significantly different.
##
##   yield groups
## B  28.6      a
## D  25.0     ab
## C  22.4      b
## A  20.4      b
```

13.2.4 Duncan's Multiple Range Test (DMRT)

```r
library(agricolae)
duncan.test(model9.1, "variety", console=TRUE)
```

```
##
## Study: model9.1 ~ "variety"
##
## Duncan's new multiple range test
## for yield
##
## Mean Square Error:  11.025
##
## variety,  means
##
##   yield      std r  Min  Max
## A  20.4 3.708773 5 16.1 25.2
## B  28.6 4.118859 5 24.1 34.8
## C  22.4 2.700926 5 18.4 25.9
## D  25.0 2.466779 5 21.7 28.2
##
## Alpha: 0.05 ; DF Error: 16
##
## Critical Range
##        2        3        4
## 4.451801 4.668308 4.803648
##
## Means with the same letter are not significantly different.
##
##   yield groups
## B  28.6      a
## D  25.0     ab
## C  22.4      b
## A  20.4      b
```

13.2.5 Waller-Duncan's Bayes MSD test

```
library(agricolae)
waller.test(model9.1, "variety", console=TRUE)
```

```
##
## Study: model9.1 ~ "variety"
##
## Waller-Duncan K-ratio t Test for yield
##
## This test minimizes the Bayes risk under additive loss and certain other assumptions
##                                 ......
## K ratio                   100.000000
## Error Degrees of Freedom   16.000000
## Error Mean Square          11.025000
## F value                     5.690098
## Critical Value of Waller    2.153000
##
## variety,  means
##
##   yield      std r  Min  Max
## A  20.4 3.708773 5 16.1 25.2
## B  28.6 4.118859 5 24.1 34.8
## C  22.4 2.700926 5 18.4 25.9
## D  25.0 2.466779 5 21.7 28.2
##
## Minimum Significant Difference 4.5213
## Treatments with the same letter are not significantly different.
##
##   yield groups
## B  28.6      a
## D  25.0     ab
## C  22.4     bc
## A  20.4      c
```

13.4.1 Testing for a Linear Trend in Treatment Means

Example 13.3

```
example13.3 = data.frame(block=rep(c("A","B","C","D"),each=6),
                         N=rep(c(0,25,50,75,100,125),4),
                         yield=c(3.6,4.8,4.4,5.3,4.8,5.0,
                                 4.1,5.1,5.2,5.9,5.5,5.4,
                                 3.2,4.0,4.6,4.6,5.2,4.8,
                                 3.9,3.9,4.8,5.0,5.1,4.6))
```

Analysis of Variance with N as factor (qualitative)

```
example13.3$N = as.factor(example13.3$N)
model13.3 = aov(yield~block+N,data=example13.3)
anova(model13.3)
```

```
## Analysis of Variance Table
##
## Response: yield
##            Df Sum Sq Mean Sq F value    Pr(>F)
## block       3   2.19   0.730   9.359  0.000988 ***
## N           5   6.32   1.264  16.205 1.397e-05 ***
## Residuals 15   1.17   0.078
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Analysis of Variance with N as numeric for testing a linear trend. It is necessary to include the treatment at factor (`factor(N)`) to count for the treatment deviations in the ANOVA table after `N` in the formula, i.e., `N + factor(N)`. **Attention!** The opposite `factor(N) + N` does not result in the same result, here the order of the summation matters.

```r
example13.3 = data.frame(block=rep(c("A","B","C","D"),each=6),
                         N=rep(c(0,25,50,75,100,125),4),
                         yield=c(3.6,4.8,4.4,5.3,4.8,5.0,
                                 4.1,5.1,5.2,5.9,5.5,5.4,
                                 3.2,4.0,4.6,4.6,5.2,4.8,
                                 3.9,3.9,4.8,5.0,5.1,4.6))
model13.3linear = aov(yield~block+N+factor(N),data=example13.3)
anova(model13.3linear)
```

```
## Analysis of Variance Table
##
## Response: yield
##            Df Sum Sq Mean Sq F value    Pr(>F)
## block       3 2.1900  0.7300  9.3590  0.000988 ***
## N           1 4.4251  4.4251 56.7326 1.794e-06 ***
## factor(N)  4 1.8949  0.4737  6.0733  0.004115 **
## Residuals 15 1.1700  0.0780
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

13.4.2 Testing for a Quadratic Trend in Treatment Means

Analysis of Variance with N as numeric for testing a linear and quadratic trends. Note that here we need to use the `I()` function to avoid the direct sum of the terms before going into the function. It is necessary to include the treatment at factor (`factor(N)`) to count for the treatment deviations in the ANOVA table.

```r
example13.3 = data.frame(block=rep(c("A","B","C","D"),each=6),
                         N=rep(c(0,25,50,75,100,125),4),
                         yield=c(3.6,4.8,4.4,5.3,4.8,5.0,
                                 4.1,5.1,5.2,5.9,5.5,5.4,
                                 3.2,4.0,4.6,4.6,5.2,4.8,
                                 3.9,3.9,4.8,5.0,5.1,4.6))

model13.3quadratic = aov(yield~block+I(N)+I(N^2)+factor(N),data=example13.3)
anova(model13.3quadratic)
```

```
## Analysis of Variance Table
##
## Response: yield
##            Df Sum Sq Mean Sq F value    Pr(>F)
## block       3 2.1900  0.7300  9.3590 0.0009880 ***
## I(N)        1 4.4251  4.4251 56.7326 1.794e-06 ***
## I(N^2)      1 1.8011  1.8011 23.0907 0.0002315 ***
## factor(N)  3 0.0938  0.0313  0.4008 0.7544699
## Residuals 15 1.1700  0.0780
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(model13.3quadratic)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## block        3  2.190   0.730   9.359 0.000988 ***
## I(N)         1  4.425   4.425  56.733 1.79e-06 ***
## I(N^2)       1  1.801   1.801  23.091 0.000231 ***
## factor(N)    3  0.094   0.031   0.401 0.754470
## Residuals   15  1.170   0.078
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

13.4.4 Regression in a Two-Factor Experiment

Now we need to have the factors as numeric. The following example test linear trends for each factor and also for the interaction.

```
example12.3 = data.frame(
  block = rep(c("B1","B2","B3"),each=15),
  P = rep(rep(c(1,2,3),each=5),3),
  N = rep(c(1,2,3,4,5),9),
  yield = c(0.9,1.2,1.3,1.8,1.1,0.9,1.1,1.3,1.6,1.9,0.9,1.4,1.3,1.4,1.2,
            0.9,1.3,1.5,1.9,1.4,0.8,0.9,1.5,1.3,1.6,1.0,1.2,1.4,1.5,1.1,
            1.0,1.2,1.4,2.1,1.2,0.8,0.9,1.1,1.1,1.5,0.7,1.0,1.4,1.4,1.3)
  )

model12.3 = aov(yield~block+P+factor(P)+N+factor(N)+P*N+factor(P*N),data=example12.3)
summary(model12.3)
```

```
##               Df Sum Sq Mean Sq F value   Pr(>F)
## block          2 0.0640  0.0320   1.688   0.2026
## P              1 0.1333  0.1333   7.032   0.0128 *
## factor(P)      1 0.0360  0.0360   1.899   0.1788
## N              1 1.7921  1.7921  94.512 1.25e-10 ***
## factor(N)      3 0.6981  0.2327  12.272 2.33e-05 ***
## factor(P * N)  7 1.0146  0.1449   7.644 3.08e-05 ***
## Residuals     29 0.5499  0.0190
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

13.5 Treatment with Structure (Contrasts)

Example 13.5

```
example13.5 = data.frame(
  block = rep(c("B1","B2","B3","B4"),each=5),
  herbicide = rep(c("A","B","C","D","E"),4),
  yield = c(64,68,71,75,69,
            66,64,69,74,71,
            68,60,75,78,65,
            59,61,68,72,60)
  )
```

```r
model13.5 = aov(yield~block+herbicide,data=example13.5)
anova(model13.5)
```

```
## Analysis of Variance Table
##
## Response: yield
##           Df Sum Sq Mean Sq F value    Pr(>F)
## block      3  99.75   33.25  3.9118 0.0368135 *
## herbicide  4 370.80   92.70 10.9059 0.0005752 ***
## Residuals 12 102.00    8.50
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Building the contrasts table (as table 13.14)

```r
c1 <- c(4,-1,-1,-1,-1)
c2 <- c(0,3,-1,-1,-1)
c3 <- c(0,0,2,-1,-1)
c4 <- c(0,0,0,1,-1)

## Combine in a matrix
contrasts <- cbind(c1,c2,c3,c4)

## Define contrasts in the object herbicides
contrasts(example13.5$herbicide) <- contrasts

## ANOVA w/ contrasts
model13.5 = aov(yield~block+herbicide,data=example13.5)
summary.aov(model13.5,split =list(herbicide=list(1,2,3,4)))
```

```
##                 Df Sum Sq Mean Sq F value   Pr(>F)
## block            3   99.8   33.25   3.912 0.036814 *
## herbicide        4  370.8   92.70  10.906 0.000575 ***
##   herbicide: C1  1   64.8   64.80   7.624 0.017245 *
##   herbicide: C2  1  161.3  161.33  18.980 0.000934 ***
##   herbicide: C3  1    0.2    0.17   0.020 0.890961
##   herbicide: C4  1  144.5  144.50  17.000 0.001413 **
## Residuals       12  102.0    8.50
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
## If you want to put the contrast names:
summary.aov(model13.5,split =list(herbicide=list("A vs B,C,D,E"=1, "B vs C,D,E" = 2, "C vs D,E"=3, "D v
```

```
##                            Df Sum Sq Mean Sq F value   Pr(>F)
## block                       3   99.8   33.25   3.912 0.036814 *
## herbicide                   4  370.8   92.70  10.906 0.000575 ***
##   herbicide: A vs B,C,D,E   1   64.8   64.80   7.624 0.017245 *
##   herbicide: B vs C,D,E     1  161.3  161.33  18.980 0.000934 ***
##   herbicide: C vs D,E       1    0.2    0.17   0.020 0.890961
##   herbicide: D vs E         1  144.5  144.50  17.000 0.001413 **
## Residuals                  12  102.0    8.50
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Including orthogonal contrasts

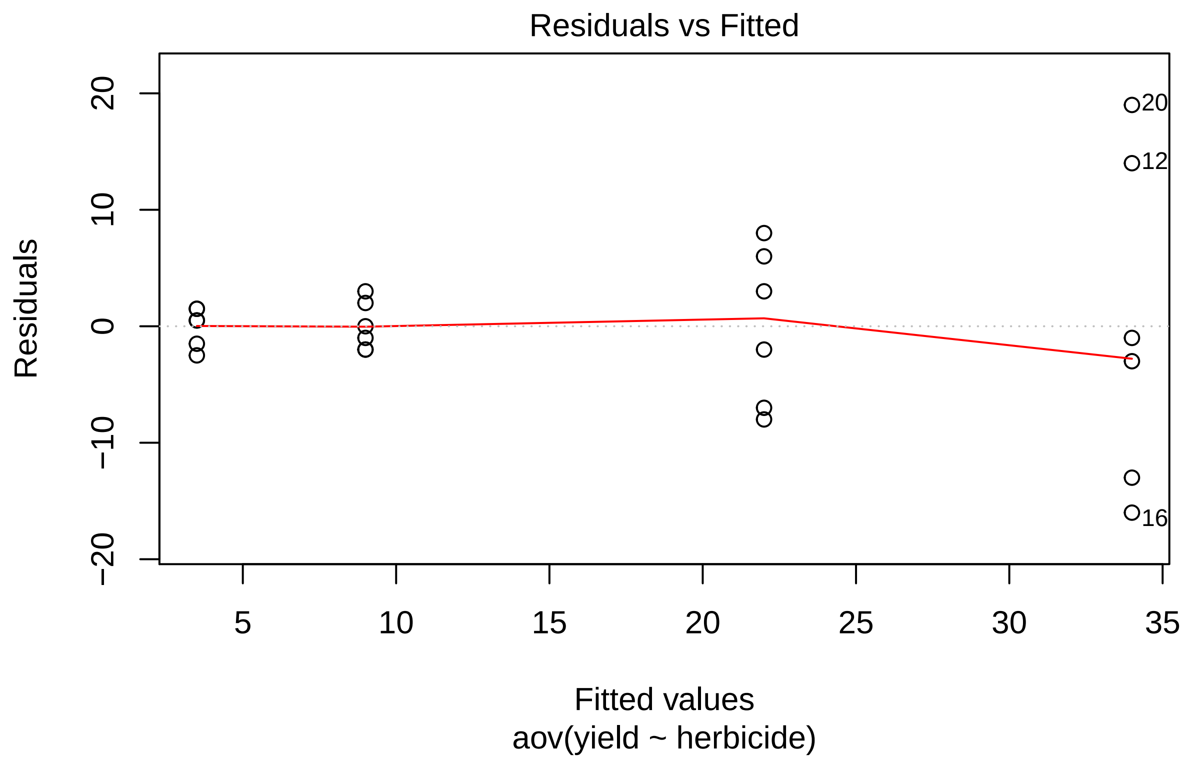
Chapter 14: Checking the Assumptions and Transformation of Data

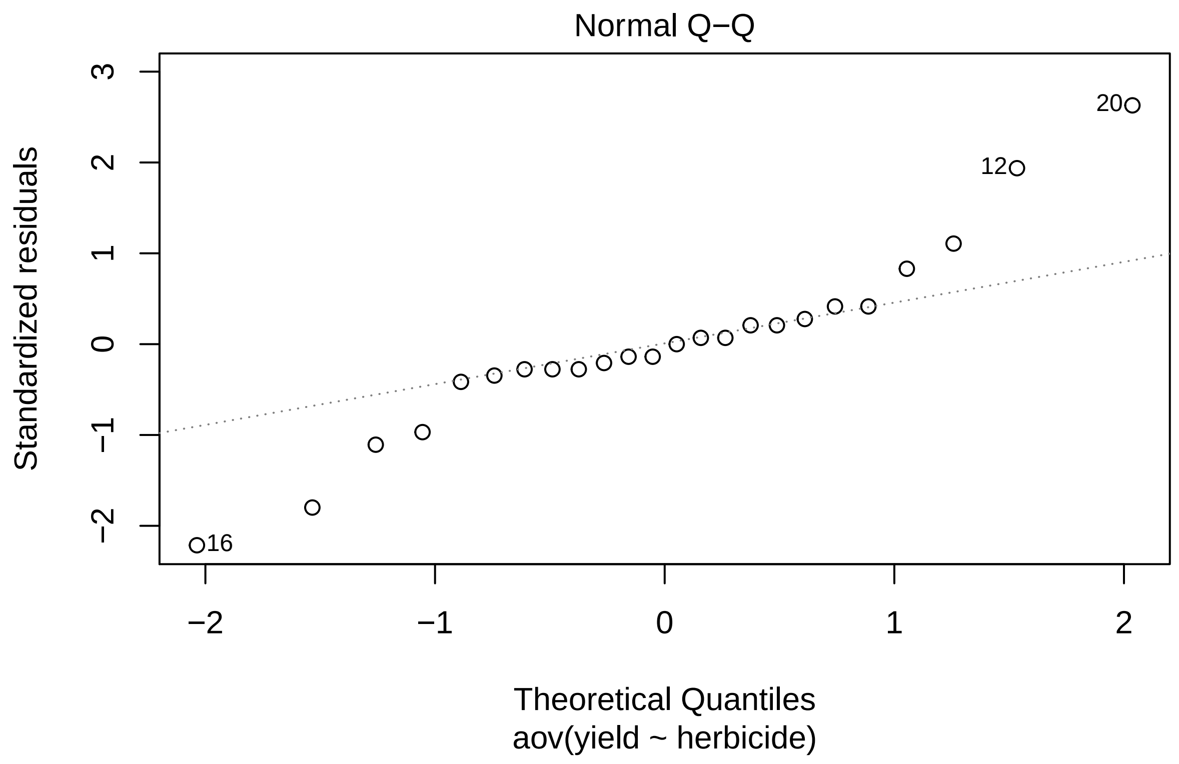
Example 14.1

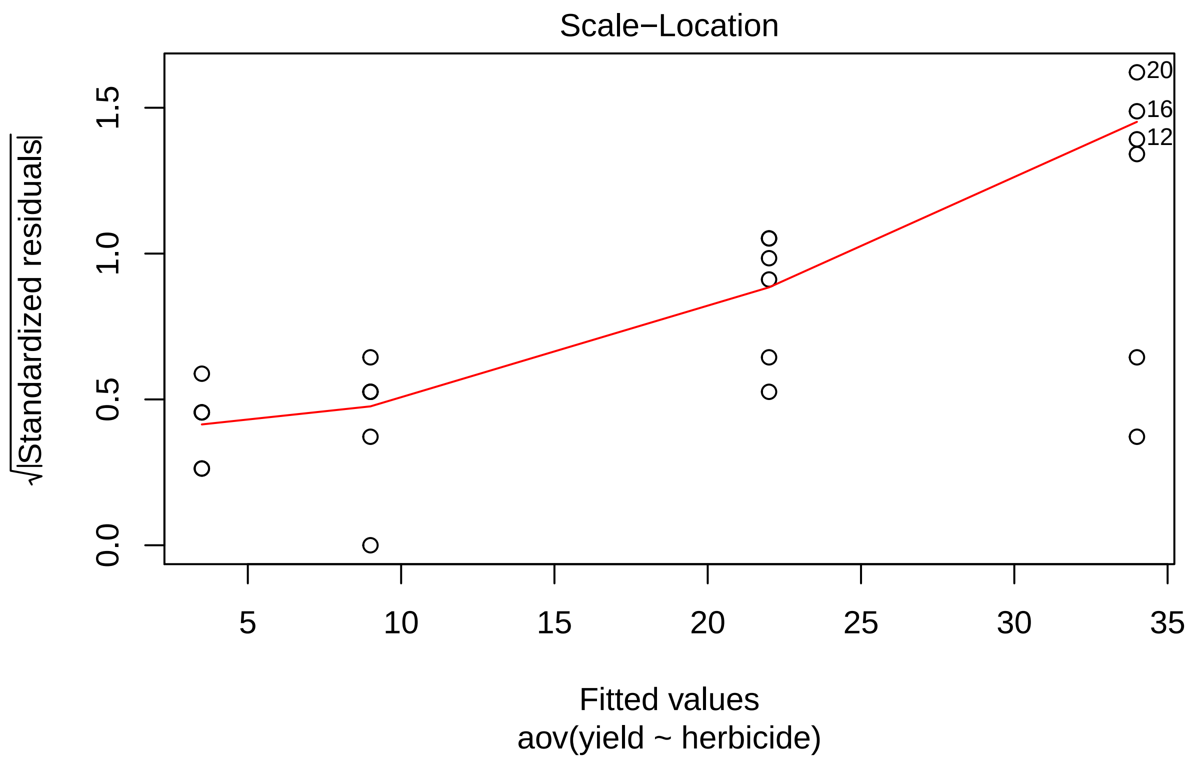
```r
example14.1 = data.frame(
  herbicide = rep(c("A","B","C","D"),6),
  yield = c(4,8,25,33,
            5,11,28,21,
            2,9,20,48,
            5,12,15,18,
            4,7,14,53,
            1,7,30,31)
)

model14.1 = aov(yield~herbicide,data=example14.1)
anova(model14.1)
```

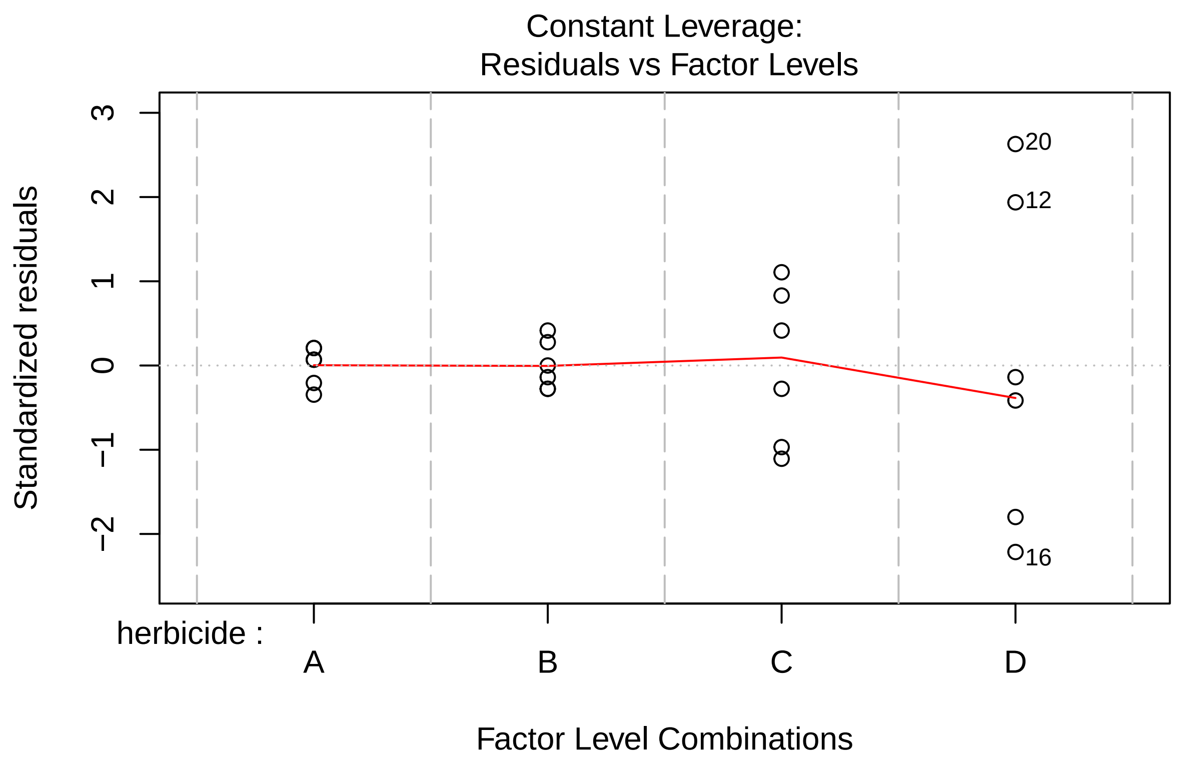
```
## Analysis of Variance Table
##
## Response: yield
##           Df Sum Sq Mean Sq F value    Pr(>F)
## herbicide  3 3361.1 1120.38  17.876 7.022e-06 ***
## Residuals 20 1253.5   62.67
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
plot(model14.1)
```


Residuals vs Fitted
Residuals
20
10
0
−10
−20
5
10
15
20
25
30
35
Fitted values
aov(yield ~ herbicide)
20
12
16


Normal Q−Q
Standardized residuals
Theoretical Quantiles
aov(yield ~ herbicide)
20
12
16
−2
−1
0
1
2
3


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
5
10
15
20
25
30
35
Fitted values
aov(yield ~ herbicide)
20
16
12

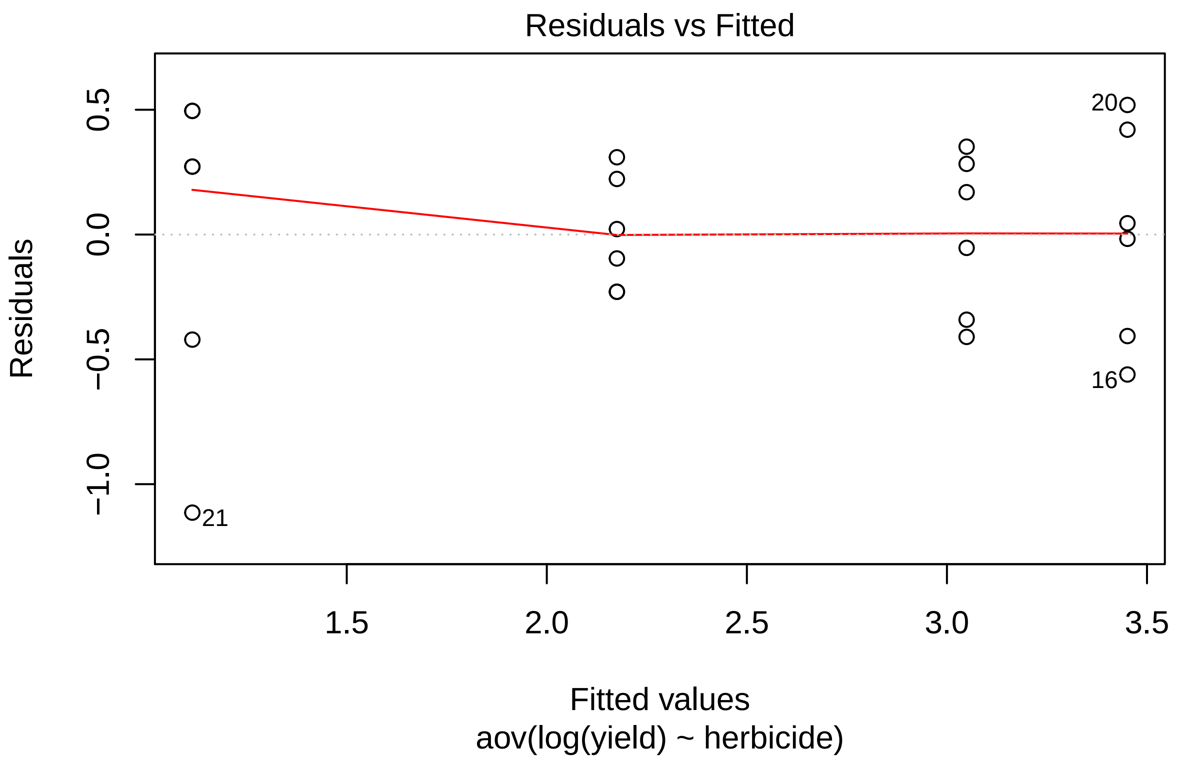


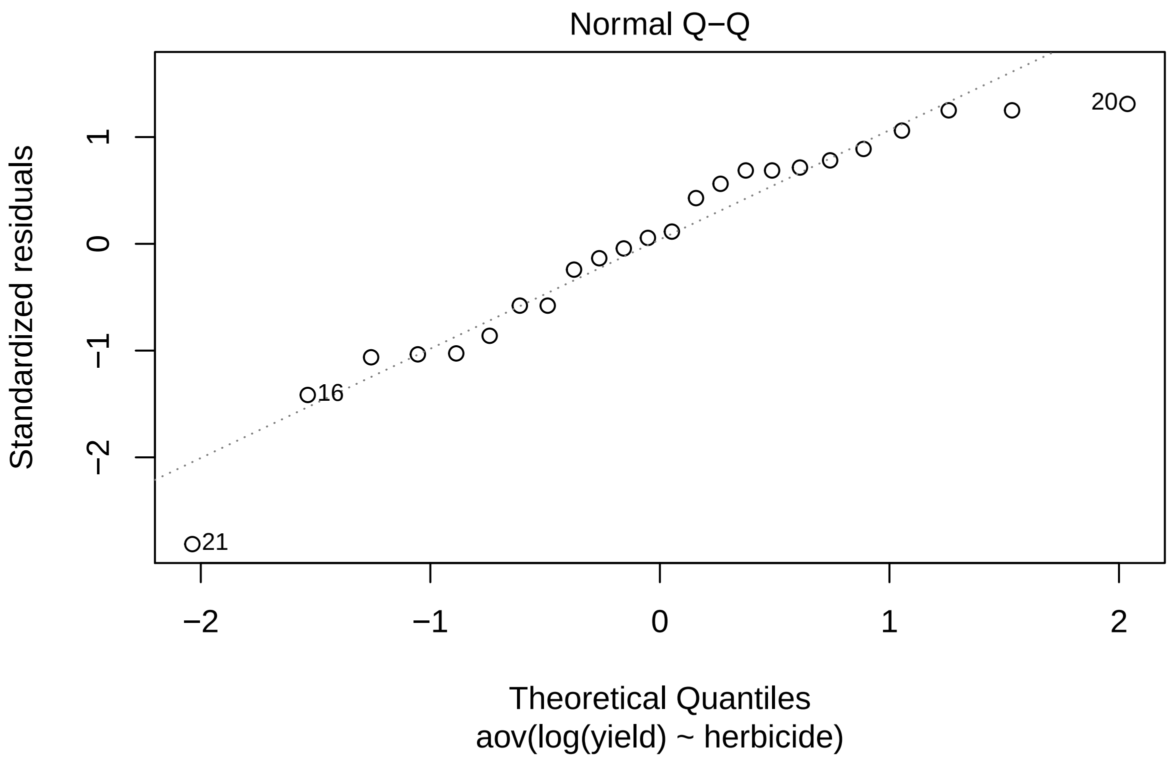
Log transformation

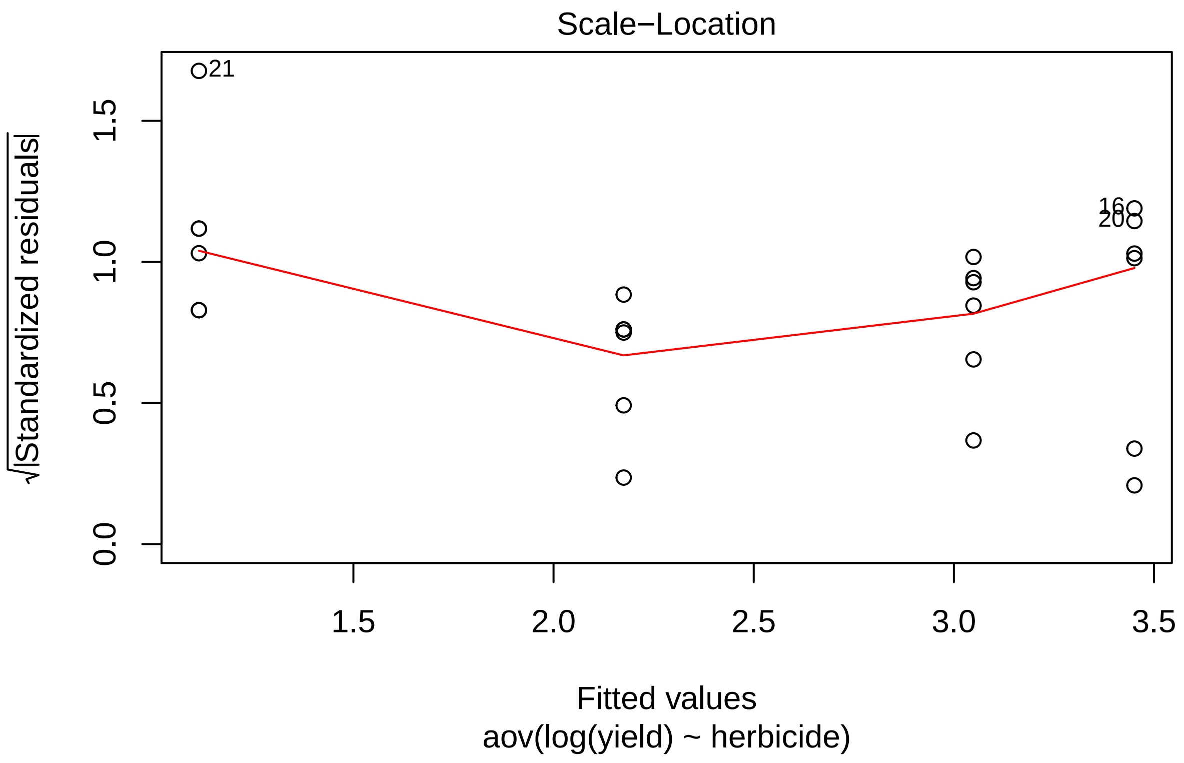
```
model14.1log = aov(log(yield)~herbicide,data=example14.1)
anova(model14.1log)
```

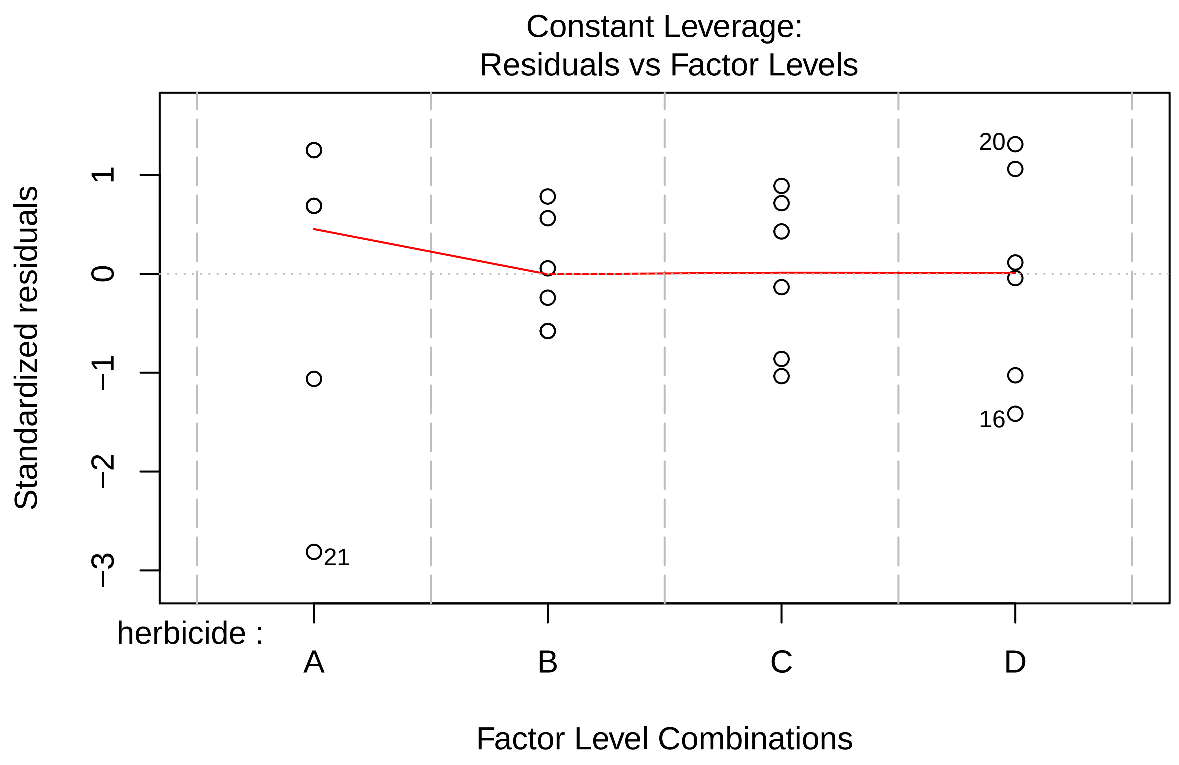
```
## Analysis of Variance Table
##
## Response: log(yield)
##           Df Sum Sq Mean Sq F value   Pr(>F)
## herbicide  3 19.328  6.4428  34.216 4.54e-08 ***
## Residuals 20  3.766  0.1883
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(model14.1log)
```


Residuals vs Fitted
Residuals
0.5
0.0
−0.5
−1.0
1.5
2.0
2.5
3.0
3.5
20
16
21
Fitted values
aov(log(yield) ~ herbicide)


Normal Q−Q
Standardized residuals
Theoretical Quantiles
aov(log(yield) ~ herbicide)
21
16
20
−2
−1
0
1
2


Scale−Location
21
16
20
1.5
1.0
0.5
0.0
1.5
2.0
2.5
3.0
3.5
√|Standardized residuals|
Fitted values
aov(log(yield) ~ herbicide)

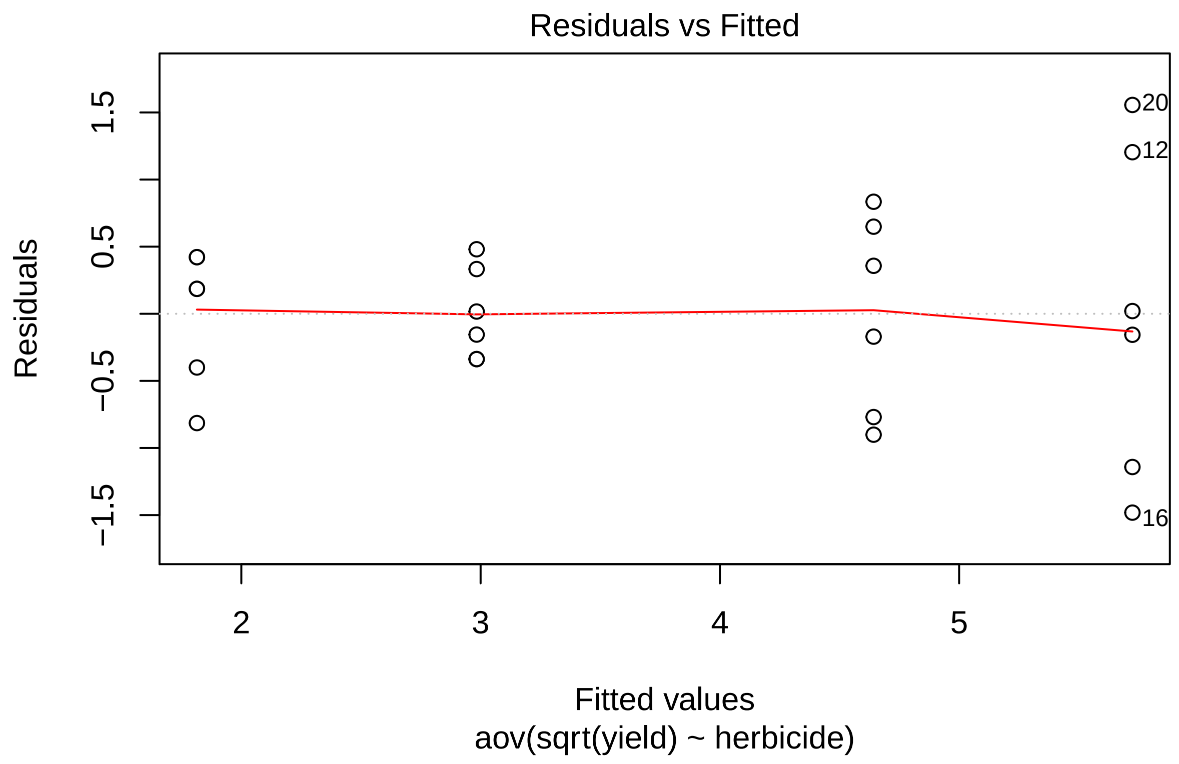


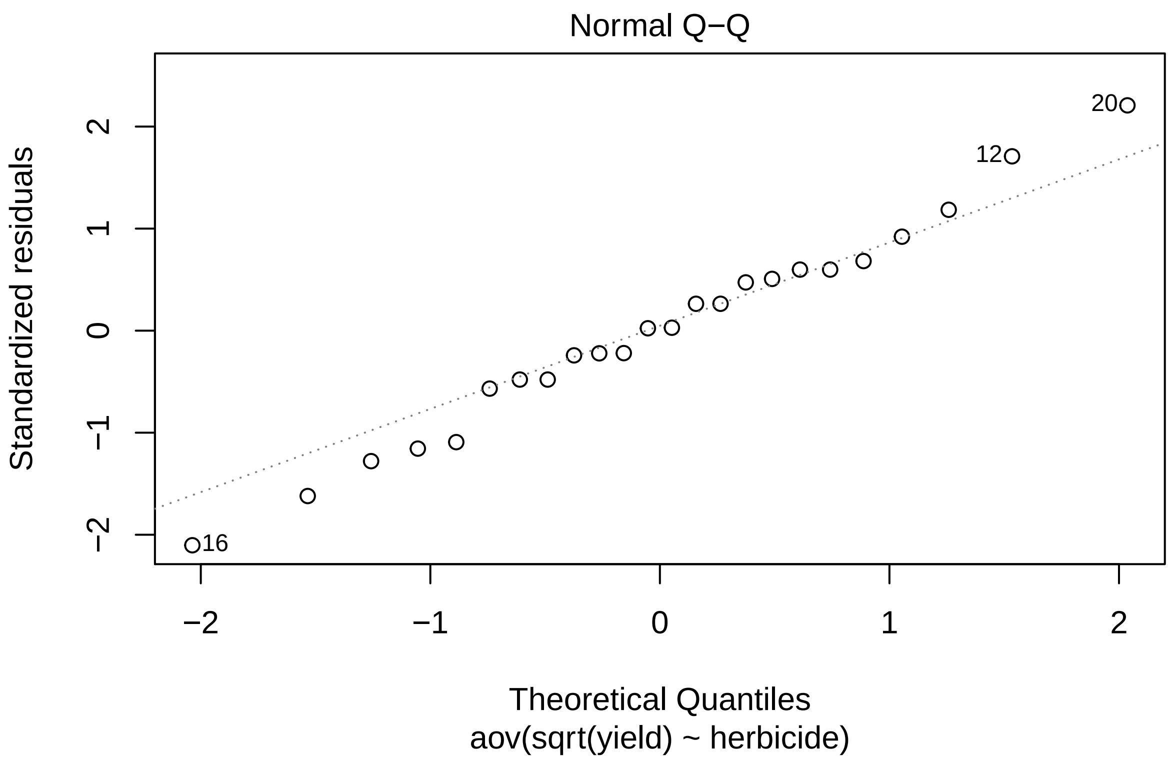
Square-Root transformation

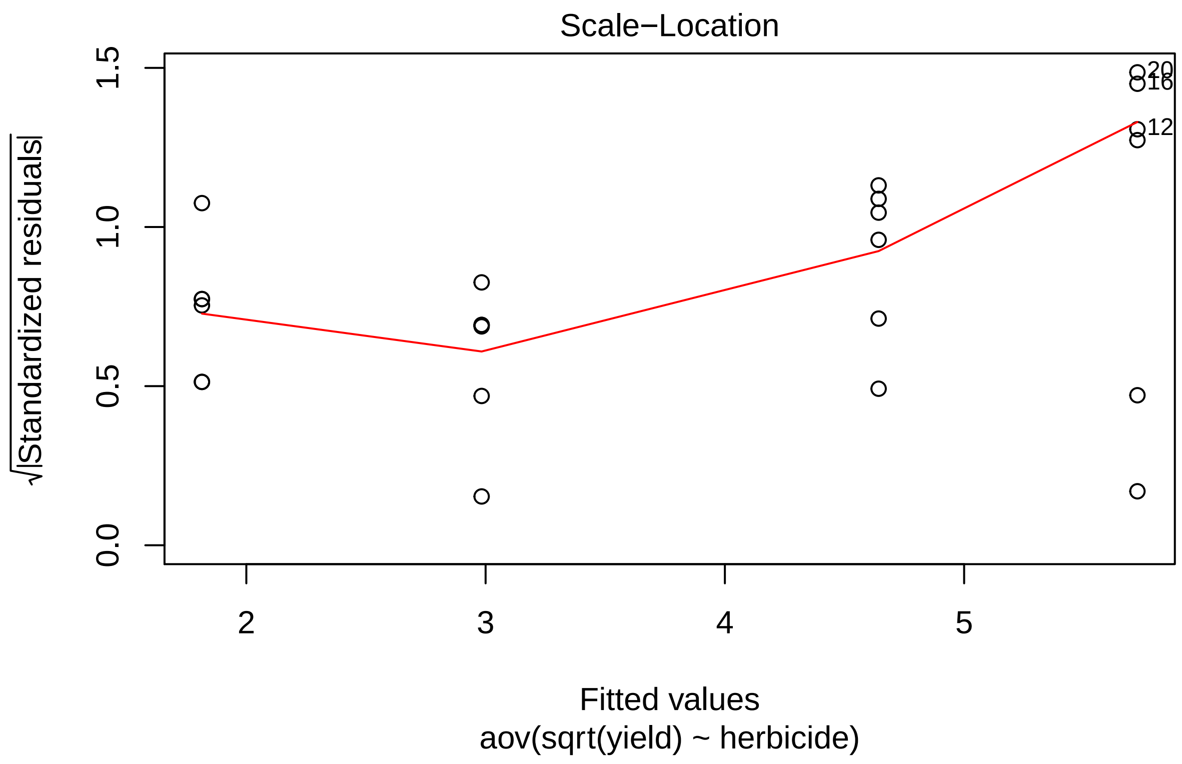
```
model14.1sqrt = aov(sqrt(yield)~herbicide,data=example14.1)
anova(model14.1sqrt)
```

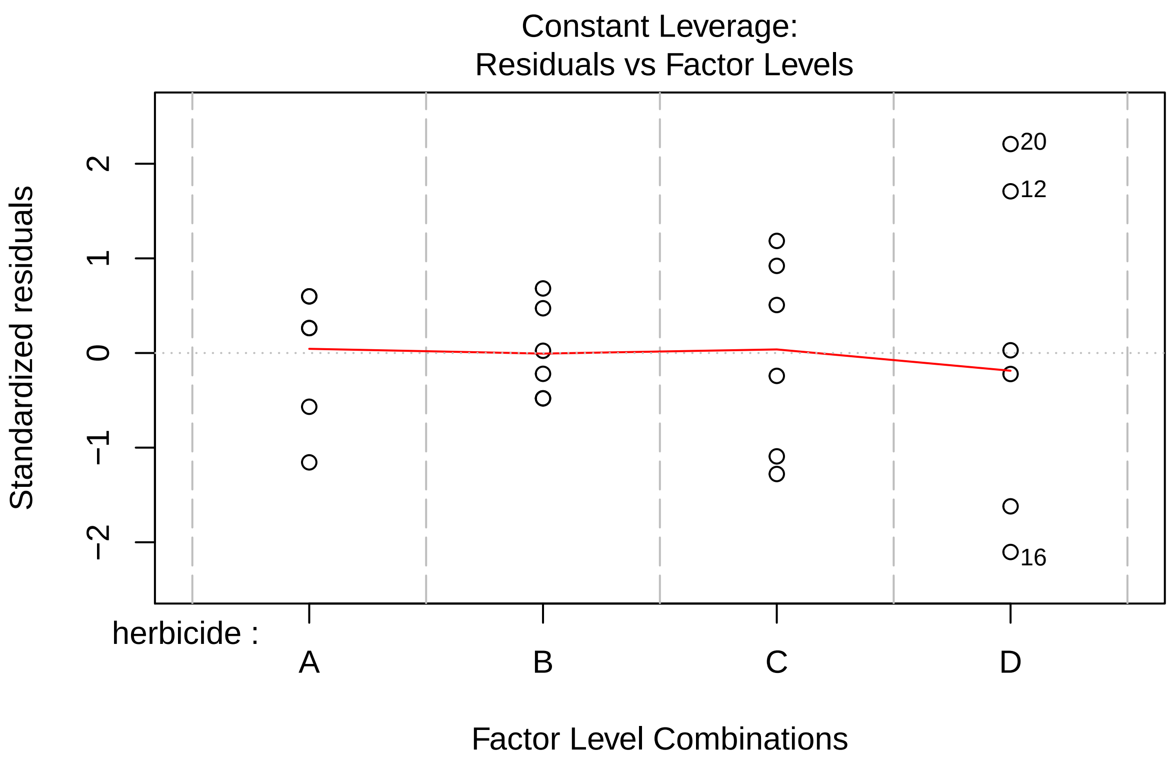
```
## Analysis of Variance Table
##
## Response: sqrt(yield)
##           Df Sum Sq Mean Sq F value    Pr(>F)
## herbicide  3 54.132 18.0440  30.289 1.235e-07 ***
## Residuals 20 11.914  0.5957
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(model14.1sqrt)
```


Residuals vs Fitted
Residuals
1.5
0.5
−0.5
−1.5
20
12
16
2
3
4
5
Fitted values
aov(sqrt(yield) ~ herbicide)


Normal Q−Q
Standardized residuals
Theoretical Quantiles
aov(sqrt(yield) ~ herbicide)
20
12
16
−2
−1
0
1
2


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2
3
4
5
20
16
12
Fitted values
aov(sqrt(yield) ~ herbicide)


Constant Leverage:
Residuals vs Factor Levels
Standardized residuals
2
1
0
−1
−2
20
12
16
herbicide :
A
B
C
D
Factor Level Combinations

Chapter 15: Missing Values and Incomplete Blocks

Example 15.1

Missing values in R are storage as NA

```
example15.1 = data.frame(
  treatment = rep(c("1","2","3","4"),each=3),
  Y1 = c(6.4,NA,5.6,
         9.8,8.7,7.2
         ,7.3,6.1,6.4,
         NA,8.0,9.4),
  Y2 = c(12.2,13.4,NA,
         15.5,16.3,17.8,
         10.4,NA,10.6,
         NA,16.8,17.8)
  )

model15.1 = aov(Y1~treatment,data=example15.1)
anova(model15.1)
```

```
## Analysis of Variance Table
##
## Response: Y1
##           Df  Sum Sq Mean Sq F value Pr(>F)
## treatment  3 13.2223  4.4074  4.8198 0.0487 *
## Residuals  6  5.4867  0.9144
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Example 15.2

The R default is to use the Adjusted SS when missing values are present.

```
example15.2 = data.frame(variety = c("V1", "V1", "V1",
                                     "V2", "V2", "V2",
                                     "V3", "V3", "V3",
                                     "V4", "V4", "V4"),
                         block = c("B1", "B2", "B3",
                                   "B1", "B2", "B3",
                                   "B1", "B2", "B3",
                                   "B1", "B2", "B3"),
                         yield = c(NA, 6.5, 5.6,
                                   9.8, 6.8, 6.2,
                                   7.3, 6.1, NA,
                                   9.5, 8.0, 7.4))

model15.2 = aov(yield~block+variety,data=example15.2)
anova(model15.2)
```

```
## Analysis of Variance Table
##
## Response: yield
##           Df  Sum Sq Mean Sq F value   Pr(>F)
## block      2 10.5993  5.2997 18.8792 0.009176 **
## variety    3  6.2938  2.0979  7.4736 0.040742 *
## Residuals  4  1.1229  0.2807
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Pairwise comparison

```r
library(agricolae)
LSD.test(model15.2,"variety",console=TRUE)
```

```
##
## Study: model15.2 ~ "variety"
##
## LSD t Test for yield
##
## Mean Square Error:  0.2807143
##
## variety,  means and individual ( 95 %) CI
##
##    yield       std r      LCL      UCL Min Max
## V1  6.05 0.6363961 2 5.009825 7.090175 5.6 6.5
## V2  7.60 1.9287302 3 6.750701 8.449299 6.2 9.8
## V3  6.70 0.8485281 2 5.659825 7.740175 6.1 7.3
## V4  8.30 1.0816654 3 7.450701 9.149299 7.4 9.5
##
## Alpha: 0.05 ; DF Error: 4
## Critical Value of t: 2.776445
##
## Groups according to probability of means differences and alpha level( 0.05 )
##
## Treatments with the same letter are not significantly different.
##
##    yield groups
## V4  8.30      a
## V2  7.60     ab
## V3  6.70     bc
## V1  6.05      c
```

```r
HSD.test(model15.2,"variety",console=TRUE)
```

```
##
## Study: model15.2 ~ "variety"
##
## HSD Test for yield
##
## Mean Square Error:  0.2807143
##
## variety,  means
##
##    yield       std r Min Max
## V1  6.05 0.6363961 2 5.6 6.5
## V2  7.60 1.9287302 3 6.2 9.8
## V3  6.70 0.8485281 2 6.1 7.3
## V4  8.30 1.0816654 3 7.4 9.5
##
## Alpha: 0.05 ; DF Error: 4
## Critical Value of Studentized Range: 5.757058
```

```
##
## Groups according to probability of means differences and alpha level( 0.05 )
##
## Treatments with the same letter are not significantly different.
##
##    yield groups
## V4  8.30      a
## V2  7.60     ab
## V3  6.70     ab
## V1  6.05      b
```

15.5 Incomplete Block Designs

15.5.1 Balanced Incomplete Blocks

Randomization

```
library(agricolae)
treatments = c(1:9)
field.design = design.bib(t=treatments,k=3,r=12)
```

```
##
## Parameters BIB
## ==============
## Lambda     : 3
## treatmeans : 9
## Block size : 3
## Blocks     : 36
## Replication: 12
##
## Efficiency factor 0.75
##
## <<< Book >>>
```

```
field.design
```

```
## $parameters
## $parameters$design
## [1] "bib"
##
## $parameters$trt
## [1] 1 2 3 4 5 6 7 8 9
##
## $parameters$k
## [1] 3
##
## $parameters$serie
## [1] 2
##
## $parameters$seed
## [1] 535947531
##
## $parameters$kinds
## [1] "Super-Duper"
##
##
```

```
## $statistics
##        lambda treatmeans blockSize blocks  r Efficiency
## values      3          9         3     36 12       0.75
##
## $sketch
##       [,1] [,2] [,3]
##  [1,] "9"  "2"  "7"
##  [2,] "9"  "3"  "6"
##  [3,] "2"  "1"  "7"
##  [4,] "5"  "8"  "2"
##  [5,] "7"  "5"  "3"
##  [6,] "8"  "2"  "5"
##  [7,] "2"  "3"  "7"
##  [8,] "1"  "9"  "5"
##  [9,] "6"  "4"  "2"
## [10,] "2"  "4"  "9"
## [11,] "7"  "6"  "8"
## [12,] "7"  "9"  "4"
## [13,] "3"  "8"  "1"
## [14,] "3"  "8"  "9"
## [15,] "3"  "4"  "9"
## [16,] "9"  "5"  "7"
## [17,] "6"  "5"  "3"
## [18,] "3"  "8"  "7"
## [19,] "5"  "2"  "6"
## [20,] "4"  "1"  "3"
## [21,] "9"  "6"  "8"
## [22,] "1"  "2"  "8"
## [23,] "2"  "1"  "9"
## [24,] "8"  "4"  "6"
## [25,] "6"  "2"  "3"
## [26,] "7"  "1"  "6"
## [27,] "8"  "4"  "7"
## [28,] "6"  "7"  "1"
## [29,] "4"  "5"  "1"
## [30,] "5"  "4"  "6"
## [31,] "5"  "8"  "9"
## [32,] "1"  "8"  "4"
## [33,] "5"  "7"  "4"
## [34,] "5"  "3"  "1"
## [35,] "1"  "6"  "9"
## [36,] "2"  "4"  "3"
##
## $book
##     plots block treatments
## 1     101     1          9
## 2     102     1          2
## 3     103     1          7
## 4     201     2          9
## 5     202     2          3
## 6     203     2          6
## 7     301     3          2
## 8     302     3          1
## 9     303     3          7
## 10    401     4           5
## 11    402     4           8
## 12    403     4           2
## 13    501     5           7
## 14    502     5           5
## 15    503     5           3
## 16    601     6           8
## 17    602     6           2
## 18    603     6           5
## 19    701     7           2
## 20    702     7           3
## 21    703     7           7
## 22    801     8           1
## 23    802     8           9
## 24    803     8           5
## 25    901     9           6
## 26    902     9           4
## 27    903     9           2
## 28   1001    10           2
## 29   1002    10           4
## 30   1003    10           9
## 31   1101    11           7
## 32   1102    11           6
## 33   1103    11           8
## 34   1201    12           7
## 35   1202    12           9
## 36   1203    12           4
## 37   1301    13           3
## 38   1302    13           8
## 39   1303    13           1
## 40   1401    14           3
## 41   1402    14           8
## 42   1403    14           9
## 43   1501    15           3
## 44   1502    15           4
## 45   1503    15           9
## 46   1601    16           9
## 47   1602    16           5
## 48   1603    16           7
## 49   1701    17           6
## 50   1702    17           5
## 51   1703    17           3
## 52   1801    18           3
## 53   1802    18           8
## 54   1803    18           7
## 55   1901    19           5
## 56   1902    19           2
## 57   1903    19           6
## 58   2001    20           4
## 59   2002    20           1
## 60   2003    20           3
## 61   2101    21           9
## 62   2102    21           6
## 63   2103    21           8
## 64   2201    22           1
## 65   2202    22           2
## 66   2203    22           8
## 67   2301    23           2
## 68   2302    23           1
## 69   2303    23           9
## 70   2401    24           8
## 71   2402    24           4
## 72   2403    24           6
## 73   2501    25           6
## 74   2502    25           2
## 75   2503    25           3
## 76   2601    26           7
## 77   2602    26           1
## 78   2603    26           6
## 79   2701    27           8
## 80   2702    27           4
## 81   2703    27           7
## 82   2801    28           6
## 83   2802    28           7
## 84   2803    28           1
## 85   2901    29           4
## 86   2902    29           5
## 87   2903    29           1
## 88   3001    30           5
## 89   3002    30           4
## 90   3003    30           6
## 91   3101    31           5
## 92   3102    31           8
## 93   3103    31           9
## 94   3201    32           1
## 95   3202    32           8
## 96   3203    32           4
## 97   3301    33           5
## 98   3302    33           7
## 99   3303    33           4
## 100  3401    34           5
## 101  3402    34           3
## 102  3403    34           1
## 103  3501    35           1
## 104  3502    35           6
## 105  3503    35           9
## 106  3601    36           2
## 107  3602    36           4
## 108  3603    36           3
```

Example 15.3

```
example15.3 = data.frame(block = rep(c("1","2","3","4"),each=3),
                         treatment = c("T1","T3","T2",
                                       "T4","T2","T1",
                                       "T2","T3","T4",
                                       "T3","T1","T4"),
```

```r
                         height = c(14,12,13,
                                    7.3,12.5,15.5,
                                    12.8,15.5,11,
                                    12.5,18.3,9.3))

model15.3 = aov(height ~ block+treatment,data=example15.3)
anova(model15.3)
```

```
## Analysis of Variance Table
##
## Response: height
##           Df Sum Sq Mean Sq F value   Pr(>F)
## block      3  4.556  1.5186   0.710 0.586366
## treatment  3 77.619 25.8731  12.097 0.009934 **
## Residuals  5 10.694  2.1388
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
HSD.test(model15.3,"treatment",console=TRUE)
```

```
##
## Study: model15.3 ~ "treatment"
##
## HSD Test for height
##
## Mean Square Error:  2.138833
##
## treatment,  means
##
##      height       std r  Min  Max
## T1 15.93333 2.1825062 3 14.0 18.3
## T2 12.76667 0.2516611 3 12.5 13.0
## T3 13.33333 1.8929694 3 12.0 15.5
## T4  9.20000 1.8520259 3  7.3 11.0
##
## Alpha: 0.05 ; DF Error: 5
## Critical Value of Studentized Range: 5.218325
##
## Minimun Significant Difference: 4.406147
##
## Treatments with the same letter are not significantly different.
##
##      height groups
## T1 15.93333      a
## T3 13.33333     ab
## T2 12.76667     ab
## T4  9.20000      b
```

Chapter 16: Split Plot Designs

Randomization

```
treatment1 = c("A","B","C","D")
treatment2 = c("I","II","III")
field_design = design.split(trt1 = treatment1, trt2 = treatment2,r = 3)
field_design$book
```

```
##    plots splots block treatment1 treatment2
## 1    101      1     1          A          I
## 2    101      2     1          A         II
## 3    101      3     1          A        III
## 4    102      1     1          C         II
## 5    102      2     1          C          I
## 6    102      3     1          C        III
## 7    103      1     1          D         II
## 8    103      2     1          D          I
## 9    103      3     1          D        III
## 10   104      1     1          B         II
## 11   104      2     1          B          I
## 12   104      3     1          B        III
## 13   105      1     2          A         II
## 14   105      2     2          A        III
## 15   105      3     2          A          I
## 16   106      1     2          D          I
## 17   106      2     2          D         II
## 18   106      3     2          D        III
## 19   107      1     2          B        III
## 20   107      2     2          B         II
## 21   107      3     2          B          I
## 22   108      1     2          C          I
## 23   108      2     2          C         II
## 24   108      3     2          C        III
## 25   109      1     3          D          I
## 26   109      2     3          D        III
## 27   109      3     3          D         II
## 28   110      1     3          C         II
## 29   110      2     3          C        III
## 30   110      3     3          C          I
## 31   111      1     3          A         II
## 32   111      2     3          A          I
## 33   111      3     3          A        III
## 34   112      1     3          B          I
## 35   112      2     3          B         II
## 36   112      3     3          B        III
```

Example 16.2

```
example16.2 = data.frame(block=rep(c("1","2","3","4"),each=6),
                         factorA = c("A2","A2","A2","A1","A1","A1",
                                     "A1","A1","A1","A2","A2","A2",
                                     "A1","A1","A1","A2","A2","A2",
                                     "A2","A2","A2","A1","A1","A1"),
                         factorB = c("B2","B1","B3","B3","B2","B1",
                                     "B1","B2","B3","B1","B3","B2",
                                     "B3","B2","B1","B2","B3","B1",
                                     "B3","B2","B1","B2","B3","B1"),
                         yield = c(25.5,24.9,25.8,26.1,18.0,21.7,
                                   21.1,17.9,28.9,27.6,29.4,29.3,
                                   28.6,19.5,23.2,29.7,30.3,29.5,
                                   29.3,29.2,29.8,21.3,31.0,25.8))
```

In order to correctly split the stratum, we create the **plotA** variable which represents the plot units for factor A.

```
example16.2$plotA = rep(c("1","2","3","4","5","6","7","8"),each=3)

model16.2 = aov(yield~block+factorA*factorB+Error(plotA),data=example16.2) #or exp
summary(model16.2)
```

```
##
## Error: plotA
##           Df Sum Sq Mean Sq F value  Pr(>F)
## block      3  55.06   18.35   5.158 0.10550
## factorA    1 136.33  136.33  38.318 0.00849 **
## Residuals  3  10.67    3.56
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Error: Within
##                 Df Sum Sq Mean Sq F value   Pr(>F)
## factorB          2  98.37   49.18  102.23 2.90e-08 ***
## factorA:factorB  2  84.80   42.40   88.13 6.71e-08 ***
## Residuals       12   5.77    0.48
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The pairwise test should be by stratum as follows and setting the MSerror and its degrees of freedom in the function.

```
HSD.test(example16.2$yield,example16.2$factorA,
         MSerror=3.56,DFerror=3,console=TRUE)
```

```
##
## Study: example16.2$yield ~ example16.2$factorA
##
## HSD Test for example16.2$yield
##
## Mean Square Error:  3.56
##
## example16.2$factorA,  means
##
##    example16.2.yield      std  r  Min  Max
## A1          23.59167 4.420090 12 17.9 31.0
## A2          28.35833 1.901415 12 24.9 30.3
##
## Alpha: 0.05 ; DF Error: 3
## Critical Value of Studentized Range: 4.500659
##
```

```
## Minimun Significant Difference: 2.451379
##
## Treatments with the same letter are not significantly different.
##
##    example16.2$yield groups
## A2          28.35833      a
## A1          23.59167      b
```

```
HSD.test(example16.2$yield,example16.2$factorB,
         MSerror=0.48,DFerror=12,console=TRUE)
```

```
##
## Study: example16.2$yield ~ example16.2$factorB
##
## HSD Test for example16.2$yield
##
## Mean Square Error:  0.48
##
## example16.2$factorB,  means
##
##    example16.2.yield      std r  Min  Max
## B1            25.450 3.347067 8 21.1 29.8
## B2            23.800 5.213992 8 17.9 29.7
## B3            28.675 1.848358 8 25.8 31.0
##
## Alpha: 0.05 ; DF Error: 12
## Critical Value of Studentized Range: 3.772929
##
## Minimun Significant Difference: 0.9241751
##
## Treatments with the same letter are not significantly different.
##
##    example16.2$yield groups
## B3            28.675      a
## B1            25.450      b
## B2            23.800      c
```

```
HSD.test(example16.2$yield,paste0(example16.2$factorB,example16.2$factorA),
         MSerror=0.48,DFerror=12,console=TRUE)
```

```
##
## Study: example16.2$yield ~ paste0(example16.2$factorB, example16.2$factorA)
##
## HSD Test for example16.2$yield
##
## Mean Square Error:  0.48
##
## paste0(example16.2$factorB, example16.2$factorA),  means
##
##      example16.2.yield      std r  Min  Max
## B1A1            22.950 2.095233 4 21.1 25.8
## B1A2            27.950 2.254625 4 24.9 29.8
## B2A1            19.175 1.594522 4 17.9 21.3
## B2A2            28.425 1.961929 4 25.5 29.7
## B3A1            28.650 2.007486 4 26.1 31.0
## B3A2            28.700 1.984943 4 25.8 30.3
##
## Alpha: 0.05 ; DF Error: 12
## Critical Value of Studentized Range: 4.750231
##
## Minimun Significant Difference: 1.645528
##
## Treatments with the same letter are not significantly different.
##
##      example16.2$yield groups
## B3A2            28.700      a
## B3A1            28.650      a
## B2A2            28.425      a
## B1A2            27.950      a
## B1A1            22.950      b
## B2A1            19.175      c
```