# Introduction to R: basic programming

Ziv Shkedy, Hasselt University

```
## Warning: package 'mvtnorm' was built under R version 3.6.2
```

# Introduction

## Slides, code and tutorials

This interactive book contains all R code that was used to produce the results and output presented in chapter 3 (programming) in the course's slides. We include YouTube tutorials as a part of the book and links to the relevant tutorials are provided in different chapters. Note that these tutorials were not developed especially for this book, they cover the same topics using different examples.

## R ?

No previous knowledge about R is required. We start from the basic and follow a user approach and not a programmer approach. The datasets used for illustrations are available in R and one of them (the law school data) is part of the `bootstrap` R package. To run the code smoothly, this package need to be installed.

```
library(bootstrap)
```

## Slides

Slide for this part of the course are avilable online in the >eR-BioStat website. See RcoursePrograming.

# R Objects

## YouTube tutorial: objects in R

For a short YouTube introduction, by Mike Marin, about objects in R see YTobjects1.

## Introduction

R works with objects.An object in R could be a scaler, for example

```
x<-1
```

We can print x

```
print(x)
```

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

The object x can be a vector

```
x<-c(1,2,3,4,5,6,7,8,9,10)
x
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

The data structure that will be discussed in this chapter are

- scaler
- vector
- matrix
- data frame

### Functions

Another important concept in R is a function. For example, if we want to calculate the mean of a sample we can use the function mean()

```
mean(x)
```

```
## [1] 5.5
```

This will be discuused from Chapter 3 onward.

## Scaler

The most simple object in Splus is a scaler. In order to define $X = 5$ we use

```
x <- 5
```

x is an object and the value of this object is 5. In order to see the object we type x

```
x
```

```
## [1] 5
```

Note that the arrow in R means equal (=). We can define a new object y which is equal to $x^2$:

```
 y <- x^2
 y
```

```
## [1] 25
```

The object $z = x + 5$.

```
z <- x + 5
z
```

```
## [1] 10
```

### Practical session

Define two objects $w = 10$ and $z = 3$, calculate $w \times z$, $z + w$, $w^z$, $w \times (z + 5)$.

## vectors

### YouTube tutorial: vectors and matrices in R

For a YouTube tutorial by Mike Marin about objects in R see YTobjects2.

### Numerical vectors

A vector in R can be defined by using the function c() .For example the vector $x = (10, 15, 15, 25, 100)$ can be defined in the following way

```
x <- c(10, 15, 15, 25, 100)
x
```

```
## [1]  10  15  15  25 100
```

The first element in $x$, $x_1$, is

```
x[1]
```

```
## [1] 10
```

The second element in $x$:

```
x[2]
```

```
## [1] 15
```

The first three elements in $x$ are 10,15,15.

```
x[1:3]
```

```
## [1] 10 15 15
```

We can define a new vector, $y$ which equal to $x + 5$ by adding to $x$ the constant 5. Note that this constant is added for all the element in $x$

```
y <- x + 5
y
```

```
## [1]  15  20  20  30 105
```

### Practical session

Define two vectors: $x = (1, 2, 3, 4, 5)$ and $y = (10, 15, 25, 5, 12)$.

- Define a new vector $w$, $w = x + y$.
- Produce a scatterplot of $x$ versus $y$ with the function `plot()`.

## Factors

A character vector or a factor is define by

```
z <- c("A", "B", "A", "A", "A", "A", "B")
z
```

```
## [1] "A" "B" "A" "A" "A" "A" "B"
```

In the followig example, z is a numerical vector.

```r
z <- c(1, 2, 1, 1, 1, 1, 2)
```

We can use the function as.factor() to change z from numerical vector to a factor.

```r
x <- as.factor(z)
```

### index vectors

Consider a dataset with 5 subjects, for each we have information about the gender and the height. Let us define two vectors. The first is a factor represents the gender: $sex = (M, M, M, F, F)$.

```r
sex <- c("M", "M", "M", "F", "F")
sex
```

```
## [1] "M" "M" "M" "F" "F"
```

The second is a numeric vector represents the heights: $height = (190, 180, 192, 165, 170)$.

```r
height <- c(190, 180, 192, 165, 170)
height
```

```
## [1] 190 180 192 165 170
```

We can combine the two vectors into one data structure. This will be discussed further in Section 2.7.

```r
cbind(sex,height)
```

```
##      sex height
## [1,] "M" "190"
## [2,] "M" "180"
## [3,] "M" "192"
## [4,] "F" "165"
## [5,] "F" "170"
```

The first subject is a male,

```r
sex[1]
```

```
## [1] "M"
```

and his height is 190.

```r
height[1]
```

```
## [1] 190
```

The last subject is a female

```r
sex[5]
```

```
## [1] "F"
```

and her height is equal to 170.

```r
height[5]
```

```
## [1] 170
```

We can print the heights of the males with

```r
height[sex == "M"]
```

```
## [1] 190 180 192
```

Equivalently, the heights of the females

```r
height[sex == "F"]
```

```
## [1] 165 170
```

Note that the vector sex == “M” is a factor with lvels equal to TRUE and FALSE

```r
sex == "M"
```

```
## [1]  TRUE  TRUE  TRUE FALSE FALSE
```

## Data frame

### YouTube tutorial: vectors and matrices in R

For a YouTube tutorial by LearnR about data frames in R see YTobjects3.

### Example: data frame in R

A data frame in R is a data structure which combines few vectors together. We can create a data frame by using the function data.frame() . For example, we define a new object in R, z which is a data frame which contains the vectors sex and height .

```
z<-data.frame(sex, height)
```

To see the data frame we type z

```
z
```

```
##   sex height
## 1   M    190
## 2   M    180
## 3   M    192
## 4   F    165
## 5   F    170
```

The data frame **z** contains two vectors: sex and height . To print the vector sex we use

```
z$sex
```

```
## [1] M M M F F
## Levels: F M
```

The $ means: the vector sex in the data frame z .

The vector height.

```
z$height
```

```
## [1] 190 180 192 165 170
```

## Matrix

A matrix in R is an object which combines together few vectors.

```
z <- c(1, 2, 3, 4, 5, 6, 7, 8, 9)
z
```

```
## [1] 1 2 3 4 5 6 7 8 9
```

Use the function matrix() to create a $3 \times 3$ matrix from the vector z .

```
w <- matrix(z, 3, 3)
```

The matrix w is a $3 \times 3$ matrix

```
w
```

```
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    5    8
## [3,]    3    6    9
```

The entery $W_{11}$.

```r
w[1, 1]
```

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

$W_{1,3}$.

```r
w[1, 3]
```

```
## [1] 7
```

$W_{2,3}$.

```r
w[2, 3]
```

```
## [1] 8
```

The first column.

```r
w[, 1]
```

```
## [1] 1 2 3
```

The second row.

```r
w[2,  ]
```

```
## [1] 2 5 8
```

# Basic plots in R

## Introduction

In previous sections we created R objects, we used R statements and function in order to define vectors and data frames. In this section we focus on two real datasets: the airquality } and the law school datasets. Both datasets are entered to R as data frame. The aim of this section is to illustrate the use of R in working with the data.

#### YouTube tutorial: basic plots in R

#### Basic plots in R

For a YouTube tutorial by Jonathan Tuke about basic plots in R see YTplots1.

#### Basic plots in R (line, scatter, histogram, box, matrix plots)

For a YouTube tutorial by Research HUB about basic plots in R see YTplots2.

## Graphical functions: the airquality data

We focus on two graphical functions plot() and hist() . The airquality dataset is a data frame in R which contains information about 4 variables: ozone level, radiation, temperature and wind speed. We use the function print(airquality) to see the data. The first column is the ozone level, the second column is the radiation, the third is the temperature and the last column is the wind speed. A partial printout in given below.

```
head(airquality)
```

```
##   Ozone Solar.R Wind Temp Month Day
## 1    41     190  7.4   67     5   1
## 2    36     118  8.0   72     5   2
## 3    12     149 12.6   74     5   3
## 4    18     313 11.5   62     5   4
## 5    NA      NA 14.3   56     5   5
## 6    28      NA 14.9   66     5   6
```

In order to get more information about the data use the function help() . The wind speed is shown below.

```
airquality$Wind
```

```
##   [1]  7.4  8.0 12.6 11.5 14.3 14.9  8.6 13.8 20.1  8.6  6.9  9.7  9.2 10.9 13.2
##  [16] 11.5 12.0 18.4 11.5  9.7  9.7 16.6  9.7 12.0 16.6 14.9  8.0 12.0 14.9  5.7
##  [31]  7.4  8.6  9.7 16.1  9.2  8.6 14.3  9.7  6.9 13.8 11.5 10.9  9.2  8.0 13.8
##  [46] 11.5 14.9 20.7  9.2 11.5 10.3  6.3  1.7  4.6  6.3  8.0  8.0 10.3 11.5 14.9
##  [61]  8.0  4.1  9.2  9.2 10.9  4.6 10.9  5.1  6.3  5.7  7.4  8.6 14.3 14.9 14.9
##  [76] 14.3  6.9 10.3  6.3  5.1 11.5  6.9  9.7 11.5  8.6  8.0  8.6 12.0  7.4  7.4
##  [91]  7.4  9.2  6.9 13.8  7.4  6.9  7.4  4.6  4.0 10.3  8.0  8.6 11.5 11.5 11.5
## [106]  9.7 11.5 10.3  6.3  7.4 10.9 10.3 15.5 14.3 12.6  9.7  3.4  8.0  5.7  9.7
## [121]  2.3  6.3  6.3  6.9  5.1  2.8  4.6  7.4 15.5 10.9 10.3 10.9  9.7 14.9 15.5
## [136]  6.3 10.9 11.5  6.9 13.8 10.3 10.3  8.0 12.6  9.2 10.3 10.3 16.6  6.9 13.2
## [151] 14.3  8.0 11.5
```

A scatterplot of ozone versus the wind speed can be produced with the function plot() shown in Figure~@ref(fig:fig1)

```r
plot(airquality$Wind,airquality$Ozone)
```

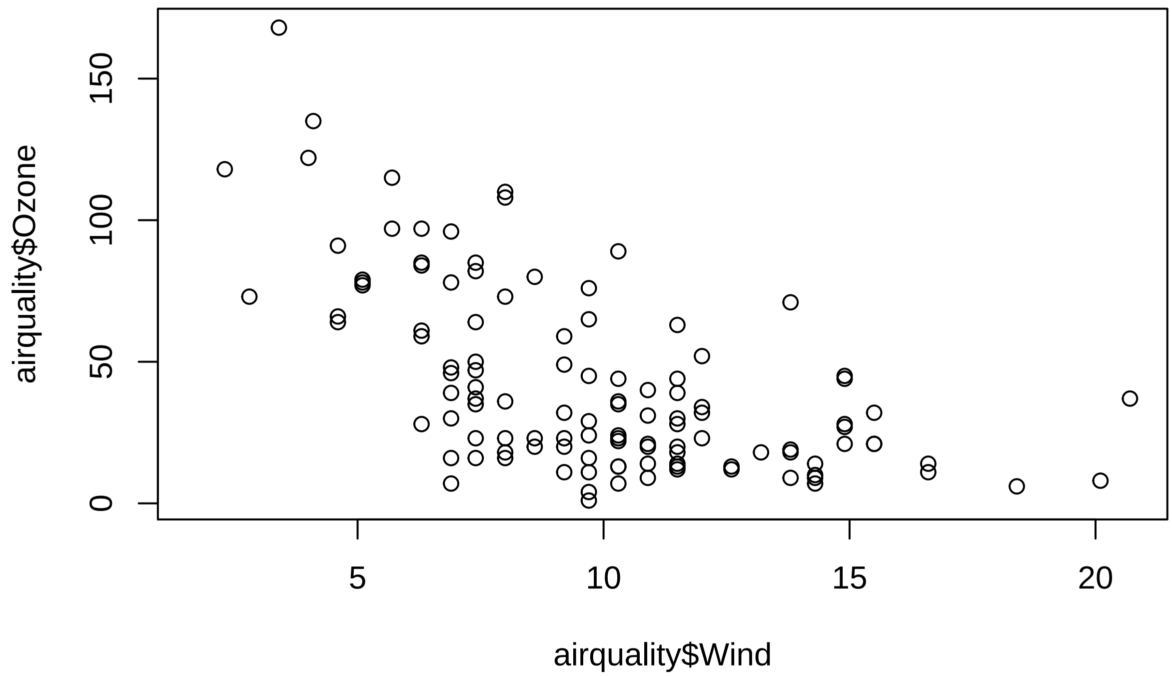



Figure 1: Ozone level versus wind speed.

Figure~@ref(fig:fig1b) presents an Histogram of the ozone.

```r
hist(airquality$Ozone)
```

We can use the function par() to plot the two figures in one page. A general call of the function par() has the form par(mfrow=c(number of rows, number of columns)) .For example par(mfrow=c(1,2)) will produce a graphical page with two figures in one row as shown in Figure~@ref(fig:fig2)

```r
par(mfrow=c(1,2)) #put 2 figures in the same page
plot(airquality$Wind,airquality$Ozone)
hist(airquality$Ozone)
```

## Graphical functions: the law school data

The low school data gives information about LSAT and GPA scores in 82 low schools in USA. It a part of the R package bootstrap so you need to install the package. Figure~@ref(fig:fig3) shows a scatterplot for the scores and a random sample of 15 schools from the population of 82 schools.

```r
par(mfrow=c(1,2))
plot(law82$LSAT,law82$GPA,xlim=c(450,700),ylim=c(2.5,3.5))
points(law$LSAT,law$GPA,pch="o",col=2)
plot(law$LSAT,law$GPA,pch="o",xlim=c(450,700),ylim=c(2.5,3.5))
```

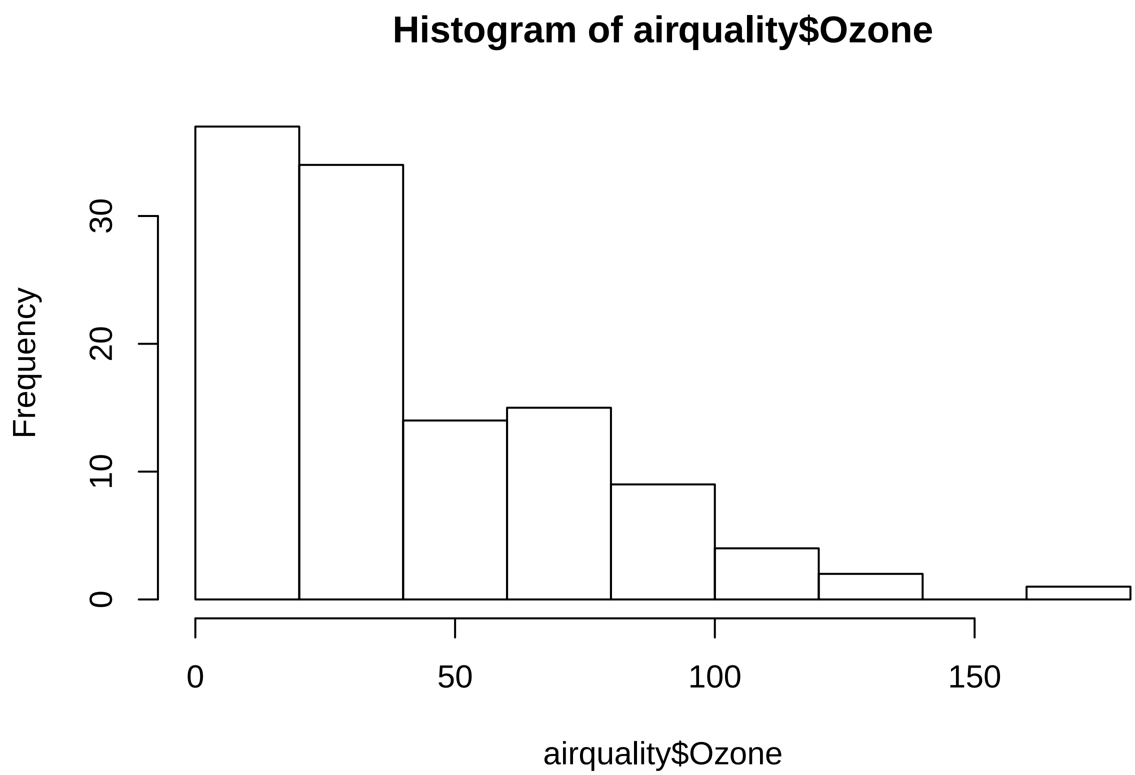



Figure 2: Histogram for the Ozone level.

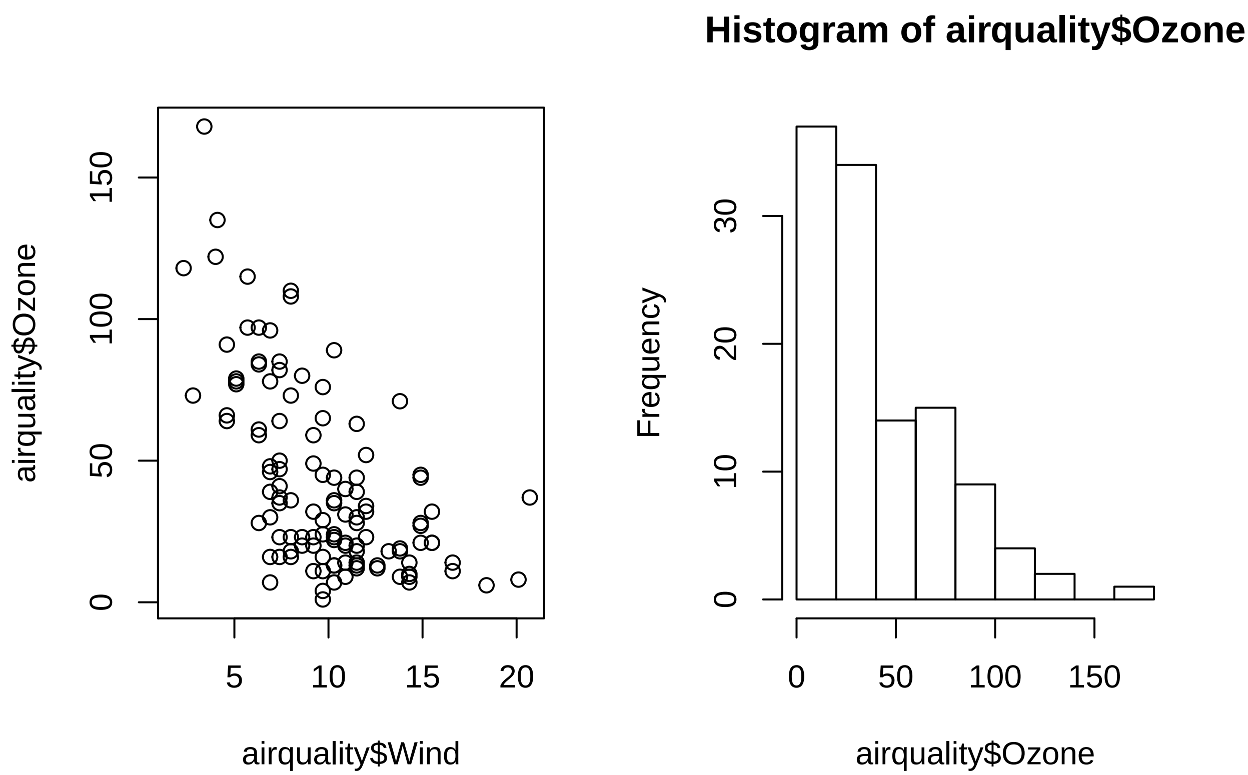



Figure 3: Two figures in the same page.

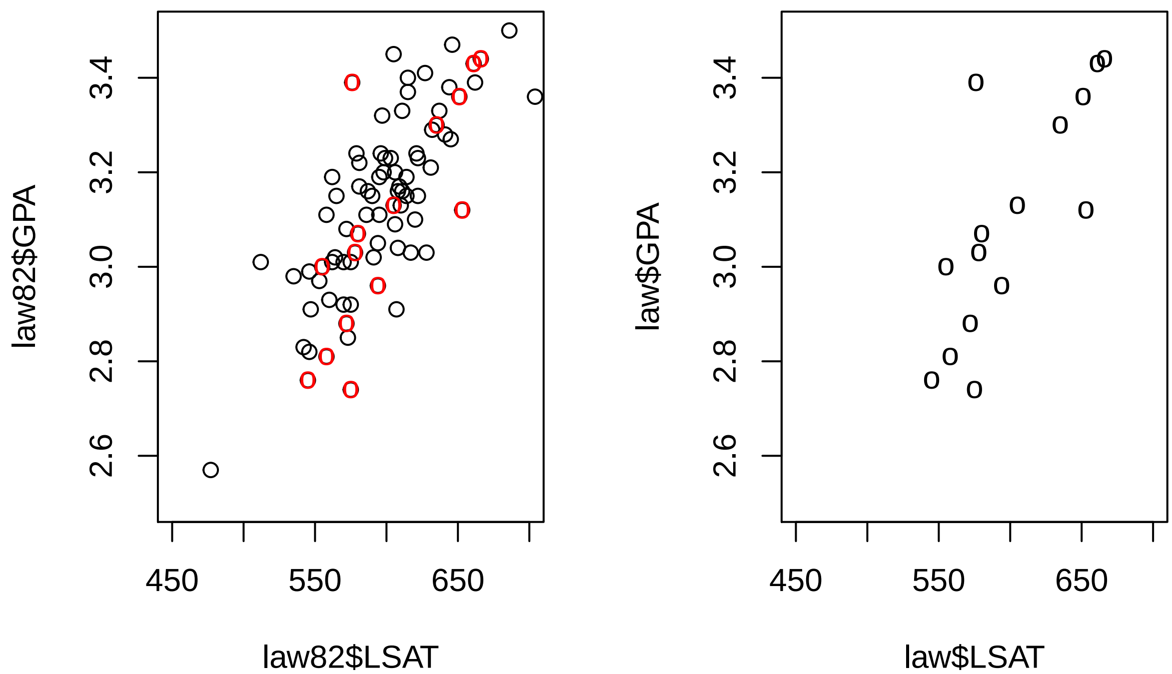



Figure 4: The low schools data

The population correlation is equal to

```r
cor(law82$LSAT,law82$GPA)
```

```
## [1] 0.7599979
```

and the sample correlations.

```r
cor(law$LSAT,law$GPA)
```

```
## [1] 0.7763745
```

# Summary statistics

## introduction

In this section we focus on R functiona that can be used in order to calculate summary statistics of a sample. The summary statistics that will be discussed

- Sample mean.
- Sample median.
- Variance.
- Minimum and Maximum.

### Summary Statistics In R Software (Pt. 1 of 3)

For a YouTube tutorial by economicurtis about Summary Statistics in R see YTstat1.

### Summary Statistics in R

For a YouTube tutorial by LawrenceStats about Summary Statistics in R see YTstat2.

## Calculation of summary statistics in R

We will use the Chicken Weights (the chickwts object ) data frame. The first few lines are shown below

```
head(chickwts)
```

```
##   weight      feed
## 1    179 horsebean
## 2    160 horsebean
## 3    136 horsebean
## 4    227 horsebean
## 5    217 horsebean
## 6    168 horsebean
```

We use the function mean() to calculate the mean of a vector.The mean of the Chicken Weights:

```
mean(chickwts$weight)
```

```
## [1] 261.3099
```

The median of the Chicken Weights

```
median(chickwts$weight)
```

```
## [1] 258
```

The variance the Chicken Weights

```
var(chickwts$weight)
```

```
## [1] 6095.503
```

The minimum of the Chicken Weights

```
min(chickwts$weight)
```

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

The maximum of the Chicken Weights

```r
max(chickwts$weight)
```

```
## [1] 423
```

A box plot of the Chicken Weights by diet group is shown in Figure~@ref(fig:fig5)

```r
boxplot(split(chickwts$weight,chickwts$feed))
```

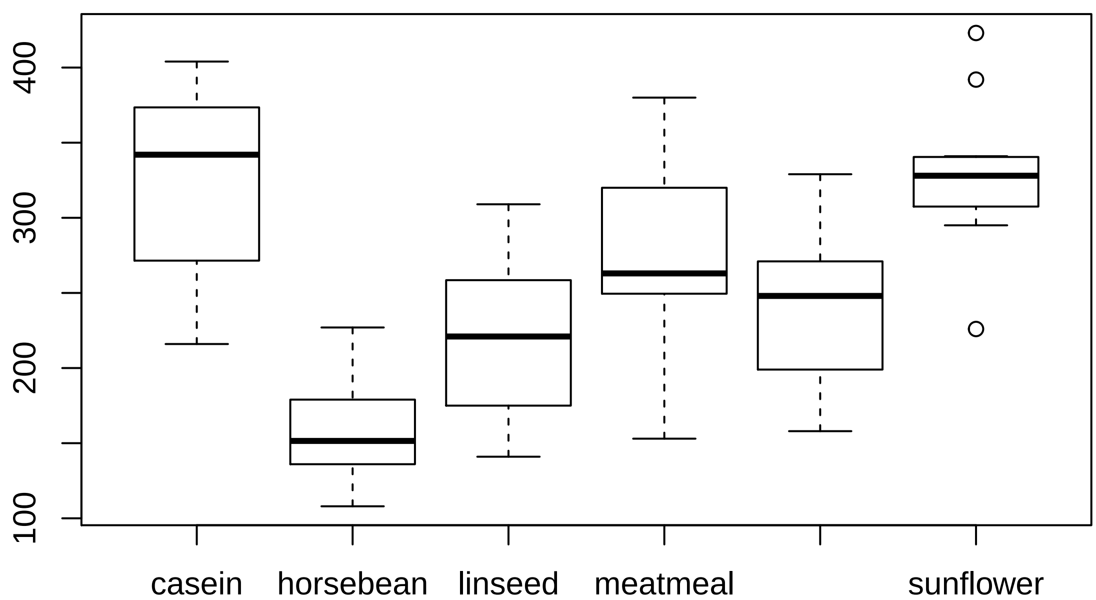



Figure 5: Chicken weights by diet group.

The mean of the Chicken Weights per group can be calculated using the function tapply() . A genela call of the function has the form
tapply(numerical vector, factor, statistics) .

For the Chicken Weights dataset we use

```r
tapply(chickwts$weight,chickwts$feed,mean)
```

```
##    casein horsebean   linseed  meatmeal   soybean sunflower
##  323.5833  160.2000  218.7500  276.9091  246.4286  328.9167
```

**Practical session**

The data frame `mtcars` contains information about fuel consumption of cars.

- What are the variables names in the data frame

- calculate the mean, median minimum and maximun of the variable mile per gallon (the R object mpg )

# A foor loop

## YouTube tutorial: the for loop in R

For a short online YouTube introduction by Richard Webster about a for loop in R see YTforlorloop.

## Example 1: distribution of the sample means

A for loop in R is a loop in which we repeatedly ask R to do the same action in each step of the loop. For example, suppose that we would like to draw a sample of 10 observations from $N(0,1)$. In R this can be done using the code

```
x<-rnorm(10,0,1)
```

The sample is

```
x
```

```
##  [1] -0.5871780  0.1527135 -0.4896620  1.2366098  0.1488369  0.7685010
##  [7]  0.6483991  1.6354646  0.6077612  0.2158710
```

and the sample mean

```
mx<-mean(x)
mx
```

```
## [1] 0.4337317
```

Suppose that we would like to draw a sample of 10 observations from $N(0,1)$ 1000 times. To do this we can use a "for loop" in the following way. First, we define a vector that will be used to store the sample means

```
mx<-c(1:1000)
```

The for loop we use

```
for(i in 1:1000)
{
x<-rnorm(10,0,1)
mx[i]<-mean(x)
}
```

A histogram of the sample means is shown in Figure~@ref(fig:fig6)

```
hist(mx,nclass=20)
```

## Example 2: print on the screen

Print a text on the screen $K$ times:

```
for(i in 1:10)
{
print("Text")
}
```

```
## [1] "Text"
## [1] "Text"
## [1] "Text"
## [1] "Text"
```

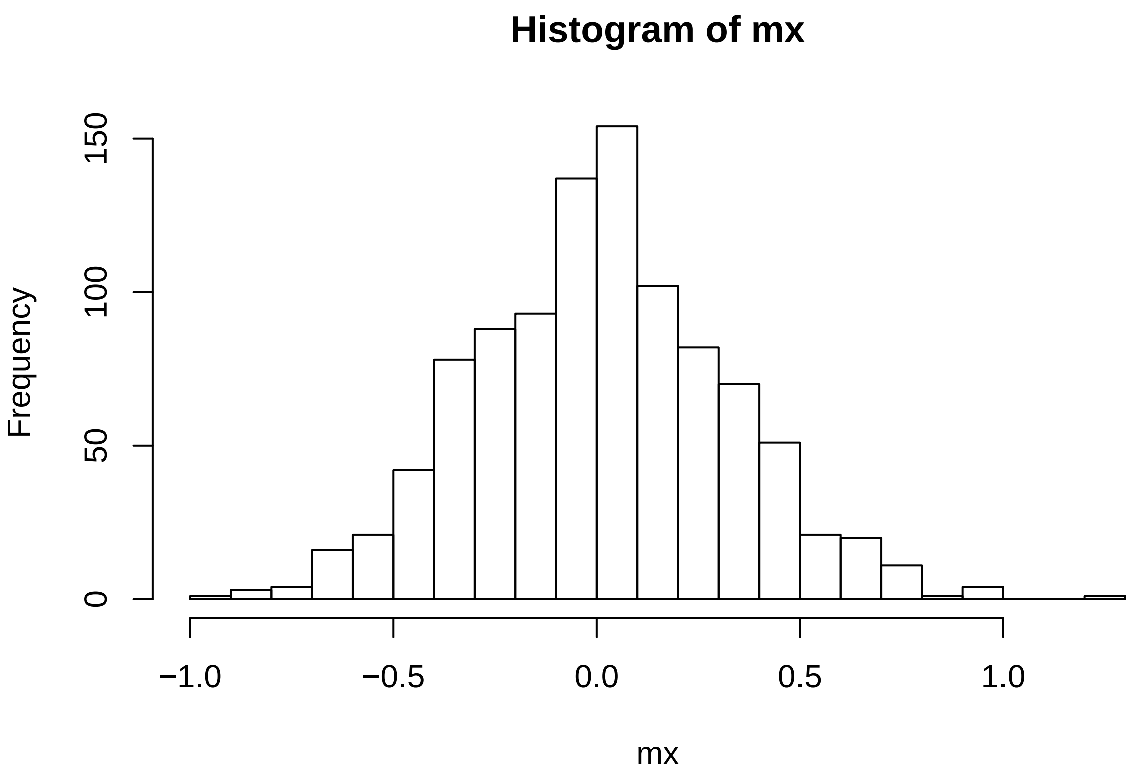



Figure 6: Histogram for 1000 sample means

```
## [1] "Text"
## [1] "Text"
## [1] "Text"
## [1] "Text"
## [1] "Text"
## [1] "Text"
```

We can print both test and iteration number using the R function paste:

```
for(i in 1:10)
{
print(paste("Step:",i))
}
```

```
## [1] "Step: 1"
## [1] "Step: 2"
## [1] "Step: 3"
## [1] "Step: 4"
## [1] "Step: 5"
## [1] "Step: 6"
## [1] "Step: 7"
## [1] "Step: 8"
## [1] "Step: 9"
## [1] "Step: 10"
```

# User functions in R

## YouTube tutorials: user function in R

### Making Functions in R

For a YouTube tutorial about user function in R by Richard WebsterR see YTfunctions1.

### User-defined functions in R

For a YouTube tutorial about user function in R by Jonatan Lindh see YTfunctions2.

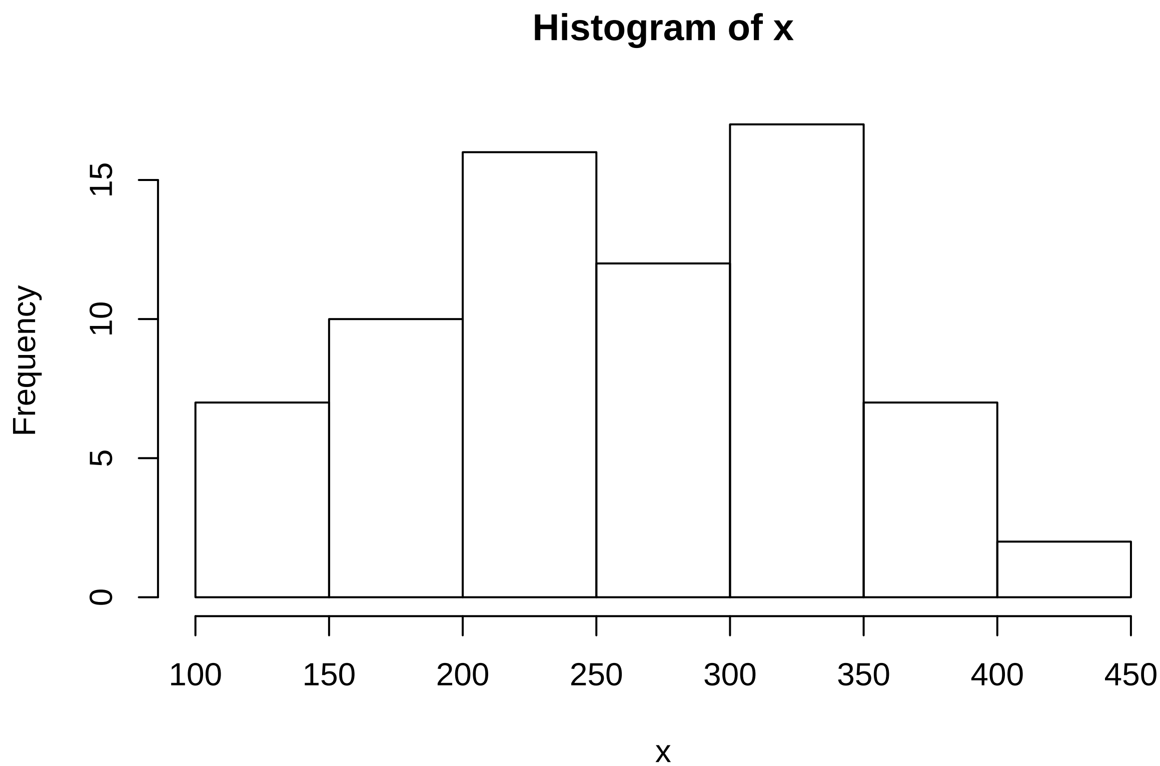
## Example 1: descriptive statistics for a sample

A function that calculates the sample mean and median and plot a histogram of the sample can be defined by

```
fun20<-function(x)
{
mean.x<-mean(x)
med.x<-median(x)
hist(x)
return(c(mean.x,med.x))
}
```

The R object fun20 is the function. For the Chicken Weights we have

```
results.1<-fun20(chickwts$weight)
```



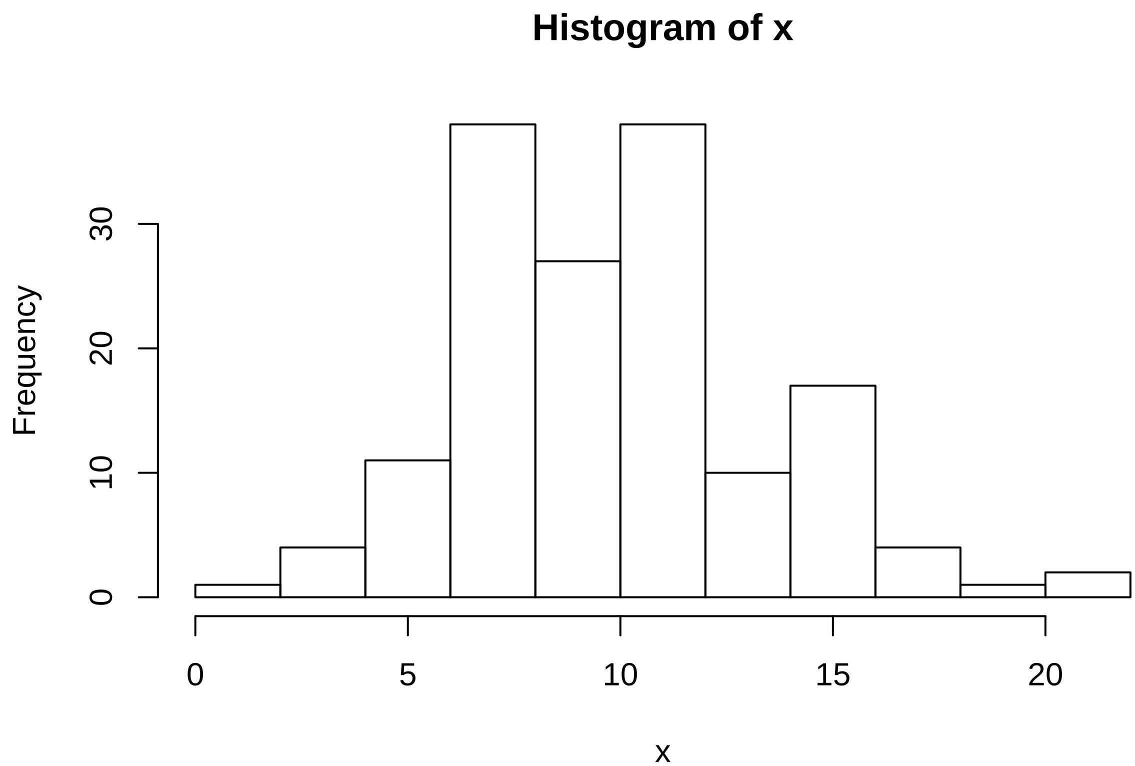

The object results.1 contains the results. To see the mean and median we print the object

```
results.1
```

```
## [1] 261.3099 258.0000
```

In the same way, the same output (the mean of the sample) can be produced for the wind speed of the aitquality data

```
fun20(airquality$Wind)
```




```
## [1] 9.957516 9.700000
```

### Example 2: simple linear regression model

For this example we use the cars dataset that gives the speed of cars and the distances taken to stop. We would like to write a function that

- Calculate the correlation between two variables.
- Fit a regression model of the form $y_i = \alpha + \beta x_i + \varepsilon_i$.
- Plot the variable x and y with the regression line.

For the cars data set the corresponding R code can be used to produce the results. First we define the variables $x$ and $y$:

```
x<-cars$speed
y<-cars$dist
```

The correlation

```
cor(x,y)
```

```
## [1] 0.8068949
```

The regression model and output

```
fit.lm<-lm(y~x)
fit.lm
```

```
##
```

```
## Call:
## lm(formula = y ~ x)
##
## Coefficients:
## (Intercept)            x
##     -17.579        3.932
```

and the scaterplot with the fitted model.

```
plot(x,y)
lines(x,fit.lm$fit)
```

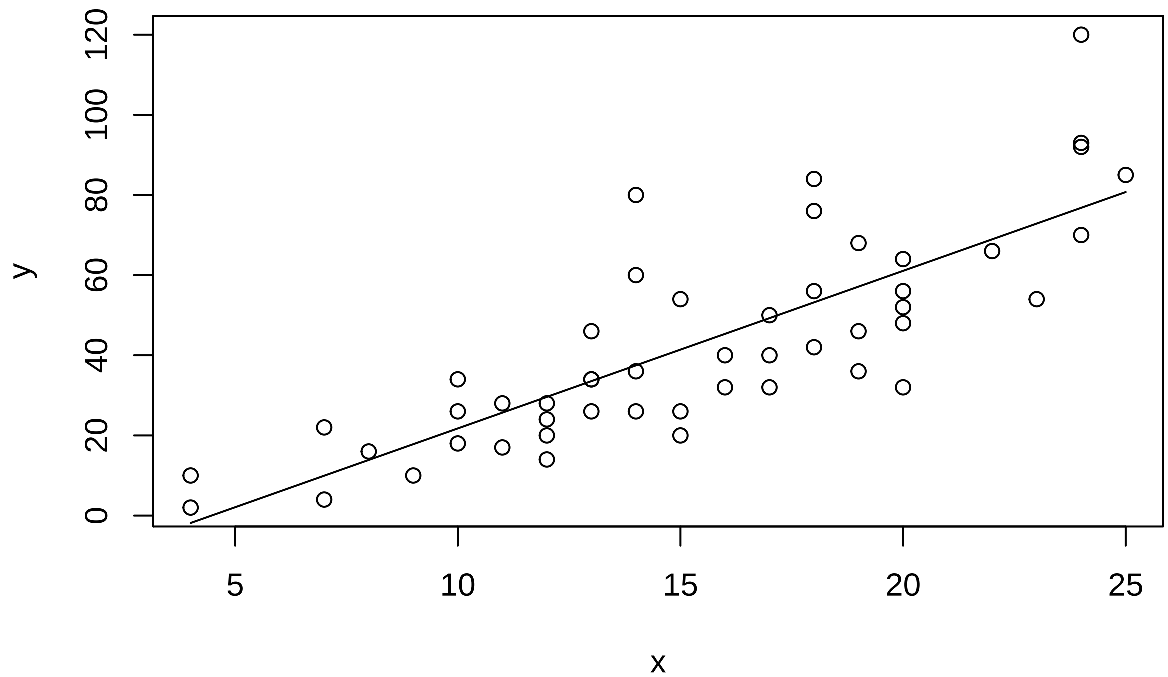



Figure 7: The cars dataset: data and fitted model

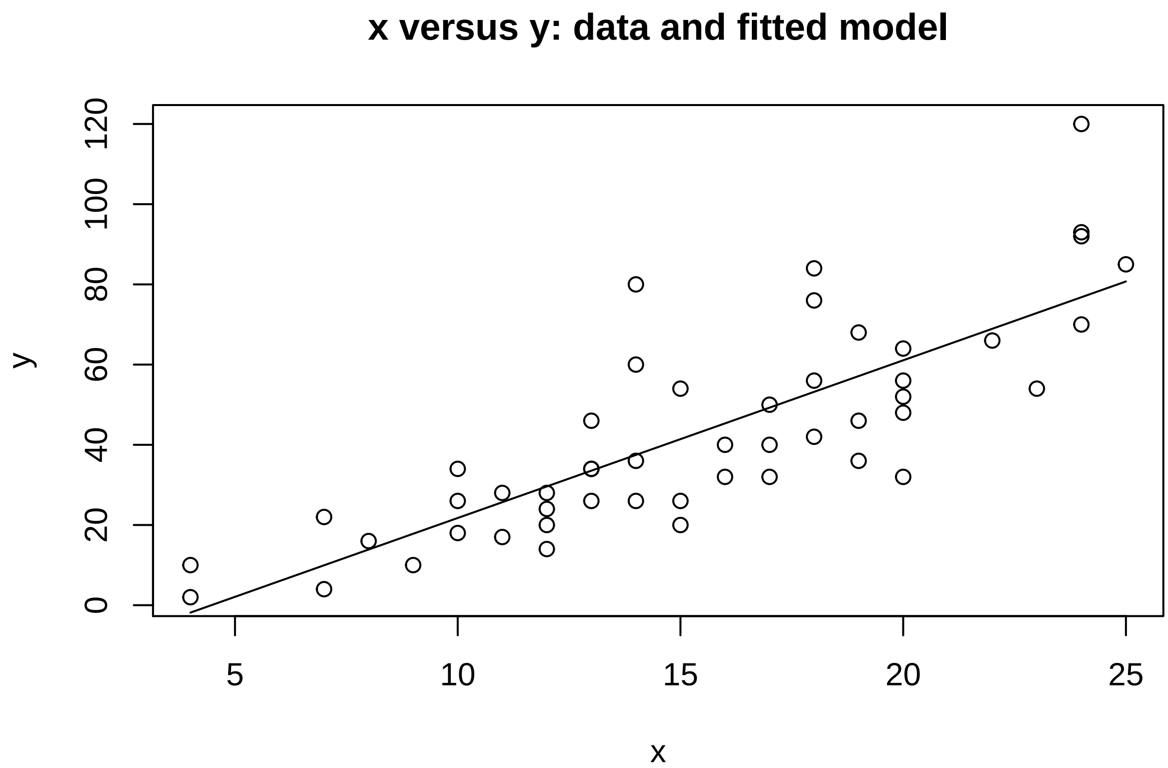
The function fun21 is a user function that was written to produce the output above.

```
fun21<-function(x,y)
{
print(cor(x,y))
fit.lm<-lm(y~x)
plot(x,y)
lines(x,fit.lm$fit)
title("x versus y: data and fitted model")
return(fit.lm)
}
```

We define an R object model.1 that contains the output

```
model.1<-fun21(x,y)
```

```
## [1] 0.8068949
```




Note that the correlation is printed on the screen as requested. To see the parameter estimate we use

```
summary(model.1)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.069  -9.525  -2.272   9.215  43.201
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -17.5791     6.7584  -2.601   0.0123 *
## x             3.9324     0.4155   9.464 1.49e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.38 on 48 degrees of freedom
## Multiple R-squared:  0.6511, Adjusted R-squared:  0.6438
## F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```