

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/351246118>

INTRODUCTION TO STATISTICS USING R (for archaeologists)

Book · January 2021

CITATIONS

0

READS

381

1 author:



[Efthymia Nikita](#)

The Cyprus Institute

84 PUBLICATIONS 893 CITATIONS

[SEE PROFILE](#)

Some of the authors of this publication are also working on these related projects:



SACHROFICS: SACRIFICIAL CLASSICAL HELLENISTIC AND ROMAN FOOD IN CYPRUS [View project](#)



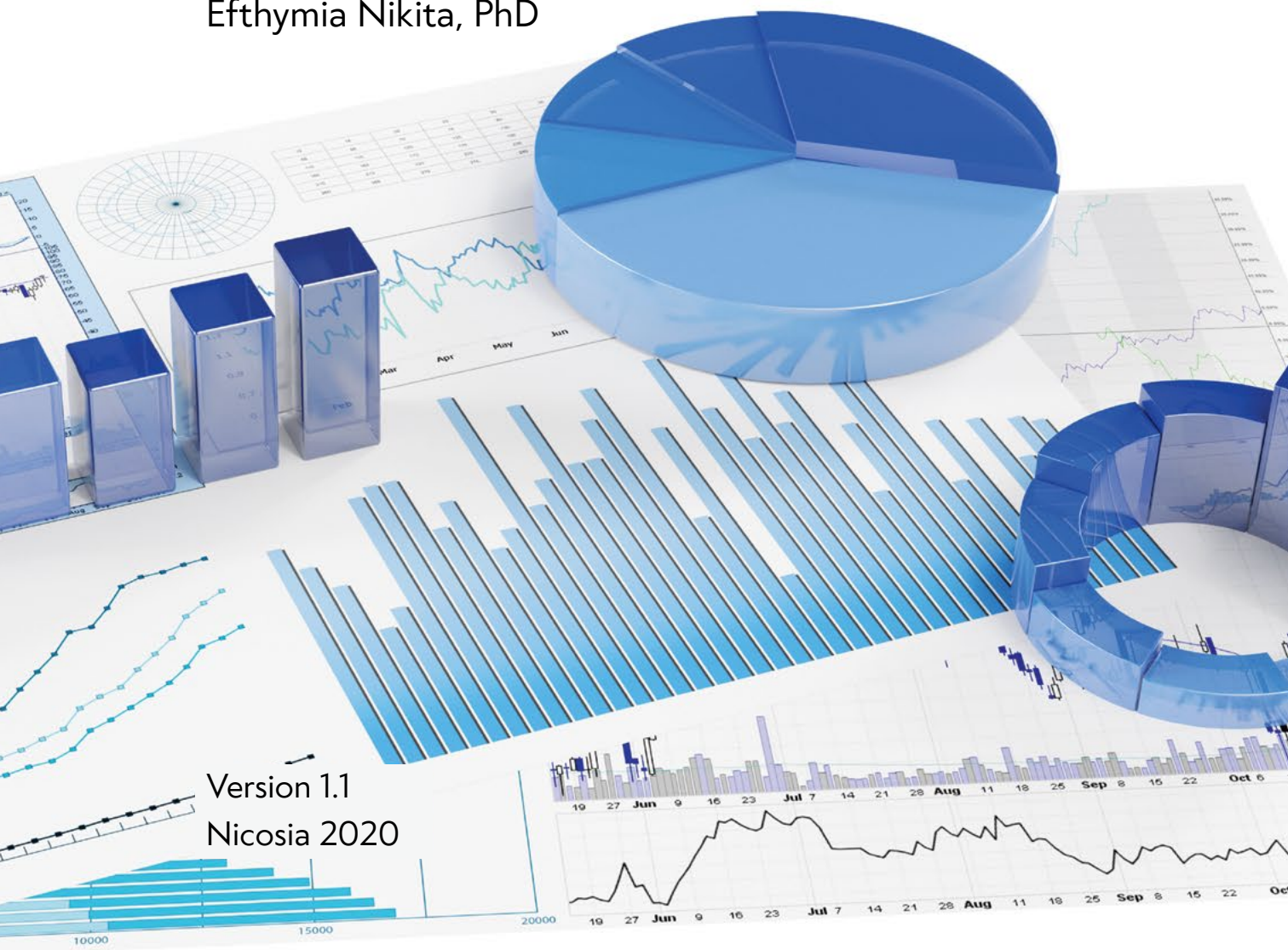
Ancient Kition Project [View project](#)

INTRODUCTION TO STATISTICS USING R

(for archaeologists)

The Cyprus Institute
Science and Technology in Archaeology and
Culture Research Center (STARC)

Efthymia Nikita, PhD



Version 1.1
Nicosia 2020

INTRODUCTION TO STATISTICS USING R

(for archaeologists)

The Cyprus Institute
Science and Technology in Archaeology and
Culture Research Center (STARC)

Efthymia Nikita, PhD

Version 1.1
Nicosia 2020
(layout edited 2021)

ISBN 978-9963-2858-6-0

CONTENTS

PREFACE / 1

1. THE SCIENTIFIC PROCESS / 3

- 1.1. Sample and population 3
- 1.2. Variables 5

2. THE R INTERFACE / 6

- 2.1. Installing R 6
- 2.2. Using R 7
- 2.3. Types of objects 9
- 2.4. Data input 11
- 2.5. Selecting parts of a dataset 13
- 2.6. Packages 14
- 2.7. Getting help 14

3. DESCRIPTIVE STATISTICS / 15

- 3.1. Summary statistics 16
- 3.2. Summary statistics in R 19
- 3.3. Contingency tables 21
- 3.4. Contingency tables in R 22
- 3.5. Graphs 23
- 3.6. Graphs in R 27

4. INFERENCE STATISTICS / 42

- 4.1. Probability distributions 42
- 4.2. Point estimation 53
- 4.3. Confidence intervals 56
- 4.4. Hypothesis testing 61

5. HYPOTHESIS TESTING: ONE SAMPLE / 67

- 5.1. Normality test 67
- 5.2. Outliers 68

6. HYPOTHESIS TESTING: TWO SAMPLES / 71

- 6.1. Homogeneity of variance 71
- 6.2. Independent-samples t-test 73
- 6.3. Paired samples t-test 75

7.

HYPOTHESIS TESTING: MANY SAMPLES / 77

7.1.	One-way analysis of variance (ANOVA)	77
7.2.	Repeated measures ANOVA	83
7.3.	Analysis of covariance (ANCOVA)	88

8.

NON-PARAMETRIC TESTS / 95

8.1.	Wilcoxon rank-sum test	95
8.2.	Wilcoxon signed-rank test	97
8.3.	Kruskal-Wallis test	98
8.4.	Friedman test	101

9.

CORRELATION / 103

9.1.	Bivariate correlation	103
9.2.	Partial correlation	107
9.3.	Correlation of matrices	109

10.

REGRESSION / 114

10.1.	Basic definitions	115
10.2.	Linear regression	118
10.3.	Logistic regression	134
10.4.	Generalized Linear Models	141
10.5.	Generalized Estimating Equations	155

11.

CATEGORICAL DATA / 158

11.1.	Chi-squared test	158
11.2.	Fisher's exact test	159
11.3.	Correspondence analysis	160

12.

MULTIVARIATE ANALYSES / 162

12.1.	Multivariate normality test and detection of outliers	162
12.2.	Principal Component Analysis	166
12.3.	Metric and non-metric multidimensional scaling	173
12.4.	Cluster analysis	178
12.5.	Linear Discriminant Analysis	184
12.6.	Multivariate Analysis of Variance	187

PREFACE

Statistical analysis allows the processing of data in order to explore patterns and draw conclusions as to whether the observed similarities, differences, correlations etc. are ‘true’ or the result of random factors. Moreover, statistical analysis aims to present data in a meaningful and broadly acceptable manner; results can be re-evaluated by other scholars and compared to those from other studies, provided that the raw data is available. Another key reason why statistics is important, is that in most scientific disciplines the material that each researcher can study is merely a fraction of all possible material. For example, in archaeology, we study samples of the material culture and organic remains of past civilizations. However, what we are trying to explore is the life of past people, which would require the preservation, excavation and study of every single organic remain of the past as well as every single artefact produced by every single individual that ever lived. In order to draw generalizable conclusions based on the samples that we can actually study, it is essential to apply statistical analysis.

For statistics to constitute a helpful component of any research, there are three crucial prerequisites:

1. The data we have collected must be relevant to the research question we wish to address and representative of the broader assemblage they come from.
2. The correct type of statistical analysis should be employed according to the nature of our data and our research question.
3. The results should be interpreted correctly taking into account both the mathematical principles underlying the analyses and the context from which the data has been drawn.

The emphasis of the current guide is on ‘Applied Statistics’. This means that even though the mathematical principles of different statistical methods are briefly described, the key aim is to make sure the reader understands what kind of data is being analyzed, what type of test is the most appropriate, how to actually implement this test correctly using the R language, and how to properly interpret the results. The examples given throughout this guide have been drawn from the archaeological sciences. However, these examples may be easily understood by non-archaeologists and it should be straightforward for specialists from other disciplines to see how the analyses presented could be used with their type of data and research questions.

These notes were written especially for users running the Windows version of R, but most of the material applies to the Mac and Linux versions as well. All the sample data files used in this guide are provided in the folder *R_Files*, which may be freely downloaded from the Zenodo repository (DOI: 10.5281/zenodo.4641950). This document is an open resource and it is anticipated to be updated at regular intervals. I would greatly appreciate any feedback and recommendations for future improvement.*

Efthymia Nikita

* For feedback/recommendations, please contact me at e.nikita@cyi.ac.cy

1.

THE SCIENTIFIC PROCESS

The scientific process begins when we pose research questions in order to better understand a phenomenon that we study. For example, in an ancient cemetery, we observe that females are twice as many as males. To understand this phenomenon, we generate possible explanations, from which we can make predictions. For example, males may have died in the battlefield and were buried there instead of the settlement cemetery. Now, we need to collect data in order to test our hypothesis. First, we should identify what type of data we need to collect so that they are indeed relevant to our research question, and then we need to analyze those data. In our example, if the increased representation of females is indeed due to the fact that males were buried near the battlefield, we would expect very few, if any, young males in the cemetery we examine; instead, most interred males in our sample should be nonadults or old adults, thus too young or too old to have participated in warfare. To test if this is indeed the case, we need to collect osteoarchaeological data on the age-at-death of the male and female sample of our assemblage. The analysis of the data may support our hypothesis or suggest that we need to modify it. In our example, if we find that the proportion of young to old males is similar to that of females, we need a different explanation for the underrepresentation of males in the cemetery. Therefore, there is a strong interrelationship between data collection and analysis, and the generation of research hypotheses: hypotheses dictate data collection/analysis and data collection/analysis generates new hypotheses. The scientific process is summarized in Figure 1.1.

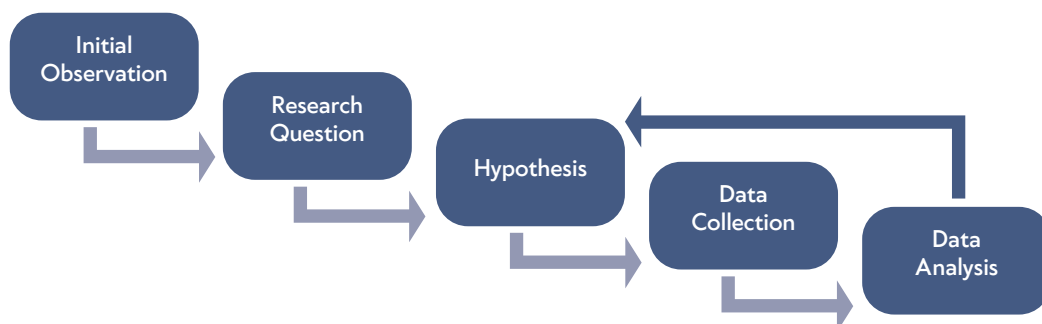


Figure 1.1.
The scientific
process
(adapted from
Figure 1.2. in
Field et al.
2012¹)

1.1. SAMPLE AND POPULATION

All archaeologists, irrespective of their specialization, examine a *sample* of the original archaeological record. This original record that includes all evidence of past life in each site is called *population* and we can never access it fully. What we define as population and sample will differ depending on our research question. For example, if you are an osteoarchaeologist and your research relates to the health and disease patterns of a specific ancient group, then your population would be all the humans who constituted part of that group throughout the group's existence. In this case the sample consists of the skeletons that we have found during the excavation of the cemetery used by this population. If the aim is to examine the patterns of osteoarthritis in this group, there is no point in including children in the research design as this is a pathological condition that we see only in adults. Therefore, the population and sample now are the same as previously but excluding all children.

¹ Field A, Miles J, Field Z. 2012. *Discovering Statistics using R*. Sage Publishing

When we study any sample, what we are actually interested in doing, is reaching conclusions concerning the population from which the sample was drawn. To do this, we need to make sure that the sample under study is representative of the population. The first step in this process is to adopt an appropriate excavation strategy; however, this is beyond the scope of the present guide. What we will examine here is how to sample effectively from the material that has already been excavated; thus, we will treat all excavated material as the population.

Let's assume that we got a small grant to examine the stone tools from a Roman site. Our grant can cover fieldwork expenses for three weeks; however, the stone tools excavated require several months of study. The way to approach this issue is to select a sample from the available excavated material. In order for our sample to be representative of all excavated material, we must select it randomly. *A random sample is defined as one in which each member of the population has the same probability of being selected as any other member.*

To make this clearer, assume that the excavated dataset of stone tools has the following composition:

Table 1.1.
Composition of
a hypothetical
stone tool
assemblage

Chipped stone tools	N	Ground stone tools	N
Scrapers	153	Querns	302
Sickle blades	222	Axes	177
Borers	51	Pounders	216
Daggers	178	Grinders	77
Burins	39	Polishers	21
Pins	80	Adzes	10
Drills	12	Chisels	45
Awls	90	Pestles	183
		Hammerstones	69
Total	825	Total	1100

If we believe that chipped stone tools are far more interesting than ground stone tools and select to study only them, then although our sample of stone tools will include 825 items, that is, almost 43% of all excavated material, it is in no way representative of the stone tools of the Roman site under study, as it implies that the occupants of the site were not making or using ground stone tools!

If we want our sample to be representative of the excavated material, we need to make a random selection of the items to be studied. In this way, all stone tool categories will be represented and their relative frequency will also be demonstrated in our sample. In other words, by randomly selecting a sub-set of tools, given that querns are far more abundant in the original assemblage, they will also be more frequent in our sample, while adzes will be rare. It would be wrong to create our sample by selecting the same number of tools from each category, that is, 10 scrapers, 10 sickle blades, 10 querns, etc. A non-random selection like this may have been appropriate if we were interested in issues such as the technological characteristics of the stone tools found in the site. However, if we wish to explore the relative use and importance of each tool type and the activities in which the Roman population was involved, then the relative frequency of each tool type is very important and this information is being lost when we select the same sample size for each tool type.

To conclude, the best way to ensure that our sample is representative of the population is to select the items that will constitute our sample in a random fashion, meaning that every element of the population has an equal probability of being incorporated in the sample. The sample size also plays

a role in how representative our data will be of the population. In general, bigger is better, especially when the population is heterogeneous and complex statistical tests will be used. However, a large sample will not compensate for a poor sampling procedure.

1.2. VARIABLES

During data collection, we need to identify what kind of data we will collect in order to test our hypotheses. In every dataset, we have cases and variables. *Cases* express the objects, individuals or other items that we examine, while *variables* are the characteristics of the cases. For example, in a pottery assemblage, each pot is a case, while the capacity, height, color and any other trait of the pottery is a variable. In a data matrix, each case occupies a single row, while each variable occupies a single column.

Variables are divided in different categories depending on the relationship between them and the levels of measurement:

Relationship between variables

In certain statistical tests, we treat some of the variables as a proposed cause (*independent or predictor variable*) and other variables as a proposed outcome (*dependent or outcome variable*). For example, there are many equations with which we can estimate an individual's stature based on the length of his/her long bones. In this case, the length of each long bone examined is the independent variable (or predictor), whereas the stature of the individual is the dependent variable (or outcome) because its value depends upon the length of the long bones (the independent variable).

Levels of measurement

Depending on how they are measured, variables are broadly divided into categorical and numerical. *Categorical variables*, as their name implies, consist of categories and these categories are mutually exclusive. For example, variable 'sex' consists of the categories 'males' and 'females,' and an individual can be either male or female. In their simplest form, categorical variables have only two categories (e.g. 'males,' 'females') and they are called *binary variables*. Another example of a binary variable is 'disease expression', which may consist of the categories 'present' and 'absent.' When there are more than two categories, the variable is called *nominal*. For example, when we study an archaeological feature, the variable that expresses the type of this feature may be a nominal variable with three categories: 'tomb', 'wall', and 'pit.' Note that in binary and nominal variables there is no intrinsic ordering of the categories. When the categories of a categorical variable express an order, the variable is called *ordinal variable*. For example, when studying 'osteoarthritis', instead of using the categories 'present' and 'absent', we may want to be more detailed and use the categories 'absent', 'mild', 'moderate', 'extreme.' In this case each category expresses a greater 'degree' than the previous category; an individual with 'extreme' osteoarthritis suffered more than an individual with 'mild' osteoarthritis.

Numerical variables express properties that can be measured. Numerical variables are *continuous* if they may take any of an infinite number of values within a given range, or *discrete* if they may take only a specific set of numeric values (e.g. only integers). In addition, numerical variables may be interval or ratio. In *interval variables* equal intervals on the scale represent equal differences in the property being measured; however, there is no meaningful fixed 0-point. A typical example is chronology: we can say that between 600 BC, 400 BC and 200 BC the difference is 200 years so there is an equal distance. However, we cannot say that 400 BC is twice as long ago as 200 BC. *Ratio variables* also require that equal intervals on the scale represent equal differences in the property being measured but, in addition, the ratios of values along the scale should be meaningful. When we measure the thickness of sherds, not only is it true that the difference between 2 cm and 2.3 cm is the same as the difference between 3.5 cm and 3.8 cm, but also it is true that a sherd measuring 2 cm is twice as thick as a sherd measuring 1 cm.

2.

THE R INTERFACE

R is a language and software environment for statistical computing and graphics. It was created by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand². Very importantly, the R code is accessible to everyone. This open source philosophy allows anyone, anywhere to contribute to the software. Consequently, the capabilities of R dynamically expand as people from all over the world add to it.

2.1. INSTALLING R

R is freely downloadable at: <http://www.r-project.org/>

When you visit this website, click on 'download R' (Figure 2.1) and a new window will appear so that you select a so-called CRAN mirror (Figure 2.2). The initials stand for Comprehensive R Archive

Figure 2.1.
Home page of
the R language

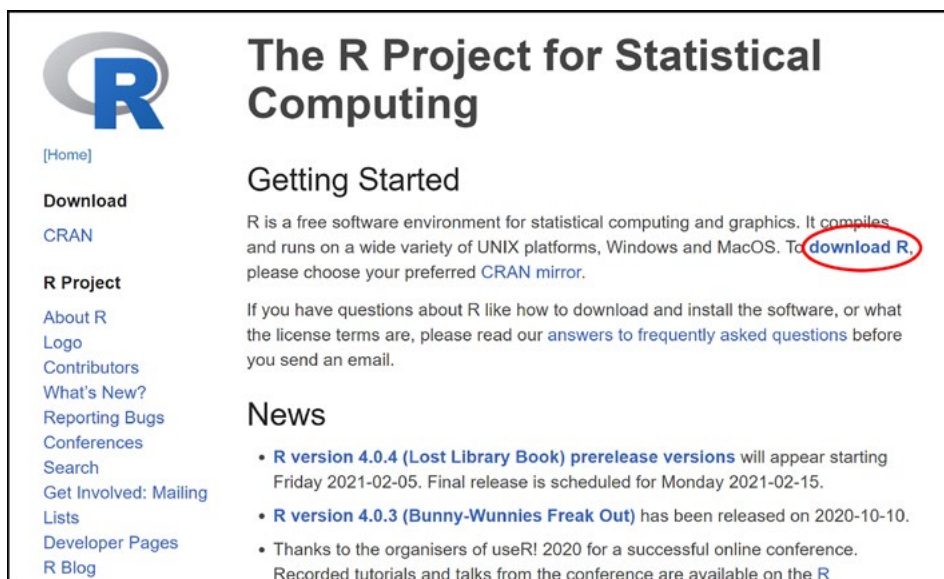
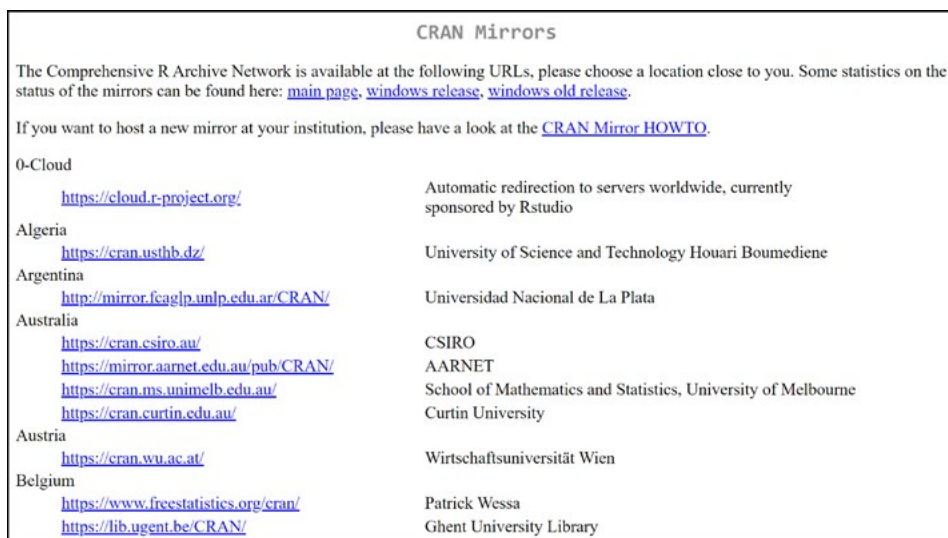


Figure 2.2.
Window to select
CRAN mirror for
downloading R



² Ihaka R, Gentleman R. 1996. R: a language for data analysis and graphics. *Journal of Computational and Graphical Statistics* 5: 299-314

Network and it represents a network of web servers around the world that store identical versions of code and documentation for R. It is advisable to scroll down and select the CRAN mirror geographically nearest to you to minimize network load.

Subsequently, you select the platform you use (Linux, MacOS or Windows). If you click on 'Download R for Windows', you will be taken to another page where you should click on 'base: install R for the first time' (Figure 2.3), which will redirect you to the webpage with the link to the setup file. Click on 'Download R 4.0.3³ for Windows' in order to download the R setup file. Once this file has been downloaded, double-click on it and follow the installation steps.

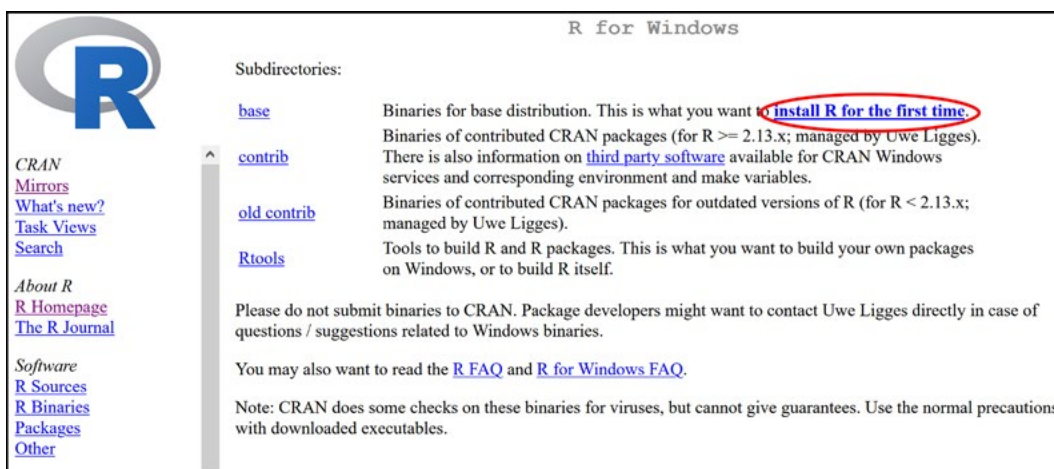


Figure 2.3.
Window to
install R for the
first time

2.2. USING R

When we run R, the window of Figure 2.4 appears. This is called the *R Console window* and initially provides some basic information about this language. This is the window where we can both type commands and see the results of executing these commands. If we draw any plots, these will appear in a separate window called the *graphics window*.

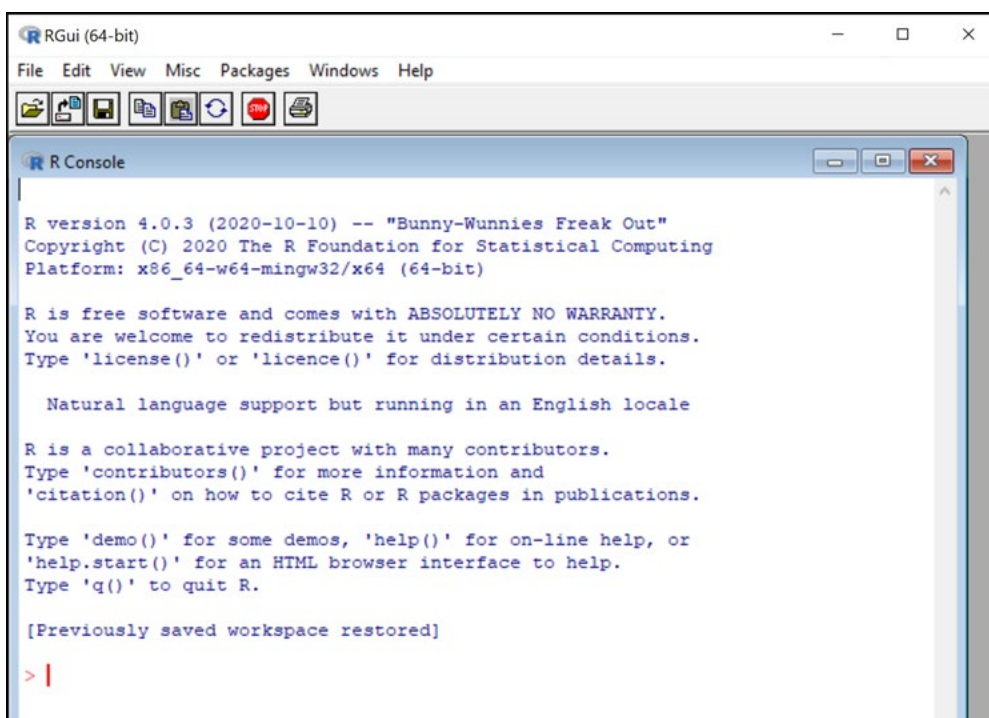


Figure 2.4.
R Console window

³ This is the latest version of R at the time of writing these notes

Now that we have R installed and running, we can directly start typing commands in the R Console window. The ‘>’ sign is the prompt sign. It means that the R Console is ready to accept commands. In the R Console each command is shown in red letters preceded by the ‘>’ sign and all results are shown underneath each command in blue letters. For example, if we want to make a simple multiplication (five times two), we type in the R Console: 5*2 and hit the Enter key. We obtain the results seen in Figure 2.5. Basic operators used in R are given in Table 2.1.

Table 2.1.
Basic operators
in R

Arithmetic operators	Relational operators	Logical operators
+ addition	<= less than or equal to	or
- subtraction	< less than	& and
* multiplication	== equal to	! not
/ division	>= greater than or equal to	
^ exponentiation	!= not equal to	
	> greater than	

If we wish to add a comment, which will not be read by R as part of the code, we use the hash symbol (#), as seen also in Figure 2.5. Everything on a line that is preceded by “#” is not interpreted by R.

Figure 2.5.
Simple multi-
plication in R

```
> 5*2 # 2 multiplied by 5
[1] 10
```

When R is running, data, results, functions etc., are stored in the active memory of the computer in the form of *objects*. The R user can perform various actions on these objects using operators and functions (which are also objects themselves).

Instead of typing commands directly in the R Console window, it is often simpler to write the code in a Word document and then copy and paste it in R. Alternatively, we can open the *R Editor* window from the menu *File → New script*. Now we can type our commands in the Editor window and send them to the R Console to run by selecting the line(s) we want to run and going *Edit → Run line or selection* from the menu bar of the Editor window. We can save the commands we have typed in the R Editor window using *File → Save* from the menu bar. In this way, the commands are saved as an R File.

OTHER WAYS TO INTERACT WITH R

Besides the R Console and the R Editor windows, we may use R through the R Commander or RStudio.

R Commander combines the R Editor and the R Console windows by placing one over the other and offers many additional commands through the menu bar for interacting with the dataset and performing various statistical analyses. R Commander also has a window (R Markdown) that builds files which include any comments we wish to add. When the file is processed, the commands are run and the results are

embedded into a single HTML file. In this way, one can effectively combine a description of what was done, with the commands used and the output.

RStudio arranges the R Editor, R Console and other windows that contain information about the variables, commands, files in the current directory, plots, R packages and others into four panes so that we have everything at our disposal while we work. To install RStudio, go to the website www.rstudio.com

2.3. TYPES OF OBJECTS

Now that we have a means of performing computations, we need a way to store the results. The values produced by R can be stored by assigning names to them. R uses `<-` or `=` to denote assignment. An assignment creates an object and gives it a value. As mentioned above, R objects include data and functions (sequences of commands). We will examine different types of functions/commands throughout this guide. R can handle different types of data and data structures:

NUMERIC VARIABLES contain data that are numbers. `b <- 3.14` or `b = 3.14`
In order to assign a numeric value to a variable, say value 3.14 to variable `b`, we can use the command:

The symbol `"<-"` tells R to assign the value that lies to its right to the variable that lies to its left. If we type the above command in the R Console, then type `b` and hit Enter, we obtain:

```
[1] 3.14
```

STRING VARIABLES consist of data that are text. Such values are assigned to a variable in a similar manner:

```
a <- "hello"
```

If we type `a` in the R Console and press Enter, we obtain:

```
[1] "hello"
```

R IS CASE SENSITIVE!

Note that R is case sensitive. This means that if we name a variable 'length', it is not the same as naming it 'Length'. Be prepared to get many error messages on the R Console because of small misprints like that.

VECTORS contain a series of values, which may be numeric or string or other type. Be careful, all the values of a vector should be of the same type. Assume that we want variable `x` to take the values: 2.45, 3, 7.28, and the value of variable `b` (`=3.14`). To create a vector, we use the concatenate function, `c()`, which groups things together:

```
x <- c(2.45, 3, 7.28, b)
```

If we type `x` in the R Console and hit Enter, we obtain:

```
[1] 2.45 3.00 7.28 3.14
```

MATRICES contain data of the same type (numeric, string etc.) organized in a rectangular format. We can convert a vector into a matrix with `r` rows and `c` columns, provided that the vector contains `rx` data, using the function:

```
y <- matrix(vector, nrow=r, ncol=c)
```

For example, if we type in the R Console:

```
a <- c(3.2, 4.5, 6.2, 9.1, 5, 11)
y <- matrix(a, 2, 3)
y
```

we obtain:

```
      [,1] [,2] [,3]
[1,]  3.2  6.2   5
[2,]  4.5  9.1  11
```

DATAFRAMES have the same structure as matrices, that is, they contain data organized in a rectangular format. However, the columns may contain different types of data, for example one column may have a string variable and another column a numeric variable. Dataframes are created through the combination of variables using the `data.frame()` function:

```
a <- c(2.1, 5.2, 3.3, 4)
e <- c("first", "second", "third", NA)
f <- c(TRUE, TRUE, TRUE, FALSE)
y <- data.frame(a, e, f)
```

If we type `y` in the R Console, we obtain:

```
      a      e      f
1 2.1 first  TRUE
2 5.2 second TRUE
3 3.3 third  TRUE
4 4.0  <NA> FALSE
```

In a dataframe, each row represents a case and each column a variable. As mentioned above, the cases may be archaeological sites, pots, skeletons, metal objects etc. The variables may be the length, depth, texture and other qualitative and quantitative properties.

LISTS contain variables that may be of a different type from each other and have different sample sizes. For example, the following commands create the list named `y`:

```
a <- c(1, 2, 3)
b <- c(TRUE, TRUE, FALSE, FALSE, FALSE)
y <- list(a, b, 1.2, 5, "first")
```

If we type `y` in the R Console, we obtain:

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

[[2]]
[1] TRUE TRUE FALSE FALSE FALSE

[[3]]
[1] 1.2

[[4]]
[1] 5

[[5]]
[1] "first"
```

FACTOR VARIABLES

Sometimes a numeric variable is actually a factor, that is, it expresses the group to which each of the cases of the dataset belong. For example, variable 'Context' may take the values 1, 2, 3, etc. and these values may express stratigraphic sequence. In such cases, prior to using this variable in statistical analyses, we need to specify that it is a factor (i.e. a grouping variable). If, for example, the dataframe `myData` contains a variable named `Context`, coded numerically but to be treated as a factor, we can use the command:

```
myData$Context <- factor(myData$Context)
```

2.4. DATA INPUT

We may input data in R by simply typing the data directly in the R Console, for example using vectors, matrices or dataframes, as shown above. However, when we have large datasets, it is much more efficient to input the data to R from the files where we have them already stored.

In order to input our data to R, we need to *set the working directory*; in other words, we need to let R know where our data is. For this purpose, we create a folder in our local disc C, say *R_Files*, where we store all our R data. Note that we could have our data on our desktop or any other location, but for the purposes of this guide, we will store everything in the *R_Files* folder inside the local disc C. At the beginning of any R session, we go *File* → *Change dir* and browse for this folder in the window *Browse For Folder*. We select it and press *OK* (Figure 2.6).

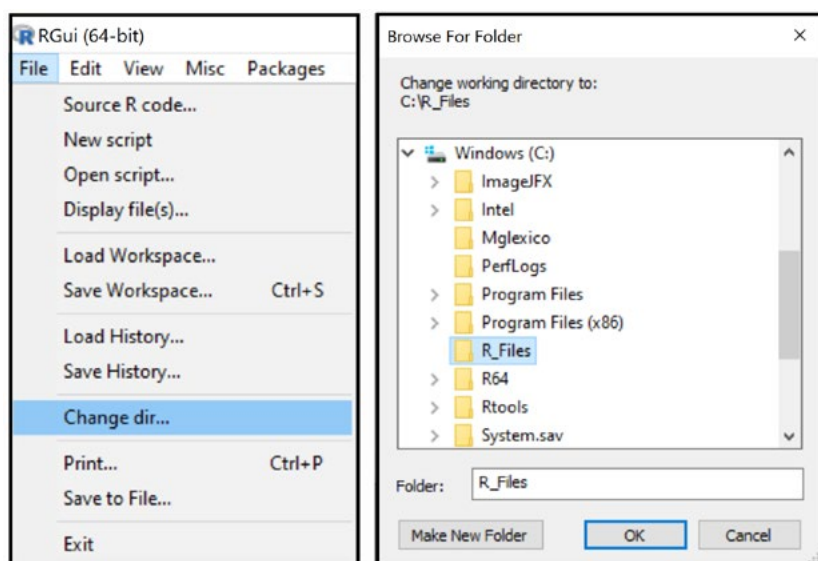


Figure 2.6.
The *Change dir* command in R to browse the working directory

In case we are not certain what our current working directory is, we can find out by typing in the R Console:

```
getwd()
```

You should get in the habit of setting your working directory whenever you use R. Create a directory for the project you are working on so that all the files relating to that project are in one place and do not get mixed up with other projects.

The simplest way to input data in R is through the `read.table()` function when the data is stored in .txt format and the `read.csv()` function when the data is stored in .csv format. Thus, if we have our original data in an Excel spreadsheet, we convert it to .txt (Tab delimited) or CSV (Comma delimited), using the *Save as* command through the Excel *File* menu.

Now, we can open any .txt file saved inside the *R_Files* folder using the `read.table()` function:

```
data <- read.table("SampleData.txt", head=1, sep="\t")
```

where *head=1* means that the first row of the dataset contains the variable names, otherwise we use *head=0*; *sep="\t"* means that the columns are tab delimited.

In addition, we can open any .csv file inside the *R_Files* folder using the `read.csv()` function:

```
data <- read.csv("SampleData.csv", head=1, sep=",")
```

For simplicity, we will save most of the datasets used in this guide in .txt format. As an example, we have the Excel spreadsheet *SampleData.xlsx* (Figure 2.7 – left). We save it as .txt (tab delimited) inside the *R_Files* folder that we have created inside the local disk C (Figure 2.7 – right).

Figure 2.7.
Sample data in
Excel (left) and
.txt (right) format

	A	B	C	D
1	c1	c2	c3	c4
2	a	2.3	4	2.35
3	a	5.6	2	5.78
4	a	5.5	9	5.55
5	b	0.538	11	4.29
6	b	1.312	24	3.65
7	b	-0.401		
8	b	0.973		

*SampleData - Notepad				
File	Edit	Format	View	Help
c1	c2	c3	c4	
a	2.3	4	2.35	
a	5.6	2	5.78	
a	5.5	9	5.55	
b	0.538	11	4.29	
b	1.312	24	3.65	
b	-0.401			
b	0.973			

To input this datafile to R, we type the following function in the R Console:

```
data <- read.table("SampleData.txt", head=1, sep="\t")
```

Now, if we type *data* in the R Console, we obtain:

Figure 2.8.
The datafile
SampleData
entered in R
under the name
data

```
> data
```

	c1	c2	c3	c4
1	a	2.300	4	2.35
2	a	5.600	2	5.78
3	a	5.500	9	5.55
4	b	0.538	11	4.29
5	b	1.312	24	3.65
6	b	-0.401	NA	NA
7	b	0.973	NA	NA

We observe that R automatically fills in the final cells of columns c3 and c4 with NA, which means that these are actually empty cells.

BROWSING FOR OUR FILES

Instead of specifying the name of the file we want to open, we may use the `file.choose()` function inside the `read.table()` or `read.csv()` function. In the example above, instead of typing: `data <- read.table("SampleData.txt", head=1, sep="\t")`, we could have typed `data <- read.table(file.choose(), head=1, sep="\t")`. This will open a new window that allows us to browse our folders and find the file we want to open. If we use the `file.choose()` function, we do not need to set the working directory and our file may be stored in any folder in our computer.

MISSING VALUES

To denote missing values in R, we use the code NA (in capital letters), which stands for 'not available'. For example, in variable *data*:

```
data <- c(3, NA, 7, 4, NA, 10)
```

the second and fifth values are missing.

2.5. SELECTING PARTS OF A DATASET

It is often useful to extract subsets of variables from a dataframe. For example, if we have a dataframe titled *myData* and it contains a variable titled *Weight*, which we want to extract in order to use in further analysis, we may use the command:

```
Weight <- myData$Weight
```

If variable *Weight* occupies the third column in the dataframe, we may alternatively use the command:

```
Weight <- myData[,3]
```

When we want to select multiple columns, say columns 2, 4, 5, we use:

```
Columns2.4.5 <- myData[, c(2, 4, 5)]
```

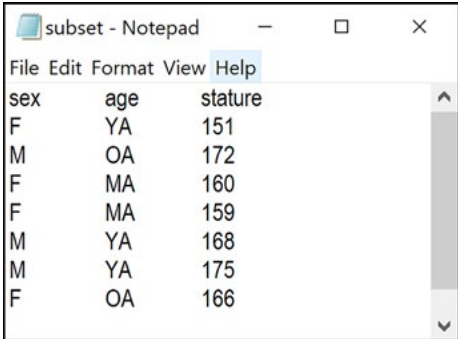
When we want to select multiple consecutive columns, say columns 2, 3, 4, we use:

```
Columns2.3.4 <- myData[,c(2:4)]
```

Rows can be similarly selected by placing the selection before the comma in `[,]`, and we can combine row and column selections in an obvious way, e.g. `myData[3, 5]` will select the third case of the fifth column.

If we want to select a subset of our dataset, we can also use the `subset()` function. For example, if we have the dataset of Figure 2.9, we can select only the females using the command:

```
data <- read.table("subset.txt", head=1,
  sep="\t")
dataF <- subset(data, sex=="F")
```



sex	age	stature
F	YA	151
M	OA	172
F	MA	160
F	MA	159
M	YA	168
M	YA	175
F	OA	166

Figure 2.9.
Dataset of
stature per sex
and age

If we type in the R Console `dataF`, we obtain:

```
> dataF
  sex age stature
1  F  YA    151
3  F  MA    160
4  F  MA    159
7  F  OA    166
```

Figure 2.10.
Subset of females

Similarly, if we want to select young adult females, we use the command:

```
newdataFYA <- subset(data, sex=="F" &
  age=="YA")
```

If we want to select females with stature over 159, we type:

```
newdataFST <- subset(data, sex=="F" & stature>159)
```

If we want to select individuals that are either YA or OA, we use the command:

```
dataYAorOA <- subset(data, age=="YA" | age=="OA")
```

If we want to select the cases 2, 3, 5, we use the command:

```
cases2.3.5 <- data[c(2, 3, 5),]
```

We obtain:

```
> cases2.3.5 <- data[c(2, 3, 5),]
> cases2.3.5
  sex age stature
2  M  OA    172
3  F  MA    160
5  M  YA    168
```

Figure 2.11.
Subset of cases

2.6. PACKAGES

When we install R, we automatically install the basic packages of this language, that is, the code to perform basic functions. However, additional packages are often needed in order to perform specific statistical analyses. Anyone can write a package for other people to use. In order to install and subsequently use a package that is not part of the basic packages already installed in R, we go *Packages* → *Install package(s)*. In the window that pops up we select a *CRAN mirror*. A new window will pop up so that we select the package we want to install.

Alternatively, we may install a package by typing in the R Console the command:

```
install.packages("NameOfPackage")
```

In order to use a package, it is not enough to install it, we also need to load it. For this purpose, we go *Packages* → *Load package* and select it. **Whereas we only need to install each package once, we have to load it every time we open R and intend to use it.**

Alternatively, we may load a package by typing in the R Console the command:

```
library("NameOfPackage")
```

2.7. GETTING HELP

In order to get information about a function in R, we can use the `help()` function:

```
help(NameOfFunction)
```

Alternatively, we may type:

```
?NameOfFunction
```

A new window will open with the help documentation for that function. Note that help files for a specific function are active only if we have loaded the package to which the function belongs.

In addition, the main R Project webpage (www.r-project.org) has manuals, lists of books, a journal, and links to free beginners' guides in multiple languages. Finally, there are a number of blogs and other websites with free advice on R troubleshooting.

3.

DESCRIPTIVE STATISTICS

When a dataset is particularly large, it is very difficult to discern patterns and communicate these to other scholars. It is the purpose of *descriptive statistics* to summarize effectively the dataset and reveal patterns. In particular, descriptive statistics are used for the effective presentation of data by means of summary statistics, contingency tables, and graphs. *Summary statistics* are adopted to express the properties of a sample of continuous data in a more succinct way, whereas for the same purpose *contingency tables* are used for samples of nominal or ordinal data. *Graphs* are used for the visualization of the data properties and concern the majority of the statistical methods.

Assume that we have measured the diameter of 75 postholes from 8 different structures found in an archaeological site (Table 3.1). We assume that the bigger the posthole, the bigger the structure it belonged to. How can we make sense of the size and subsequent function of our sample of postholes by simply visually examining the data in Table 3.1? Even if we are really observant and can discern patterns in this table, would we be able to draw any conclusions if the sample consisted of 1075 postholes? Here is where descriptive statistics comes in handy!

structure	diameter	structure	diameter	structure	diameter	structure	diameter
1	7.2	3	10.0	5	17.3	7	10.5
1	9.0	3	9.1	5	14.7	7	9.3
1	9.8	3	13.4	5	16.8	7	11.4
1	5.5	3	11.5	5	14.9	7	10.9
1	8.9	3	10.6	5	18.0	7	11.8
1	8.2	3	11.0	5	16.1	7	11.2
1	7.5	3	10.8	5	16.3	7	8.8
1	6.2	3	8.9	6	11.6	8	10.3
2	11.2	4	11.3	6	12.1	8	6.3
2	12.3	4	13.6	6	12.1	8	7.8
2	10.7	4	11.3	6	10.9	8	15.1
2	12.7	4	11.1	6	13.0	8	16.6
2	9.9	4	9.2	6	10.5	8	19.2
2	10.8	4	10.8	6	11.4	8	9.9
2	8.2	4	17.9	6	13.7	8	10.0
2	11.0	5	15.0	6	9.8	8	14.7
2	10.0	5	14.2	6	12.5	8	17.2
2	12.8	5	14.4	7	11.5	8	10.7
3	13.2	5	13.6	7	9.1		

Table 3.1.
Posthole
diameter (in cm)
per structure

3.1. SUMMARY STATISTICS

Summary statistics consists of measures that summarize data and can be calculated for *quantitative variables*. These measures express the properties of a sample and can be classified in three groups:

- a. *measures of central tendency or measures of location*
- b. *measures of dispersion*, and
- c. *measures of shape*

Measures of central tendency give information concerning the location of the center of the distribution of the data if we arrange it along an axis. *Measures of dispersion*, as their name suggests, show how dispersed the data is around the midpoint along this axis. *Measures of shape* relate to the shape of the distribution of the data, that is, they show how the data is arranged around a central value. **Table 3.2** presents the most common measures of summary statistics.

Table 3.2.
Summary
statistics

Measures of central tendency	Measures of dispersion	Measures of shape
Mean	Variance	Skewness
Trimmed mean	Standard deviation	Kurtosis
Median	Range	
Mode	Interquartile range	

Measures of central tendency

The *mean or average value* of a sample is often labelled $\bar{x}$ and it is calculated by summing all the values in the dataset, and then dividing this sum by the number of observations:

$$\bar{x} = \frac{\sum x}{m} = \frac{(x_1 + x_2 + \dots + x_m)}{m}$$

In the above equation the Greek letter Σ (capital sigma) stands for “the sum of”, $x_1, x_2, \dots, x_m$ are the sample values (x_1 is the first sample value, x_2 the second sample value and so on), and m is the sample size.

For example, let's assume that inside one of our excavation trenches we found 15 amphorae and their capacity is given in **Table 3.3**.

Table 3.3.
Amphora
capacity

Amphora	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Capacity (lt)	28.1	22.7	24.9	19.4	21.3	30.2	24.4	27.5	29.1	28.5	29.4	27.3	22.8	26.3	30.0

To calculate the average capacity of these amphorae, we add all capacity values and divide the sum by the number of amphorae:

$$\text{Average capacity} = \frac{(28.1+22.7+24.9+\dots+26.3+30.0)}{15} = \frac{391.9}{15} \approx 26.1 \text{ litres}$$

The basic meaning of the mean is that it is the value around which all other values of the sample cluster. This is valid provided that the sample does not contain *outliers*, i.e. values that differ significantly from the rest. For example, for the hypothetical sample (2, 3, 60, 4, 6, 2, 4, 3, 4) the mean is 9.78 due to the extreme (and possibly wrong) value 60. Because of this outlier, 9.78 is not really the value around which the other sample values cluster. In this case we should use either the trimmed mean or the median. The *trimmed mean* is the mean value calculated after removing a percentage of the smallest and largest values.

The *median* is the value that lies in the middle of the sample values. In other words, 50% of the sample values are smaller than or equal to the median and 50% of the sample values are larger than or equal to the median. In order to find the median, we need to arrange the sample values from the smallest to the largest and locate the value that lies in the middle. In the amphora capacity example, the median is equal to 27.3 since this is the value lying in the middle of the dataset with 7 values above it and 7 values below it:

19.4 21.3 22.7 22.8 24.4 24.9 26.3 27.3 27.5 28.1 28.5 29.1 29.4 30.0 30.2

Let's assume that our sample consists of 16 amphorae and the capacity of the 16th amphora is 25.2 litres. Our sample now becomes:

19.4 21.3 22.7 22.8 24.4 24.9 25.2 26.3 27.3 27.5 28.1 28.5 29.1 29.4 30.0 30.2

The new median is equal to the mean value of the two middle values: $(26.3 + 27.3)/2 = 26.8$ litres

The *mode* is the value that appears most often in the dataset. This summary statistic is mostly useful for discrete variables. In cases where two or more values are most frequent and appear an equal number of times in a dataset, the samples are called *bimodal* (if two modes are present) or *multimodal* (when more than two modes exist). When one single mode exists in the dataset, the distribution is called *unimodal*.

Measures of dispersion

The simplest way to determine the dispersion of the sample values is to calculate the range. The *range* is the difference between the maximum and the minimum values in the dataset. In the amphora capacity example, based on the data of Table 3.3, the maximum value is 30.2 and the minimum 19.4 litres. So, the range is equal to $30.2 - 19.4 = 10.8$ litres. Among the limitations of this measure is that the range is affected by extreme values (outliers) and it does not effectively express the variability in the data as it only takes into consideration the largest and smallest value.

To overcome these limitations, we can instead calculate the quartiles and the interquartile range. Every sample can be divided in four equal parts using three *quartiles*. The *first quartile (Q1)* is the sample value below which 25% of the sample values lie. The *third quartile (Q3)* is the sample value below which 75% of the sample values lie. The *second quartile (Q2)* is the median. The difference between the third and first quartile ($Q3 - Q1$) is called *interquartile range (IQR)*. The advantage of the interquartile range (compared to the range) is that it is not affected by extreme scores at either end of the distribution. However, it has the limitation that it is estimated using only half of the data. Quartiles are used primarily in the construction of boxplots, as will be described in section 3.5. In the sample of amphora capacities, the quartiles are as follows:

Q1 **Q2** **Q3**
 ↓ ↓ ↓

19.4 21.3 22.7 **22.8** 24.4 24.9 26.3 **27.3** 27.5 28.1 28.5 **29.1** 29.4 30.0 30.2

The *variance* shows how dispersed the sample values are around the mean. It is calculated as:

$$s^2 = \frac{\sum (x - \bar{x})^2}{m - 1} = \frac{(x_1 - \bar{x})^2 + (x_2 - \bar{x})^2 + \dots + (x_m - \bar{x})^2}{m - 1}$$

where s^2 is the variance of x , $\bar{x}$ is the mean of x , and m is the sample size.

If the value of the variance is high, then the distribution of the sample values is substantially spread out, thus the sample is more heterogeneous. In order to estimate the variance of the capacity of the amphorae (Table 3.3), we calculate:

$$s^2 = \frac{(28.1-26.1)^2 + (22.7-26.1)^2 + \dots + (22.8-26.1)^2 + (26.3-26.1)^2 + (30.0-26.1)^2}{15-1} = \frac{159.8}{14} \approx 11.4 \text{ litres}^2$$

The variance has the disadvantage that it is expressed in units squared. In the example of amphora capacities, the variance is in litres². What is the physical meaning of squared litres? To overcome this problem, we calculate the *standard deviation*, s , which is the square root of the variance. For the amphora capacities example, the standard deviation is:

$$s = \sqrt{s^2} = \sqrt{11.4} \approx 3.4 \text{ litres}$$

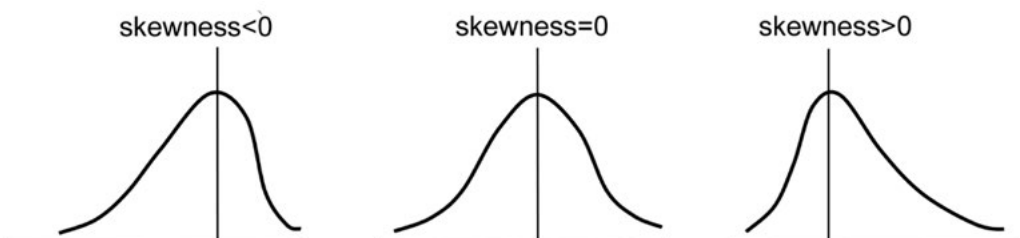
Both the variance and the standard deviation have the disadvantage that they are affected by outliers, thus, their interpretation should be cautious.

To distinguish the mean, $\bar{x}$, variance, s^2 , and standard deviation, s , of a sample from those of the corresponding population, the mean, variance and standard deviation of the population are denoted by μ , σ^2 and σ , respectively.

Measures of shape

Sample values are distributed around the mean value. This distribution may be symmetric or asymmetric. The measures used to describe the shape of distribution of the data, that is, how the data is arranged around a central value, are *skewness* and *kurtosis*. Skewness shows how symmetric the distribution of the data around a central value (mean or median) is, while kurtosis shows how pointy this distribution is. When skewness = 0, the distribution of values around the mean is symmetric, when skewness < 0, the distribution is asymmetric and extends more to the left, which means that most values in the sample are smaller than the mean, while when skewness > 0, the distribution is asymmetric and extends more to the right (Figure 3.1). When kurtosis = 3, the distribution is neither too narrow nor too broad, when kurtosis < 3, the distribution is too broad, and when kurtosis > 3, the distribution is too narrow (Figure 3.2). In a *normal distribution* the value of skewness is 0 and kurtosis is equal to 3.

Figure 3.1.
Distributions
with positive,
zero and negative
skewness



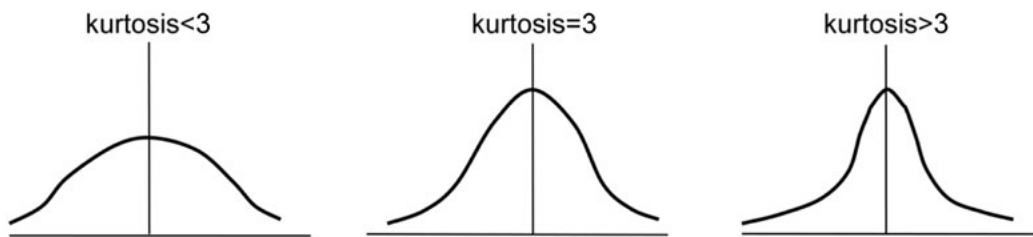


Figure 3.2.
Distributions
with negative,
zero and
positive kurtosis

3.2. SUMMARY STATISTICS IN R

We may use various R functions to calculate summary statistics. The most important of them are listed in **Table 3.4**.

Function	Output
<code>mean(x)</code>	Mean
<code>mean(x, trim = percentage)</code>	Trimmed mean
<code>median(x)</code>	Median
<code>quantile(x)</code>	Three quartiles, minimum and maximum
<code>var(x)</code>	Variance
<code>sd(x)</code>	Standard deviation
<code>range(x)</code>	Minimum and maximum
<code>min(x)</code>	Minimum
<code>max(x)</code>	Maximum
<code>skewness(x)</code> (in <i>moments</i> package)	Skewness
<code>kurtosis(x)</code> (in <i>moments</i> package)	Kurtosis

Table 3.4.
R functions
for summary
statistics

Three other interesting functions are the functions `summary(x)`, `describe(x)` and `describeBy(x, group)`. The function `summary(x)` estimates the minimum and maximum value, first and third quartile, median, and mean. The functions `describe(x)` and `describeBy(x, group)` belong to the *psych* package and, therefore, in order to use them, we should have first installed and subsequently loaded package *psych*. The function `describe(x)` estimates the minimum and maximum value, the range, median, mean, trimmed mean, standard deviation, skewness and kurtosis. If in the dataset there is a grouping variable and we want to estimate the descriptive statistics per group, we can use the function `describeBy(x, group)`, where *group* is the grouping variable.

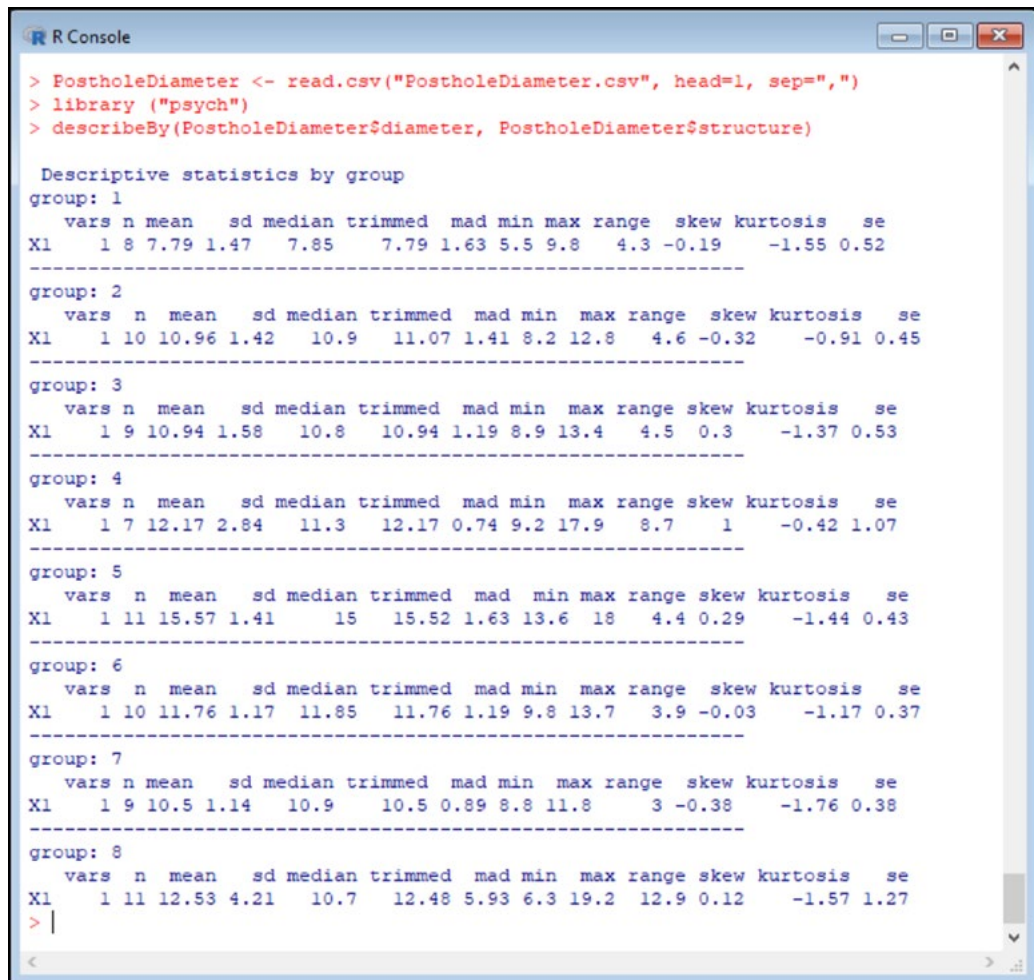
As an example, consider the data of **Table 3.1**. We arrange the data in an Excel spreadsheet, in two columns, one defining the structure and the other the posthole diameter, and save this file in the *R_Files* folder as *PostholeDiameter.csv*.

To calculate the summary statistics for each structure, we type in the R Console:

```
PostholeDiameter <- read.csv("PostholeDiameter.csv", head=1,
sep=",")
library("psych")
describeBy(PostholeDiameter$diameter,
PostholeDiameter$structure)
```

Figure 3.3.

Summary statistics for the posthole diameter per structure in R



```
> PostholeDiameter <- read.csv("PostholeDiameter.csv", head=1, sep=",")
> library("psych")
> describeBy(PostholeDiameter$diameter, PostholeDiameter$structure)

Descriptive statistics by group
group: 1
  vars n mean   sd median trimmed  mad min max range skew kurtosis   se
Xl    1  8 7.79 1.47   7.85   7.79 1.63 5.5 9.8   4.3 -0.19   -1.55 0.52
-----
group: 2
  vars n mean   sd median trimmed  mad min max range skew kurtosis   se
Xl    1 10 10.96 1.42   10.9   11.07 1.41 8.2 12.8   4.6 -0.32   -0.91 0.45
-----
group: 3
  vars n mean   sd median trimmed  mad min max range skew kurtosis   se
Xl    1  9 10.94 1.58   10.8   10.94 1.19 8.9 13.4   4.5  0.3   -1.37 0.53
-----
group: 4
  vars n mean   sd median trimmed  mad min max range skew kurtosis   se
Xl    1  7 12.17 2.84   11.3   12.17 0.74 9.2 17.9   8.7  1   -0.42 1.07
-----
group: 5
  vars n mean   sd median trimmed  mad min max range skew kurtosis   se
Xl    1 11 15.57 1.41   15   15.52 1.63 13.6 18   4.4 0.29   -1.44 0.43
-----
group: 6
  vars n mean   sd median trimmed  mad min max range skew kurtosis   se
Xl    1 10 11.76 1.17   11.85   11.76 1.19 9.8 13.7   3.9 -0.03   -1.17 0.37
-----
group: 7
  vars n mean   sd median trimmed  mad min max range skew kurtosis   se
Xl    1  9 10.5 1.14   10.9   10.5 0.89 8.8 11.8   3 -0.38   -1.76 0.38
-----
group: 8
  vars n mean   sd median trimmed  mad min max range skew kurtosis   se
Xl    1 11 12.53 4.21   10.7   12.48 5.93 6.3 19.2  12.9 0.12   -1.57 1.27
> |
```

The first command reads the data we have in the .csv file. The second command loads the *psych* package and the third command estimates the summary statistics and it does so for each structure.

Note that the first command `PostholeDiameter <- read.csv(file.choose(), head=1, sep=",")` may be replaced by:

which offers more flexibility since it allows us to browse our folders and find the file we want to open. If we use the command `PostholeDiameter <- read.csv("PostholeDiameter.csv", head=1, sep=",")`, we first need to set the working directory via *File → Change dir.*

We obtain the results shown in Figure 3.3.

From these results, we can conclude that the mean posthole diameter is similar in structures 2, 3 and 7, it is the smallest in structure 1 and the largest in structure 5. The sample values for structures 4 and 8 are broadly dispersed based on the standard deviation values, which suggests that postholes of various sizes were employed in these cases. This feature may imply that these structures included various sub-structures. Skewness values vary per structure but all kurtosis values are negative, which implies that most sample values are asymmetrical and spread out. The above properties can become clearer if we draw boxplots. This issue will be discussed in section 3.5.

3.3. CONTINGENCY TABLES

When we study qualitative data, whether ordinal or nominal, the above summary statistics cannot be calculated. In such cases, we calculate *frequencies*, that is, we assess how many times different values appear in the sample. Let's assume that $x_1, x_2, \dots, x_m$ are the values of a variable in a sample. We call *frequency of value x_i* the integer v_i that shows how many times the value x_i appears in the sample. If $v = v_1 + v_2 + \dots + v_m$ then the ratio:

$$f_i = \frac{v_i}{v}$$

is called *relative frequency of x_i* .

For example, if we study the color of different pots in an assemblage and end up with the sample {red, black, orange, red, yellow, white, red}, the frequency of red pots is 3 since we have 3 red pots in our sample. The relative frequency of red pots (f_{red}) is equal to $\frac{3}{7}$ since we have 3 red pots in a sample of 7 pots. The *percentage frequency* of red pots is:
 $\frac{3}{7} \times 100 \approx 42.9\%$.

A table that displays the frequency of a categorical variable is called a *frequency table*. **Table 3.5** is a frequency table that displays the frequency of some of the ceramic typologies from the Middle Minoan III palace at Knossos⁴:

Ceramic typology	Frequency
Broad HC 1A	37
Broad HC 1B	137
Conical HC 2A	35
Tall HC 3A	150
Tall HC 3B	25
Carinated cups	16
Piriform jars	23
Imported jars	26

Table 3.5.
Frequency of
selected ceramic
typologies at
Knossos

A table that displays the relationship between two categorical variables is called a *contingency table*. Each value in this table represents the number of times a particular combination of variable outcomes occurred. **Table 3.6** is a contingency table and presents the frequency of males and females that had been interred or cremated in an ancient cemetery. In this table we have two categorical variables: sex and post-mortem treatment. Sex has two categories: male and female, while post-mortem treatment also has two categories: interment and cremation.

		Sex		Total
		Males	Females	
Post-mortem treatment	Interment	13	5	18
	Cremation	6	12	18
	Total	19	17	36

Table 3.6.
Contingency
table for
post-mortem
treatment per
sex (frequencies)

⁴ Data obtained from Figure 1.2 in Knappett C, Mathioudaki I, Macdonald CF. 2013. Stratigraphy and ceramic typology in the Middle Minoan III palace at Knossos. *British School at Athens Studies* 21: 9-19

In the above contingency table, the values in the centre (13, 5, 6, 12) are called *joint frequencies*, whereas the values in the margins (18, 18, 19, 17, 36) are called *marginal frequencies*.

In cases where we wish to explore the relationship between categorical variables, it is more helpful to use relative or percentage frequencies in order to standardize the data, especially when the sample sizes in each category are different. For the data given in **Table 3.6**, we have 19 males and 17 females, thus the difference in the sample sizes is small and in this instance we can see some clear patterns in post-mortem treatment even using the raw frequencies. However, if the number of males and females in the dataset had been very different, it would have been hard to tell whether there is a distinct pattern in the post-mortem treatment of each sex based solely on raw frequencies. **Table 3.7** presents the percentage frequencies calculated based on the data of **Table 3.6**.

Table 3.7.
Contingency
table for
post-mortem
treatment per
sex (percentage
frequencies)

		Sex	
		Males	Females
Post-mortem treatment	Interment	68.4	29.4
	Cremation	31.6	70.6
	Total	100%	100%

3.4. CONTINGENCY TABLES IN R

Assume that we have the data shown in **Table 3.8** and want to generate the contingency **Table 3.6**. We arrange the data in an Excel spreadsheet, in two columns, one defining the sex and the other the post-mortem treatment, and save this file in the *R_Files* folder as *SexPMtreat.txt* tab delimited file.

Table 3.8.
Data to be
organized in
a contingency
table

Sex	PM_treatment	Sex	PM_treatment
1	1	1	1
1	2	2	1
2	1	2	2
2	2	1	1
2	1	1	2
1	1	2	2
1	1	2	1
2	2	1	1
1	2	2	2
1	1	1	1
2	2	1	1
2	1	2	2
1	2	1	2
2	2	1	1
1	2	2	2
1	1	2	2
2	2	2	2
1	1	1	1

Key:

sex:

1 = males,

2 = females;

post-mortem treatment:

1 = interment,

2 = cremation

To calculate the contingency table, we type in the R Console:

```
data <- read.table("SexPMtreat.txt", head=1, sep="\t")
levels(data$sex) <- c("males", "females")
levels(data$PM_treatment) <- c("interment", "cremation")
x <- table(data$PM_treatment, data$sex)
rownames(x) <- levels(data$PM_treatment)
colnames(x) <- levels(data$sex)
x
```

Here, the first command opens the file *SexPMtreat.txt* in R using the name *data*, the second and third commands define the levels of the categorical variables *sex* and *PM_treatment*, respectively, the next command calculates the contingency table, and the fifth and sixth commands are used to label the rows and columns of the contingency table.

We obtain the output of
Figure 3.4:

	males	females
interment	13	5
cremation	6	12

Figure 3.4.
Contingency
table

For relative frequencies we can use the command:

```
px <- prop.table(x, 2)
```

where *x* is the contingency table we have already created and 2 suggests that we estimate the relative frequency with regard to variable *sex*. If we had typed 1, we would have estimated the relative frequency with regard to *PM treatment*.

If we type in the R Console *px*, we obtain the output of Figure 3.5:

```
> px <- prop.table(x, 2)
> px

           males    females
interment 0.6842105 0.2941176
cremation  0.3157895 0.7058824
```

Figure 3.5.
Contingency
table of relative
frequencies

3.5. GRAPHS

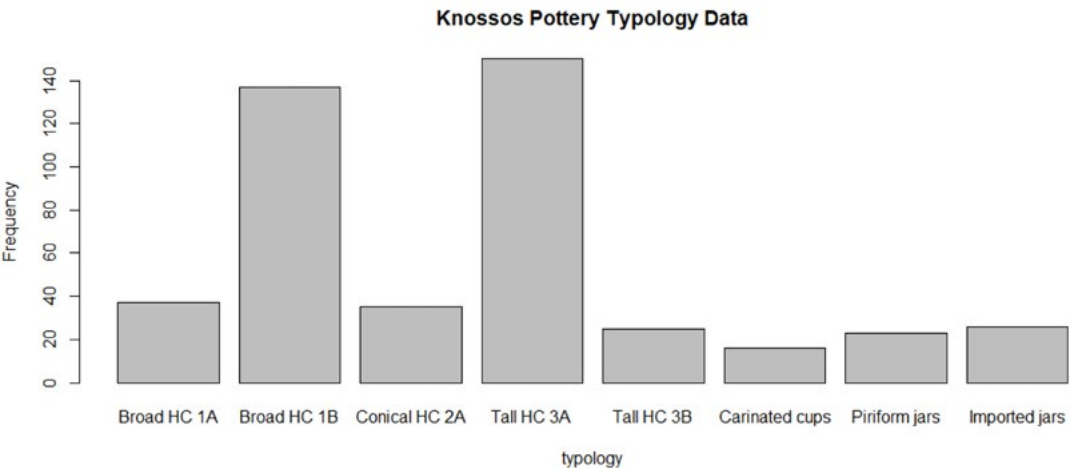
The most effective way to present quantitative and qualitative data is by means of graphs. There are many different types of graphs. The most commonly used ones in descriptive statistics are:

1. Bar charts
2. Pie charts
3. Histograms
4. Boxplots

The first two are used for qualitative variables and the last two for quantitative variables.

BAR CHARTS are plotted based on the frequency table of a qualitative variable. The different categories of the variable are placed in the horizontal axis and a rectangular column corresponds to each category with height proportional to the frequency of each category. All columns are of the same width. As an example, Figure 3.6 shows the bar chart of the Minoan ceramic typologies given in Table 3.5.

Figure 3.6.
Bar chart with the frequency of the Minoan ceramic typologies given in Table 3.5.



A **SEGMENTED BAR CHART** is a type of bar plot where two variables are depicted together. For example, segmented bar charts representing the data of the contingency **Table 3.7** are shown in **Figures 3.7** (clustered bar chart) and **3.8** (stacked bar chart). It is seen that graphs often visualize much more efficiently frequency data compared to contingency tables. In these segmented plots, we observe immediately that females are more often cremated while males are more often interred.

Figure 3.7.
Clustered bar chart of post-mortem treatment per sex (left)

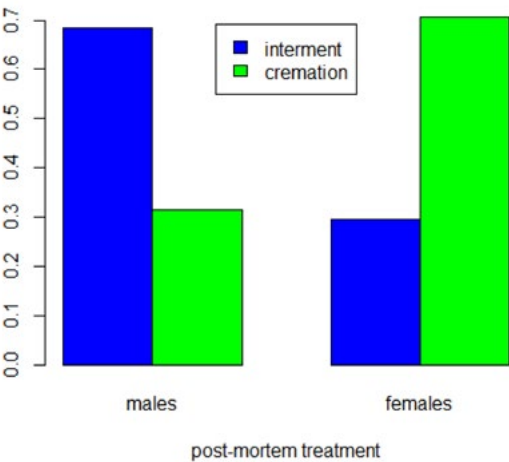
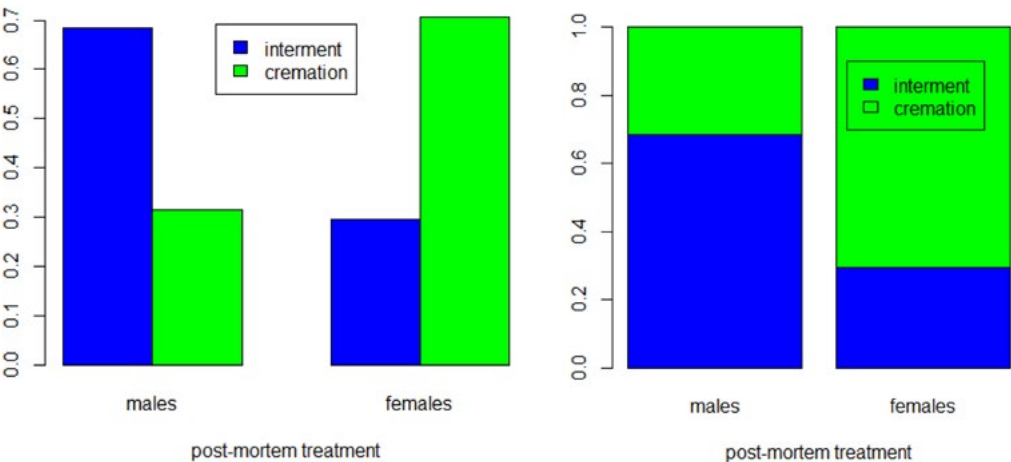


Figure 3.8.
Stacked bar chart of post-mortem treatment per sex (right)



PIE CHARTS are also used for qualitative variables and are created based on frequency tables. They are circular graphs divided in sectors. They look like pizzas with unequal slices. Each sector (slice) represents one category and has area proportional to the frequency of this category. **Figure 3.9** is the pie chart of the data of **Table 3.9** for the average percentage frequency of various lithic debitage categories from archaeological sites in Arizona⁵.

A limitation of such plots is that it is generally difficult to compare categories with nearly identical counts or proportions. In addition, pie charts are problematic to use when there are many categories.

⁵ Data from Table 2, column Technological Group IA, in Sullivan AP, Rozen KC. 1985. Debitage Analysis and Archaeological Interpretation. *American Antiquity* 50: 755-779

Debitage Category	Percentage
Complete Flakes	53.4
Broken Flakes	6.7
Flake Fragments	16.0
Debris	6.1
Cores	14.7
Retouched Pieces	3.1

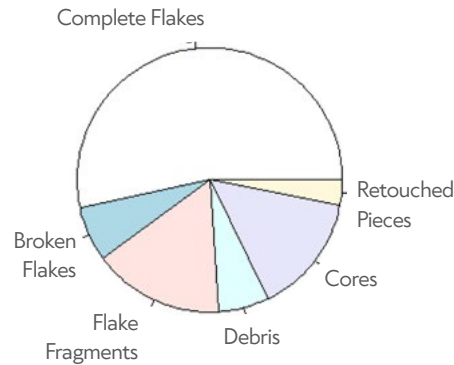


Table 3.9.
Average
percentage
frequency of
lithic debitage
categories

Figure 3.9.
Pie chart for
the data of
Table 3.9

HISTOGRAMS are similar to bar charts but for continuous data. The problem in this case is that the data does not fall in specific predefined categories. To overcome this problem, we create arbitrary categories as follows: If $X_{\min}$ and $X_{\max}$ are the minimum and maximum sample values, we divide $X_{\max} - X_{\min}$ into k sectors with range $\Delta x = (X_{\max} - X_{\min})/k$. These sectors are called *classes* or *bins*. Subsequently, we count the number of sample values that fall within each class. This number is called *class (or bin) frequency*. Similarly, we determine relative and percentage class frequencies. Thus, histograms provide a view of the data density. Higher bars show where the data are relatively more common.

Males	49.9	52.5	49.7	52.3	51.2	54.2	55.2	53.4	57.8	50.9
Females	43.9	50.1	44.9	40.2	44.7	45.8	44.1	39.2	47.1	48.1

Table 3.10.
Sample of
femoral
lengths (in cm)
of individuals
excavated in a
Medieval site

As an example, consider the data sample of **Table 3.10**, consisting of the femoral lengths (in cm) of 20 individuals excavated in a Medieval site. We observe that the minimum ($X_{\min}$) is 39.2 cm and the maximum ($X_{\max}$) is 57.8 cm. So, the range of values is equal to $57.8 - 39.2 = 18.6$ cm. The next step is to divide this range into equal segments (classes). A crucial issue is how many classes we should use. It has been proposed that the number of classes should be equal to the square root of the sample size. However, it is advisable that each researcher determines the number of classes to be used based on the heterogeneity of the data. Note that all commercial statistics software packages calculate the number of classes automatically when plotting histograms. Nonetheless, many of them allow the researcher to manually modify the number of classes in order to achieve the clearest results.

In the present example, given that the sample size is small, we will use 5 classes: 1) 35-40 cm, 2) 40-45 cm, 3) 45-50 cm, 4) 50-55 cm, 5) 55-60 cm. The next step is to calculate the frequency of each class; in other words, count how many femora fall in each class. We obtain the class frequencies shown in **Table 3.11**. Finally, we plot this data and obtain **Figure 3.10**.

Class	f
35-40	1
40-45	5
45-50	5
50-55	7
55-60	2

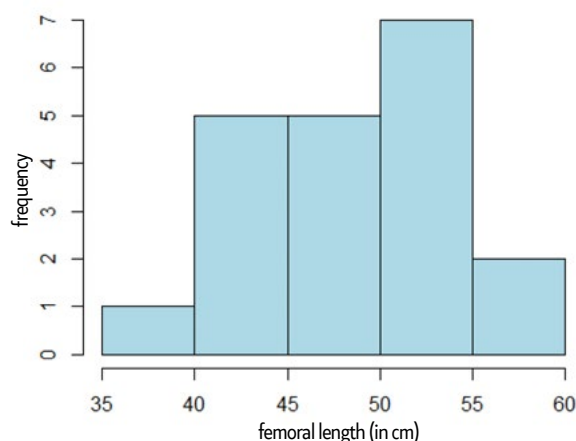


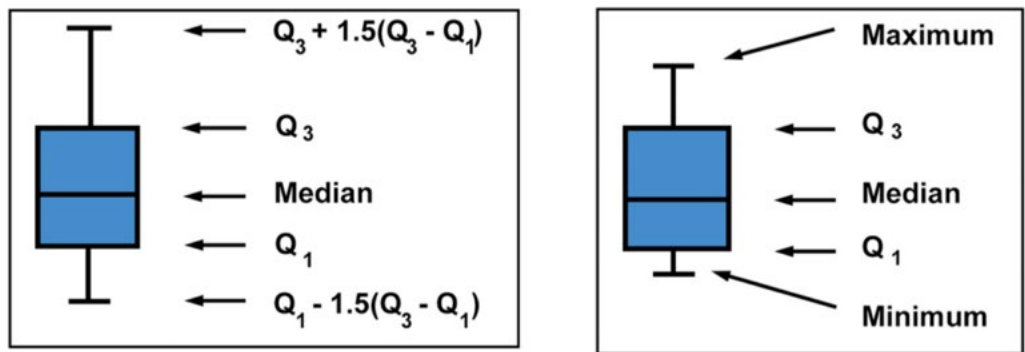
Table 3.11.
Frequency
distribution
of femoral
lengths

Figure 3.10.
Histogram
based on
femoral
lengths

A visual inspection of a sample through its histogram may offer useful information about the distribution of the sample values. However, this information is valid only if the sample is relatively large since in small samples the shape of the histogram may be strongly affected by the number of bins.

BOXPLOTS or **BOX-WHISKER DIAGRAMS** consist of a rectangle with two antennas, one at the lower base and the other at the upper one. The antennas are T and inverse T-shaped, respectively. The lower base of the rectangle lies on the first quartile (Q_1) and the upper base marks the third quartile (Q_3). The median is represented by a horizontal line in the interior of the rectangle (Figure 3.11). The antennas are called *whiskers* and extend up to $Q_3 + 1.5(Q_3 - Q_1)$ and $Q_1 - 1.5(Q_3 - Q_1)$. If the maximum and minimum sample values lie within this range, then the whiskers shift to the maximum and minimum values. If there are outliers, these appear as points beyond the whiskers.

Figure 3.11.
Boxplots



Boxplots have the advantage that they visually display the distribution of sample values, show outliers, and allow for the effective comparison between samples. The main drawback is that many of the details of the distribution are not as clearly identifiable with boxplots as they are with histograms.

Figure 3.12.
Boxplot of male
and female
femoral length
values

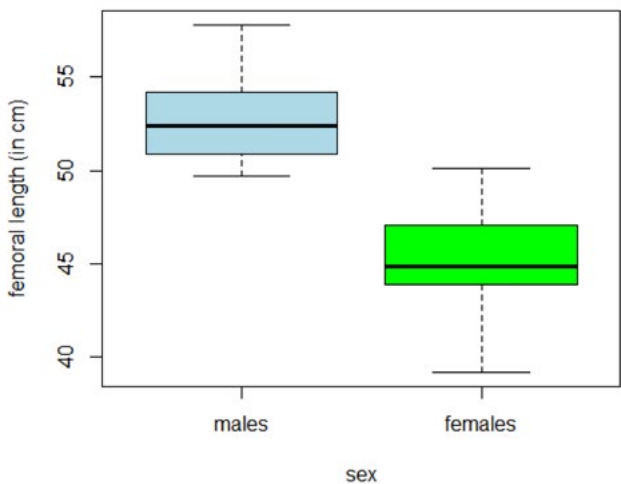


Figure 3.12 shows the boxplots of the femoral lengths of the 20 Medieval individuals given in Table 3.10. It can be seen that male values are greater than female ones; therefore, there is pronounced sexual dimorphism in long bone length in this assemblage.

3.6. GRAPHS IN R

The basic version of R includes the `plot()` function, which can create a wide variety of graphs. For bar plots, pie plots, boxplots, and histograms we may use the functions `barplot()`, `pie()`, `boxplot()`, and `hist()`, respectively. Details for the use of these functions may be found in many websites as well as via the commands `?barplot`, `?pie`, `?boxplot`, and `?hist`.

A more advanced way to make graphs in R is through the *ggplot2* package. In *ggplot2*, any graph is made up of a series of layers. Each layer contains one of the visual elements of the graph: text, data points, axes, lines, etc. For the final image, these layers are placed on top of each other.

In *ggplot2* the visual elements are known as *geoms* (short for ‘geometric objects’). Therefore, when we define a layer, we have to tell R what geom we want displayed on that layer. The *geoms* have aesthetic properties (`aes()`) that determine their appearance (e.g. their colour, size, style, location). Aesthetics may be defined for the whole plot or for each layer.

Geometric objects

The most common *geoms* are the following:

- `geom_bar()`: creates a layer with bars
- `geom_point()`: creates a layer with data points
- `geom_line()`: creates a layer with straight lines that connect data points
- `geom_smooth()`: creates a layer with a ‘smoother’ (i.e., instead of straight lines, it uses a smooth line that passes through the data points)
- `geom_histogram()`: creates a layer with a histogram
- `geom_boxplot()`: creates a layer with a boxplot

For each *geom*, we can specify various aesthetics that determine what the *geom* will look like. Some of these aesthetics are required and others are optional, as shown in Table 3.12.

Geom	Required properties	Optional properties	
<code>geom_bar()</code>	x: the variable to plot on the x-axis	• colour • size • fill	• linetype • alpha (transparency)
<code>geom_point()</code>	x: the variable to plot on the x-axis y: the variable to plot on the y-axis	• shape • colour • size	• fill • alpha
<code>geom_line()</code>	x: the variable to plot on the x-axis y: the variable to plot on the y-axis	• colour • size • line	• type • alpha
<code>geom_smooth()</code>	x: the variable to plot on the x-axis y: the variable to plot on the y-axis	• colour • size • fill	• linetype • alpha
<code>geom_histogram()</code>	x: the variable to plot on the x-axis	• colour • size • fill	• linetype • alpha
<code>geom_boxplot()</code>	x: the variable to plot	• colour • size • fill • alpha	

Table 3.12.
Required and optional aesthetics for different *geoms* (from Table 4.1. in Field et al. 2012⁶)

⁶ Field A, Miles J, Field Z. 2012. *Discovering Statistics using R*. Sage Publishing

Aesthetics

Aesthetics control the appearance of graphs and may be specified for the plot as a whole or for individual *geoms*. Aesthetics can be set to a specific value (e.g., set the color of the bars to red). In such cases, the aesthetic is not specified within the `aes()` function but it needs to be specified within the `geom()` used to create the particular layer of the plot. Alternatively, aesthetics can be set to vary as a function of a variable (e.g., set the bar color to red for females and to blue for males). In these cases, the aesthetic should be specified within `aes()`. Table 3.13 lists the main aesthetics and how to specify each one.

Table 3.13.
Specifying
key aesthetics
(from Table
4.2. in Field et
al. 2012⁷)

Aesthetic	Option	Outcome
Linetype	linetype = 1	Solid line (default)
	linetype = 2	Hashed line
	linetype = 3	Dotted line
	linetype = 4	Dot and hash
	linetype = 5	Long hash
	linetype = 6	Dot and long hash
	...	...
Size	size = value	Replace 'value' with a value in mm (default size = 0.5). Values larger than 0.5 give fatter lines/larger text/bigger points than the default whereas smaller values will produce thinner lines/smaller text and points than the default
Shape	shape = integer, shape = "x"	The integer is a value between 0 and 25, each of which specifies a particular shape. Some common examples are given below. Alternatively, specify a single character in quotes to use that character (shape = "A" will plot each point as the letter A)
	shape = 0	Hollow square
	shape = 1	Hollow circle
	shape = 2	Hollow triangle
	shape = 3	Cross
	shape = 5	Hollow rhombus
	shape = 6	Hollow inverted triangle
	...	...
Colour	colour = "Name"	Simply type the name of a standard colour. For example, colour = "Red" will make the geom red
	colour = "#RRGGBB"	Specify colours using the RRGGBB system. For example, colour = "#3366FF" produces a shade of blue, whereas colour = "#336633" produces a dark green
Alpha	alpha(colour, value)	Colours can be made transparent by specifying alpha, which can range from 0 (fully transparent) to 1 (fully opaque). For example, alpha("Red", 0.5) will produce a half transparent red

⁷ Field A, Miles J, Field Z. 2012. *Discovering Statistics using R*. Sage Publishing

Scatterplots

Although scatterplots are not closely related to descriptive statistics, we start from this type of plots in order to understand the functionality of `ggplot2`. A scatterplot is a type of plot that uses points placed on a two-dimensional Cartesian coordinate system to display values from two or more variables.

Consider the dataset shown in **Table 3.14**, which is a collection of femoral maximum lengths and midshaft diameters (in cm) of individuals from three archaeological sites. This dataset is named *sampleGraph* and it is saved in .txt tab delimited format in the *R_Files* folder.

Site	FemLength	FemMidDiam
1	48	2.3
1	47	2.1
1	43	2
1	40	2
1	52	2.5
1	51	2.2
1	54	2.4
1	50	2.1
2	51	2.4
2	55	2.7
2	45	2.2
2	53	2.6
2	50	2.5
2	51	2.4
2	54	2.3
2	49	2.1
2	48	2
2	47	2
3	48	2.3
3	49	2.5
3	46	2.1
3	45	2
3	49	2.2

Table 3.14.
Femoral measurements per archaeological site

Assume that we want to plot femoral midshaft diameter versus femoral length.

First, we input the dataset *sampleGraph.txt* to R using the command:

```
dataplot <- read.table("sampleGraph.txt", head=1, sep="\t")
```

or, for more flexibility, using the command:

```
dataplot <- read.table(file.choose(), head=1, sep="\t")
```

Then we use the commands: `library(ggplot2)`
`plot <- ggplot(data=dataplot, aes(x=FemLength, y=FemMidDiam))`

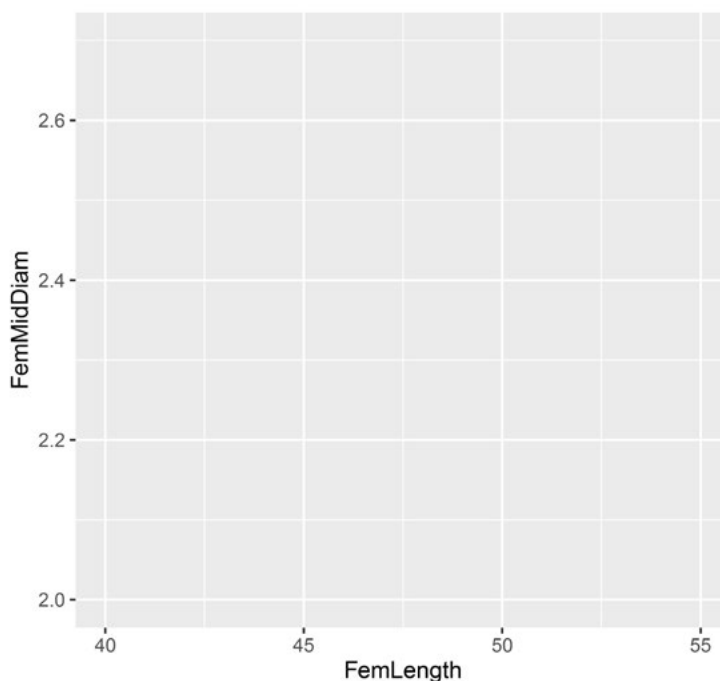
The first command loads the *ggplot2* package and the second command creates a new graph object called *plot*. In fact, we have told ggplot to use the dataframe called *dataplot*, which is the file we have input to R, and we define the variables to be plotted on the x (horizontal) and y (vertical) axes via the *aes()* function.

To see the created object *plot*, we need to type the command:

and press Enter.

We obtain Figure 3.13. It is seen that we do not get the final scatterplot but only the Cartesian coordinate system for the plot of femoral midshaft diameter versus femoral length.

Figure 3.13.
Cartesian
coordinate
system for the
plot of femoral
midshaft
diameter
versus femoral
length



If we want to create the scatterplot of the femoral midshaft diameter versus femoral length, we need to specify the type of plot that we want; in this case, a scatterplot.

Thus, in the command: `plot <- ggplot(dataplot, aes(x=FemLength, y=FemMidDiam))`

we add *geom_point()*: `plot <- ggplot(dataplot, aes(x=FemLength, y=FemMidDiam)) + geom_point()`
`plot`

We obtain the scatterplot of Figure 3.14. It is seen that all points have the same color and size. Therefore, we cannot identify which point belongs to each site and, therefore, we cannot identify trends among the archaeological sites.

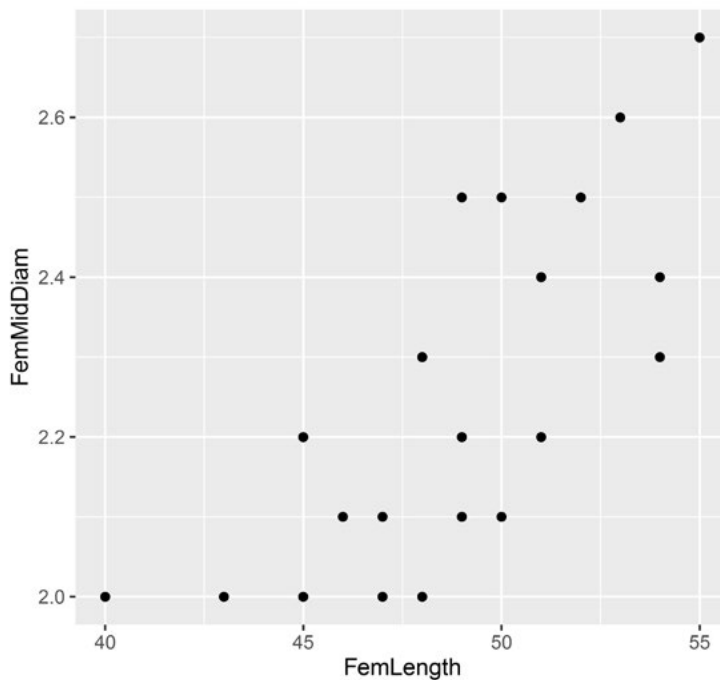


Figure 3.14.
Scatterplot
of femoral
midshaft
diameter
versus femoral
length
irrespective of
archaeological
site

If we want to code the points from each archaeological site using different colors, we convert the variable Site into factor and color the points per site within the `aes()` function using the following commands:

```
dataplot$Site <- factor(dataplot$Site)
splot <- ggplot(dataplot, aes(x=FemLength,
y=FemMidDiam, colour=Site)) + geom_point()
splot
```

We obtain Figure 3.15, which shows three overlapping scatterplots, one for each site.

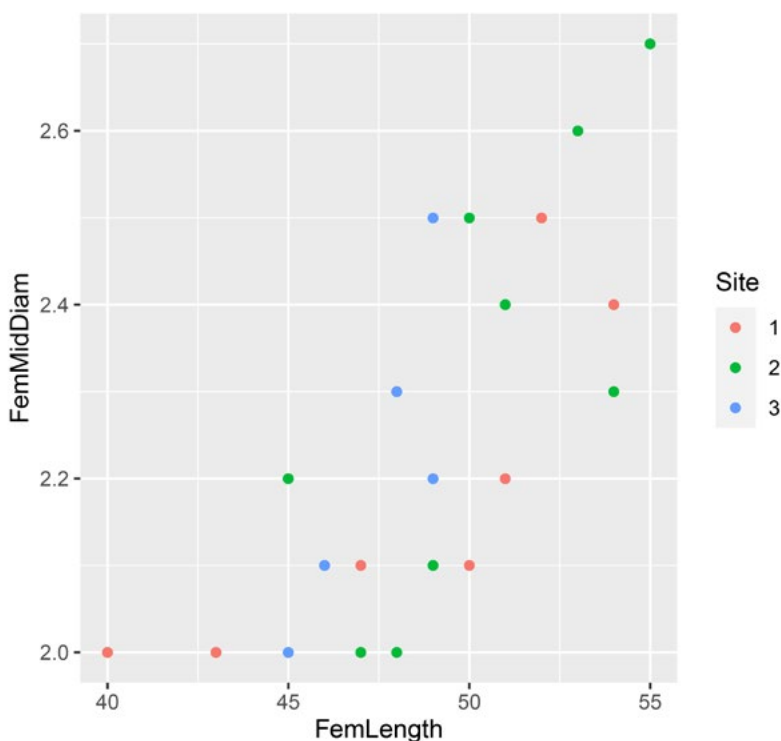


Figure 3.15.
Overlapping
scatterplots
of femoral
midshaft
diameter
versus femoral
length at
three different
archaeological
sites

If we do not like the circles that appear by default, we can change the shape of the points by specifying this for the geom.

For example, executing the following code, we replace the circles by triangles (Figure 3.16):

```
splot <- ggplot(dataplot, aes(x=FemLength, y=FemMidDiam,
colour=Site)) + geom_point(shape=2)
splot
```

If we additionally want to change the size of the triangles, this is done by specifying a value (in mm) for 'size' (Figure 3.17):

```
splot <- ggplot(dataplot, aes(x=FemLength, y=FemMidDiam,
colour=Site)) + geom_point(shape=2, size=4)
splot
```

Figure 3.16.
Change in
the shape of
the points of
Figure 3.15

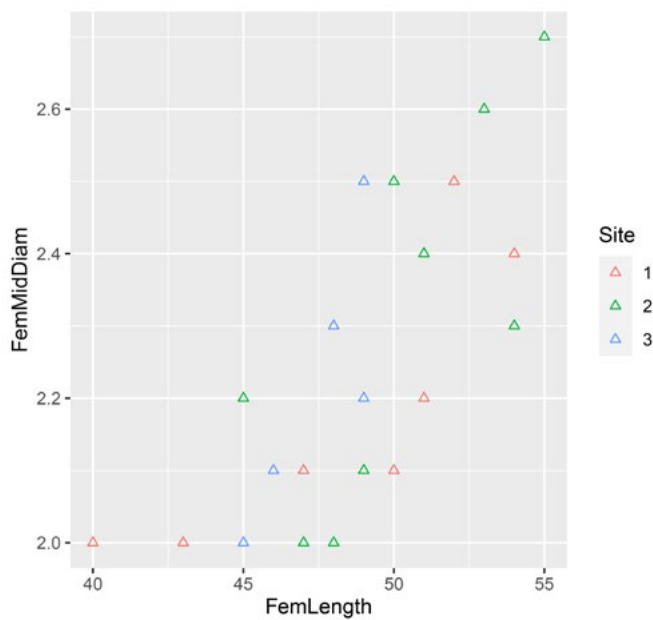
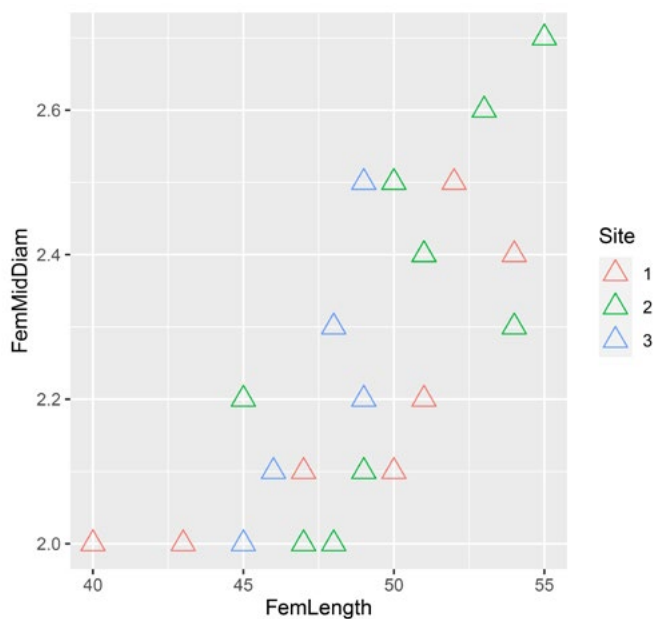


Figure 3.17.
Change in the
size of the
triangles of
Figure 3.16



To add custom labels to the x and y axes of the plot of Figure 3.15, we may use the commands:

```
splot <- ggplot(dataplot, aes(x=FemLength, y=FemMidDiam,
  colour=Site)) + geom_point() +
  xlab("Femoral Maximum Length (in cm)") +
  ylab("Femoral Midshaft Diameter (in cm)")
splot
```

which yield the plot of Figure 3.18:

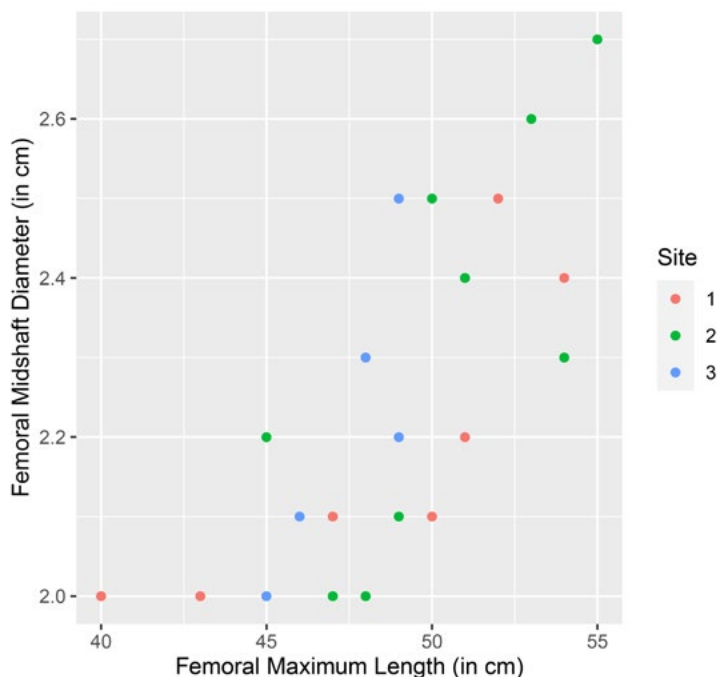


Figure 3.18.
Change in the
axes labels of
Figure 3.15

To add lines in the above plot, we simply add `geom_line()`:

```
splot <- ggplot(dataplot, aes(x=FemLength, y=FemMidDiam,
  colour=Site)) + geom_point() + geom_line() +
  xlab("Femoral Maximum Length (in cm)") +
  ylab("Femoral Midshaft Diameter (in cm)")
splot
```

In Figure 3.19 we observe that the `geom_line` connects the points by straight lines. To plot smooth lines, `geom_line()` should be replaced by `geom_smooth()` and the smoothing method should be specified.

The commands below create the plot of Figure 3.20:

```
splot <- ggplot(dataplot, aes(x=FemLength, y=FemMidDiam,
  colour=Site)) + geom_smooth(method="auto", se=FALSE) +
  geom_point() + xlab("Femoral Maximum Length (in cm)") +
  ylab("Femoral Midshaft Diameter (in cm)")
splot
```

Note that in these commands the argument `se=FALSE` is used to avoid displaying confidence intervals around the smooth lines.

Figure 3.19.
Adding
lines in the
scatterplots
of Figure 3.18

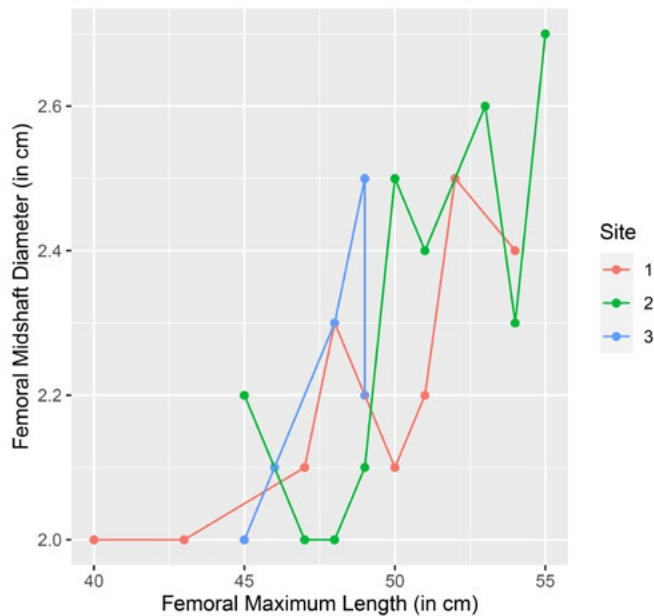
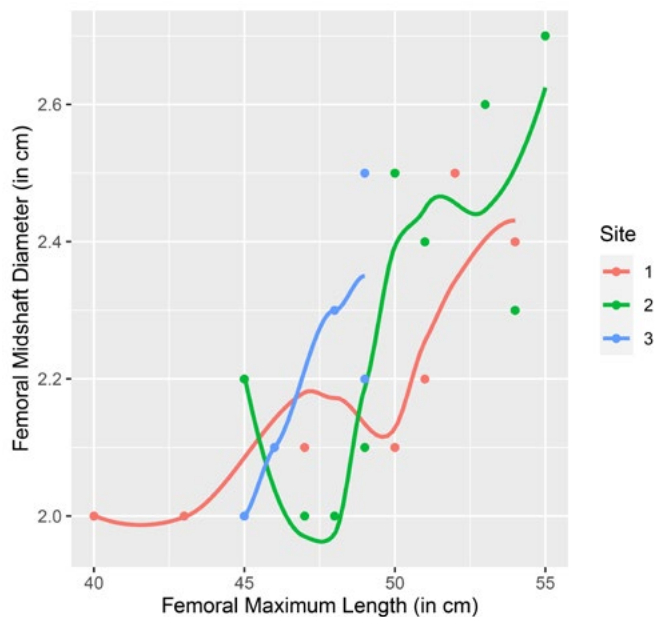


Figure 3.20.
The
scatterplots
of Figure 3.18
with smooth
lines



Histograms

To plot a histogram, we need to use `geom_histogram()`.

Thus, for the histogram of the femoral length values in the sample given in Table 3.14, we can use the commands:

```
histogram <- ggplot(dataplot, aes(x=FemLength)) +  
  geom_histogram(binwidth=0.5) +  
  xlab("Femoral Maximum Length (in cm)") + ylab("Frequency")  
histogram
```

Note that the data we have used is in the file *sampleGraph.txt* and has been input to R using the command `dataplot <- read.table("sampleGraph.txt", head=1, sep="\t")`. Note also that the argument *binwidth* determines the width of each bin in the histogram.

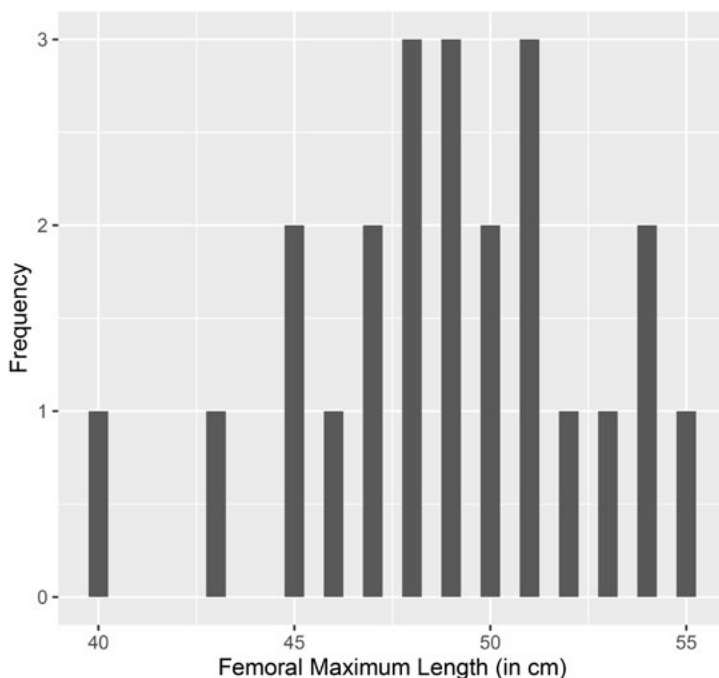


Figure 3.21.
Histogram
of femoral
length values
presented in
Table 3.14

We may use colors and adjust the bin width of the above histogram as follows:

```
histogram <- ggplot(dataplot, aes(x=FemLength,
  color=I("red"), fill=I("blue"), alpha=I(.2))) +
  geom_histogram(binwidth=1) +
  xlab("Femoral Maximum Length (in cm)") +
  ylab("Frequency")

histogram
```

Note that in these commands the function `I(value)` is used to indicate a specific value, avoiding the appearance of a label with this value. The argument *color* defines the color of the bins outline, the *fill* concerns the bin fill color, and *alpha* controls the transparency of the colors. The histogram obtained is shown in Figure 3.22.

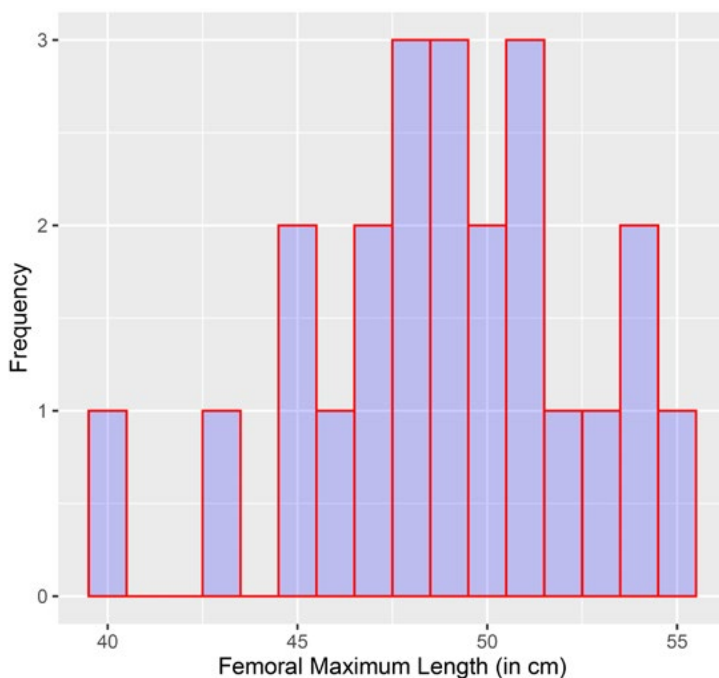


Figure 3.22.
Histogram of
Figure 3.21
with modified
bin width and
color

Consider now the dataset of **Table 3.10**, which is a sample of femoral lengths (in cm) of male and female individuals excavated in a Medieval site. The values of this sample may be pooled and give just one histogram, such as the histogram of [Figure 3.10](#), or we may create a double histogram where male and female values are shown in different color. To plot such histograms, we first arrange the data in an Excel spreadsheet, in two columns, one defining the sex with values 1 for males and 2 for females, and the other defining the femoral lengths. We name these columns Sex and FemLength, respectively, and save the file in the *R_Files* folder as *FemoralLength.txt* tab delimited.

To open this file in R, we may use either the command:

```
FL <- read.table(file.choose(), head=1, sep="\t")
```

or the command:

```
FL <- read.table("FemoralLength.txt", head=1, sep="\t")
```

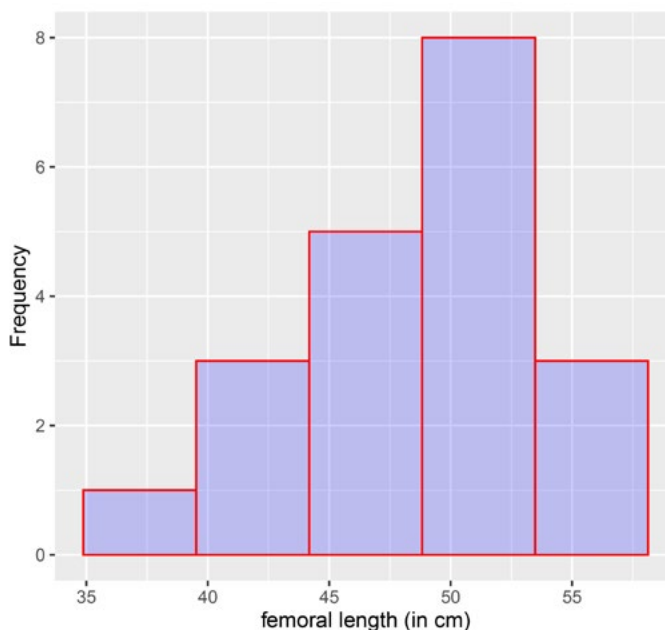
For the histogram of the pooled values, we can use:

```
hist1 <- ggplot(data=FL, aes(x=FemLength, color=I("red"),
fill=I("blue", alpha=I(.2)))) + stat_bin(bins=5) +
xlab("femoral length (in cm)") + ylab("Frequency")

hist1
```

Note that here `stat_bin(bins=5)` is used to set manually the number of bins in ggplot. The obtained histogram is given in [Figure 3.23](#):

Figure 3.23.
Histogram of
the pooled
values of
Table 3.10



To obtain the double histogram composed of the histograms of males and females, we may use the following commands, where the variable Sex is converted into factor and the color of the two histograms is specified within the `aes()` function:

```
FL$Sex <- factor(FL$Sex, levels=c(1,2),
labels=c("males", "females"))
hist2 <- ggplot(data=FL, aes(x=FemLength, fill=Sex, color=
Sex)) +
geom_histogram(binwidth=2, position="identity", alpha=0.5) +
xlab("femoral length (in cm)") +
ylab("Frequency")

hist2
```

The double histogram is shown in Figure 3.24:

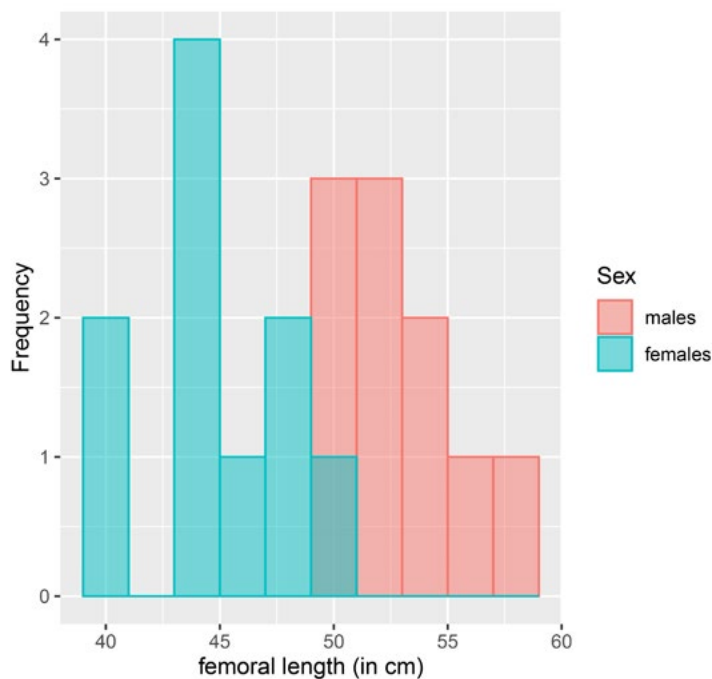


Figure 3.24.
Double
histogram of
the sample of
Table 3.10

Boxplots

For boxplots, we use `geom_boxplot()`.

Thus, for the boxplots of the femoral length values in the sample presented in Table 3.14, we can use the commands:

```
data <- read.table("sampleGraph.txt", head=1, sep="\t")
data$Site <- factor(data$Site)
boxplot <- ggplot(data, aes(x=Site, y=FemLength)) +
  geom_boxplot() + xlab("Archaeological site") +
  ylab("Femoral maximum length (in cm)")
boxplot
```

We obtain Figure 3.25.

To add colors, we may use the commands:

```
boxplot <- ggplot(data, aes(x=Site, y=FemLength, fill=Site)) +
  geom_boxplot() + xlab("Archaeological site") +
  ylab("Femoral maximum length (in cm)") +
  theme(legend.position='none')
boxplot
```

and obtain Figure 3.26.

Similar commands can be used for the boxplots of Figure 3.12. Since the data are in the file *FemoralLength.txt*, we can use the following commands, which yield Figure 3.27:

```
FemoralLength <- read.table("FemoralLength.txt", head=1,
  sep="\t")
FemoralLength$Sex <- factor(FemoralLength$Sex,
  levels=c(1,2), labels= c("males", "females"))
bp <- ggplot(FemoralLength, aes(x=Sex, y=FemLength,
  fill=Sex)) +
  geom_boxplot(alpha=0.5) + xlab("sex") +
  ylab("Femoral length (in cm)") +
  theme(legend.position='none')
bp
```

Figure 3.25.
Boxplots for
the dataset of
Table 3.14

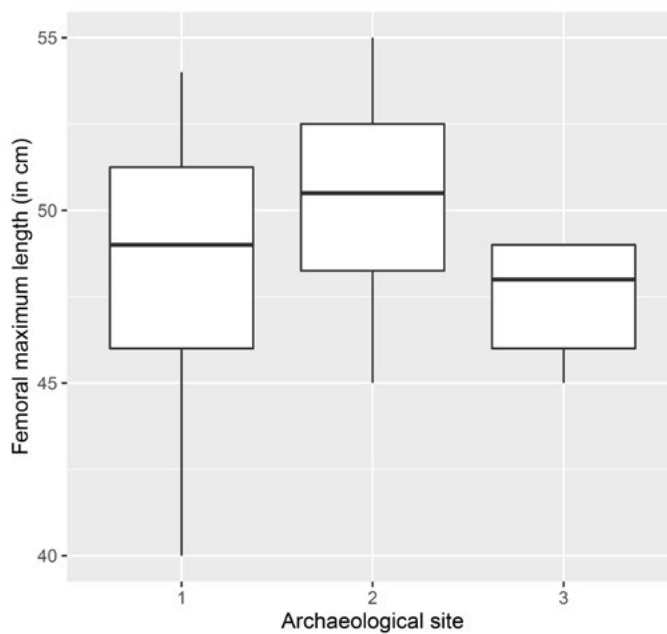
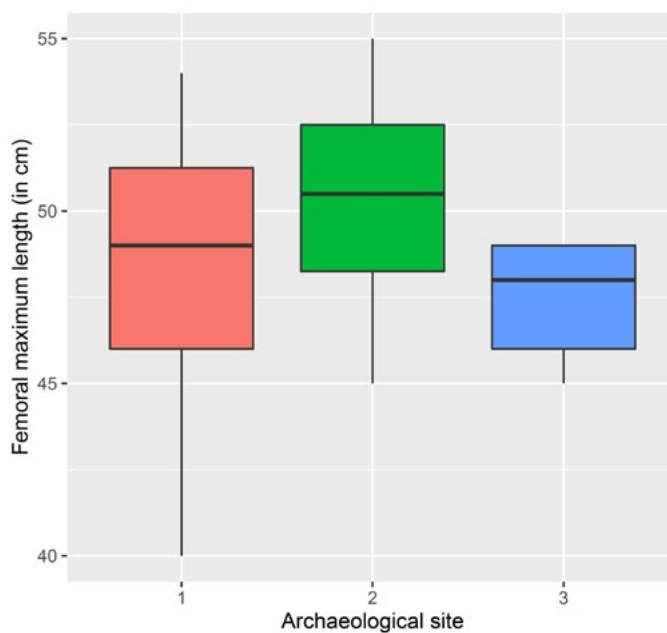


Figure 3.26.
As in Figure
3.25 with
different color
per site



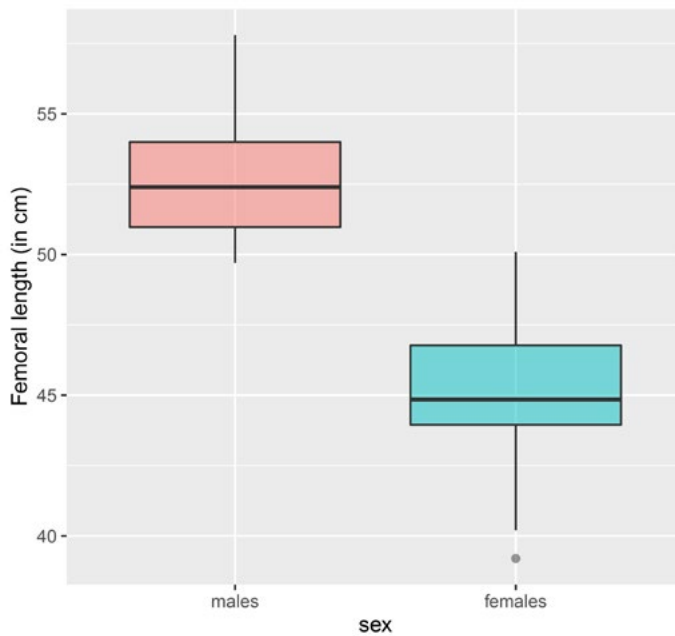


Figure 3.27.
Boxplots of
Figure 3.12

Bar and pie plots

To plot bar charts, we should use `geom_bar()`. In addition, we should use the argument `stat="identity"`, which implies that you will not perform any statistical transformation in your y data.

The following commands create the bar chart of Figure 3.28 (see also Figure 3.6):

```
typology <- read.table("typologies.txt", head=1, sep="\t")
barplot <- ggplot(typology, aes(x=CeramicTypology,
y=Frequency, color=l("red"), fill=l("blue"), alpha=l(.2))) +
geom_bar(stat="identity") + xlab("typology")

barplot
```

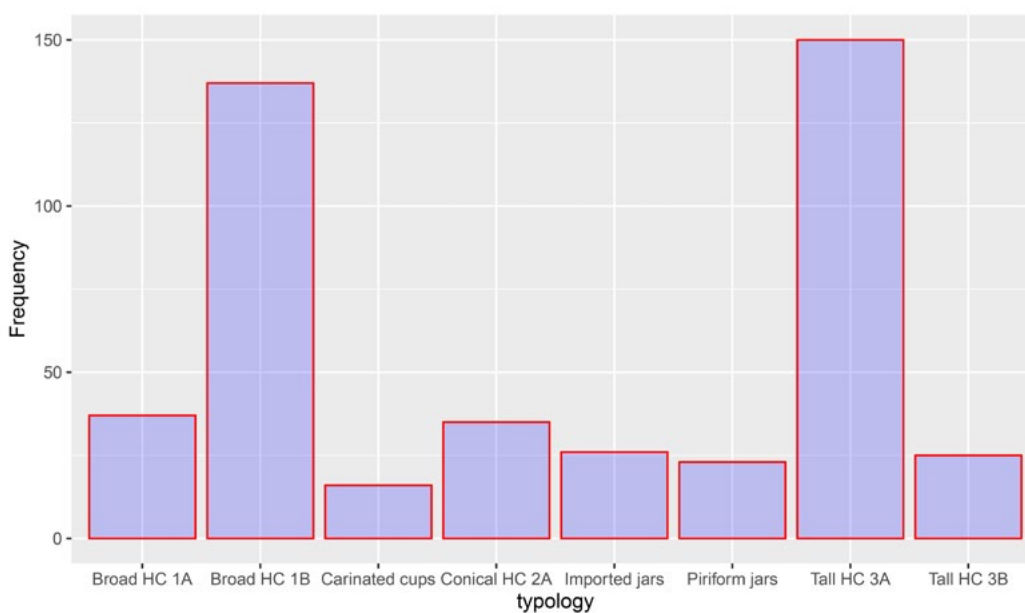


Figure 3.28.
Bar chart with
the frequency
of the Minoan
ceramic
typologies
given in Table
3.5.

For stacked barplots with multiple groups, we should use the *fill* argument, as in the following commands, which yield Figure 3.29 (see also Figure 3.8):

```
PMTreatment <- read.table("PostMortemTreatment.txt",
  head=1, sep="\t")
```

```
bp1 <- ggplot(PMTreatment, aes(x=Sex, y=Frequency,
  fill=PostMortemTreatment)) +
  geom_bar(stat="identity") + xlab("sex")
```

```
bp1
```

For clustered bar charts, we should use the argument *position=position_dodge()*, as in the commands:

```
bp2 <- ggplot(data=PMTreatment, aes(x=Sex,
  y=Frequency, fill=PostMortemTreatment)) +
  geom_bar(stat="identity", position=position_dodge()) +
  xlab("sex")
```

```
bp2
```

We obtain Figure 3.30 (see also Figure 3.7).

Figure 3.29.
Stacked bar
chart of
post-mortem
treatment
per sex for
the data of
contingency
Table 3.5

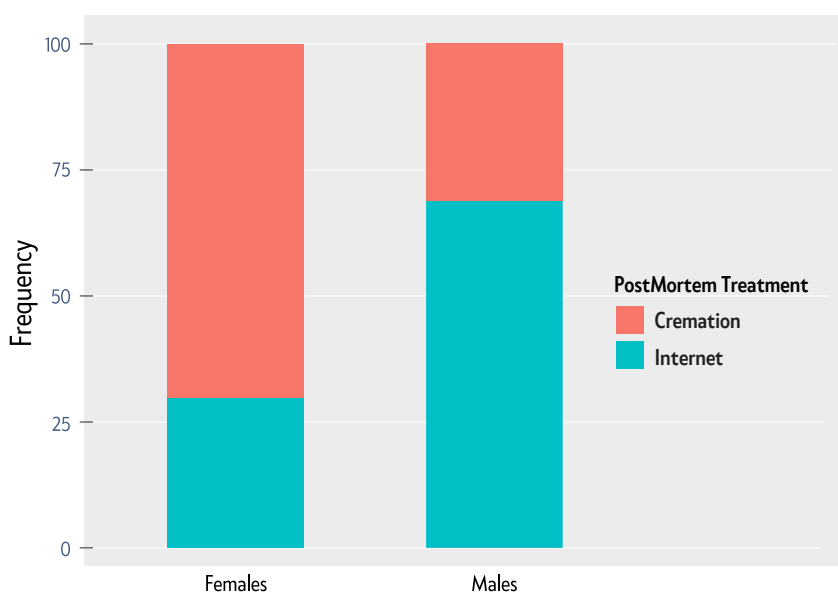
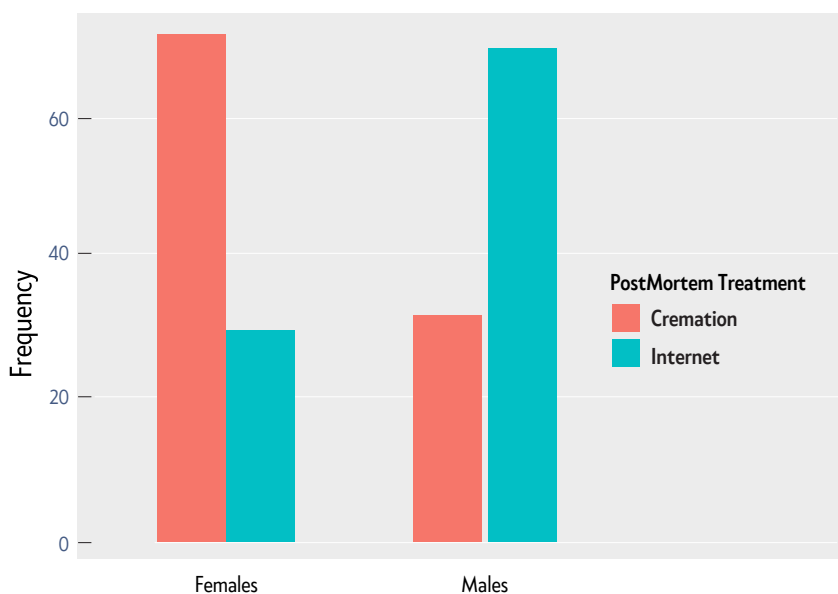


Figure 3.30.
Clustered
bar chart of
post-mortem
treatment
per sex for
the data of
contingency
Table 3.5



For a pie chart, we first create a basic bar plot and then convert it to a pie and add labels. If we use the following commands, we obtain the pie chart of Figure 3.31:

```
artifacts <- read.table("pie.txt", head=1, sep="\t")
pie <- ggplot(artifacts, aes(x="", y=Percentage, fill=
ArtifactCategory)) + geom_bar(stat = "identity", width=1)
pie <- pie + coord_polar("y", start=0)
pie <- pie + geom_text(aes(label=paste0(round(Percentage),
"%")), position = position_stack(vjust=0.5))
pie <- pie + xlab("") + ylab("")
pie
```

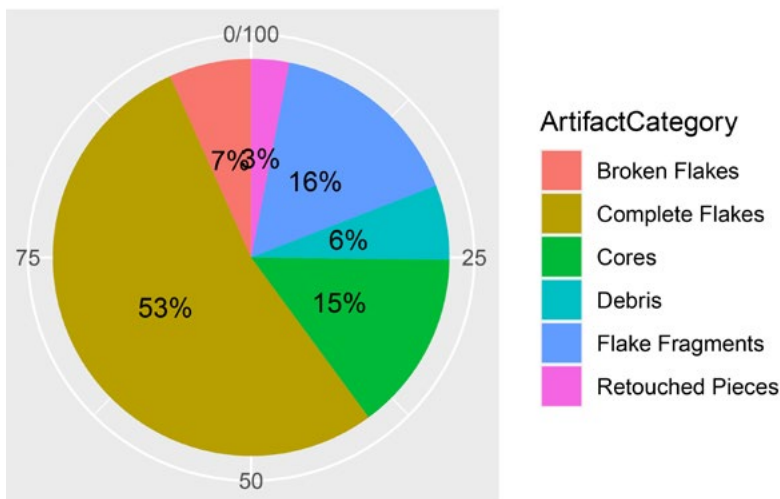


Figure 3.31.
Pie chart for
the data of
Table 3.9

SAVING GRAPHS

Once we have created our graph, we can save it as TIFF (or other format) by clicking on it and then going *File* → *Save as*, and selecting the format of our choice from the drop down menu that appears.

4.

INFERENCE STATISTICS

Inferential statistics employs probability theory for deducing the properties of populations from observations and analyses of samples derived from these populations. In other words, we use *sample statistics* in order to estimate *population parameters*.

There are two broad approaches to estimate population parameters from samples: point and interval estimation. In *point estimation*, we calculate a certain descriptive measure, for example the mean value, and use it as the "best estimate" of the corresponding unknown population parameter, for example the population mean value. In *interval estimation*, we determine a *confidence interval* for the parameter under study, i.e. a range of values from a to b, which includes the unknown population parameter with a given probability. The confidence intervals are closely related to *hypothesis testing*. Hypothesis testing refers to the statistical methods used to accept or reject *statistical hypotheses*, i.e. assumptions about population parameters. Thus, inferential statistics involves point and confidence interval estimation as well as statistical hypothesis testing.

Since the goal of inferential statistics is to use the sample to get information about the population, the sample should be an unbiased representation of the population. This unbiased representation is achieved by using a *random sample*. As has already been pointed out, this is a sample in which every member of the population from which the sample originates has the same probability of being included. Thus, a random sample consists of cases randomly selected from the population. However, despite this randomness, the sample values as well as the population values, follow specific distributions. These distributions have properties that allow making inferences about the population based on sample data.

4.1. PROBABILITY DISTRIBUTIONS

As discussed in section 3.5, histograms offer the most effective graphical display of the frequency distribution of a set of continuous data, provided that the sample size is relatively large. If the sample size is small, the shape of the histogram may be strongly affected by the number of bins, resulting in false conclusions.

Figure 4.1 shows the histogram of some artificial data on the capacity of 139 Roman era amphorae.

It has been obtained using the following commands:

```
ac <- read.table("AmphoraeCapacity.txt", head=1, sep="\t")
library(ggplot2)

hist1 <- ggplot(data=ac, aes(x=Amphora_capacity,
color=l("red"), fill=l("blue"), alpha=l(.2))) +
geom_histogram(binwidth=5) + xlab("capacity (in litres)") +
ylab("Frequency")

hist1
```

We observe that the distribution of the values in this sample is not symmetric around a central value (mean or median) and that it is skewed right. Most amphorae have a capacity between about 20 and 25 litres (the highest bars), whereas very few amphorae have a capacity greater than 50 litres.

A histogram can acquire interesting properties if we alter the y axis from *Frequency* to *Density*, where the density is defined as:

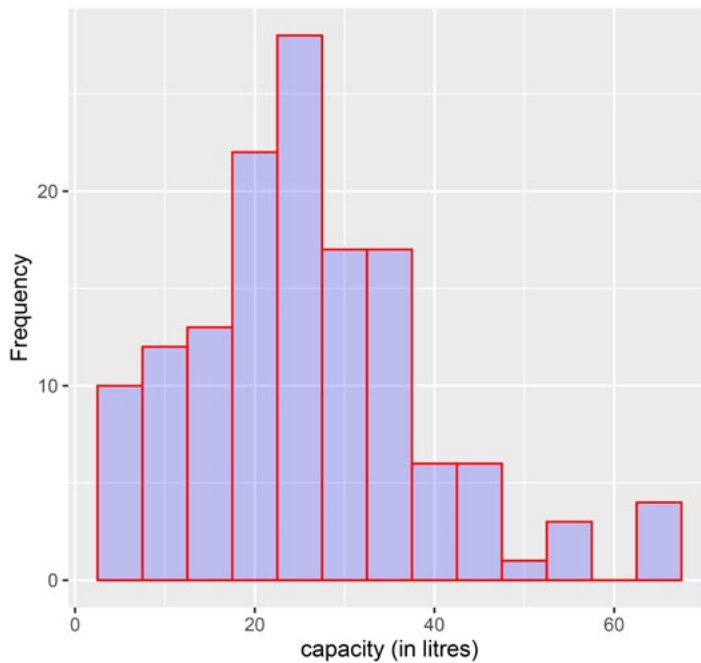


Figure 4.1.
Histogram
of amphorae
capacity (in
litres)

$$\text{Density} = \text{Frequency} / (n \cdot dx)$$

where n is the sample size and dx the bin width. For example, in Figure 4.1 $n = 139$ and $dx = 5$. Therefore, in order to obtain the density histogram, we should divide the values in the histogram of Figure 4.1 by $139 \times 5 = 695$. We obtain the histogram of Figure 4.2. Note that the histogram of Figure 4.2 shows also the density curve, which is a smoothed version of the histogram.

In fact, Figure 4.2 has been created via the following commands:

```
hist2 <- ggplot(data=ac, aes(x=Amphora_capacity,
color=l("red"), fill=l("blue"), y = ..density..)) +
  xlab("capacity (in litres)") + geom_histogram(aes(alpha=l(.2)),
  binwidth=5, position="identity") +
  stat_density(aes(color=l("blue")), geom="line", size=1.2)
```

hist2

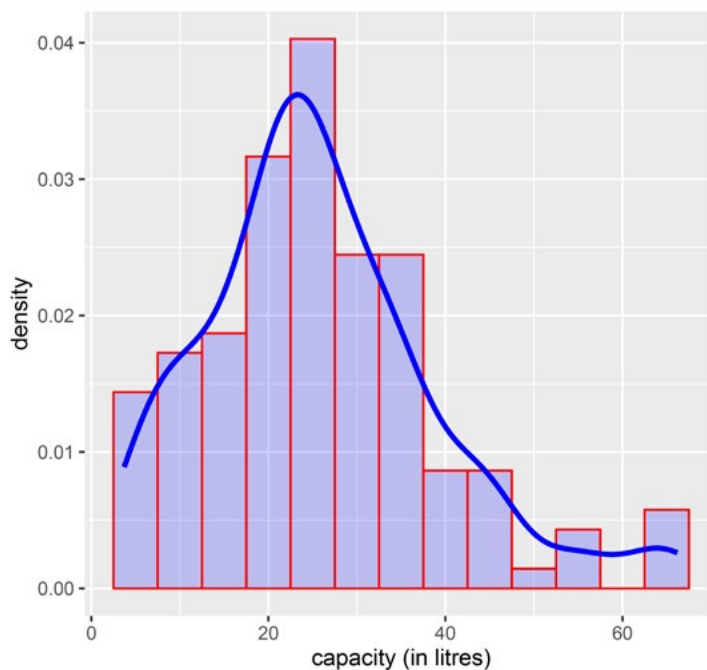


Figure 4.2.
Density
histogram and
density curve
for amphorae
capacity (in
litres)

When we replace the frequency by the density, the area of the rectangles that form the histogram becomes equal to 1 and, in addition, the area of the histogram between two values, a and b , of the x axis is equal to the probability of the variable of the histogram being between a and b . For example, from the plots of Figures 4.1 and 4.2 we obtain that the probability of finding an amphora in the sample with capacity from 17.5 to 22.5 litres is equal to $22/695=0.032$, whereas this probability becomes $(22+28)/695=0.072$ for an amphora with capacity from 17.5 to 27.5 litres. Thus, by altering the y axis of a histogram from frequency to density, we change the graphical display from a frequency distribution to a probability distribution.

From Figure 4.2 we also obtain that a density histogram can be effectively replaced by its density curve as the area under this curve is equal to 1. In this case, the area under the density curve between two values, a and b , of the x axis is equal to the probability of the variable of the histogram being between a and b . Such a curve is called *probability density curve*.

BE CAREFUL

The density histogram, as well as the probability density curve, can be used to estimate the probability within a certain range of x values and not at a certain x value, when x is a continuous variable. For example, in Figure 4.2 we observe that $x=20$ litres corresponds to the value density = 0.032. However, this does not entail that the probability of finding an amphora in the sample with capacity exactly equal to 20 litres is 0.032 because in strict mathematical language “exactly equal to 20 litres” means the capacity is 20.00000...0 and the probability in this case is equal to 0.

The experimental and theoretical study of histograms has shown that when the samples are large enough, the probability density curve is described by certain mathematical functions. Such a function is simply called *probability distribution* or even simpler, *distribution*. There are many different distributions, the most important of which are the following:

- Normal distribution
- Log-normal distribution
- Student or t distribution
- Chi-square distribution
- Fisher or F distribution
- Binomial distribution
- Poisson distribution

The first five distributions concern continuous variables, whereas the last two discrete variables.

Normal distribution

The normal distribution is the most common distribution. It is denoted by $N(\mu, \sigma)$ and is a very basic distribution since many statistical tests are only valid under the assumption that the samples involved in these tests come from normal populations, i.e. populations with values that are distributed according to the normal distribution. If we study a variable that follows the normal distribution, the following properties apply under the condition that the sample is relatively large:

- The histogram is symmetric, bell-shaped
- The mean and median are the same value and lie at the peak of the distribution
- The tails of the distribution never touch the X -axis; they come incrementally ever closer to zero but without reaching it
- The majority of values lie around the centre of the distribution, while as we get further away from the centre, the frequency of cases decreases. In specific:
 - 68.26% of the sample values lie within one standard deviation from the mean (mean ± 1 standard deviation).

- 95.44% of the sample values lie within two standard deviations from the mean (mean ± 2 standard deviations) or 95% of the sample values lie within 1.96 standard deviations from the mean (mean ± 1.96 standard deviations).
- 99.73% of the sample values lie within three standard deviations from the mean (mean ± 3 standard deviations) or 99% of the sample values lie within 2.58 standard deviations from the mean (mean ± 2.58 standard deviations).

Examples of normal distributions are shown in Figures 4.3 to 4.5. Figure 4.3 shows the density histogram and the corresponding probability density curve of a simulated sample of 100,000 values that follow the normal distribution with mean = 5 and standard deviation = 2.

It has been created using the commands:

```
library(ggplot2)

sn <- data.frame(PF=rnorm(100000, mean=5, sd=2))
nplot <- ggplot(sn, aes(x=PF)) + geom_histogram(aes
(y = ..density..), colour="black", fill="white") +
stat_function(fun=dnorm, args=list(mean=mean(sn$PF),
sd=sd(sn$PF)), size=1)

nplot
```

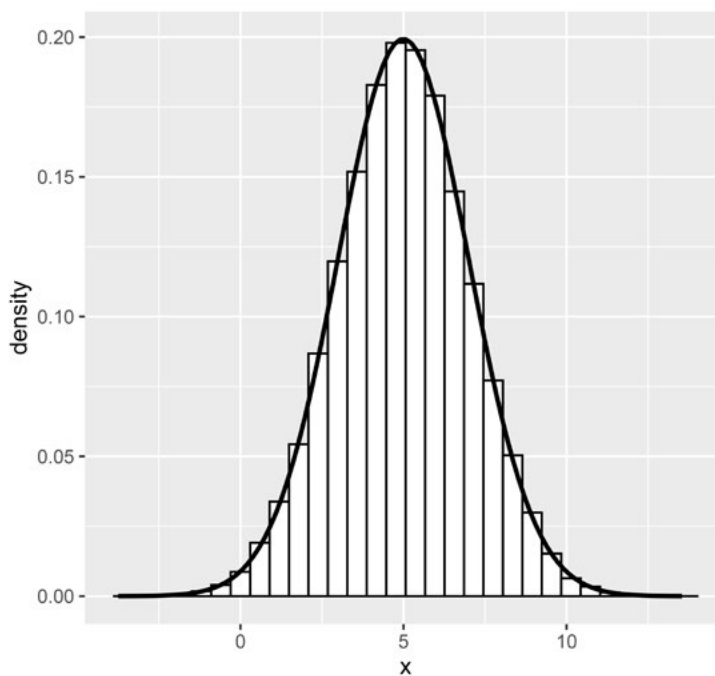


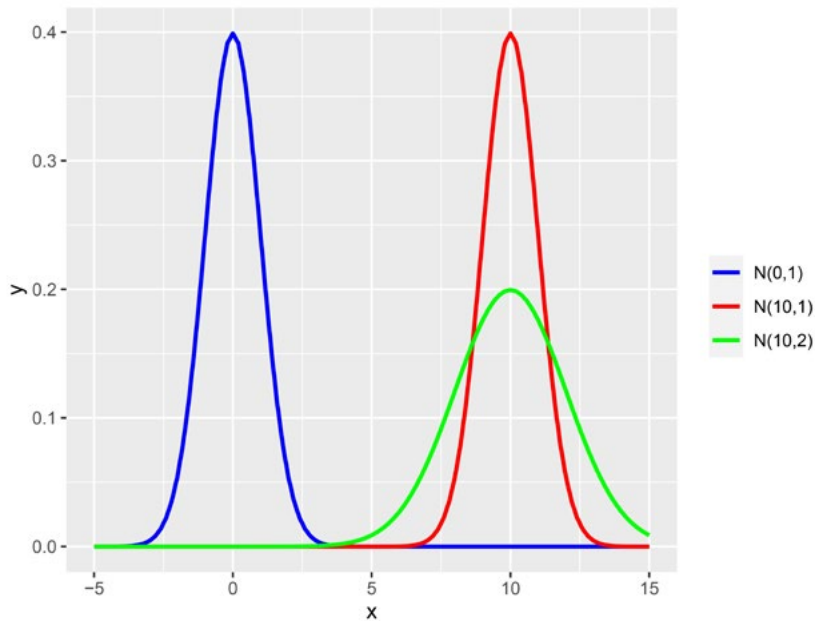
Figure 4.3.
Density
histogram and
probability
density curve
of the normal
distribution
with mean=5
and standard
deviation=2

Figure 4.4 depicts three normal distribution curves for different means and standard deviations created via the commands:

```
nplot2 <- ggplot(data.frame(x=c(-5, 15)), aes(x=x)) +
stat_function(fun=dnorm, args=list(0, 1), aes(colour="N(0,1)"),
size=1) + stat_function(fun=dnorm, args=list(10, 1),
aes(colour="N(10,1)"), size=1) + stat_function(fun=dnorm,
args=list(10, 2), aes(colour="N(10,2)"), size=1) +
scale_colour_manual("", values=c("blue", "red", "green"))

nplot2
```

Figure 4.4.
Normal distribution curves for different means and standard deviations



Finally, Figure 4.5 shows an interesting limiting case of the normal distribution where the mean is equal to 0 and the standard deviation is 1. This is called the *standard normal distribution* or *the z distribution*. Note that according to the properties of the normal distribution, 68.26% of the sample values lie within -1 and 1, 95% of the sample values lie within -1.96 and 1.96, and 99% of the sample values lie within -2.58 and 2.58.

Figure 4.5 has been drawn using the commands:

```
Shaded <- function(x) {
  y <- dnorm(x, mean = 0, sd = 1)
  y[x < -1 | x > 1] <- NA
  return(y)}
splot1 <- ggplot(data.frame(x = c(-4, 4)), aes(x = x)) +
  stat_function(fun = dnorm, color="blue",
  size=1) + stat_function(fun=Shaded, geom="area",
  fill="#84CA72", alpha=0.5, size=1)+
  geom_hline(yintercept=0) + annotate("text", x = 0, y = 0.2,
  label = "68.3%")

splot1

Shaded <- function(x) {
  y <- dnorm(x, mean = 0, sd = 1)
  y[x < -1.96 | x > 1.96] <- NA
  return(y)}
splot2 <- ggplot(data.frame(x = c(-4, 4)), aes(x = x)) +
  stat_function(fun = dnorm, color="blue", size=1) +
  stat_function(fun=Shaded, geom="area", fill="#84CA72",
  alpha=0.5, size=1) + geom_hline(yintercept=0,size=1) +
  annotate("text", x = 0, y = 0.2, label = "95%")

splot2
```

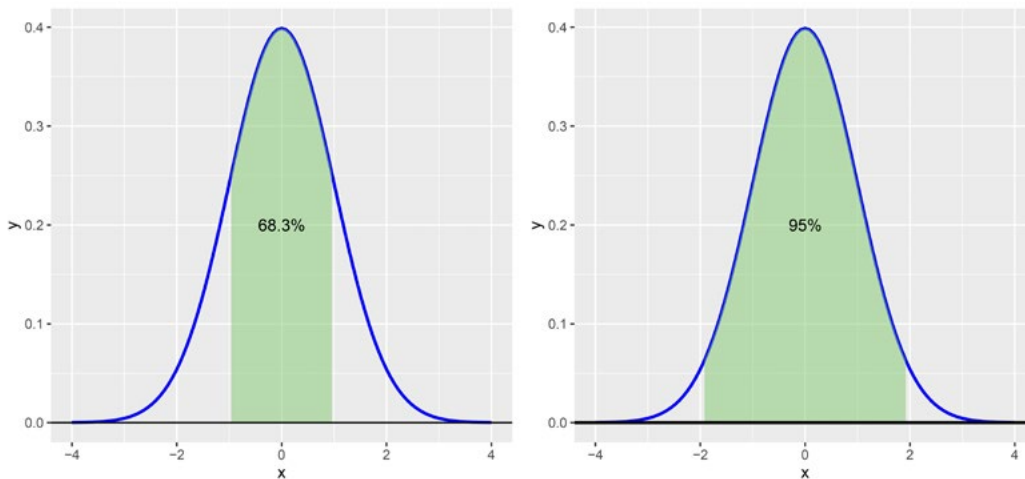


Figure 4.5.
Standard
normal
distribution
curve

Although the standard normal distribution is a specific case of the normal distribution, any dataset can be converted into a dataset that has a mean of 0 and a standard deviation of 1. For example, consider a dataset that has a mean of 28.5 and a standard deviation of 3.18. To transform this sample to a sample with 0 mean and 1 standard deviation, first we centre the data around zero by subtracting from each sample value the mean of all values ($\bar{x} = 28.5$). Then, we divide the resulting score by the standard deviation ($s = 3.18$) to ensure the data have a standard deviation of 1. The resulting values are known as *z-scores* and follow the standard normal distribution, provided that the original sample, i.e. the x values, follow the normal distribution.

$$z = \frac{x - \bar{x}}{s}$$

Log-normal distribution

When there are deviations from the normal distribution, the data may follow the log-normal distribution. A sample comes from a log-normal population if the logarithms of its values follow the normal distribution. An example is shown in Figure 4.6. On the left, this figure depicts the density histogram and the corresponding probability density curve of a simulated sample of 100,000 values that follow the log-normal distribution with mean = 2 and standard deviation = 0.3. On the right, it depicts the density histogram and the corresponding probability density curve of the logarithms of the sample values.

Figure 4.6 has been created using the commands:

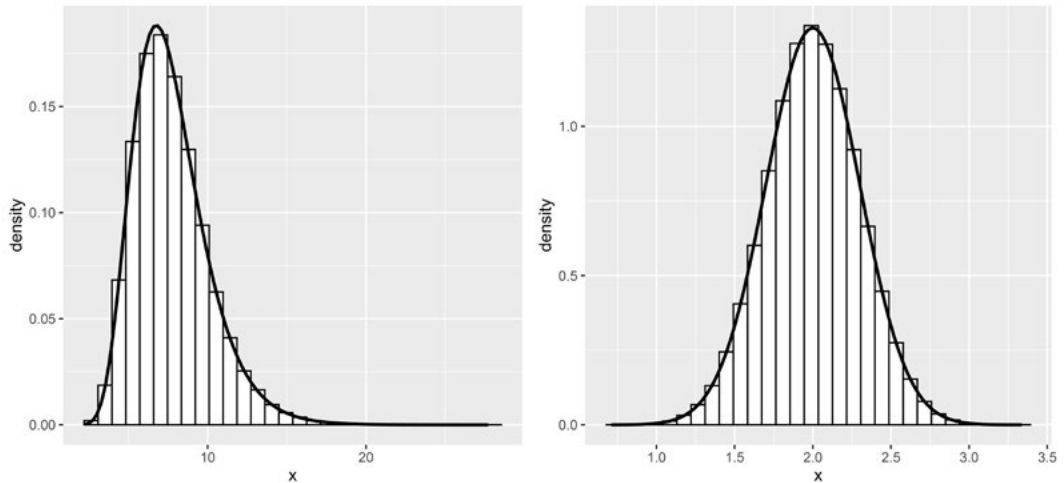
```
sn <- data.frame(PF=rlnorm(100000, mean=2, sd=0.3))
lnplot1 <- ggplot(sn, aes(x=PF)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  stat_function(fun=dlnorm, args=list(2, 0.3), size=1) + xlab("x")

lnplot1

sn <- data.frame(PF=log(rlnorm(100000, mean=2, sd=0.3)))
lnplot2 <- ggplot(sn, aes(x=PF)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  stat_function(fun=dnorm, args=list(2, 0.3), size=1) + xlab("x")

lnplot2
```

Figure 4.6.
Log-normal distribution (left) and the corresponding normal distribution of its logarithmic values (right) when mean=2 and standard deviation=0.3



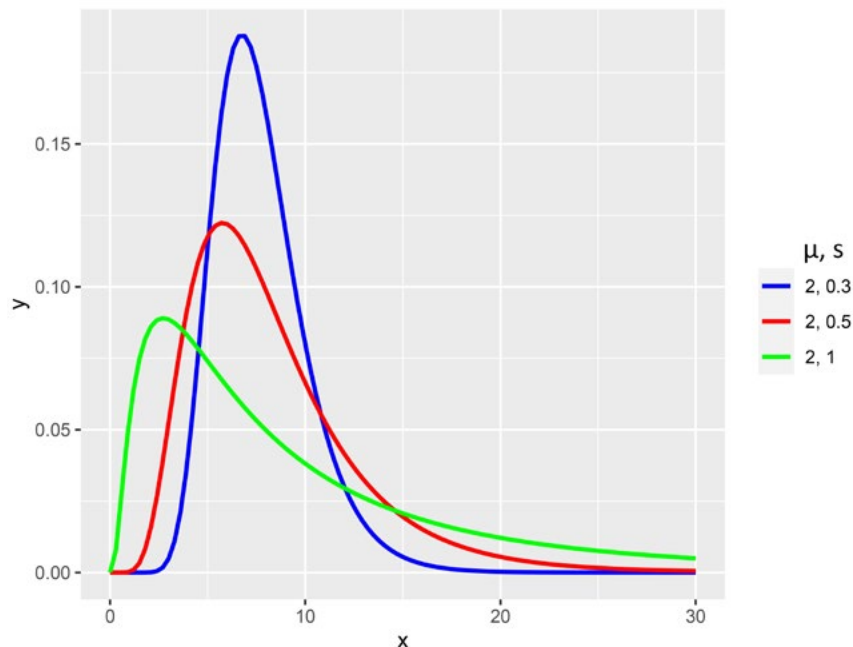
It is seen that the log-normal distribution is not symmetric around a central value and it is skewed right, whereas its logarithmic values are normally distributed. Examples of log-normal distributions with the same mean but different standard deviations are shown in Figure 4.7.

They have been drawn using the commands:

```
Inplot <- ggplot(data.frame(x=c(0, 30)), aes(x=x)) +
  stat_function(fun=dlnorm, args=list(2, 0.3), aes(colour="2,
  0.3"),size=1) + stat_function(fun=dlnorm, args=list(2, 0.5),
  aes(colour="2, 0.5"), size=1) + stat_function(fun=dlnorm,
  args=list(2, 1), aes(colour="2, 1"), size=1) +
  scale_colour_manual("μ, σ", values=c("blue", "red", "green"))

Inplot
```

Figure 4.7.
Log-normal distribution curves with mean=2 and different standard deviations



Student, Chi-square and Fisher distributions

The distributions Student, Chi-square, and Fisher are artificial distributions, i.e. they do not appear in real samples but in samples that are created after various mathematical transformations of the values of real samples.

The shape of these distributions is shown in Figures 4.8-4.10, which have been drawn using the following code:

```
# Student distribution
stplot <- ggplot(data.frame(x=c(-4, 4)), aes(x=x)) +
  stat_function(fun=dt, args=list(df=1), aes(colour="df=1"),size=1) +
  stat_function(fun=dt, args=list(df=2), aes(colour="df=2"),
  size=1) + stat_function(fun=dt, args=list(df=100),
  aes(colour="df=100"), size=1) + scale_colour_manual("", values
  = c("blue", "red", "green"))
stplot <- stplot+ geom_hline(yintercept=0)
stplot

# Chi-square distribution
chplot <- ggplot(data.frame(x=c(0, 20)), aes(x=x)) +
  stat_function(fun=dchisq, args=list(df=3), aes(colour="
  df=3"),size=1) + stat_function(fun=dchisq, args=list(df=5),
  aes(colour="df=5"), size=1) + stat_function(fun=dchisq,
  args=list(df=10), aes(colour="df=10"), size=1) +
  scale_colour_manual("", values=c("blue", "red", "green"))
chplot <- chplot+ geom_hline(yintercept=0)
chplot

# Fisher distribution
fplot <- ggplot(data.frame(x=c(0, 4)), aes(x=x)) +
  stat_function(fun=df, args=list(df1=10, df2=5),
  aes(colour="df(10,5)"), size=1) + stat_function(fun=df,
  args=list(df1=10, df2=10), aes(colour="df(10,10)"),size=1) +
  stat_function(fun=df, args=list(df1=15, df2=15),aes(colour
  ="df(15,15)"),size=1) + scale_colour_manual("", values=c("blue",
  "red", "green"))
fplot <- fplot+ geom_hline(yintercept=0)
fplot
```

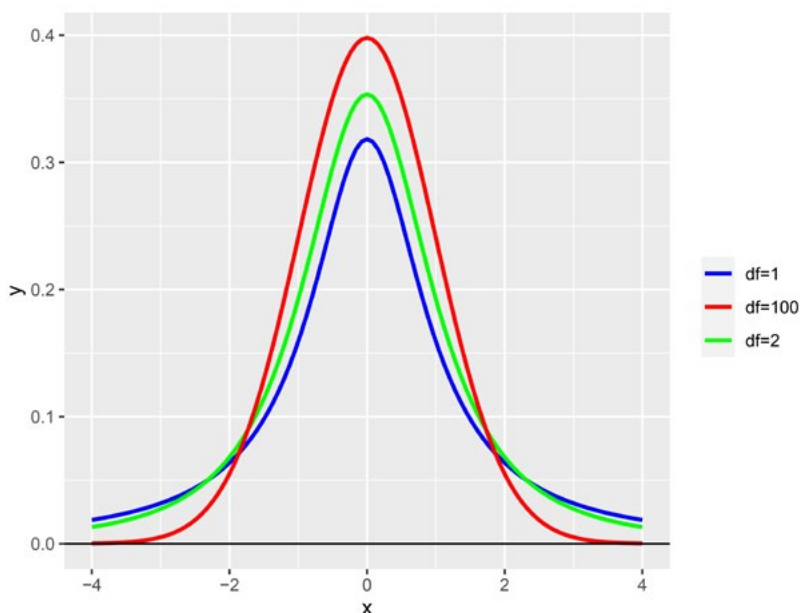


Figure 4.8.
Student
distribution
curves for
different
degrees of
freedom (df)

Figure 4.9.
Chi-square
distribution
curves for
different
degrees of
freedom (df)

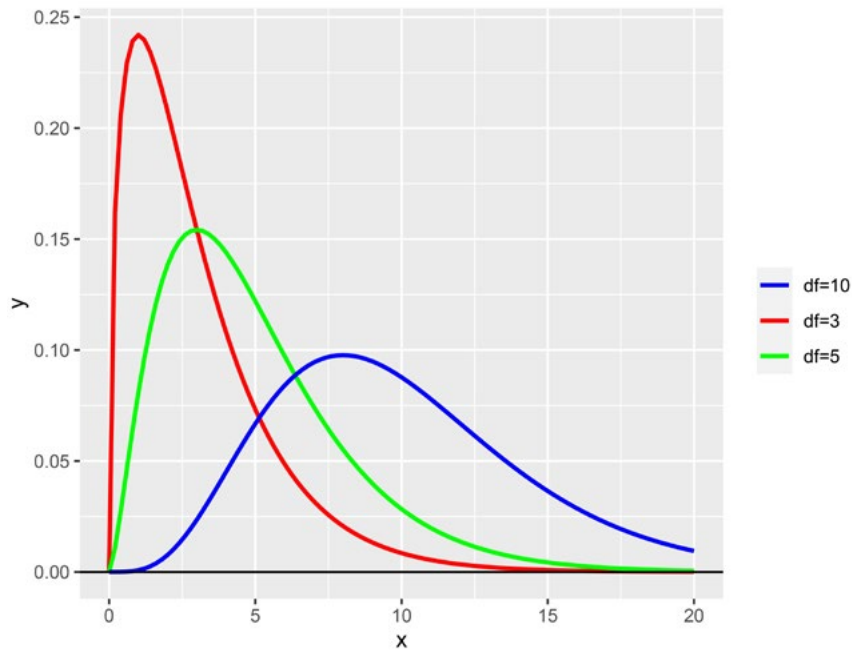
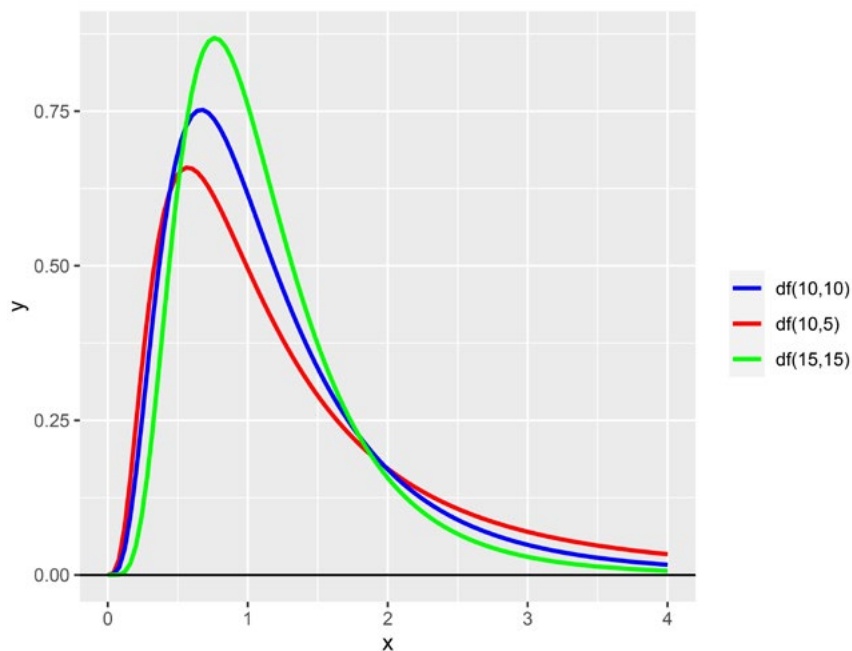


Figure 4.10.
Fisher
distribution
curves for
different
degrees of
freedom (df1,
df2)



DEGREES OF FREEDOM

The degrees of freedom, df , are the number of values (observations) in a sample that are free to vary when computing statistical parameters. For example, consider a sample of five values, 4, 5, 3, 7, 6 and our aim is to estimate the sample variance. This estimation presumes the estimation of the sample mean. In this example, the mean value is equal to 5 and this value should be used for the calculation of s^2 . In the original sample we are free to vary all five observations. But if the target is to estimate the sample variance, we need first to calculate the mean value. Since the mean value of the original sample is 5, then we should assume that the population mean is also 5 and we should keep this value constant. Now, we are not free to vary all five observations but only four of them. For example, if we alter the first four to 5, 5, 8, 2, the fifth observation is necessarily equal to 5 to give a mean value equal to 5. Consequently, the degrees of freedom are 4. In general, if we have a sample of N observations and need to calculate the variance or standard deviation, the degrees of freedom are equal to $N - 1$.

Binomial distribution

When we study a discrete variable that takes only two possible values, then the data follows the *binomial distribution*. An example of such a variable is the existence of burnishing in certain pots, which may take the values 'present' or 'absent'. Another example would be the sex of individuals assessed from their skeletal remains, which may take the values 'male' or 'female'. The probability of the data falling into one category is denoted as p , while the probability of the data falling to the second category is $1-p$. The binomial distribution gives the probability that the discrete variable will take exactly x times one of its values in n observations. The distribution is denoted by $B(n, p)$.

As an example, consider the nonmetric trait *cusp 7* in the mandibular first molar, which may be present or absent. In an ancient cemetery this trait was recorded with probability $p = 0.2$. In a certain tomb cluster of the cemetery this trait was present in five individuals ($x = 5$) among the nine individuals that comprised the cluster ($n = 9$). To estimate the probability of finding exactly five individuals with the trait under study is a limiting case of estimating the probability of finding exactly $x = 0, 1, 2, \dots, 9$ individuals with the trait *cusp 7* among the nine individuals, which is given by the binomial distribution, $B(9, 0.2)$.

This distribution is shown in Figure 4.11, created using the commands:

```
bplot <- ggplot(data.frame(x=0:9), aes(x=x)) +
  stat_function(fun=dbinom, args=list(9, 0.2), n=10,
    aes(colour=l("red")), size=2, geom="point")
bplot
```

Note that the probability of finding exactly five individuals with the trait under study is very small, approximately equal to 0.017, and it is estimated using the R command:

```
dbinom(5, 9, 0.2)
```

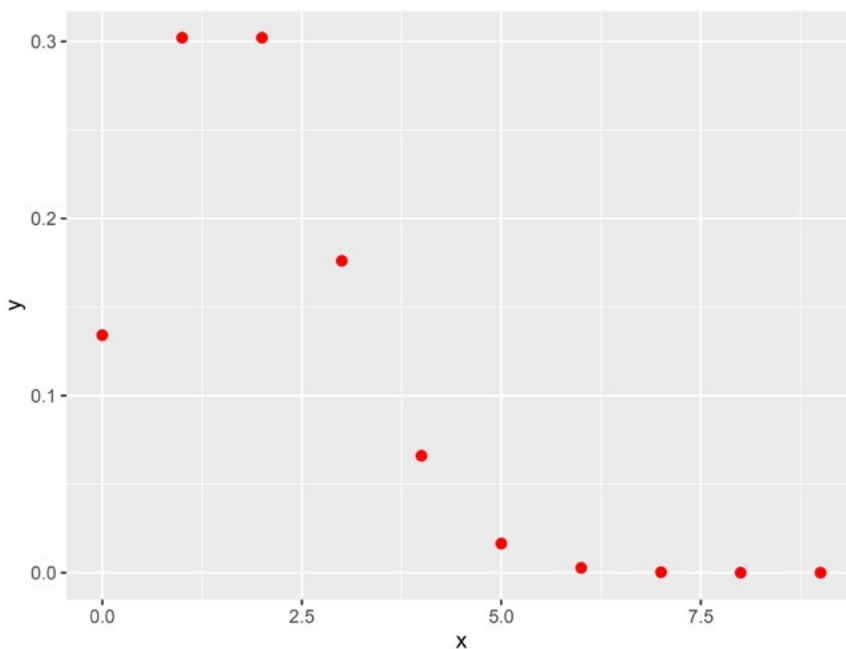


Figure 4.11.
Binomial
distribution
curve $B(9, 0.2)$

Poisson distribution

The *Poisson distribution* concerns systems with a large number of possible events, each of which is rather rare. In archaeology the Poisson distribution is usually adopted to calculate the probability of the number of events (e.g., artifacts) found within a given spatial unit. This probability depends upon λ , a positive real number, which is equal to the expected number of events, i.e. the average number of events occurring per space. Therefore, the Poisson distribution depends only on λ and, for this reason, it is denoted by $\text{Pois}(\lambda)$.

For example, consider that 20 pots were found in 10 square meters of excavated land. In this case λ is $20/10 = 2$ and the distribution $\text{Pois}(2)$ may be created using the commands:

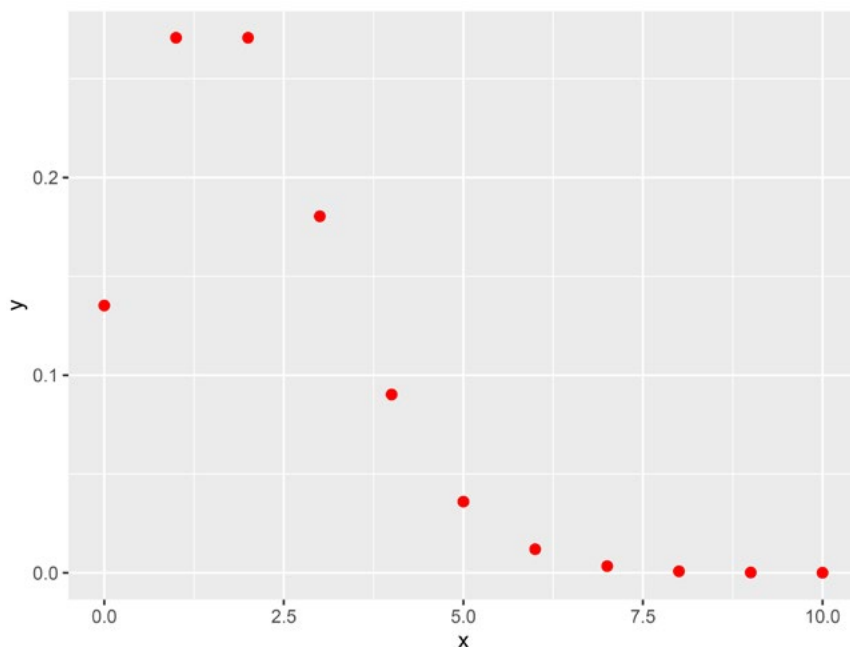
```
pplot <- ggplot(data.frame(x=0:10), aes(x=x)) +  
  stat_function(fun=dpois,args=list(2), n=11,  
  aes(colour=!("red")), size=2, geom="point")  
pplot
```

From this plot, we obtain that the maximum probability is 0.27 and it is observed when $x=1$ or 2 , i.e. when a square meter contains 1 or 2 pots, whereas the probability of an empty square is 0.135.

These individual probabilities can be calculated using the function:

```
dpois(x,  $\lambda$ )
```

Figure 4.12.
Poisson
distribution
curve $\text{Pois}(2)$



We should stress again that in continuous distributions we estimate the probability within a certain range of x values, whereas in discrete distributions, the probability is estimated at a certain x value. However, even in discrete distributions, we can estimate probabilities for a range of discrete values. For example, the probability of a square to contain either 0 or 1 or 2 pots is the sum $\text{Pois}(0,2) + \text{Pois}(1,2) + \text{Pois}(2,2)$, which is equal to $0.135 + 0.271 + 0.271 = 0.677$.

4.2. POINT ESTIMATION

As defined above, in point estimation, based on the sample data, we calculate the value of a certain measure (*statistic*) and use this value as the "best estimate" of the corresponding unknown population parameter. This approach for making inferences about the population is strictly valid when the statistic under consideration is an *unbiased estimator*. By this term we mean that the statistic has the following property: If we take a great number of samples from the same population, in each sample we calculate the value of the statistic under study and then average all these values, the mean value is the same as the corresponding value of the parameter in the population. It can be shown that the mean and the variance of a sample are unbiased estimators, while the standard deviation is not an unbiased estimator.

For a comparison of population parameters to sample statistics, we may use the function `bias()`, the code of which is given below. This function creates a log-normal or a normal population of 1,000,000 members and calculates the population parameters mean, standard deviation, variance, median, first and third quartile, skewness, and kurtosis. Then it generates samples of predefined size (`sampleSize`) from the population under study. The number of samples is defined from the variable `iter`. For each sample, the above descriptive statistics are estimated and averaged over all samples. This is repeated 20 times. The function provides the population histogram and comparison plots where the population parameters are shown by a blue horizontal line and each sample statistic by 20 points.

```
bias=function(distr="lnorm", popMean=4, popSD=0.35, sampleSize=30, iter=10000){
# This function compares sample statistics to population parameters
# Each population is created using the function rnorm() or rlnorm() with 1,000,000 members
# The samples are generated using the function sample()
# distr = "lnorm" or distr = "norm" is the population distribution
# popMean is the mean of the population
# popSD is the standard deviation of the population
# sampleSize is the size of the samples we will draw from the population
# iter is the number of iterations used to estimate the mean value of the sample statistic
# The function provides the population histograms and comparison plots where
# the population parameters are shown by a blue horizontal line
# and each sample statistic by 20 points estimated by averaging iter values
library(moments)
if(distr=="norm")
population <- rnorm(1000000, mean=popMean, sd=popSD)
else
population <- rlnorm(1000000, mean=popMean, sd=popSD)
par(mfrow=c(1,1))
hist(population)
pm=mean(population); psd=sd(population)
pvar=var(population); pmed=median(population)
pq=quantile(population,prob=0.25); pq2=quantile(population,prob=0.75)
psk=skewness(population); pku=kurtosis(population)
m=c(); s=m; v=m; med=m; q=m; q2=m; sk=m; ku=m
sm=m; ssd=m; svar=m; smed=m; sq=m; sq2=m; ssk=m; sku=m
for(jj in 1:20){
for(j in 1:iter){
samp <- sample(population, sampleSize, replace=TRUE)
m[j]=mean(samp); s[j]=sd(samp)
v[j]=var(samp); med[j]=median(samp)
q[j]=quantile(samp,prob=0.25); q2[j]=quantile(samp,prob=0.75)
sk[j]=skewness(samp); ku[j]=kurtosis(samp)
```

```

}
sm[jj]=mean(m); ssd[jj]=mean(s)
svar[jj]=mean(v); smed[jj]=mean(med)
sq[jj]=mean(q); sq2[jj]=mean(q2)
ssk[jj]=mean(sk); sku[jj]=mean(ku)
}
dev.new()
par(mfrow=c(2,4))
plot(x=1:20, y=sm, ylim=c(0.999*min(sm,pm),1.001*max(sm,pm)), xlab="n", ylab="mean")
abline(h=pm, col="blue")
plot(x=1:20, y=ssd, ylim=c(0.999*min(ssd,psd),1.001*max(ssd,psd)), xlab="n", ylab="sd")
abline(h=psd, col="blue")
plot(x=1:20, y=svar, ylim=c(0.999*min(svar,pvar),1.001*max(svar,pvar)), xlab="n", ylab="var")
abline(h=pvar, col="blue")
plot(x=1:20, y=smed, ylim=c(0.999*min(smed,pmed),1.001*max(smed,pmed)), xlab="n",
ylab="median")
abline(h=pmed, col="blue")
plot(x=1:20, y=sq, ylim=c(0.999*min(sq,pq),1.001*max(sq,pq)), xlab="n", ylab="quantile 0.25")
abline(h=pq, col="blue")
plot(x=1:20, y=sq2, ylim=c(0.999*min(sq2,pq2),1.001*max(sq2,pq2)), xlab="n",
ylab="quantile 0.75")
abline(h=pq2, col="blue")
plot(x=1:20, y=ssk, ylim=c(0.999*min(ssk,psk),1.001*max(ssk,psk)), xlab="n", ylab="skewness")
abline(h=psk, col="blue")
plot(x=1:20, y=sku, ylim=c(0.999*min(sku,pku),1.001*max(sku,pku)), xlab="n", ylab="kurtosis")
abline(h=pku, col="blue")
}

```

To run this function, copy the above code and `bias()` paste it into the R Console. Then, to use the function with its default values, type in the R Console the command:

We obtain the histogram of [Figure 4.13](#) (left) and the comparisons of [Figure 4.14](#). Note that these comparisons correspond to a log-normal population with $\mu=4$ and $\sigma=0.35$. To alter the arguments and examine a normal population with, say, $\mu=4$ and $\sigma=2$, we should use the command:

We obtain the histogram of [Figure 4.13](#) (right) and the comparisons of [Figure 4.15](#). It is seen that the mean and the variance of a sample are unbiased estimators, whereas all the other descriptive measures are biased estimators, at least for a log-normal population. For a normal population, the median and the skewness also become unbiased estimators. However, the difference between population parameters and sample statistics is not particularly large and, therefore, in the absence of any other information about a population parameter, we may (cautiously) use the corresponding sample statistic.

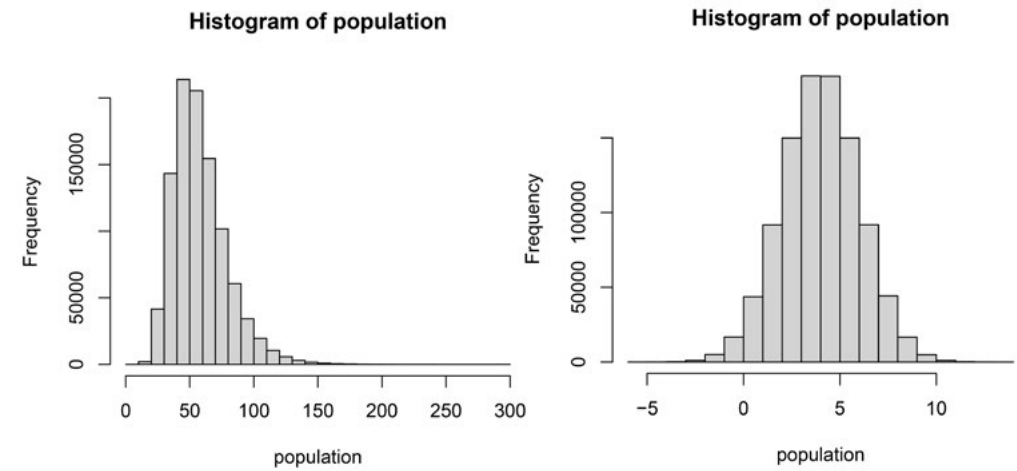


Figure 4.13.
Density histograms of a log-normal population with $\mu=4$ and $\sigma=0.35$ (left) and a normal population with $\mu=4$ and $\sigma=2$ (right)

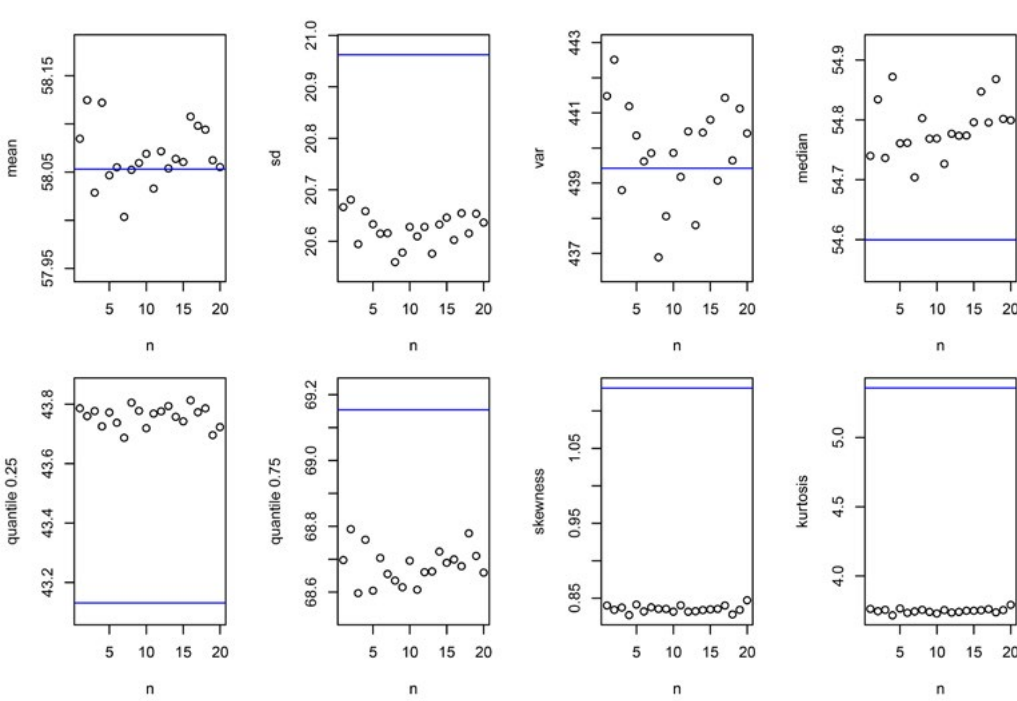


Figure 4.14.
Comparison of population parameters, shown with blue lines, to sample descriptive measures, shown by points, for the log-normal population of Figure 4.13

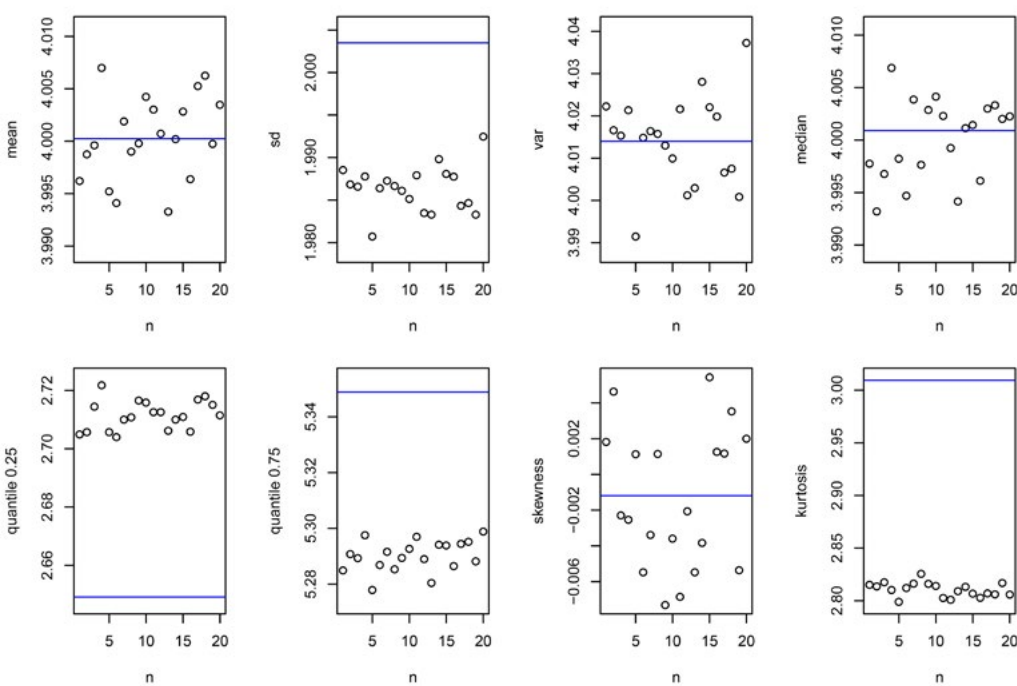


Figure 4.15.
Comparison of population parameters, shown with blue lines, to sample descriptive measures, shown by points, for the normal population of Figure 4.13

4.3. CONFIDENCE INTERVALS

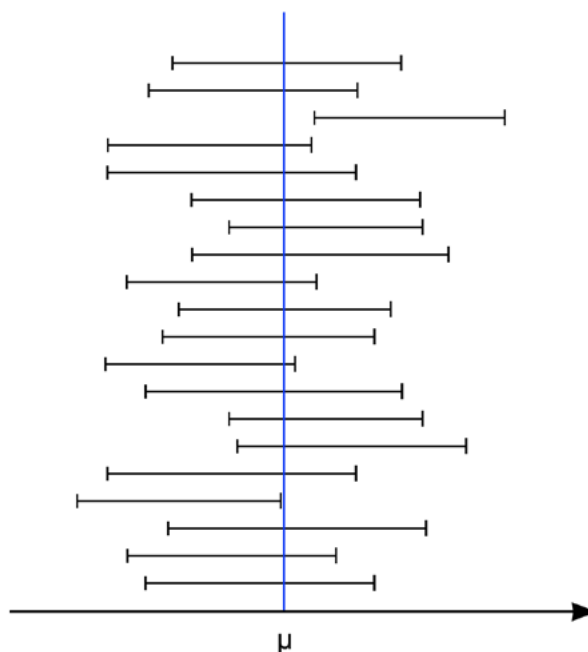
A confidence interval for a population parameter is a range of values, $[a, b]$, within which we believe the true value of that parameter will fall. A confidence interval is always related to a probability $P\%$ and for this reason, it is usually called *P% confidence interval*. The $P\%$ probability can also be written as $P = 100(1-\alpha)$, where α is the *significance level*. The most common confidence interval is that of 95%. However, confidence intervals of 90% and 99% are also used.

In most textbooks the $P\%$ confidence interval for a population parameter is the range of values which includes the unknown population parameter with a probability equal to $P\%$, whereas α gives the probability that the population parameter lies outside the estimated confidence interval. Thus, a 95% confidence interval means that there is a 95% probability that the population parameter lies within that interval and $\alpha=0.05=5\%$ probability that it lies outside this interval.

However, this definition of the confidence intervals has been questioned on the basis that once an interval is calculated, it either contains the population parameter value or it does not; so we cannot estimate probabilities. In this approach, the meaning of the $P\%$ confidence interval for a population parameter is the following: Consider that we draw a great number of samples from the population, in each sample we calculate the statistic related to the population parameter under study and construct the $P\%$ confidence interval for that parameter. Then $P\%$ of the confidence intervals we constructed would contain the true value of the population parameter. This is schematically shown in Figure 4.16 for a 90% confidence interval for the population mean.

Usually we are interested in constructing a $P\%$ confidence interval for the mean, that is, a range of plausible values for the population mean. This concerns continuous data. For samples of binary data, we create $P\%$ confidence intervals for the proportion. They are both easily constructed if the sampling distribution of the mean/proportion is known and especially if this distribution is normal.

Figure 4.16.
Meaning
of a 90%
confidence
interval for the
mean



Sampling distribution

First, we need to distinguish between the sample distribution and the sampling distribution. The *sample distribution* is the distribution of the observations in the sample, whereas the *sampling distribution* of a statistic is the distribution of that statistic when it is computed from many samples of the same size, n . Therefore, the sampling distribution of the mean is the probability distribution of a sample consisting of the mean values of a great number of samples of size n drawn from a population of a continuous variable. Similarly, the sampling distribution of the proportion is the probability distribution of a sample consisting of the proportions of a great number of samples of size n drawn from a population of a binary variable. Note that the sample proportion of a binary variable, π , is mathematically equivalent to the sample mean of the binary variable. For example, if a sample consists of n_0 values equal to 0 and n_1 values equal to 1, then the proportion π is equal to $(0 \times n_0 + 1 \times n_1) / (n_0 + n_1) = n_1 / n$, since $n = n_0 + n_1$.

The shape of the sampling distribution depends upon the original population and the size n of the samples drawn from the original population. However, according to the *Central Limit Theorem*, the sampling distribution of the mean and the proportion will be normal or nearly normal provided that the sample size n is greater than 30. Moreover, if μ and σ^2 are the mean and the variance of a normal population of a continuous variable, then the sampling distribution is also normal with mean value μ and variance σ^2/n . Similarly, if p is the population proportion of a binary variable, then for samples with $n > 30$, the sampling distribution of the proportion will be nearly normal with mean equal to p and variance equal to $p(1-p)/n$. Note that the variance of the population of a binary variable is equal to $p(1-p)$.

Confidence intervals for the mean

The sampling distribution of the mean has a direct application to the construction of confidence intervals of the mean. Consider a great number of samples of size n is drawn from a normal population with mean and variance equal to μ and σ^2 , respectively. The mean values of these samples, $\bar{x}$, follow the normal distribution with mean value equal to μ and variance σ^2/n . Note that $\bar{x}$ is an unbiased estimator of μ and, thus, we can interchange $\bar{x}$ and μ . Therefore, μ follows the normal distribution with mean value equal to $\bar{x}$ and variance σ^2/n . However, from the properties of the normal distribution we know that 95% of the sample values lie within 1.96 standard deviations from the mean (mean $\pm$ 1.96 standard deviations). From this observation we obtain that the 95% confidence interval for the mean is:

$$\bar{x} \pm 1.96 \frac{\sigma}{\sqrt{n}}$$

Note that the population standard deviation σ is usually unknown. For this reason, we approximate it by the sample standard deviation s , despite the fact that s is not an unbiased estimator for σ :

$$\bar{x} \pm 1.96 \frac{s}{\sqrt{n}}$$

Thus, in order to estimate the 95% confidence interval for the population mean from a sample of size n , we calculate the sample mean value and standard deviation and apply the above expression. This expression is usually adopted for rather large samples, $n > 30$, and it can be extended for any $P\%$ confidence interval via the following expression:

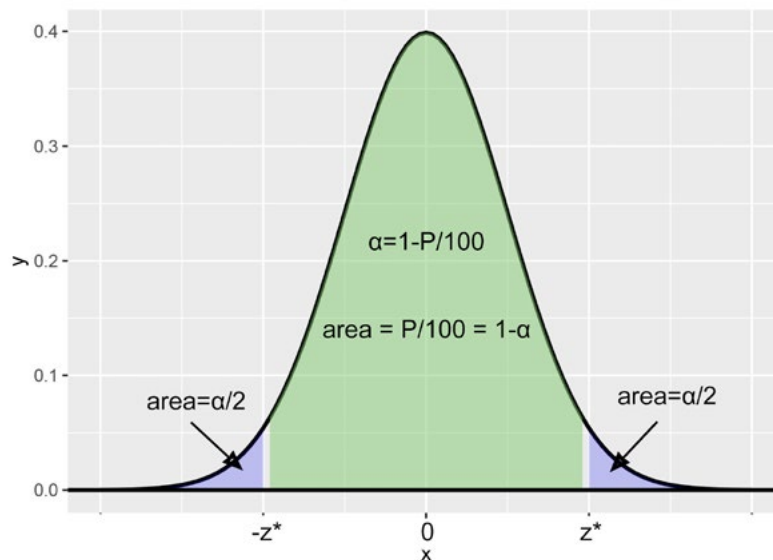
$$\bar{x} \pm z^* \frac{s}{\sqrt{n}}$$

where z^* is the upper $\alpha/2$ critical value for the standard normal distribution, defined in [Figure 4.17](#).

In R this critical value is calculated using the function:

`qnorm(1-a/2)`

Figure 4.17.
Standard normal distribution and definition of the upper $\alpha/2$ critical value



For relatively small samples, we should replace the standard normal distribution by the student distribution with $n-1$ degrees of freedom:

$$\bar{x} \pm t^* \frac{s}{\sqrt{n}}$$

where t^* is the upper $\alpha/2$ critical value for the t distribution with $n-1$ degrees of freedom.

In R it is calculated using the function:

`qt(1-a/2, n-1)`

For example, when $n=30$, the value of t^* for the 95% confidence interval is equal to 2.045, very close to 1.97, which is the corresponding z^* value of the standard normal distribution.

As an example, consider that we have used dendrochronology in order to date an archaeological site. We have taken eight samples, which gave the dates: 1252, 1188, 1305, 1273, 1195, 1231, 1287, 1267. Based on these data, we want to estimate the date of the site. In R, we first create a vector to include all our sample values:

`dates <- c(1252, 1188, 1305, 1273, 1195, 1231, 1287, 1267)`

Then we compute the mean value using the function:

`mean(dates)`

and the confidence limits using the command:

`qt(1-a/2, n-1)*sd(dates)/sqrt(n)`

Thus, in the R Console we type the commands:

```
mean(dates)
P <- 95
a <- 1-P/100
n <- 8
qt(1-a/2, n-1)*sd(dates)/sqrt(n)
```

and obtain the results shown in Figure 4.18:

```
> dates <- c(1252, 1188, 1305, 1273, 1195, 1231, 1287, 1267)
> mean(dates)
[1] 1249.75
> P <- 95
> a <- 1-P/100
> n <- 8
> qt(1-a/2, n-1)*sd(dates)/sqrt(n)
[1] 35.25123
```

Figure 4.18.
Computation
of 95%
confidence
interval

It is seen that the 95% confidence interval is:

$$1249.75 \pm 35.25 \text{ AD}$$

or

$$1250 \pm 35 \text{ AD}$$

This interval may be also expressed as:

$$1250 \pm 35 = [1250-35, 1250+35] = [1215, 1285]$$

An alternative and more direct approach to estimate confidence intervals is via the function `t.test()`:

The results are given in Figure 4.19 and we can see that the obtained interval is the same as that given above.

```
> t.test(dates, conf.level = 0.95)

One Sample t-test

data:  dates
t = 83.832, df = 7, p-value = 9.048e-12
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 1214.499 1285.001
sample estimates:
mean of x
 1249.75
```

Figure 4.19.
Calculation
of confidence
interval using
the function
`t.test()`

We need to stress that confidence intervals are only accurate when the sample data follow the normal distribution and have no outliers (extreme values).

If the data are not normally distributed, the best solution to construct confidence intervals is to use bootstrapping. *Bootstrapping* uses multiple random samples from the original data to construct confidence intervals. The samples are the same size as the original dataset, but they are drawn randomly with replacement so that some observations do not appear in a particular sample and others appear more than once. This method computes the mean for each sample and then estimates the 2.5 and 97.5 percent quantiles of those samples, i.e. the values below which 2.5% and 97.5% of the sample values lie, which give the 95% confidence interval.

To obtain confidence intervals with bootstrapping, we can use the `boot()` function of the *boot* package. Once we install this package, we can use the commands:

```
library(boot)
wmean <- function(x, d) {return(c(mean(x[d]),
var(x[d])/length(d)))}
means.boot <- boot(dates, wmean, R=5000)
boot.ci(means.boot, type="stud")
```

#we obtain Studentized confidence intervals

Note that the second line creates a function called `wmean()`. The function has two arguments, `x`, the variable to be used in creating the bootstrapped confidence interval, and `d`, an index that `boot()` will use to draw the samples. It returns the mean of `x[d]` and the sample variance divided by the number of observations. Next we pass `wmean()` to `boot()` along with the data we plan to use and the number of samples to generate.

The output is shown in Figure 4.20. It is seen that the 95% interval is now from 1202 to 1282 years.

Figure 4.20.
Confidence
interval
estimated with
bootstrapping

```
> library(boot)
> wmean <- function(x, d) {return(c(mean(x[d]), var(x[d])/length(d))) }
> means.boot <- boot(dates, wmean, R=5000)
> boot.ci(means.boot, type="stud") #we obtain Studentized confidence intervals
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 5000 bootstrap replicates

CALL :
boot.ci(boot.out = means.boot, type = "stud")

Intervals :
Level      Studentized
95%      (1202, 1282 )
Calculations and Intervals on Original Scale
```

Confidence intervals for binary data

The extension of confidence intervals for the mean of a continuous variable to confidence intervals for the proportion of a binary variable is straightforward. The sampling distribution of the proportion for samples with $n > 30$ is nearly normal with mean equal to p and variance equal to $p(1-p)/n$, where p is the population proportion. Since this is usually unknown, we can approximate it by π , the sample proportion. Therefore, the expression $\bar{x} \pm z^*s/\sqrt{n}$ is extended to:

$$\pi \pm z^* \sqrt{\frac{\pi(1-\pi)}{n}}$$

For example, suppose we examine 100 skeletons and find that 44% of them are male and 56% are female. Thus, we have a sample proportion of $n=0.44$ males and $1-n=0.56$ females. Therefore, for the 95% confidence interval for the proportion of males we have:

$$\pi \pm z^* \sqrt{\frac{\pi(1-\pi)}{n}} = 0.44 \pm 1.96 \sqrt{\frac{0.44(1-0.44)}{100}} = 0.0973$$

In R we may use the commands:

```
P <- 95
a <- 1-P/100
n <- 100
p <- 0.44
qnorm(1-a/2)*sqrt(p*(1-p)/n)
```

which yield the results of Figure 4.21. It is seen that the 95% confidence interval is:

0.44 ± 0.097 or $[0.343, 0.537]$

```
> P <- 95
> a <- 1-P/100
> n <- 100
> p <- 0.44
> qnorm(1-a/2)*sqrt(p*(1-p)/n)
[1] 0.09729005
```

Figure 4.21.
Computation
of 95%
confidence
interval for the
proportion of
males

4.4. HYPOTHESIS TESTING

Hypothesis testing includes the methods used by statisticians to test assumptions regarding population parameters. In a broader sense, hypothesis testing allows statisticians to make decisions based on experimental data. To make statistical decisions, it is necessary to make statistical hypotheses. However, the statistical hypotheses we can use and test are specific and depend on the type of problem we are examining. In other words, for every problem, we select the statistical hypotheses that have already been proposed for the study of that specific problem; we cannot use any hypothesis we want.

A very important hypothesis is called *null hypothesis* and is symbolized by H_0 . This hypothesis supports that there is no statistically significant effect in our data. For every null hypothesis there is an *alternative hypothesis*, H_1 , which supports the opposite of H_0 ; that is, that a significant effect is present.

For example, assume we want to study the nitrogen isotope values in the skeletal material found in two different archaeological sites. The null hypothesis is that there is no significant difference in the values between the two sites, whereas the alternative hypothesis is that the difference in nitrogen values between the two sites is statistically significant. However, **we must be careful**: The null and the alternative hypotheses are always stated using population and not sample terms. Thus, in the example of the nitrogen isotopes, consider the two samples from the two different archaeological sites and let $\bar{x}_1$ and $\bar{x}_2$ be their mean values⁸. The question that arises is whether $\bar{x}_1$ and $\bar{x}_2$ are significantly different or not. These values will be statistically the same if the samples come from populations with the same means, $\mu_1 = \mu_2$, and different if $\mu_1 \neq \mu_2$. Therefore, the statistical hypotheses we can make are:

H_0 : the samples come from populations with the same mean, $\mu_1 = \mu_2$

⁸ Note again that the symbol $\bar{x}$ is used to denote the sample mean, whereas μ symbolizes the population mean

For the alternative hypothesis, H_1 , we have the following options:

H_1 : The samples come from populations with $\mu_1 \neq \mu_2$ or

H_1 : The samples come from populations with $\mu_1 > \mu_2$ or

H_1 : The samples come from populations with $\mu_1 < \mu_2$

Consider another example of a statistical test and the corresponding hypotheses. Assume that we want to test if the data under study follow the normal distribution, that is, if the population from which the sample we examine comes follows the normal distribution. In this case, in mathematical terms the statistical hypotheses are:

H_0 : The sample comes from a normal population

H_1 : The sample does not come from a normal population

In simpler and less mathematical terms, the null hypothesis states that there is no statistically significant difference between the distribution of our sample and the normal distribution, while the alternative hypothesis supports that the difference between our sample distribution and the normal distribution is significant.

An important issue in hypothesis testing is that *we cannot prove the alternative hypothesis; we can only reject the null hypothesis*. Even if we reject the null hypothesis, this does not prove the alternative hypothesis, it only supports it.

Significance level and errors

In order to test the null hypothesis, we first define a level that is called *significance level*, it is symbolized by the Greek letter α and expresses the probability of rejecting a valid null hypothesis. The values usually used for the significance level are $\alpha = 0.05$ and $\alpha = 0.01$. This means that the probability of rejecting a correct null hypothesis is 5% when $\alpha = 0.05$ and 1% when $\alpha = 0.01$.

The role of the significance level is crucial. As already mentioned, inferential statistics are based on probabilities; so we need to define an acceptable probability of making a mistake when we conclude based on our results that there is a statistically significant pattern when there is not one and the observed differences are the result of random factors.

At this point we should stress again the following:

When we reject the null hypothesis, there is $\alpha\%$ probability that we are making a mistake. However, if our results fail to reject the null hypothesis, then we DO NOT accept it, because in this case there is no way of knowing the probability of making a mistake. In other words, when we reject the alternative hypothesis, this does not mean that the null hypothesis is true.

If, based on our statistical analysis, we reject the null hypothesis when it is actually valid, we are making a *Type I error*. The probability of performing a type I error is equal to α (usually 5%). In contrast, if we fail to reject a false null hypothesis, we are making a *Type II error*. Unfortunately, when we try to reduce the probability of making a type I error, we increase the probability of making a type II error and vice versa. The only way of reducing both types of error is by increasing the sample size.

The p-value and test statistic

The *p-value* is a way of quantifying the strength of the evidence against the null hypothesis and in favour of the alternative hypothesis. The smaller the p-value, the stronger the data favour H_1 over H_0 . A small p-value, usually smaller than 0.05, corresponds to sufficient evidence to reject H_0 in favour of H_1 .

The p-value is calculated using the proper test statistic. A *statistic* is any function among random variables that is calculated from sample data. A *test statistic* is a statistic that can be used to test the null hypothesis. Depending on the probability distribution of the test statistic, we can define the ranges of its values, where the null hypothesis is rejected or not rejected, and estimate the p-value. An example of this procedure is given below.

The results of a hypothesis test, i.e. the p-value, are affected by the sample(s) size. Small sample sizes tend to give unreliable results, unless they support the rejection of the null hypothesis.

One-tailed and two-tailed tests

When the alternative hypothesis, H_1 , is stated using the symbol $\neq$, the test is called *two-tailed*. In contrast, when the alternative hypothesis is stated using the symbol $>$ or $<$, the test is called *one-tailed*.

As an example, consider that we want to compare the body mass between males and females in a sample, as calculated from the skeletal remains of the individuals. In this case, we know in advance that in general males tend to be bigger and heavier than females, although in every sample there will be big women and small men. In this case, we are not really interested in testing if there is a significant difference between the male and the female body mass but whether male body mass is significantly bigger than the female one. Thus, the alternative hypothesis should be expressed as:

H_1 : The samples come from populations with $\mu_{\text{male}} > \mu_{\text{female}}$

This is a one-tailed test, because we have a specific 'direction' in mind in advance.

In contrast, if we want to compare the body mass of male individuals from two archaeological sites, there may be no reason to assume in advance that the men from site A will be bigger than the men from site B. In other words, we are comparing our samples without knowing in advance which one should be bigger than the other. In this case we are performing a two-tailed test and the alternative hypothesis is stated as:

H_1 : The samples come from populations with $\mu_{\text{siteA}} \neq \mu_{\text{siteB}}$

Parametric and non-parametric tests

Statistical tests may be *parametric* or *non-parametric*. Parametric tests presuppose that the samples under study come from populations that follow a known distribution, usually the normal distribution. When the samples come from populations of unknown distribution, we have to apply non-parametric tests, since these tests do not require any assumptions concerning the distribution of the data.

An interesting difference between parametric and non-parametric tests is that in non-parametric tests we do not analyze the raw data. Instead, the raw data are usually transformed into *ranks* or we conduct a *runs* test. If $(x_1, x_2, \dots, x_m)$ is a sample of quantitative data, we call *rank* of value x_i the number r_i of sample data that are smaller than or equal to x_i . For example, the sample (0.12, 0.09, 0.11, 0.10) is turned into (4, 1, 3, 2). A runs test is used to decide if the sample values come from a random process. A run is a sequence of two symbols, for example the symbols + and -. To transform a sample into a run, we may correspond the symbol + to each sample value lower than the median and the symbol - to values greater than the median. Values equal to the median are omitted. Non-parametric tests include also the *bootstrap* and *Monte Carlo tests with permutations*.

Parametric methods generally produce more accurate and precise estimates, i.e. they have more statistical *power*, where the power of a statistical test is the probability that it correctly rejects the null hypothesis when the null hypothesis is false. However, this holds provided that the assumptions on which they are based are met. If these assumptions are not met, parametric methods can produce misleading results.

An example of testing the null hypothesis

In an African archaeological site animal mandibular remains that likely belong to dwarf goats were discovered. The mandibular lengths are given in **Table 4.1**. Taking into consideration that among modern dwarf goats the mean mandibular length is 12.5 cm, what conclusions can be drawn?

Table 4.1.
Maximum
mandibular
lengths of
dwarf goats

11.2	12.8	13	10.7	10.5	11.8
12.2	11.5	12.1	12.7	11.9	12.4

In this example we want to examine if dwarf goats have become significantly differentiated over time due to palaeoenvironmental factors. Thus, the statistical hypotheses are:

$$H_0: \mu = 12.5, H_1: \mu \neq 12.5$$

The null hypothesis suggests that the sample comes from a population with mean $\mu = 12.5$, since this is the mean mandibular length among modern dwarf goats. If the null hypothesis is rejected, we can conclude that dwarf goats have become differentiated over time with $\alpha\%$ probability that this conclusion is wrong. If the null hypothesis cannot be rejected, then we would conclude that there is no evidence of a systematic differentiation of dwarf goats over time. However, this conclusion does not mean that there is no systematic differentiation of dwarf goats, only that none has been demonstrated.

In what concerns the significance level, as mentioned above, this is usually set to $\alpha = 0.05$, which means that we accept a 5% probability of making a mistake when rejecting a correct null hypothesis.

In order to test the null hypothesis, we should use a proper test statistic. In the problem under study the test statistic is:

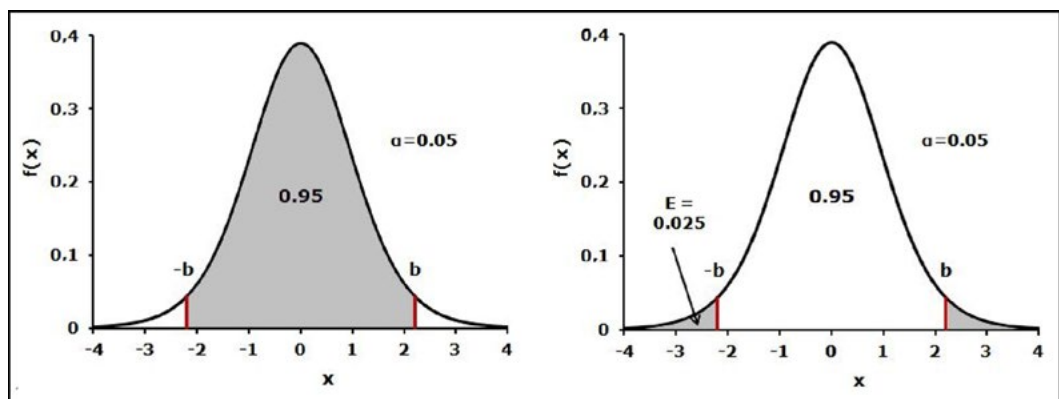
$$t = \frac{\bar{x} - \mu}{s} \sqrt{n}$$

It can be theoretically proven that t follows the student distribution with $n-1 = 12-1 = 11$ degrees of freedom. Subsequently, we calculate the t value:

$$t = \frac{\bar{x} - \mu}{s} \sqrt{n} = \frac{11.9 - 12.5}{0.803} \sqrt{12} = -2.587$$

where $\bar{x} = 11.9$ and $s = 0.803$ are the sample mean value and standard deviation, respectively. In order to understand what the meaning of $t = -2.587$ is, see [Figure 4.22](#). This figure displays the student distribution with 11 degrees of freedom. It can be proven that in this graph when $b = 2.2$, the area below the curve from $-b$ to b is 0.95. Moreover, because the area below the curve of any test statistic is equal to 1, the area from $-\infty$ to $-b$ is 0.025.

Figure 4.22.
Plot of the
student
distribution
with 11 degrees
of freedom.
Note that b
 $= 2.2$



This suggests that if we create samples with 12 values of mandibular length from modern populations of dwarf goats, then the t value that is calculated for each sample lies in the interval $[-2.2, 2.2]$ with 95% probability. However, if the samples have not been drawn from the modern population and are significantly differentiated from the modern samples due to environmental and other factors, then the mean of each sample will be systematically different from value 12.5, thus the estimated t values will most likely lie outside the interval $[-2.2, 2.2]$. Of course, the t value of a sample from a modern population may lie outside the $[-2.2, 2.2]$ interval with a probability 5%, due to random factors. Therefore, a t value larger than 2.2 or smaller than -2.2, may indicate the impact of some systematic factor. There is merely 5% probability that our finding is due to random factors and, because this probability is so small, we can reject the null hypothesis (always bearing in mind that there is 5% chance of being mistaken).

The value $b = 2.2$ (and $b = -2.2$) is called *critical value* and it is defined as shown in Figure 4.21. In specific, this value is such so that the area below the curve of the student distribution from $-b$ to b is equal to 0.95 when $\alpha = 0.05$ or the area from $-\infty$ to $-b$ is equal to 0.025 or from b to ∞ is equal to 0.025.

In the example under study $t = -2.578 < -2.2$ (Figure 4.23). Therefore, the null hypothesis is rejected at the 5% significance level. This leads to the conclusion that dwarf goats have become differentiated over time, though there is 5% probability of being wrong. On the other hand, if the estimated t value had lied within interval $[-2.2, 2.2]$, we would have concluded that there is no evidence of a systematic differentiation of dwarf goats over time. Note again that this would not have meant that there is no systematic differentiation of dwarf goats over time, only that no such differentiation has been demonstrated.

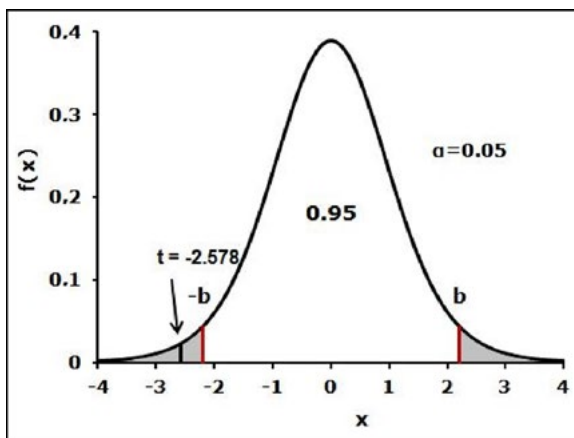


Figure 4.23.
Testing the null hypothesis

In commercial statistical software packages, null hypothesis testing is not based on critical values, but rather on the so-called p -value. The definition of the p -value is given in Figure 4.24. We observe that the p -value is equal to two times the area from $-\infty$ to the estimated t value. However, because the area from $-\infty$ to $-b$ is equal to $\alpha/2 = 0.025$ (Figure 4.24), we can conclude that the null hypothesis is rejected when $p\text{-value} < \alpha$. In general, the smaller the p -value in comparison to the α value, the more confident we can be when we reject the null hypothesis.

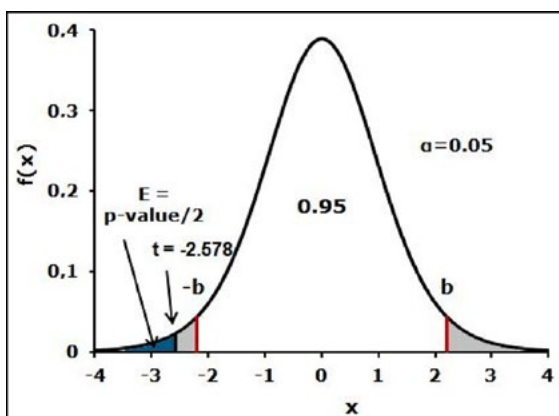


Figure 4.24.
Definition of the p -value

In statistics the above procedure is called *one-sample t-test* and in R it can be conducted easily by means of the `t.test()` function:

```
x <- c(11.2, 12.8, 13, 10.7, 10.5, 11.8, 12.2, 11.5, 12.1,
12.7, 11.9, 12.4)
t.test(x, mu=12.5)
```

As expected, we obtain the same p-value = 0.025 (Figure 4.25).

Figure 4.25.
Application
of one-sample
t-test in R

```
> x <- c(11.2, 12.8, 13, 10.7, 10.5, 11.8, 12.2, 11.5,
+ 12.1, 12.7, 11.9, 12.4)
> t.test(x, mu=12.5)

      One Sample t-test

data:  x
t = -2.5871, df = 11, p-value = 0.02527
alternative hypothesis: true mean is not equal to 12.5
95 percent confidence interval:
 11.38954 12.41046
sample estimates:
mean of x
      11.9
```

5.

HYPOTHESIS TESTING: ONE SAMPLE

The most important statistical tests that can be conducted in a single sample are the normality test, tests for the existence of outliers and the one sample t-test. The latter has already been examined above. The first two tests are described in this section. They are both important for the selection of the correct procedure that should be adopted in testing statistical hypotheses. For this reason, they usually precede most statistical analyses.

5.1. NORMALITY TEST

Most of the parametric tests presume that the sample(s) involved in the tests come from normal population(s). Although small deviations from normality are allowable, the normality test is a necessary tool in hypothesis testing. As already pointed out, the null hypothesis and its alternative are stated as:

H_0 : The sample comes from a normal population

H_1 : The sample does not come from a normal population

There are several normality tests. Among them, the Shapiro-Wilk test has the greatest power. In R this test is implemented via the `shapiro.test()` function.

Consider the sample of mandibular lengths of dwarf goats of **Table 4.1**.

To run the Shapiro-Wilk test, we can use the commands:

```
x <- c(11.2, 12.8, 13, 10.7, 10.5, 11.8, 12.2, 11.5,
12.1, 12.7, 11.9, 12.4)
shapiro.test(x)
```

We obtain the results shown in **Figure 5.1**, where it can be seen that the test statistic W is equal to 0.95565 and the p-value is 0.72. Since the p-value is greater than 0.05, the sample does not appear to violate the normality assumption.

```
> x <- c(11.2, 12.8, 13, 10.7, 10.5, 11.8, 12.2, 11.5, 12.1, 12.7, 11.9, 12.4)
> shapiro.test(x)

      Shapiro-Wilk normality test

data:  x
W = 0.95565, p-value = 0.7204
```

Figure 5.1.
Results of the
normality test

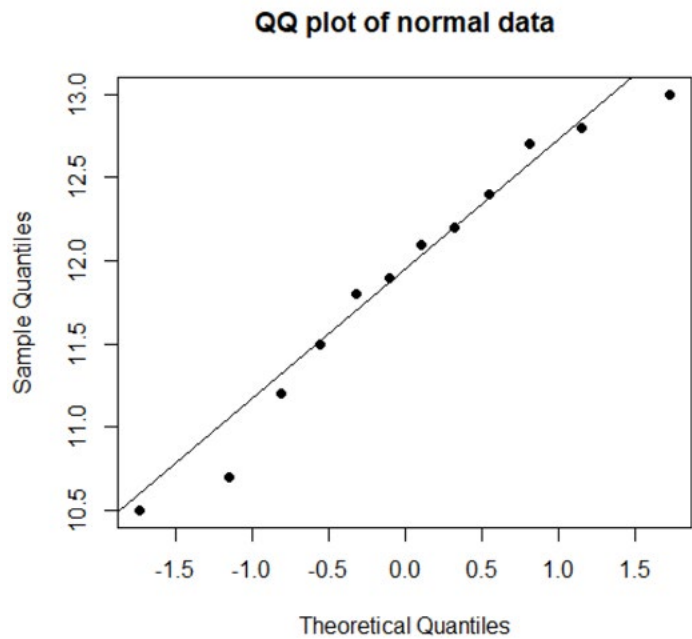
An alternative method to test the normality is via a normal Q–Q (quantile-quantile) plot. In this plot, quantiles from a theoretical normal distribution are compared to sample quantiles. For normally distributed data, the points of this plot should lie approximately on a straight line. If the data is non-normal, the points deviate markedly from a straight line.

In R the normal Q-Q plot is created by means of the following commands:

```
qqnorm(x, main="QQ plot of normal data",
pch=19)
qqline(x)
```

We obtain the plot of Figure 5.2, which clearly shows that the points lie (approximately) on a straight line and, therefore, we can conclude that the data of Table 4.1 are normally distributed.

Figure 5.2.
Normal Q-Q
plot for the
data of Table
4.1



WARNING

In large samples the Shapiro-Wilk test can be significant even when the scores are only slightly different from a normal distribution. Therefore, they should be interpreted in conjunction with histograms, Q-Q plots, and the values of skewness and kurtosis.

5.2. OUTLIERS

In statistics, an outlier is a value that is distant from other values in the dataset. Outliers are important to identify because they may signify an error (either during obtaining the value or during data input) or they may indeed suggest a case in the dataset that stands out from the rest and requires further examination and interpretation. To examine if there are outliers in a sample, we can use boxplots or check the z sample values. Outliers stand out of the whiskers of a boxplot and produce z scores conventionally smaller than -3 or larger than 3.

Consider the sample of strontium isotope values ($^{87}\text{Sr}/^{86}\text{Sr}$) from human teeth shown in Table 5.1. Such values can indicate if an individual is local or nonlocal through comparison with the values obtained from local plants, animals, and other sources of bioavailable strontium. In this example, we want to see if the $^{87}\text{Sr}/^{86}\text{Sr}$ values of Table 5.1 are homogeneous or not. If some individuals stand out from the rest, this suggests a potential different place of origin for them.

Table 5.1.
 $^{87}\text{Sr}/^{86}\text{Sr}$ values
for human
teeth

0.70866	0.70575	0.70764	0.70459	0.70628	0.7036	0.70306	0.70876
0.70318	0.70368	0.70325	0.70318	0.70375	0.7059	0.70760	0.70759
0.707611	0.70367	0.7150	0.70575	0.70874	0.70576	0.70481	0.70278

If we draw a boxplot of the values of **Table 5.1**, using the following commands, we obtain **Figure 5.3**.

```
SrData <- read.table("SrValues.txt", head=1,
  sep="\t")
library(ggplot2)
boxplot <- ggplot(SrData, aes(y=SrValues)) +
  geom_boxplot() + ylab("87Sr/86Sr values")
boxplot
```

In this Figure, it is seen that there is one value that stands out of the whiskers of the boxplot. This value (at 0.715) is shown as a black dot. The individual to whom this value corresponds likely originated from a different place compared to the rest of the individuals in the cemetery.

To use the method of z scores, we should convert the data of **Table 5.1** into z-scores by subtracting the sample mean from each sample value and dividing the result by the sample standard deviation:

$$z = \frac{x - \bar{x}}{s}$$

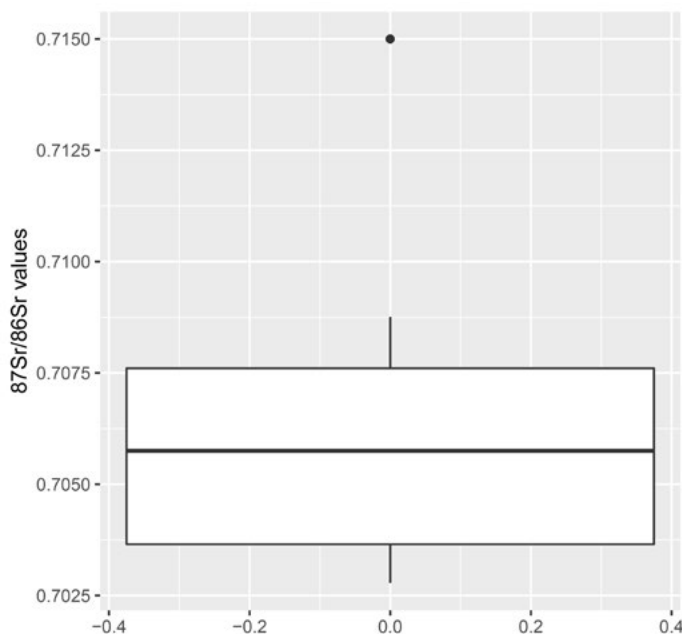


Figure 5.3.
Boxplot of
 $^{87}\text{Sr}/^{86}\text{Sr}$ values
from Table 5.1

In R we may use the commands:

```
x <- as.matrix(SrData)
z <- (x-mean(x))/sd(x)
xz <- cbind(x,z)
colnames(xz) <- c("SrValues", "z")
xz
```

The first command converts the data frame *SrData* into the variable *x* with a matrix format, the second command calculates the z scores, the third command creates the table *xz* with two columns, the variables *x*, *z*, and the fourth command gives the names "SrValues" and "z" to these columns.

We obtain the results of Figure 5.4. Note that z-scores basically rescale the data so that their distribution has a mean value of 0 and a standard distribution of 1. Thus, z values that are distant from 0 may indicate an outlier. In most cases, we set a threshold of -3 and 3, which means that all z scores smaller than -3 or larger than 3 are considered outliers. Figure 5.4 shows that the $^{87}\text{Sr}/^{86}\text{Sr}$ value = 0.715 with a z-score equal to 3.2598 is an outlier, in agreement with the result obtained from the boxplot of Figure 5.3.

Table 5.4.
Results of the
z test

```
> x <- as.matrix(SrData)
> z <- (x-mean(x))/sd(x)
> xz <- cbind(x,z)
> colnames(xz) <- c("SrValues", "z")
> xz
```

	SrValues	z
[1,]	0.7086600	0.99912478
[2,]	0.7057500	-0.03849936
[3,]	0.7076400	0.63542147
[4,]	0.7045900	-0.45212273
[5,]	0.7062800	0.15048373
[6,]	0.7036000	-0.80512888
[7,]	0.7030600	-0.99767769
[8,]	0.7087600	1.03478196
[9,]	0.7031800	-0.95488907
[10,]	0.7036800	-0.77660313
[11,]	0.7032500	-0.92992903
[12,]	0.7031800	-0.95488907
[13,]	0.7037500	-0.75164310
[14,]	0.7058900	0.01142070
[15,]	0.7076030	0.62222831
[16,]	0.7075960	0.61973231
[17,]	0.7076110	0.62508089
[18,]	0.7036710	-0.77981228
[19,]	0.7150000	3.25979043
[20,]	0.7057490	-0.03885593
[21,]	0.7087390	1.02729395
[22,]	0.7057650	-0.03315078
[23,]	0.7048060	-0.37510321
[24,]	0.7027813	-1.09705427

6.

HYPOTHESIS TESTING: TWO SAMPLES

To test hypotheses concerning two samples, we should first determine the type of samples. In statistics, two or more samples may be *independent* or *dependent*. The samples are *independent* when each of them has been obtained independently from the others. A main characteristic of such samples is that they may have different sizes (number of cases). The samples are *dependent* if we know in advance that each observation in one dataset is directly related to a specific observation in the other dataset. Therefore, dependent samples always have the same number of observations (cases). Two dependent samples are usually called *paired* or *matched samples*.

For two independent samples, we can use different tests, among which the most important is the *two-samples t-test*. It is used to test whether two sample means are significantly different or not and, therefore, whether the samples come from populations with different or the same means. Thus, the statistical hypotheses are:

H_0 : the samples come from populations with the same mean, $\mu_1 = \mu_2$

H_1 : The samples come from populations with $\mu_1 \neq \mu_2$

or H_1 : The samples come from populations with $\mu_1 > \mu_2$

or H_1 : The samples come from populations with $\mu_1 < \mu_2$

To conduct the above t-tests, the samples should be normal or nearly normal. Therefore, we should first test the normality of the samples. For independent samples, we should additionally test the *homogeneity of variance*, that is, whether the samples come from populations with the same variance or not. For this test the statistical hypotheses are stated as:

H_0 : the samples come from populations with the same variance, $\sigma_1^2 = \sigma_2^2$

H_1 : The samples come from populations with $\sigma_1^2 \neq \sigma_2^2$

6.1. HOMOGENEITY OF VARIANCE

The equality of variances of two samples may be tested using the F-test or/and the Levene test of equality of variances. The F-test is a parametric test, it can be used only for two samples, and assumes that the samples come from normal populations, whereas Levene's test is non-parametric and can be used for two or more samples.

In R the F-test can be conducted by means of the `var.test()` function, whereas for Levene's test, we may use the `leveneTest()` function from the `car` package. The basic expression of the `var.test` is `var.test(x, y)`, where `x`, `y` are numeric vectors of data values of the first and second sample, respectively. For the `leveneTest()` function, the corresponding expression is `leveneTest(x, group, center=mean)`, where `x` is a numeric vector of the data values of all samples, `group` is a factor defining the samples, i.e. a grouping variable, and `center=mean` gives the original Levene's test, whereas the default, `center=median`, provides a more robust test, the Brown-Forsythe test. Thus, in order to apply the `leveneTest()` function, we should organize the data in two vectors, `x` and `group`, where `x` includes all the sample values and `group` is a vector with values 1, 2, ... that indicate the samples in the `x` vector (1 shows the first sample, 2 the second sample and so on).

As an example, consider the body diameter (in cm) of the jugs measured at two neighbouring sites:

- site A: 10.5, 18.4, 17, 12.3, 11.5, 11.9, 14.4, 10.2, 16.6, 17.3, 17.7, 18.1, 20.4, 22.1, 13.7
- site B: 21.1, 20.4, 18.4, 15.2, 17, 18.3, 20.7, 21.4, 23.7, 28.9, 30.5, 27.7

If we wish to examine whether the jugs in these sites are significantly different in size, we must first examine whether the samples come from normal populations and whether these populations have equal or different variances.

Figure 6.1.
Data for
testing
two-samples
hypotheses

siteA	siteB
10.5	21.1
18.4	20.4
17	18.4
12.3	15.2
11.5	17
11.9	18.3
14.4	20.7
10.2	21.4
16.6	23.7
17.3	28.9
17.7	30.5
18.1	27.7
20.4	
22.1	
13.7	

x	group
10.5	1
18.4	1
17	1
12.3	1
11.5	1
11.9	1
14.4	1
10.2	1
16.6	1
17.3	1
17.7	1
18.1	1
20.4	1
22.1	1
13.7	1
21.1	2
20.4	2
18.4	2
15.2	2
17	2
18.3	2
20.7	2
21.4	2
23.7	2
28.9	2
30.5	2
27.7	2

To perform these tests, we organize the samples in two tab delimited .txt files with names *jug_diam* and *jug_diam_group*, respectively, as shown in Figure 6.1. The first file is for testing the normality of the samples and apply the F-test for the variances, whereas the second file will be used for Levene's test of the homogeneity of variance.

The commands we use are:

```
library(car)

jug1 <- read.table("jug_diam.txt", head=1,
sep="\t")

jug2 <- read.table("jug_diam_group.txt", head=1,
sep="\t")

shapiro.test(jug1$siteA); shapiro.test(jug1$siteB)

var.test(jug1$siteA, jug1$siteB)

leveneTest(jug2$x, factor(jug2$group), "mean")
```

Figure 6.2 shows the obtained results. It is seen that the normality tests give the p-values 0.4778 and 0.3546. Since both these p-values are greater than 0.05, the samples do not appear to violate the normality assumption. The same conclusion applies to the homogeneity of variance, since the F-test gives the p-value=0.3356>0.05 and Levene's test the p-value=0.4441>0.05. Thus, no statistically significant differences in the variances of the populations from which the samples of site A and B come are observed.

```

> library(car)
> jug1 <- read.table("jug_diam.txt", head=1, sep="\t")
> jug2 <- read.table("jug_diam_group.txt", head=1, sep="\t")
> shapiro.test(jug1$siteA); shapiro.test(jug1$siteB)

      Shapiro-Wilk normality test

data:  jug1$siteA
W = 0.94695, p-value = 0.4778

      Shapiro-Wilk normality test

data:  jug1$siteB
W = 0.92752, p-value = 0.3546

> var.test(jug1$siteA, jug1$siteB)

      F test to compare two variances

data:  jug1$siteA and jug1$siteB
F = 0.58055, num df = 14, denom df = 11, p-value = 0.3356
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.1728454 1.7965792
sample estimates:
ratio of variances
 0.5805549

> leveneTest(jug2$x, factor(jug2$group), "mean")
Levene's Test for Homogeneity of Variance (center = "mean")
      Df F value Pr(>F)
group  1  0.6046 0.4441
      25

```

Figure 6.2.

Results obtained from the Shapiro-Wilk normality test and F-test and Levene's test for variances

6.2. INDEPENDENT-SAMPLES T-TEST

Usually in archaeology we have to compare the mean values of two samples, where the samples may be the artifacts or ecofacts found in two different contexts, the male and female individuals from the same or different sites, different soil samples taken during an excavation season, etc. As already pointed out, in statistics the comparison of samples is carried out via the comparison of the populations from which the samples come. When the samples under examination are independent, the null hypothesis and the alternative hypothesis are:

$$H_0: \mu_1 = \mu_2$$

$$H_1: \mu_1 \neq \mu_2 \text{ or } H_1: \mu_1 > \mu_2 \text{ or } H_1: \mu_1 < \mu_2$$

depending on whether we perform a two-tailed or a one-tailed test.

The test statistic depends on whether the samples come from populations of the same or different variance. If the samples come from normal populations with the same variance, $\sigma_1^2 = \sigma_2^2$ the test statistic is given from the equation below and follows the student distribution with $df = n_1 + n_2 - 2$ degrees of freedom:

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_p^2}{n_1} + \frac{s_p^2}{n_2}}}$$

where

$$s_p^2 = \frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}$$

Here, $\bar{x}_1$ is the mean value of the first sample, $\bar{x}_2$ is the mean value of the second sample, s_1 is the standard deviation of the first sample, s_2 the standard deviation of the second sample, n_1 is the sample size for the first sample, and n_2 the sample size for the second sample.

If the samples come from normal populations with different variances, $\sigma_1^2 \neq \sigma_2^2$, the test statistic is given from:

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$$

and it follows the student distribution with df estimated from:

$$df \approx \frac{\left(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}\right)^2}{\frac{1}{n_1 - 1} \left(\frac{s_1^2}{n_1}\right)^2 + \frac{1}{n_2 - 1} \left(\frac{s_2^2}{n_2}\right)^2}$$

To run the independent samples t-test, the steps we should follow are:

1. Test the normality of the two samples. If strong deviations from normality are detected in one or both samples, the independent-samples t-test should not be applied. In this case we can run a non-parametric test, such as the Wilcoxon rank-sum test described in section 8.1.
2. Test the homogeneity of variance using either the parametric F-test or the non-parametric Levene test.
3. Run the independent samples t-test using the `t.test(x, y, var.equal)` function. In this function, `x`, `y` are vectors with the values of the two samples, and `var.equal` takes the values TRUE or FALSE depending on whether there is homogeneity of variance or not, as tested at step 2.

As an example, let us examine whether the jugs of Figure 6.1 are different in the two sites. For the samples of these jugs we have already shown that they do not appear to violate the normality assumption and that no statistically significant difference in the variances of the populations from which the samples come is detected. Therefore, we can run the t-test to examine whether the samples come from populations with different means or not.

For this purpose, we can use the command: `t.test(jug1$siteA, jug1$siteB, var.equal=TRUE)`

Note that in this command we used `var.equal=TRUE`, because the tests of the homogeneity of variances showed no statistically significant difference in the variances of the populations.

If the homogeneity of variances assumption had been violated, we would have used the command: `t.test(jug1$siteA, jug1$siteB, var.equal=FALSE)`

The results of the t-test are given in Figure 6.3. The p-value is $0.00058 \ll 0.05$, which suggests that there is a statistically significant difference between the mean values of the two samples. Looking at the estimated mean values, we can see that the jugs from site B have an average diameter = 21.94 cm, thus, they are significantly bigger than those of site A, which have an average diameter = 15.47 cm.

```
> t.test(jugl$siteA, jugl$siteB, var.equal=TRUE)

Two Sample t-test

data:  jugl$siteA and jugl$siteB
t = -3.9386, df = 25, p-value = 0.00058
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -9.850689 -3.085977
sample estimates:
mean of x mean of y
 15.47333  21.94167
```

Figure 6.3.
Results of the
independent
samples t-test

6.3. PAIRED-SAMPLES T-TEST

The paired-samples t-test is used to compare two paired samples and examine if there is a statistically significant difference between them. For example, we may want to compare the dimensions between the right and left humeri from a population in order to examine bilateral asymmetry. In such cases we want to see if there is a statistically significant difference between the values of each pair of left-right humeri. If there is no such difference, then the mean of the difference of each pair should be zero or close to zero. A basic precondition of the paired samples t-test is that the sample that consists of the differences between each pair of values follows the normal distribution.

The statistical hypotheses are expressed in terms of the mean difference score (μ_d) in the population:

$$H_0: \mu_d = 0$$

$$H_1: \mu_d \neq 0 \text{ or } \mu_d > 0 \text{ or } \mu_d < 0$$

where μ_d is the mean value of the population from which the sample of the differences in paired values comes. In simpler terms, the null hypothesis supports that there is no significant difference between the two samples, while the alternative hypothesis claims that the difference between the two samples is significant.

The test statistic is expressed as:

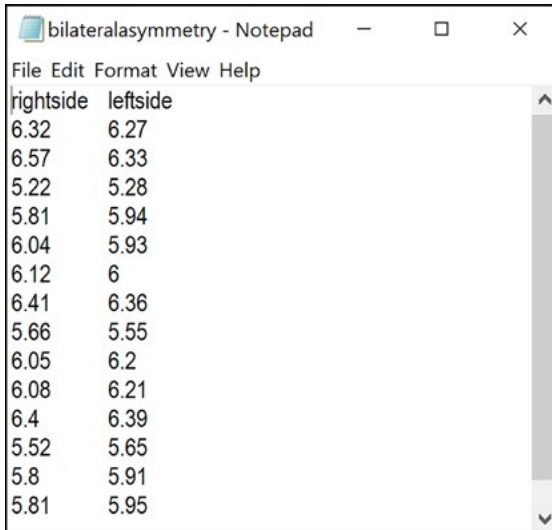
$$t = \frac{\bar{D}}{s_D / \sqrt{n}}$$

and follows the student t distribution with $df = N-1$. Here, $\bar{D}$ and s_D are the mean and standard deviation of the differences between all pairs, and n is the sample size.

For the implementation of the paired-samples t-test in R, we may still adopt the `t.test()` function, however, using the arguments `t.test(x, y, paired=TRUE)`.

As an example, consider that we have measured the circumference of the right and left humerus in a Neolithic population. The values, organized in a .txt file, are given in Figure 6.4. Given that the skeleton deposits new bone tissue when it is subjected to increased mechanical stress, we want to examine whether there is significant bilateral asymmetry, which might be attributed to unilateral daily activities. Note that the measurements of each sample are placed in a separate column. Also note that each row corresponds to one individual, so the right-left values in each row belong to the same skeleton.

Figure 6.4.
Midshaft
circumference
of right and
left humeri



rightside	leftside
6.32	6.27
6.57	6.33
5.22	5.28
5.81	5.94
6.04	5.93
6.12	6
6.41	6.36
5.66	5.55
6.05	6.2
6.08	6.21
6.4	6.39
5.52	5.65
5.8	5.91
5.81	5.95

Contrary to the independent samples t-test, in this test there is no point to check if each of the original variables follows the normal distribution. Rather, we need to calculate the difference between the right and left sides of the body and examine if this new variable follows the normal distribution. Thus, we estimate the difference between the right and the left side data and then apply the Shapiro-Wilk test.

The following commands can be used to read the data file, and run the Shapiro-Wilk test and the paired-samples t-test:

```
ba <- read.table("bilateralasymmetry.txt", head=1,
sep="\t")
shapiro.test(ba$rightside - ba$leftside)
t.test(ba$rightside, ba$leftside, paired=TRUE)
```

The output is given in Figure 6.5. It is seen that the sample of differences does not appear to violate the normality assumption ($p\text{-value} = 0.068 > 0.05$), thus we can proceed and use a paired samples t-test. The latter gives the $p\text{-value} = 0.7407$, which is greater than 0.05. Thus, there does not appear to be statistically significant bilateral asymmetry. The output also provides a 95% confidence interval for the mean difference (-0.084 to 0.0616). The importance of this interval is that it includes zero. This tells us that the true value of the mean difference may be zero. Thus, the two samples may represent random samples from the same population.

Figure 6.5.
Results of the
paired samples
t-test

```
> ba <- read.table("bilateralasymmetry.txt", head=1, sep="\t")
> shapiro.test(ba$rightside-ba$leftside)

      Shapiro-Wilk normality test

data:  ba$rightside - ba$leftside
W = 0.88488, p-value = 0.06825

> t.test(ba$rightside, ba$leftside, paired = TRUE)

      Paired t-test

data:  ba$rightside and ba$leftside
t = -0.33808, df = 13, p-value = 0.7407
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.08445735  0.06160020
sample estimates:
mean of the differences
      -0.01142857
```

7.

HYPOTHESIS TESTING: MANY SAMPLES

When we have more than two samples, then, depending on the samples, we may conduct *One-way Analysis of Variance (ANOVA)*, *Repeated Measures ANOVA*, or *Analysis of Covariance (ANCOVA)*. ANOVA is used to compare the means of three or more independent samples, repeated measures ANOVA compares means among related (dependent) samples, and ANCOVA evaluates whether the means of independent samples are equal for the various levels of a categorical variable.

7.1. ONE-WAY ANALYSIS OF VARIANCE (ANOVA)

Analysis of Variance allows the comparison of the means of more than two groups. The simplest ANOVA is one-way ANOVA, which is used to examine whether there are any significant differences among the means of three or more independent samples. Therefore, the null hypothesis states that all population means are equal. In mathematical terms this is denoted as:

$$H_0: \mu_1 = \mu_2 = \mu_3 = \dots = \mu_m$$

The alternative hypothesis states that at least one population mean differs from the rest. So, the null hypothesis supports that all samples come from populations with equal mean values and the alternative hypothesis claims that at least one sample comes from a population with significantly different mean. In simpler terms, the null hypothesis states that there is no difference between the samples under comparison, while the alternative hypothesis states that at least one sample is significantly different from the others.

The null hypothesis defined above can be tested by means of multiple t-tests, i.e. by conducting independent samples t-tests between all possible pairs of samples. However, a more elegant approach is the One-way Analysis of Variance (ANOVA), proposed by Sir Ronald Aylmer Fisher. In this approach all samples are analysed simultaneously and the null hypothesis is tested via the definition of proper variances between and within samples.

The principles of the method

Consider the following samples, which for simplicity have the same size:

Sample 1	Sample 2	...	Sample 3
x_{11}	x_{21}	...	x_{m1}
x_{12}	x_{22}	...	x_{m2}
...	...	...	...
x_{1n}	x_{2n}	...	x_{mn}
$\bar{x}_1$	$\bar{x}_2$	...	$\bar{x}_m$
s_1^2	s_2^2	...	s_m^2

In this dataset we can define two variances:

Between-groups variance:

$$s_b^2 = \frac{n \sum_{i=1}^m (\bar{x}_i - \bar{x})^2}{m - 1}$$

where $\bar{x}$ is the mean value of all sample values, n is the sample size, and m is the number of samples.

Within-groups variance:

$$s_w^2 = \frac{\sum_{i=1}^m s_i^2}{m}$$

where s_i^2 is the variance of the sample i .

When all samples come from the same population, the variances s_b^2, s_w^2 tend to become equal to the population variance, σ^2 , and, therefore, the ratio $F = s_b^2/s_w^2$ tends to 1.

However, if the samples come from different populations which have the same variance, $\sigma_1^2 = \dots = \sigma_m^2$, then if the populations have different means, the variance within the samples is not affected by the diversity of means, whereas the variance between the samples increases because the quantity $(\bar{x}_i - \bar{x})^2$ increases. Therefore, when there are samples with statistically significant differences in their means, F takes values greater than 1. The greater the differences in mean values, the greater the value of F . Taking into account these observations, and the fact the F follows a known distribution, the F or Fisher distribution with $df_1=m-1$ and $df_2=m(n-1)$ degrees of freedom, we define F as the test statistic of ANOVA:

$$F = s_b^2/s_w^2$$

Assumptions of ANOVA

There are three main preconditions to apply ANOVA:

1. There must be homogeneity of variance, i.e. there should be no statistically significant difference in the variance of the populations from which the samples derive.
2. Samples should follow the normal distribution, i.e. they should come from normal populations.
3. Samples should be independent.

Note that when sample sizes are equal, ANOVA is quite robust to violations of normality and homogeneity of variance. However, when sample sizes are unequal, deviations from normality or/and homogeneity of variance may yield misleading results. In this case non-parametric tests, such as those described in section 8, are necessary.

Post hoc procedures and the multiple comparisons problem

When the null hypothesis is rejected, we conclude that there is a statistically significant difference among the mean values of the samples under study. However, we do not know between which specific pairs of samples the significant differences are traced. Thus, we should perform *post hoc tests*. The simplest approach is to perform t-tests between each pair of samples. However, in this case the so-called *multiple comparisons problem* emerges.

In particular, when we have to perform multiple comparisons, there is increased probability of falsely rejecting the null hypothesis in one of these tests. It has been proven that if N is the number of pairwise comparisons between samples, then the probability of falsely rejecting the null hypothesis in one out of the N comparisons increases from α to $1 - (1 - \alpha)^N$. For example, when five samples are tested, we need to make 10 pairwise comparisons, and therefore the probability of falsely rejecting the null hypothesis in 1 out of the 10 comparisons increases from $\alpha = 0.05$ to $1 - 0.95^{10} = 0.4012$. In other words, the probability of making a mistake increases from 5% to 40.1%!

Therefore, when multiple comparisons are made, it is imperative either to lower the significance level α or to correct the obtained p-values.

A common approach is to apply the *Bonferroni correction* where the significance level is calculated from:

$$\alpha = \frac{0.05}{N}$$

For example, for 10 comparisons, α should be corrected to $0.05/10 = 0.005$, thus the probability of falsely rejecting the null hypothesis in 1 out of the 10 comparisons is $1-(1-0.005)^{10} = 1-0.995^{10} = 0.04889 = 4.9\%$, instead of 40.1%. The Bonferroni correction is equivalent to preserving the significance level to $\alpha = 0.05$ and multiplying all obtained p-values by N.

A better approach is to apply the *Holm-Bonferroni* method. Based on this method, the p-values are arranged from the smallest to the largest, $p\text{-value}(1) < p\text{-value}(2) < \dots < p\text{-value}(N)$. Subsequently, starting from the smallest, these values are multiplied by N, N-1, N-2, ..., 1, where N is the number of pairwise comparisons.

In R we may use the `p.adjust(p, method)` function, which adjusts p-values for multiple tests. Here, *p* is a vector of p-values and *method* is the correction method used. The methods are the Bonferroni correction ("bonferroni"), the Holm-Bonferroni correction ("holm"), and the methods proposed by Hochberg ("hochberg"), Hommel ("hommel"), and Benjamini-Hochberg ("fdr"). For example, assume we have conducted six t-tests and obtained p-values of 0.001, 0.008, 0.015, 0.04, 0.075, and 0.10.

To correct these values using the Holm-Bonferroni method, we can use the commands:

```
pvals <- c(0.001, 0.008, 0.015, 0.04, 0.075, 0.10)
p.adjust(pvals, "holm")
```

We obtain:

```
0.006 0.040 0.060 0.120 0.150 0.150
```

which shows that only two out of six p-values are statistically significant.

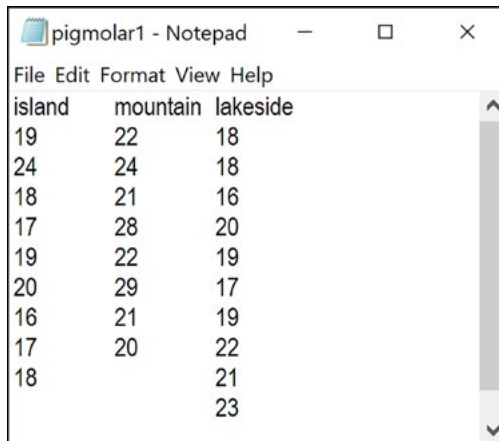
Therefore, for *post hoc tests*, we may perform t-tests between each pair of samples and correct the p-values. However, there are better approaches depending on the homogeneity of variance. If the criterion of homogeneity of variance is kept, the *Tukey test* is the best option, while in the opposite case, the *Games-Howell test* should be preferred.

An application of ANOVA in R

Consider that we have measured the length of the first molar in pigs from archaeological sites with different natural environment (island, mountain, lakeside). We expect that the tooth size will be affected by the natural environment and the available food resources in the sense that tougher food-stuff will lead to the evolution of bigger teeth. We want to examine if tooth size is indeed differentiated significantly among sites in order to determine the impact of the environment on nutrition and subsequent dental evolution. The data is given in .txt format in [Figure 7.1](#).

We observe that we have three different samples to compare and these samples are independent. This means that we have included different pigs in each sample; we are not studying the same pigs in the mountain, lakeside and island environment.

Figure 7.1.
Data on pig
molar length
(in mm)



island	mountain	lakeside
19	22	18
24	24	18
18	21	16
17	28	20
19	22	19
20	29	17
16	21	19
17	20	22
18		21
		23

To apply one-way ANOVA, first we input this file to R and run the Shapiro-Wilk test of normality:

```
x <- read.table("pigmolar1.txt", head=1,
  sep="\t")

shapiro.test(x$island); shapiro.test(x$mountain);
shapiro.test(x$lakeside)
```

Note that in the second line of the above code, we have placed two commands in the same line, separating them by semi-colons. The results are given in Figure 7.2, where it can be seen that none of the three samples appears to violate the normality assumption (p-value always > 0.05).

Now we have to test the homogeneity of variances assumption. Because we have more than two samples, we cannot use the F-test but only the Levene test. Therefore, we have to organize the data in two vectors, *length* and *group*, where *length* includes all the sample values and *group* is a grouping vector. This data layout has been adopted in the tab delimited *pigmolar2.txt* file.

Therefore, to run Levene's test, we can use the commands:

```
library(car)

xa <- read.table("pigmolar2.txt", head=1,
  sep="\t")

leveneTest(xa$length, factor(xa$group),
  "mean")
```

We obtain the results given in Figure 7.3, which show that the homogeneity of variance assumption cannot be rejected.

Figure 7.2.
Results of the
normality tests

```
> x <- read.table("pigmolar1.txt", head=1, sep="\t")
> shapiro.test(x$island); shapiro.test(x$mountain); shapiro.test(x$lakeside)

      Shapiro-Wilk normality test

data:  x$island
W = 0.86621, p-value = 0.112

      Shapiro-Wilk normality test

data:  x$mountain
W = 0.83874, p-value = 0.0731

      Shapiro-Wilk normality test

data:  x$lakeside
W = 0.97269, p-value = 0.9146
```

```
> library(car)
> xa <- read.table("pigmolar2.txt", head=1, sep="\t")
> leveneTest(xa$length, factor(xa$group), "mean")
Levene's Test for Homogeneity of Variance (center = "mean")
      Df F value Pr(>F)
group  2   1.3216 0.2854
      24
```

Figure 7.3.
Results of
the test of
homogeneity
of variance

We can now run ANOVA. The basic function that we can use is the `aov(x ~ y)` function, where `x` is a variable with all sample values and `y` is the grouping variable. Note that `y` is a factor variable. The `aov()` function is associated with the `summary()` function to obtain summary statistics, and the `TukeyHSD()` function to run pairwise tests using the Tukey criterion.

Thus, we can use the following commands:

```
f <- aov(xa$length ~ factor(xa$group))
summary(f)
TukeyHSD(f)
```

The output is given in Figure 7.4. It is seen that `Pr` (which is the p-value) is equal to 0.00241; it is smaller than 0.05, thus, there is a significant difference among the pigs from the different natural environments. Looking at the pairwise comparisons, we observe that the significant difference is found between groups 2 and 1 (p-value = 0.00339) and groups 3 and 2 (p-value = 0.0092). Thus, the mountain pigs are significantly different to the island and lakeside ones.

```
> f <- aov(xa$length ~ factor(xa$group))
> summary(f)
              Df Sum Sq Mean Sq F value    Pr(>F)
factor(xa$group)  2   109.7    54.83    7.834 0.00241 **
Residuals        24   168.0     7.00
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> TukeyHSD(f)
Tukey multiple comparisons of means
 95% family-wise confidence level

Fit: aov(formula = xa$length ~ factor(xa$group))

$`factor(xa$group)`
      diff      lwr      upr      p adj
2-1  4.7083333  1.498050  7.9186169 0.0033968
3-1  0.6333333 -2.402240  3.6689067 0.8618685
3-2 -4.0750000 -7.208837 -0.9411627 0.0092482
```

Figure 7.4.
ANOVA
output

We can visualize these results via the boxplots of the samples of Figure 7.1. To create these plots, we can use the commands:

```
library(ggplot2)
xa <- read.table("pigmolar2.txt", head=1,
sep="\t")

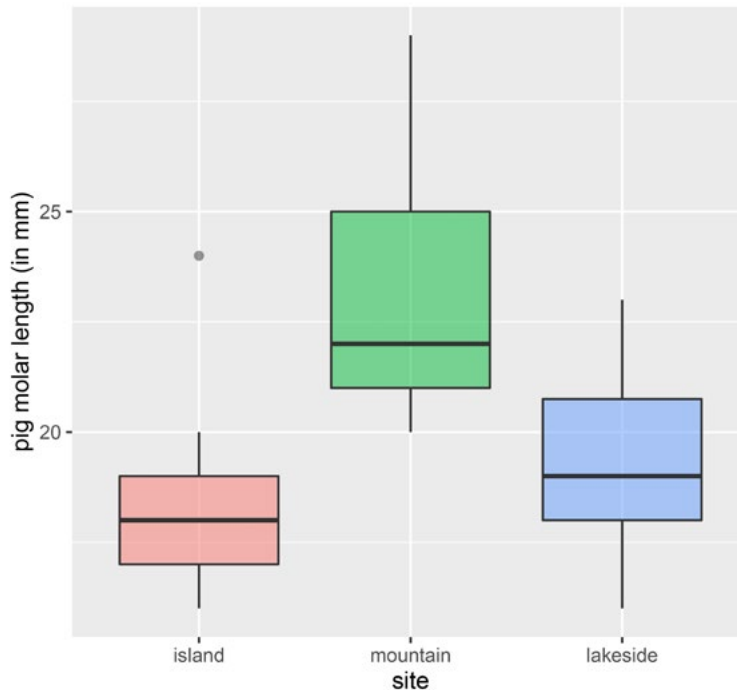
xa$group <- factor(xa$group, levels=c(1,2,3),
labels=c("island", "mountain", "lakeside"))

bp <- ggplot(xa, aes(x=group, y=length,
fill=group)) + geom_boxplot(alpha=0.5) +
xlab("site") + ylab("pig molar length (in mm)") +
theme(legend.position='none')

bp
```

We obtain the boxplots of Figure 7.5, which clearly show that the mountain pigs are different from the island and lakeside ones.

Figure 7.5.
Boxplots of
the samples of
Figure 7.1



Note that if we want to run the Games-Howell test for post hoc tests, we should install the *pairwiseComparisons* package and use the commands:

```
library(pairwiseComparisons)
```

```
xa <- read.table("pigmolar2.txt", head=1, sep="\t")
```

```
pairwise_comparisons(xa, x = group, y = length, type = "parametric", var.equal = FALSE, paired = FALSE, p.adjust.method = "none")
```

The results are given in Figure 7.6, where we observe that the p-values for the pairwise comparison of samples 1-2, 1-3 and 2-3 are 0.0158, 0.820 and 0.0316, respectively. Thus, they are similar to the Tukey test.

Figure 7.6.
Games-
Howell test

```
> library(pairwiseComparisons)
> xa <- read.table("pigmolar2.txt", head=1, sep="\t")
> pairwise_comparisons(xa, x = group, y = length, type = "parametric", var.equal = FALSE, paired = FALSE, p.adjust.method = "none")
# A tibble: 3 x 11
  group1 group2 statistic p.value alternative method distribution p.adjustment
  <chr> <chr>      <dbl>   <dbl> <chr>      <chr> <chr>      <chr>
1 1      2        4.66   0.0158 two.sided Games~ q      none
2 1      3        0.854   0.820 two.sided Games~ q      none
3 2      3       -4.16   0.0316 two.sided Games~ q      none
```

7.2. REPEATED MEASURES ANOVA

The repeated measures ANOVA is the extension of the parametric paired t-test to three or more dependent samples. As any ANOVA, repeated measures ANOVA tests the equality of means. Therefore, if we test m samples, the null hypothesis states that the population means are equal:

$$H_0: \mu_1 = \mu_2 = \mu_3 = \dots = \mu_m$$

The alternative hypothesis states that at least one mean is different to another mean.

The assumptions of repeated measures ANOVA are similar to those of simple one-way ANOVA, except that independence of the samples is not required. Thus, there must be homogeneity of variance and the samples should follow the normal distribution. An additional critical assumption, known as the assumption of *sphericity*, requires that the variances of the differences between all combinations of related groups must be equal. Repeated measures ANOVA is particularly sensitive to violations of this assumption, which may be tested using *Mauchly's test*.

An application of repeated measures ANOVA in R

The maximum cranial breadth (XCB) is defined as 'the maximum cranial breadth perpendicular to the median sagittal plane taken above the supramastoid crests'. We want to test how clear this definition is for undergraduate students who have no previous experience in craniometry. For this purpose, the above definition was given to three students without further clarifications. The students were asked to measure the maximum cranial breadth in nine different crania and the obtained data are given in **Table 7.1**.

This is a typical case of repeated measures ANOVA since we have to test three dependent samples. Thus, we first test the normality of these samples and the homogeneity of variance.

Note that these data are in the *XCB.txt* file, which is a tab delimited file with three columns with names *Student1*, *Student2* and *Student3*.

Student 1	Student 2	Student 3
13.9	13.5	13.5
12.5	12.2	12
14.3	14.4	14
14	13.9	13.5
13.2	13	12.7
13	13.2	13
13.6	13.6	13.4
12.7	12.8	12.4
13.4	13.4	13.1

Table 7.1.
Maximum
cranial breadth
of nine crania
(in cm)
measured by
three students

For the normality test, we use the commands:

```
data <- read.table("XCB.txt", head=1, sep="\t")
shapiro.test(data$Student1);
shapiro.test(data$Student2);
shapiro.test(data$Student3)
```

and obtain the results of **Figure 7.7**. We observe that the p-values of the normality tests are 0.923 (Student1), 0.999 (Student2), 0.915 (Student3), which shows that statistically significant deviations from normality are not demonstrated for any of the compared samples.

Figure 7.7.
Normality
tests results

```
> data <- read.table("XCB.txt", head=1, sep="\t")
> shapiro.test(data$Student1); shapiro.test(data$Student2); shapiro.test(

      Shapiro-Wilk normality test

data:  data$Student1
W = 0.97354, p-value = 0.9232

      Shapiro-Wilk normality test

data:  data$Student2
W = 0.99299, p-value = 0.9989

      Shapiro-Wilk normality test

data:  data$Student3
W = 0.97241, p-value = 0.9145
```

For the test of homogeneity of variance, we can create the following vectors:

```
x <- c(13.9, 12.5, 14.3, 14, 13.2, 13, 13.6, 12.7,
13.4, 13.5, 12.2, 14.4, 13.9, 13, 13.2, 13.6, 12.8,
13.4, 13.5, 12, 14, 13.5, 12.7, 13, 13.4, 12.4, 13.1)

y <- c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2,
3, 3, 3, 3, 3, 3, 3, 3, 3)
```

Note that vector x can be easily created using the command:

```
x <- c(as.matrix(data))
```

Then we can use the commands:

```
library(car)

leveneTest(x, factor(y), "mean")
```

which yield the results of Figure 7.8. It is seen that the p-value of the Levene test is 0.996, indicating that the assumption of the homogeneity of variance is not violated.

Figure 7.8.
Levene's test
results

```
> x<-c(13.9, 12.5, 14.3, 14, 13.2, 13, 13.6, 12.7, 13.4, 13.5, 12.2, 14.4, 13.9, 13, 13.
> y<-c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3)
> library(car)
> leveneTest(x,factor(y),"mean")
Levene's Test for Homogeneity of Variance (center = "mean")
      Df F value Pr(>F)
group  2  0.0039 0.9961
      24
```

Repeated measures ANOVA may be conducted using the `Anova()` function of the `car` package via the commands:

```
library(car)
studentLevels <- c(1, 2, 3)
studentFactor <- factor(studentLevels)
studentFrame <- data.frame(studentFactor)
studentBind <- cbind(data$Student1,
data$Student2, data$Student3)
studentModel <- lm(studentBind ~ 1)
analysis <- Anova(studentModel,
idata = studentFrame,
idesign = ~studentFactor)
summary(analysis, multivariate=F)
```

We obtain the results presented in Figure 7.9, which show the following: The p-value of Mauchly's test is equal to $0.436 > 0.05$, indicating that the null hypothesis that the variances of the differences between all combinations of related groups are equal cannot be rejected. Based on this result, we select the value 4.751×10^{-5} for the p-value of the repeated measures ANOVA appearing in the panel *Univariate Type III Repeated-Measures ANOVA Assuming Sphericity*. If the assumption of sphericity had not been met, we would have reported one of the p-values based on the Greenhouse-Geisser or Huynh-Feldt Correction. These values are indicated as $\text{Pr}(> F[\text{GG}])$ and $\text{Pr}(> F[\text{HF}])$, respectively. Note that if the epsilon value (GG eps) of Greenhouse-Geisser is greater than 0.75, we should use the Huynh-Feldt correction.

```
> library(car)
> studentLevels <- c(1, 2, 3)
> studentFactor <- as.factor(studentLevels)
> studentFrame <- data.frame(studentFactor)
> studentBind <- cbind(data$Student1, data$Student2, data$Student3)
> studentModel <- lm(studentBind ~ 1)
> analysis <- Anova(studentModel, idata = studentFrame, idesign = ~studentFactor)
Note: model has only an intercept; equivalent type-III tests substituted.
> summary(analysis, multivariate=F)

Univariate Type III Repeated-Measures ANOVA Assuming Sphericity

              Sum Sq num Df Error SS den Df  F value    Pr(>F)
(Intercept)  4752.1      1  9.0733      8 4189.966 3.609e-12 ***
studentFactor    0.6      2  0.2267     16   19.765 4.751e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Mauchly Tests for Sphericity

              Test statistic p-value
studentFactor    0.78893 0.43614

Greenhouse-Geisser and Huynh-Feldt Corrections
for Departure from Sphericity

              GG eps Pr(>F[GG])
studentFactor 0.82571 0.0001774 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

              HF eps  Pr(>F[HF])
studentFactor 1.013051 4.750893e-05
Warning message:
In summary.Anova.mlm(analysis, multivariate = F) : HF eps > 1 treated as 1
```

Figure 7.9.
Results of
repeated
measures
ANOVA

In the present example, the p -value = 4.751×10^{-5} of the repeated measures ANOVA shows that there are statistically significant differences among the samples. To identify between which pair(s) of samples the differences are found, we may use the `pairwise.t.test(x, group, p.adj="holm", paired=T)` function, which presumes that we have arranged the data as for the `leveneTest()` function.

Thus, we use the command:

```
pairwise.t.test(x, y, p.adj="holm", paired=T)
```

and obtain the results of Figure 7.10.

Figure 7.10.
Results of
pairwise
comparisons

```
> pairwise.t.test(x, y, p.adj="holm", paired=T)

      Pairwise comparisons using paired t tests

data:  x and y

      1      2
2 0.34659 -
3 0.00092 0.00092

P value adjustment method: holm
```

It is seen that p -value (1-2) = 0.34659, p -value (1-3) = 0.00092, and p -value (2-3) = 0.00092. That is, the measurements obtained by student 3 are significantly differentiated from those obtained by the other two students. This result implies either that student 3 has misinterpreted the definition of maximum cranial breadth, or that students 1 and 2 have made the same misinterpretation of the definition, while student 3 is correct. Another possibility is that all three students have misinterpreted the definition but in different ways.

Finally, we should stress that contrary to one-way ANOVA, in the case of repeated measures ANOVA the boxplot approach may be misleading to the viewer. Figure 7.11 shows the boxplots for the example under study.

They have been created using the commands on the right. We observe that they do not show significant differences among the samples.

```
x <- c(13.9, 12.5, 14.3, 14, 13.2, 13, 13.6, 12.7,
      13.4, 13.5, 12.2, 14.4, 13.9, 13, 13.2, 13.6, 12.8,
      13.4, 13.5, 12, 14, 13.5, 12.7, 13, 13.4, 12.4, 13.1)

y <- c(1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2,
      3, 3, 3, 3, 3, 3, 3, 3, 3)

data <- data.frame(cbind(x,y))

data$y <- factor(data$y, levels=c(1,2,3),
labels=c("student 1", "student 2", "student 3"))

library(ggplot2)

bp <- ggplot(data, aes(x= y, y= x, fill= y)) +
  geom_boxplot(alpha=0.5) +
  xlab("student") +
  ylab("maximum cranial breadth (in cm)") +
  theme(legend.position='none')

bp
```

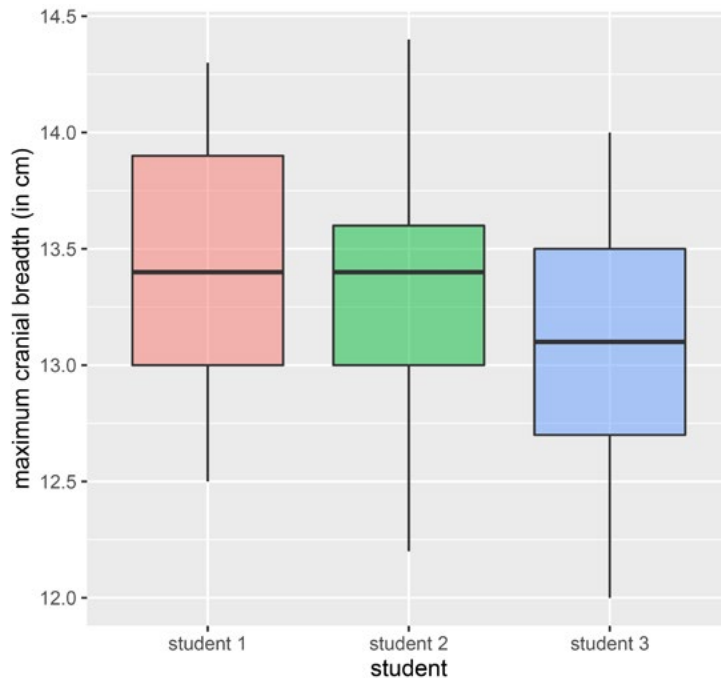


Figure 7.11.
Boxplots for
the samples
of Table 7.1

In this case, it is better to create the boxplots of the differences between all pairs of samples.

The commands that can be used and result in the boxplots of Figure 7.12 are:

```
x <- c(13.9, 12.5, 14.3, 14, 13.2, 13, 13.6, 12.7, 13.4, 13.5,
12.2, 14.4, 13.9, 13, 13.2, 13.6, 12.8, 13.4, 13.5, 12, 14,
13.5, 12.7, 13, 13.4, 12.4, 13.1)

y <- c(1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3,
3, 3, 3, 3, 3)

data <- data.frame(cbind(x,y))
x1 <- subset(data,data[,2]==1)
x2 <- subset(data,data[,2]==2)
x3 <- subset(data,data[,2]==3)

x <- c(x1[,1] - x2[,1], x1[,1] - x3[,1], x2[,1] - x3[,1])
y <- c(1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3,
3, 3, 3, 3, 3)

data <- data.frame(cbind(x,y))
data$y <- factor(data$y, levels=c(1,2,3),
labels= c("1-2", "1-3", "2-3"))

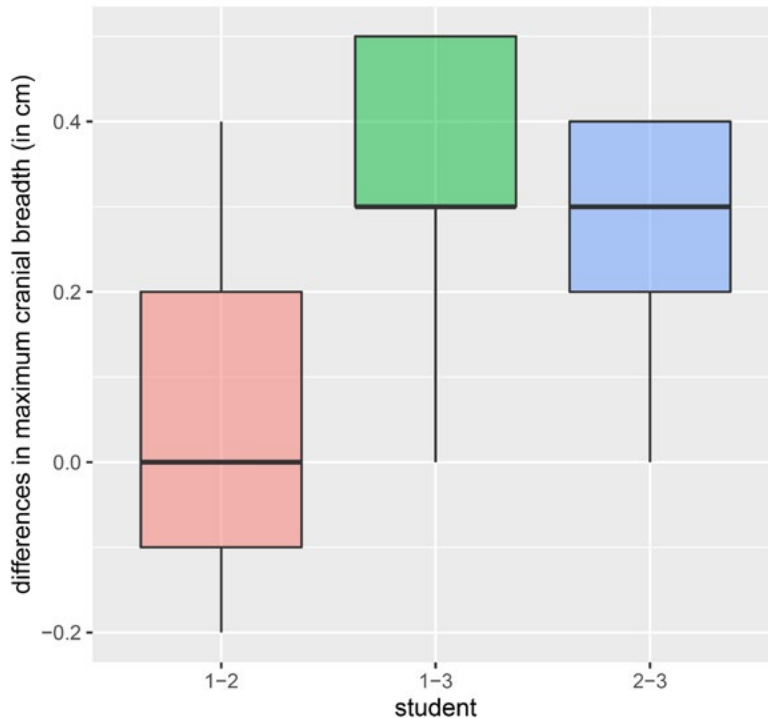
library(ggplot2)

bp <- ggplot(data, aes(x= y, y= x, fill= y)) +
geom_boxplot(alpha=0.5) +
xlab("student") +
ylab("differences in maximum cranial breadth (in cm)") +
theme(legend.position='none')

bp
```

It is seen that, in agreement with the results obtained from the pairwise comparisons of Figure 7.10, the measurements obtained by student 3 are different from those obtained by the other two students.

Figure 7.12.
Boxplots
for the
differences
between
all pairs of
samples of
Table 7.1



7.3. ANALYSIS OF COVARIANCE (ANCOVA)

The *one-way analysis of covariance (ANCOVA)* is an extension of the one-way ANOVA to incorporate a group variable and a covariate, i.e. a continuous independent variable. From a statistical point of view, ANCOVA may be considered as a combination of ANOVA and regression analysis (see section 10). In fact, it is a *general linear model*, which is a regression model that includes categorical and continuous variables as independent variables. The continuous independent variables are called *covariates*, whereas the categorical independent variables are called *factors*. Thus, ANCOVA gives prediction equations for the various levels of a categorical variable. In addition, it examines whether the means of independent samples are equal for the various levels of a categorical variable.

Since ANCOVA is a combination of ANOVA and regression analysis, ANCOVA assumes the normality of residuals (i.e. the difference between observed and predicted values). Furthermore, there are three important additional assumptions:

1. The variance of the dependent variable is equal across groups (homogeneity of variance).
2. The dependent variable varies linearly with each covariate at each level of the categorical variable.
3. The lines of the above relationships have the same slope (homogeneity of regression slopes).

These assumptions are clarified in the example below.

As an example, we want to compare the level of osteoarthritis among three groups of skeletons. However, osteoarthritis is largely dependent upon the age of the individuals since older individuals will exhibit more osteoarthritis. For this reason, we need to include age as a covariate in our tests. The data we are going to analyse is saved in file **ANCOVA.txt**, which is a tab delimited file with three columns named Group, Osteoarthritis, Age. Note that the variable Osteoarthritis expresses the level of osteoarthritis of the skeletons and it is measured on an ordinal scale from 0 to 5. However, in the analysis below using ANCOVA, it will be treated as a continuous variable.

To start the analysis, first we input the data in R using the command:

```
data <- read.table("ANCOVA.txt", head=1, se="\t")
```

In order to visualize the relationship between osteoarthritis and age, we create the scatterplot with regression lines of osteoarthritis against age for each of the three groups of skeletons.

This plot is shown in Figure 7.13 and it has been obtained using the following commands:

```
library(ggplot2)
p <- ggplot(data, aes(x=Age, y=Osteoarthritis,
color=factor(Group), shape=factor(Group)))
p1 <- p + geom_point()
p2 <- p1 + geom_smooth(method=lm,
aes(fill=factor(Group)))
p3 <- p2 + xlab("age") + ylab("osteoarthritis")
p4 <- p3 + theme_classic() +
theme(legend.title=element_blank())
p4
```

It is seen that the dependent variable, osteoarthritis, varies linearly with age at each level of the categorical variable (group) but the lines of these relationships do not have the same slope. That is, it seems that the dataset under examination fulfils the second assumption of ANCOVA but not the third one (homogeneity of regression slopes). However, in Figure 7.13 we observe that the 95% confidence bands are very wide and, therefore, it may be possible to make the regression lines parallel to each other.

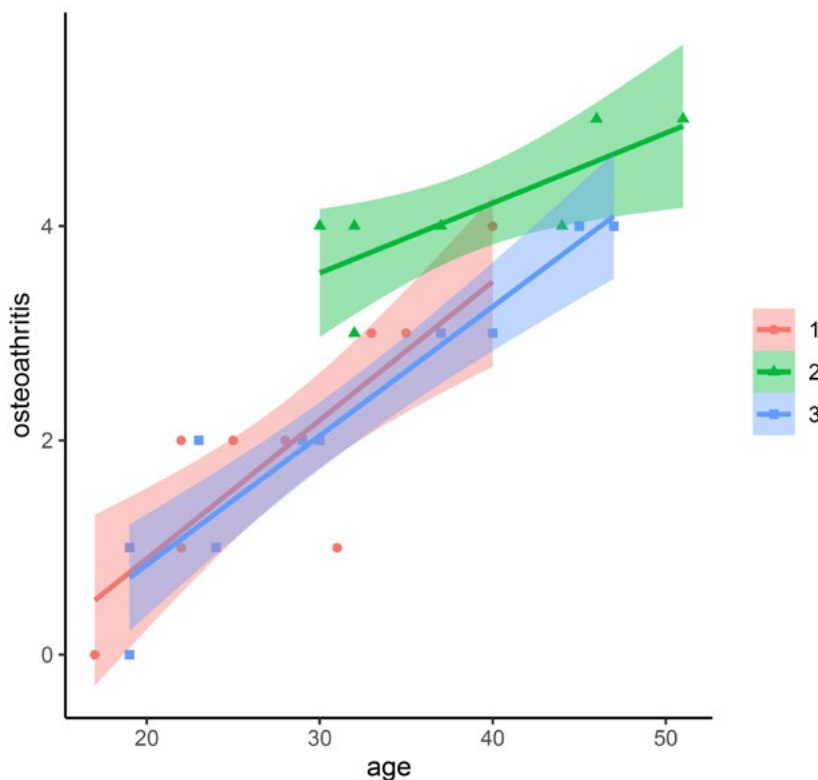


Figure 7.13.
Scatterplot and
regression lines
of osteoarthritis
against age
for each of the
three groups of
skeletons

Next, we examine the first assumption of ANCOVA, which is the homogeneity of variance.

For this purpose, we use
the commands:

```
library(car)
leveneTest(data$Osteoarthritis, factor(data$Group),
center=mean)
```

Figure 7.14.
Levene's test
results

```
> library(car)
> leveneTest(data$Osteoarthritis, factor(data$Group), center = mean)
Levene's Test for Homogeneity of Variance (center = mean)
      Df F value Pr(>F)
group  2    2.051 0.1489
      26
```

The output is given in Figure 7.14 and shows that Levene's test is non-significant, $p = 0.1489 > 0.05$. This means that for these data, the assumption of the homogeneity of variance is not violated.

Now, we check whether the covariate and any independent variables are independent. Our covariate is age-at-death and, thus, we run an ANOVA with Age as outcome and Group as predictor. If we get a significant result, we stop the analysis here.

To run this test, we use
the commands:

```
f <- aov(data$Age ~ factor(data$Group))
summary(f)
```

The output is given in Figure 7.15 and it is seen that the average age-at-death is not significantly different in the three groups ($p\text{-value} = 0.059$). This result means that it is appropriate to use age-at-death as a covariate in the analysis.

Figure 7.15.
ANOVA
output

```
> f <- aov(data$Age ~ factor(data$Group))
> summary(f)
              Df Sum Sq Mean Sq F value Pr(>F)
factor(data$Group)  2  485.1   242.6    3.158 0.0592 .
Residuals          26 1996.7    76.8
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Finally, we should check for homogeneity of regression slopes, which has been tentatively examined in Figure 7.13. For this purpose, we run the ANCOVA, including the interaction between the independent variable and the covariate. If this interaction is significant, we cannot assume homogeneity of regression slopes.

We use the command:

```
data$Group <- factor(data$Group)
slopetest <- aov(Osteoarthritis ~ Group + Age + Age:Group,
data=data)
summary(slopetest)
```

```

> data$Group<-as.factor(data$Group)
> slopetest <- aov(Osteoarthritis ~ Group + Age + Age:Group, data = data)
> summary(slopetest)
      Df Sum Sq Mean Sq F value    Pr(>F)
Group    2  24.077   12.038   44.234 1.31e-08 ***
Age       1  25.079   25.079   92.150 1.64e-09 ***
Group:Age  2    1.136    0.568    2.088   0.147
Residuals 23    6.260    0.272
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

Figure 7.16.
Summary
table for
ANCOVA
with
interaction
term

In the output of Figure 7.16 we are interested in the interaction term, so we look at the significance value of Group:Age. This value is not significant ($p = 0.147 > 0.05$), thus the assumption of homogeneity of regression slopes is not violated.

Now we can run the ANCOVA.

```

ANCOVAModel <- aov(Osteoarthritis ~ Group + Age, data=data)
summary(ANCOVAModel)

```

We will use again the `aov()` function but this time without the interaction term Age:Group.

We obtain the output of Figure 7.17. Looking at the significance value, we observe that both group membership and age-at-death significantly predict the levels of osteoarthritis (both p-values are much smaller than 0.05).

```

> ANCOVAModel<-aov(Osteoarthritis ~ Group + Age, data = data)
> summary(ANCOVAModel)
      Df Sum Sq Mean Sq F value    Pr(>F)
Group    2  24.077   12.038   40.69 1.37e-08 ***
Age       1  25.079   25.079   84.77 1.65e-09 ***
Residuals 25    7.396    0.296
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

Figure 7.17.
ANCOVA
output

If we look simply at the group means for osteoarthritis, which are easily calculated from the data-set, we can determine which groups are the ones that differ. However, we cannot interpret these group means because they have not been adjusted for the effect of the covariate. Thus, these original means tell us nothing about the group differences.

To get the adjusted means, we may use the `effect()` function in the *effects* package:

```

library(effects)
adjustedMeans <- effect("Group", ANCOVAModel, se=TRUE)
summary(adjustedMeans)

```

In the output of Figure 7.18 it is seen that the mean value of group 2 exceeds that of the other two groups when osteoarthritis is adjusted for the effect of age-at-death.

Figure 7.18.
Adjusted
means

```
> library(effects)
> adjustedMeans<-effect("Group", ANCOVAModel, se=TRUE)
> summary(adjustedMeans)
```

Group effect			
Group	1	2	3
	2.418075	3.413438	2.309367

Lower 95 Percent Confidence Limits			
Group	1	2	3
	2.067612	2.986597	1.954282

Upper 95 Percent Confidence Limits			
Group	1	2	3
	2.768539	3.840278	2.664452

Now, we can compute post hoc tests. Because we want to test differences between the adjusted means, we can use the `glht()` function in the *multcomp* package.

If we use Tukey post hoc tests, `library(multcomp)`

the commands are:

```
postHocs <- glht(ANCOVAModel, linfct=mcp(Group="Tukey"))
summary(postHocs)
```

Figure 7.19.
Post hoc
tests results

```
> library(multcomp)
> postHocs<-glht(ANCOVAModel, linfct = mcp(Group = "Tukey"))
> summary(postHocs)
```

Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Tukey Contrasts

Fit: aov(formula = Osteoarthritis ~ Group + Age, data = data)

Linear Hypotheses:

	Estimate	Std. Error	t value	Pr(> t)	
2 - 1 == 0	0.9954	0.2809	3.543	0.00429	**
3 - 1 == 0	-0.1087	0.2400	-0.453	0.89314	
3 - 2 == 0	-1.1041	0.2730	-4.045	0.00132	**

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Adjusted p values reported -- single-step method)

The output of Figure 7.19 shows that there is a significant difference between the adjusted means of groups 1 and 2, as well as groups 2 and 3 (both p-values < 0.05).

As already pointed out, ANCOVA examines whether the means of the group samples are equal for the various levels of a categorical variable and, in addition, it gives prediction equations for the various levels of a categorical variable.

The ANCOVA model can be obtained using the command:

Figure 7.20 depicts the model coefficients from which we obtain the ANCOVA model:

$$\text{Osteoarthritis} = -1.19915 + 0.99536 \cdot \text{Group2} - 0.10871 \cdot \text{Group3} + 0.11207 \cdot \text{Age}$$

Note that in order to apply this equation for:

- group 1 we should set Group2=0 and Group3=0,
- group 2 we should set Group2=1 and Group3=0, and
- group 3 we should set Group2=0 and Group3=1.

```
> summary.lm(ANCOVAModel)

Call:
aov(formula = Osteoarthritis ~ Group + Age, data = data)

Residuals:
    Min       1Q   Median       3Q      Max
-1.27509 -0.28374  0.05712  0.27662  0.84162

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.19915     0.38422  -3.121  0.00451 **
Group2       0.99536     0.28094   3.543  0.00158 **
Group3      -0.10871     0.24001  -0.453  0.65450
Age          0.11207     0.01217   9.207 1.65e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5439 on 25 degrees of freedom
Multiple R-squared:  0.8692,    Adjusted R-squared:  0.8535
F-statistic: 55.39 on 3 and 25 DF,  p-value: 3.487e-11
```

Figure 7.20.
Model
coefficients

Finally, in order to make the plot with the predicted by ANCOVA osteoarthritis against age for each of the three groups of skeletons, we can use the following commands that yield Figure 7.21.

```
y <- predict(ANCOVAModel)
#y is the predicted osteoarthritis values

data$Osteoarthritis <- y

g <- ggplot(data, aes(x=Age, y=Osteoarthritis,
color=factor(Group), shape=factor(Group)))

g1 <- g + geom_point()

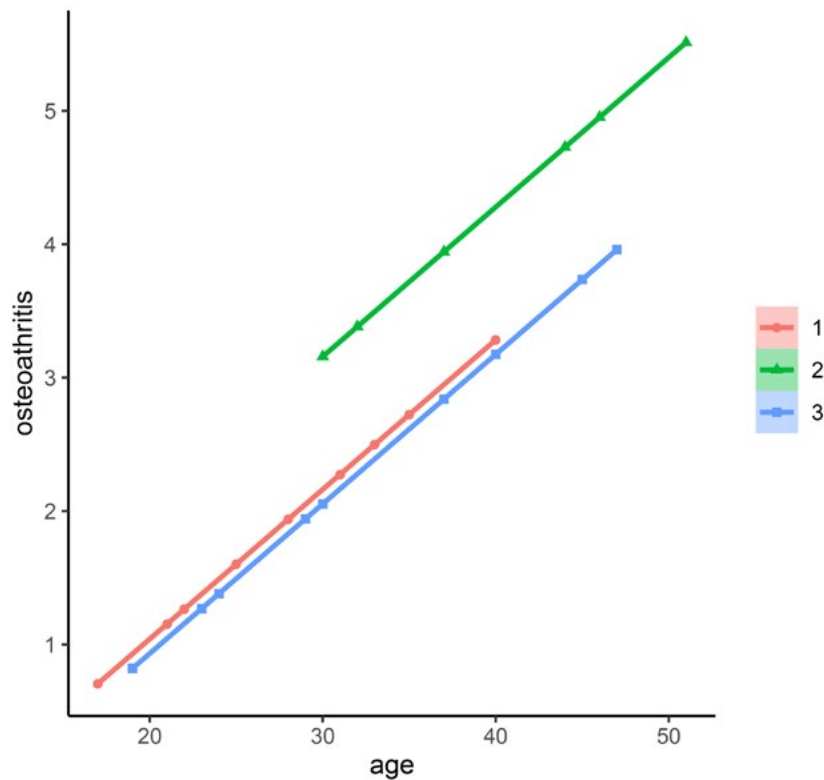
g2 <- g1 + geom_smooth(method=lm, aes(fill=factor(Group)))

g3 <- g2 + xlab("age") + ylab("osteoarthritis")

g4 <- g3 + theme_classic() +
theme(legend.title=element_blank())

g4
```

Figure 7.21.
Predicted by
ANCOVA
osteoarthritis
against age
for each of the
three groups of
skeletons



Now we can test the normality of the residuals assumption, using the commands:

```
data <- read.table("ANCOVA.txt", head=1, se="\t")
y<-predict(ANCOVAModel)
res <- data$Osteoarthritis-y
shapiro.test(res)
```

The results give a p-value equal to 0.5712, which suggests that the normality assumption for the residuals is not violated.

8.

NON-PARAMETRIC TESTS

The statistical tests we have used so far require that the data (or rather the populations from which the data derive) follow the normal distribution. If this precondition is violated, the obtained results may be wrong. In such cases we use *non-parametric tests*, in which the original raw data are transformed to *ranks* or *runs* (see section 4.4). **Table 8.1** shows the most important non-parametric tests that may be used in place of parametric tests.

Parametric test	Non-parametric test
Independent-samples t-test	Wilcoxon rank-sum test
Paired-samples t-test	Wilcoxon signed-rank test
ANOVA	Kruskal-Wallis test
Repeated measures ANOVA	Friedman test

Table 8.1.
Correspondence
between
parametric and
non-parametric
tests

8.1. WILCOXON RANK-SUM TEST

The *Wilcoxon rank-sum test*, also called *Mann-Whitney test*, compares two independent samples using the mean rank. In mathematical terms, the Wilcoxon rank-sum test examines if two samples come from the same population or not. Thus, the statistical hypotheses are:

H_0 : The samples come from the same population

H_1 : The samples come from different populations with $d_1 \neq d_2$, or $d_1 > d_2$, or $d_1 < d_2$ where d_1 and d_2 are the medians of the two populations

To run the Wilcoxon rank-sum test in R, we use the `wilcox.test(x, y, paired=FALSE)` function, where x , y are numeric vectors of the data values of the two samples, and *paired* is a logical indicating whether we want a paired test or not. Thus, consider the samples depicted in **Figure 8.1**, which again show the body diameter (in cm) of jugs in two sites but now the data violate the normality assumption.

siteA	siteB
10	17.1
25.6	18.3
28.1	18.4
12.3	15.2
11.5	17
11.9	17.3
14.4	27.7
10.2	24.4
16.6	27.7
17.3	28.9
17.7	30.5
30.5	27.7
20.4	
11	
13.7	

Figure 8.1.
Data on jug
diameter

To analyse these data, we first input them in R via the `read.table()` function and then test for normality:

```
data <- read.table("jug_diamNP.txt", head=1, se="\t")
shapiro.test(data$siteA); shapiro.test(data$siteB)
```

Figure 8.2.
Results of the normality tests

```
> data <- read.table("jug_diamNP.txt", head=1, se="\t")
> shapiro.test(data$siteA); shapiro.test(data$siteB)

      Shapiro-Wilk normality test

data:  data$siteA
W = 0.86643, p-value = 0.02992

      Shapiro-Wilk normality test

data:  data$siteB
W = 0.8438, p-value = 0.03081
```

In Figure 8.2 we observe that both variables violate the normality assumption ($p\text{-value} < 0.05$). Although the p -values do not indicate strong deviations from normality, we will compare the two assemblages using non-parametric tests.

For this purpose, we use the command:

```
wilcox.test(data$siteA, data$siteB, paired=FALSE)
```

which gives the results of Figure 8.3. It is seen that the p -value of the Wilcoxon rank-sum test is 0.023. Hence, there is a statistically significant difference between the size of the jugs in the two sites.

Figure 8.3.
Results of the Wilcoxon rank-sum test

```
> wilcox.test(data$siteA, data$siteB, paired = FALSE)

      Wilcoxon rank sum test with continuity correction

data:  data$siteA and data$siteB
W = 43, p-value = 0.02314
alternative hypothesis: true location shift is not equal to 0

Warning message:
In wilcox.test.default(data$siteA, data$siteB, paired = FALSE) :
  cannot compute exact p-value with ties
```

8.2. WILCOXON SIGNED-RANK TEST

The Wilcoxon signed-rank test is the non-parametric equivalent to the paired samples t-test. Therefore, it is used when the sample that consists of the differences between each pair of values does not follow the normal distribution. The statistical hypotheses are:

$$H_0: d = 0$$

$$H_1: d \neq 0 \text{ or } d > 0 \text{ or } d < 0$$

where d is the median of the population from which the $x_i - y_i$ differences between the two samples are derived. Thus, the null hypothesis is that the difference between the two paired samples is zero, whereas the alternative hypothesis claims that this difference is not zero.

As an example, consider that from the upper disturbed layers of a deposit we have gathered 15 objects. We sent each object to two different laboratories for dating and now we want to examine if the dates obtained from the two labs are significantly different or not. The dates (in years BP) obtained from the labs are given in **Table 8.2**.

Based on the data of **Table 8.2**, we create a tab delimited .txt file named *lab_dates* with two columns, the columns of **Table 8.2**. The names of the columns/variables are *Lab1* and *Lab2*.

Now, we use the following commands:

```
lab <- read.table("lab_dates.txt", head=1, sep="\t")
shapiro.test(lab$Lab1 - lab$Lab2)
wilcox.test(lab$Lab1, lab$Lab2, paired=TRUE)
```

Laboratory 1	Laboratory 2
875	900
663	600
1097	1021
2000	1572
15	44
729	729
3963	3201
562	488
734	724
1055	1055
999	1055
327	333
365	450
837	801
891	933

Table 8.2.
Dating data
from two
laboratories

The first command inputs the samples in R, the second command performs the Shapiro-Wilk test of normality for the differences between the two samples, and the third command runs the Wilcoxon signed-rank test. The output is given in Figure 8.4 and it is seen that the p-value of the normality test is $4.55 \times 10^{-5} \ll 0.05$, which shows strong deviations from normality. Thus, the use of a non-parametric test, such as the Wilcoxon signed-rank test, is necessary. This test gives the p-value 0.3636 (> 0.05), which shows that there does not appear to be a statistically significant difference between the results of the two labs.

Figure 8.4.
Results of
normality and
Wilcoxon
signed-rank
test

```
> lab <- read.table("lab_dates.txt", head=1, sep="\t")
> shapiro.test(lab$Lab1-lab$Lab2)

      Shapiro-Wilk normality test

data:  lab$Lab1 - lab$Lab2
W = 0.62702, p-value = 4.558e-05

> wilcox.test(lab$Lab1, lab$Lab2, paired = TRUE)

      Wilcoxon signed rank test with continuity correction

data:  lab$Lab1 and lab$Lab2
V = 59, p-value = 0.3636
alternative hypothesis: true location shift is not equal to 0

Warning message:
In wilcox.test.default(lab$Lab1, lab$Lab2, paired = TRUE) :
  cannot compute exact p-value with zeroes
```

8.3. KRUSKAL-WALLIS TEST

The Kruskal-Wallis test compares the mean ranks of more than 2 groups in order to test whether these groups come from different populations. It is a non-parametric counterpart of the one-way analysis of variance (ANOVA). In R it is implemented using the `kruskal.test(x, group)` function, where `x` is a vector with all samples and `group` is the grouping variable. For post hoc tests, we may use the `kruskalmc(x, y)` function of the *pgirmess* package.

As an example, consider that we have found stone blades in three different contexts within an archaeological site. Some blades were found in a 'domestic context', others were found as grave goods ('funerary context'), and others were found in a large building that seemed to have some symbolic function ('symbolic context'). The length of these blades is given in Table 8.3. We want to examine if the size of the blades is significantly different in these three contexts.

To analyse this dataset, we first create a tab delimited .txt file named *blade_length* with three columns, the columns of Table 8.3. The names of the columns/variables are *dc*, *fc* and *sc*, respectively.

Now, we use the following commands to test the normality:

```
data <- read.table("blade_length.txt", head=1, se="\t")
shapiro.test(data$dc); shapiro.test(data$fc);
shapiro.test(data$sc)
```

Domestic context	Funerary context	Symbolic context
5.6	2.2	8.2
6.3	3.1	9.1
5.7	4	8.7
7.7	3.5	8.8
8.9	3.9	6.8
3.8	4.4	6.5
4	4.8	4.9
6.1	4.2	9
7.2	3.7	8.3
7		8.5
8.3		7.9
7.4		

Table 8.3.
Blade length
(in cm)

We obtain the results shown in Figure 8.5, which show that the third sample exhibits some deviations from normality. Thus, we use the non-parametric Kruskal-Wallis test to examine differences among the samples.

```
> data <- read.table("blade_length.txt", head=1, se="\t")
> shapiro.test(data$dc); shapiro.test(data$fc); shapiro.test(data$sc)

      Shapiro-Wilk normality test

data:  data$dc
W = 0.95977, p-value = 0.7806

      Shapiro-Wilk normality test

data:  data$fc
W = 0.95521, p-value = 0.7471

      Shapiro-Wilk normality test

data:  data$sc
W = 0.84148, p-value = 0.03302
```

Figure 8.5.
Results of
normality
tests

To apply the Kruskal-Wallis test, we may use the commands:

```
x <- c(data$dc, data$fc, data$sc)
x <- x[!is.na(x)]

group <- c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3,
3, 3, 3, 3, 3, 3, 3, 3, 3)

kruskal.test(x, group)
```

The first command creates the vector `x` with all samples, the second command removes missing values from `x`, the third command creates the grouping variable `group`, and the last command runs the Kruskal-Wallis test. The results obtained are given in Figure 8.6. It is seen that the test statistic is $H = 18.961$ and the p-value is much smaller than 0.05 (p-value = $7.634e-05$). Therefore, we can conclude that there is a statistically significant difference in the length of the blades found in these three contexts.

Figure 8.6.
Results of the
Kruskal-Wallis
test

```
> x <- c(data$dc, data$fc, data$sc)
> x <- x[!is.na(x)]
> group <- c(1,1,1,1,1,1,1,1,1,1,1,1,2,2,2,2,2,2,2,2,3,3,3,3,3,3,3,3,3,3)
> kruskal.test(x, group)

Kruskal-Wallis rank sum test

data:  x and group
Kruskal-Wallis chi-squared = 18.961, df = 2, p-value = 7.634e-05
```

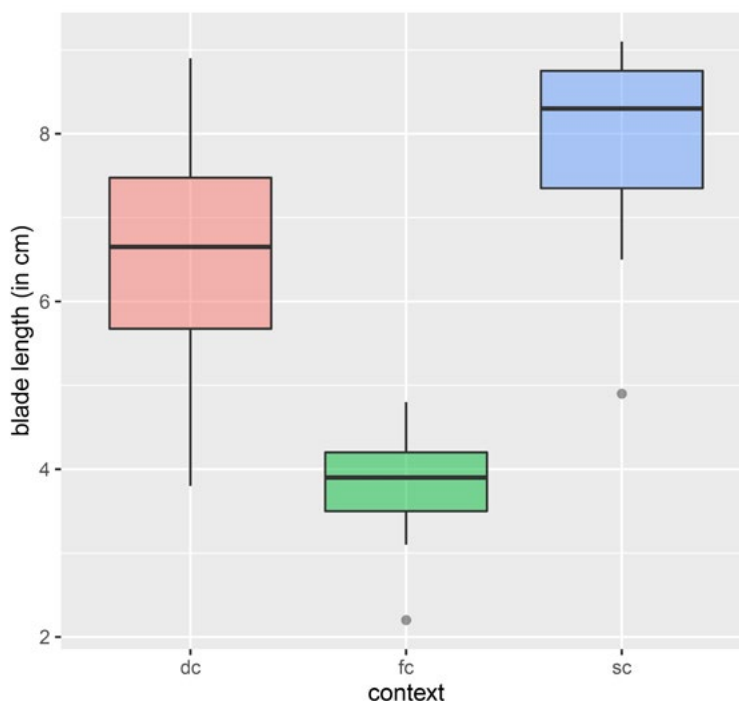
Note that, contrary to ANOVA, the Kruskal-Wallis test does not perform pairwise comparisons.

In order to see between which contexts the significant difference is found, we can plot boxplots using the commands:

```
library(ggplot2)
y <- c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3,
3, 3, 3, 3, 3, 3, 3)
xa <- data.frame(cbind(x,y))
xa$y <- factor(xa$y, levels=c(1,2,3), labels=c("dc", "fc", "sc"))
bp <- ggplot(xa, aes(x=y, y=x, fill=y)) +
  geom_boxplot(alpha=0.5) + xlab("context") +
  ylab("blade length (in cm)") + theme(legend.position='none')
bp
```

From the boxplots of Figure 8.7, we can conclude that the blades from context 2, that is, the funerary context, are notably smaller than those from the other two contexts.

Figure 8.7.
Boxplot of
blade length
in different
contexts



In addition, we can run a non-parametric post hoc test. Such a test is implemented using the `kruskalmc()` function from the *pgirmess* package:

```
library(pgirmess)
kruskalmc(x, y)
```

```
> library(pgirmess)
> kruskalmc(x, y)
Multiple comparison test after Kruskal-Wallis
p.value: 0.05
Comparisons
      obs.dif critical.dif difference
1-2 11.388889      9.902836      TRUE
1-3  6.893939      9.374298     FALSE
2-3 18.282828     10.093905      TRUE
```

Figure 8.8.
Results of non-parametric post hoc tests

The output (Figure 8.8) provides a column labelled *difference* that shows whether the observed difference is greater than the critical difference (TRUE) or not (FALSE). In other words, it shows whether the difference is significant or not. In the current example, the difference between groups 1-2 and 2-3 is bigger than the critical difference; hence they say TRUE, which means that these differences are significant.

8.4. FRIEDMAN TEST

The Friedman test is the non-parametric version of repeated measures ANOVA. Therefore, it should be adopted when we have doubts about the validity of the various assumptions on which parametric repeated measures ANOVA is based. In R this test is implemented using the `friedman.test()` function, whereas for post hoc tests we can use the function `posthoc.friedman.nemenyi.test()` of the *PMCMR* package. Therefore, we should install this package first.

Consider that we want to apply the Friedman test on the data that we used for parametric repeated measures ANOVA, i.e. the XCB measurement obtained by three students.

The commands that we can use are:

```
library(PMCMR)
data <- read.table("XCB.txt", head=1, sep="\t")
x <- as.matrix(data)
friedman.test(x)
posthoc.friedman.nemenyi.test(x)
```

We obtain the results of Figure 8.9. It is seen that there is an overall significant difference among the three students ($p\text{-value} = 0.0024$) and the pairwise comparisons suggest that the significant difference is actually found between student 3 and student 1 ($p\text{-value} = 0.009$) and student 3 and student 2 ($p\text{-value} = 0.018$), in agreement to the results of the repeated measures ANOVA.

Note that in this case we do not use the boxplots of the raw values to visualize differences among the samples. Instead, we may use boxplots of the differences between all pairs of samples (Figure 7.12).

Figure 8.9.
Results of the
Friedman test

```
> library(PMCMR)
> data <- read.table("XCB.txt", head=1, sep="\t")
> x<-as.matrix(data)
> friedman.test(x)

      Friedman rank sum test

data:  x
Friedman chi-squared = 12.062, df = 2, p-value = 0.002402

> posthoc.friedman.nemenyi.test(x)

      Pairwise comparisons using Nemenyi multiple comparison test
      with q approximation for unreplicated blocked data

data:  x

      Student1 Student2
Student2 0.970      -
Student3 0.009      0.018

P value adjustment method: none
```

9.

CORRELATION

In statistics, *correlation is any statistical relationship between two quantitative variables, irrespective of whether this relationship is causal or not*. The correlation usually describes the degree of the linear relationship between two variables, although it can be extended to matrices.

The simplest way to test for correlation is to make a scatterplot of the variables. Any point distribution except the random is an indication for the correlation between the two variables. If, in addition, the plot is linear, the slope is a measure of the correlation strength. However, if the relationship between the two variables is not linear, we need another approach to estimate the correlation strength and whether this is statistically significant or not. This is examined below.

CORRELATION IS NOT CAUSATION!

Even a strong correlation between variables does not suggest that one variable is the cause of the other variable. A typical example is the increase of the global temperature with the decrease of the number of pirates! (Figure 9.1)

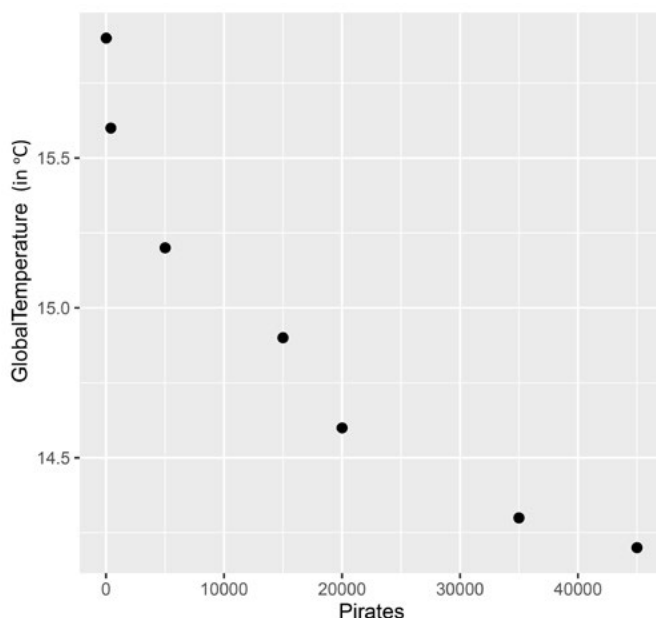


Figure 9.1.
Plot of global
temperature
vs. number of
pirates⁹

9.1. BIVARIATE CORRELATION

Bivariate correlation examines if two variables, x , y , exhibit a linear relationship or not. In other words, it explores if changes in the values of one variable produce linear changes in the values of the other variable. To define a strict correlation measure, we may start with the covariance, which is a measure of the joint variability of the two variables:

$$\text{cov}(x,y) = \frac{\sum(x - \bar{x})(y - \bar{y})}{n - 1}$$

⁹ Inspired from <https://www.sisense.com/blog/global-warming-caused-lack-pirates-bad-graph-lessons/>

A covariance of zero or close to zero indicates that the two variables are totally unrelated. If the variables increase in the same direction, the covariance is positive and it becomes negative if the variables change in opposite directions.

There is, however, one problem with covariance as a measure of the relationship between variables: it depends upon the scales of measurement used. That is, covariance is not a standardized measure. This means that we cannot compare covariances in an objective way; we cannot judge if a covariance is particularly large or small relative to another dataset unless both datasets are measured in the same units. To overcome the problem of dependence on the measurement scale, we need to convert the covariance into a standardized covariance. The standardized covariance is called *Pearson's correlation coefficient* (r) and is defined by the following equation:

$$r = \frac{\text{cov}(x, y)}{s_x s_y} = \frac{\sum (x - \bar{x})(y - \bar{y})}{(n - 1)s_x s_y}$$

where s_x is the standard deviation of the first variable, s_y is the standard deviation of the second variable, and n is the sample size.

It should be stressed again that Pearson's correlation coefficient is only used to check whether two random continuous variables are linearly related. To test the significance of r , we define the null and alternative hypotheses as:

$$H_0: r = 0$$

$$H_1: r \neq 0 \text{ or } H_1: r > 0 \text{ or } H_1: r < 0$$

To test the above null hypothesis, we use the following test statistic that follows the t distribution with $n-2$ degrees of freedom under the assumption that the two variables, x , y , follow the *bivariate normal distribution* (i.e. the normal distribution in two dimensions):

$$t = \frac{r\sqrt{n-2}}{\sqrt{1-r^2}}$$

If the data is ordinal or continuous but violate the normality assumption, i.e. if the data do not follow the bivariate normal distribution, we can use the *Spearman correlation coefficient*, ρ . Spearman's ρ first ranks the data and then applies Pearson's equation to the ranks. Alternatively, it can be computed from:

$$\rho = 1 - \frac{6\sum (r_x - r_y)^2}{n(n^2 - 1)}$$

where r_x are the values of the x variable converted to ranks and r_y are the values of the y variable converted to ranks.

The test statistic is the same as that used for Pearson's r but with ρ in place of r . This statistic follows the t distribution with $n-2$ degrees of freedom but without assumptions for the distribution of variables x , y .

Another non-parametric option is *Kendall's tau*, τ . Kendall's tau should be used rather than Spearman's coefficient when we have a small dataset, particularly when this dataset has a large number of tied ranks (i.e. many scores with the same rank). The Kendall τ coefficient is defined as:

$$\tau = \frac{n_C - n_D}{n(n-1)/2}$$

where n_C is the number of concordant pairs and n_D is the number of discordant pairs. This version of the formula assumes that there are no tied values within x or y . Note that tied observations are observations having the same value, which prohibits the assignment of unique rank numbers. Note also that the definition of concordant and discordant pairs is the following. Consider the pair of observations (X_1, Y_1) and (X_2, Y_2) . This pair is a *concordant pair* if $\text{sgn}(X_2 - X_1) = \text{sgn}(Y_2 - Y_1)$ and it is a *discordant pair* if $\text{sgn}(X_2 - X_1) = -\text{sgn}(Y_2 - Y_1)$, where sgn is the sign function, i.e. the function that gives the values -1, 0, 1 when its argument is a negative, zero or positive number, respectively.

For testing the null hypothesis $H_0: \tau = 0$, the Kendall τ coefficient is often used as a test statistic.

The correlation coefficient r , ρ or τ can take values in the range from -1 to 1. Negative values of the correlation coefficient mean that when x increases, y decreases and vice versa, values equal to 0 mean lack of linear correlation, and positive values indicate that when one variable increases, the other increases too.

For computing the correlation coefficients r , ρ , or τ and test the corresponding null hypothesis, we can use the `cor.test(x, y, method=c("pearson", "kendall", "spearman"))` function, where *method* indicates which correlation coefficient is to be used. To assess whether Pearson's correlation is appropriate or whether a non-parametric correlation coefficient should be used (Spearman or Kendall), we need to check whether the samples follow the bivariate normal distribution or not. In R there are several tests that can be used for this purpose and these are described in section 12 (multivariate normality).

As an example, consider that we want to determine if larger tombs are associated with bigger pots, a finding which might indicate that the deceased were individuals of higher status. The dataset is given in **Table 9.1**.

Tomb size (m ²)	Average capacity of pots per tomb (litres)
5.1	1.2
13.2	3.0
7.9	1.7
11.1	1.9
14.0	2.2
9.7	2.0
9.0	1.6
10.2	1.8
8.4	1.5
7.8	1.5
6.2	1.0
6.3	1.3
7.1	1.7
17.0	3.4

Table 9.1.
Data on tomb
size and
associated
pots capacity

We input the data in R using the command:

```
data <- read.table("tombsize_potcapacity.txt", head=1,
sep="\t")
```

and run the Pearson, Spearman, and Kendall tests via:

```
cor.test(data$Tomb_size, data$Pot_capacity,
method="pearson")

cor.test(data$Tomb_size, data$Pot_capacity,
method="spearman")

cor.test(data$Tomb_size, data$Pot_capacity,
method="kendall")
```

We obtain the results of Figures 9.2-9.4. It is seen that all correlation coefficients are positive and greater than 0.78, suggesting a high positive correlation between the two variables: as tomb size increases so does the capacity of the pots found inside. We also observe that all correlations are statistically significant (p-values $\ll 0.05$). Therefore, there is no need to test for bivariate normal distribution.

Figure 9.2.
Results of
Pearson's
correlation
coefficient

```
> cor.test(data$Tomb_size, data$Pot_capacity, method="pearson")

Pearson's product-moment correlation

data: data$Tomb_size and data$Pot_capacity
t = 8.4241, df = 12, p-value = 2.207e-06
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.7741717 0.9763377
sample estimates:
      cor
0.9248572
```

Figure 9.3.
Results of
Spearman's
correlation
coefficient

```
> cor.test(data$Tomb_size, data$Pot_capacity, method="spearman")

Spearman's rank correlation rho

data: data$Tomb_size and data$Pot_capacity
S = 35.076, p-value = 2.562e-06
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.9229097
```

Figure 9.4.
Results of
Kendall's
correlation
coefficient

```
> cor.test(data$Tomb_size, data$Pot_capacity, method="kendall")

Kendall's rank correlation tau

data: data$Tomb_size and data$Pot_capacity
z = 3.8986, p-value = 9.675e-05
alternative hypothesis: true tau is not equal to 0
sample estimates:
      tau
0.7889376
```

9.2. PARTIAL CORRELATION

The partial correlation is used in order to examine the correlation between two variables, while controlling for the effect of one or more other variables. In other words, we examine if two variables are correlated when the impact of other variables is considered constant. The variables may be continuous, categorical or a combination of continuous and categorical.

In the simple case where we examine the correlation between variables A and B while controlling for the effect of a third variable, C, the partial correlation coefficient is given from:

$$r_{AB,C} = \frac{r_{AB} - r_{AC}r_{BC}}{\sqrt{(1 - r_{AC}^2)(1 - r_{BC}^2)}}$$

where r_{AB} , r_{AC} and r_{BC} are the Pearson or Spearman correlation coefficients for the pair of variables A-B, A-C, and B-C, respectively. To check for statistical significance, we can use the test statistic:

$$t = \frac{r_{AB,C}\sqrt{n-3}}{\sqrt{1-r_{AB,C}^2}}$$

which follows the student distribution with $n-3$ degrees of freedom. The above relationships can be straightforwardly extended for the general case of the partial correlation between two variables, while controlling for the effect of more than one other variables.

In R we can perform partial correlation using the function of the *RVAideMemoire* package:

```
pcor.test(x, y, z, method)
```

In this function, x , y are numeric vectors of the compared variables, z is a numeric vector, or matrix, or data frame or list giving the controlling variables, and *method* is a character string that takes the values "pearson" or "spearman". For matrices, variables must be placed in columns.

As an example, assume that we want to re-examine the previous example but now we acknowledge that different tombs contain a different number of interred individuals. Therefore, we now want to examine the correlation between tomb size and average pottery capacity, while controlling for the effect of the number of individuals buried in each tomb. The data is given in **Table 9.2**.

We input our data in R using the command:

```
data <- read.table("partialcor.txt", head=1, sep="\t")
```

To compute the Pearson and Spearman partial correlation coefficients, we first install the *RVAideMemoire* package and then use the following commands:

```
library(RVAideMemoire)
pcor.test(x=data$Tomb_size, y=data$Pot_capacity,
z=data$No_individuals, method="pearson")
pcor.test(x=data$Tomb_size, y=data$Pot_capacity,
z=data$No_individuals, method="spearman")
```

Table 9.2.
Data on tomb
size, pots
capacity,
and number
of interred
individuals

Tomb size (m ²)	Average capacity of pots per tomb (litres)	No of individuals
5.1	1.2	1
13.2	3.0	4
7.9	1.7	1
11.1	1.9	3
14.0	2.2	3
9.7	2.0	2
9.0	1.6	2
10.2	1.8	2
8.4	1.5	2
7.8	1.5	1
6.2	1.0	1
6.3	1.3	1
7.1	1.7	2
17.0	3.4	5

The output is given in Figure 9.5, where it can be seen that the p-values are 0.181 and 0.344 for Pearson's ($r=0.395$) and Spearman's ($\rho=0.4286$) coefficients, respectively. Thus, we obtain a non-significant correlation between tomb size and pottery capacity when the number of individuals in each tomb is taken into consideration.

Figure 9.5.
Results of
Pearson's and
Spearman's
partial
correlation
coefficient

```
> data <- read.table("partialcor.txt", head=1, sep="\t")
> library(RVAideMemoire)
> pcor.test(x=data$Tomb_size, y=data$Pot_capacity, z=data$No_individuals, method = "pearson")

Pearson's product-moment partial correlation

data: data$Tomb_size and data$Pot_capacity, controlling for No_individuals
t = 1.4268, df = 11, p-value = 0.1814
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.1991850  0.7769807
sample estimates:
      cor 
0.3951694

> pcor.test(x=data$Tomb_size, y=data$Pot_capacity, z=data$No_individuals, method = "spearman")

Spearman's rank partial correlation

data: data$Tomb_size and data$Pot_capacity, controlling for No_individuals
S = 260, p-value = 0.344
alternative hypothesis: true rho is not equal to 0
95 percent confidence interval:
 -0.1536748  0.8440980
sample estimates:
      rho 
0.4285714
```

9.3. CORRELATION OF MATRICES

There are occasions where we want to examine the correlation between matrices, typically containing measures of distance. The term *distance* refers to a numeric score that indicates how close or far two observations are in terms of a set of variables. Larger distances suggest that the observations are less similar to each other. When two objects are more distant/different from each other, *dissimilarity* measures will be larger, while *similarity* measures will be larger when two objects are more close/similar to each other. The broader term *proximity* refers to measures of similarity and dissimilarity.

Even though *dissimilarity* and *distance* are often used interchangeably, there are certain requirements for distances that are not necessary in dissimilarities:

1. The distance between two objects should only be zero if these objects are identical to each other.
2. The distance from object 1 to object 2 should be the same as the distance from object 2 to object 1 (*symmetry*).
3. For any three objects the distance from object 1 to object 3 can never be greater than the distance from object 1 to object 2 plus the distance from object 2 to object 3 (*triangle inequality*).

In bioarchaeology, we occasionally examine the correlation between the biodistance (biological/phenotypic distance), geographic distance and temporal distance of different groups. In addition, we may examine the partial correlation between biodistances and spatial distances, controlling for temporal distances, i.e. when the effect of temporal distances is excluded, or the partial correlation between biodistances and temporal distances controlling for spatial distances, i.e. by excluding the effect of the spatial distances.

The correlation between two matrices may be performed by means of the Mantel test. While the Mantel test only allows the comparison between two matrices, the partial Mantel test can be used to compare two matrices while controlling for the effect of a third or more matrices.

In R for the simple and partial Mantel test we can use the `mantel()` function of the *ecodist* package:

```
mantel(xdist ~ ydist, nperm = 99999, mrank=FALSE)
mantel(xdist ~ ydist + zdist, nperm = 99999,
mrnk=FALSE)
```

where *xdis*, *ydis* are the matrices examined, *zdis* is the controlling matrix, *nperm* is the number of permutations when computing the p-value, and *mrnk* = *FALSE* or *TRUE* for Pearson's or Spearman's correlations. Note that matrices *xdis*, *ydis*, *zdis* should be distance matrices of class *dist*. Note also that the first command runs a simple Mantel test to examine the correlation between biodistance (*xdist*) and spatial distance (*ydist*), whereas the second command runs a partial Mantel test to examine the correlation between biodistance (*xdist*) and spatial distance (*ydist*) while controlling for temporal distance (*zdist*).

Alternatively, we may use the functions `mantel()` and `mantel.partial()` from the *vegan* package:

```
mantel(xdist, ydist, permutations=9999,
method="pearson") mantel.partial(xdist, ydist, zdist,
permutations=9999)
```

where *method* = "*pearson*" or "*spearman*" for Pearson's or Spearman's correlations.

As an example, consider the correlation between the biological distance (as given by the Mean Measure of Divergence – MMD) and the spatial and temporal distances shown in **Table 9.3**.

Table 9.3.
Distance
matrices

MMD			
0	0.3895	1.5946	0.7670
0.3895	0	0.3695	0
1.5946	0.3695	0	0.2743
0.7670	0	0.2743	0

Spatial distance			
0	15	120	100
15	0	55	25
120	55	0	30
100	25	30	0

Temporal distance			
0	200	200	100
200	0	1000	1000
200	1000	0	500
100	1000	500	0

To apply the Mantel tests, we create three .txt tab delimited files, such as the file of [Figure 9.6](#) with names `xdist`, `ydist`, `zdist`, respectively. Note that these files do not have a first row with the variable names. Therefore, in the `read.table()` function we should use the argument `head=0`.

To run the Mantel tests using the `ecodist` package, we install this package and use the following commands for Pearson’s correlations:

```
library(ecodist)
xdist <- read.table("xdist.txt", head=0, sep="\t")
ydist <- read.table("ydist.txt", head=0, sep="\t")
zdist <- read.table("zdist.txt", head=0, sep="\t")
xdist <- as.dist(xdist)
ydist <- as.dist(ydist)
zdist <- as.dist(zdist)
ecodist::mantel(xdist ~ ydist, nperm=99999,
mrnk=FALSE)
ecodist::mantel(xdist ~ ydist + zdist, nperm=99999,
mrnk=FALSE)
```

Note that the commands using the function `as.dist()` are necessary to create distance matrices. Note also that in the last two commands we define the package at the beginning of the command. This is because the `mantel()` function is also used by the *vegan* package, which will be adopted further down in this example.

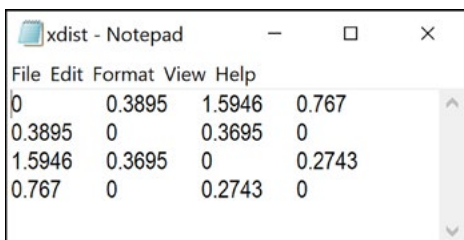
For Spearman's correlations, we can use:

```
ecodist::mantel(xdist ~ ydist, nperm=99999,
mrnk=TRUE)

ecodist::mantel(xdist ~ ydist + zdist, nperm=99999,
mrnk=TRUE)
```

In the output of Figures 9.7 and 9.8 we observe that the `mantel()` function returns the Mantel r statistic, which is the r or/and p correlation coefficients, and three p -values, *pval1*, *pval2*, *pval3*. These are the one-tailed p -value ($r \leq 0$), the one-tailed p -value ($r \geq 0$), and the two-tailed p -value ($r = 0$), respectively. *llim* is the lower and *ulim* the upper confidence limit. In the example under study, there is a strong ($r = 0.885$) and statistically significant (p -value = $0.041 < 0.05$) positive Pearson's correlation between biological distance and spatial distance. If we control for the temporal effect, the correlation between MMD and spatial distance is still strong, positive and statistically significant ($r = 0.862$, p -value = $0.041 < 0.05$). However, if we use Spearman's correlations, neither the bivariate (p -value = $0.124 > 0.05$) nor the partial correlations (p -value = $0.209 > 0.05$) are statistically significant.

At this point we should stress that the p -values in the Mantel tests are estimated using permutations.



	0	0.3895	1.5946	0.767
0	0	0.3895	1.5946	0.767
0.3895	0.3895	0	0.3695	0
1.5946	1.5946	0.3695	0	0.2743
0.767	0.767	0	0.2743	0

Figure 9.6.
Data
arrangement
in matrix

```
> ecodist::mantel(xdist ~ ydist, nperm = 99999, mrnk=FALSE)
mantelr      pval1      pval2      pval3  llim.2.5% ulim.97.5%
0.88468243 0.04104041 1.00000000 0.04104041 0.80181818 0.92057584
> ecodist::mantel(xdist ~ ydist + zdist, nperm = 99999, mrnk=FALSE)
mantelr      pval1      pval2      pval3  llim.2.5% ulim.97.5%
0.86240040 0.04112041 1.00000000 0.04112041 0.83241484 0.99417851
```

Figure 9.7.
Simple
and partial
Pearson's
correlations
among the
matrices of
Table 9.3 using
the *ecodist*
package

```
> ecodist::mantel(xdist ~ ydist, nperm = 99999, mrnk=TRUE)
mantelr      pval1      pval2      pval3  llim.2.5% ulim.97.5%
0.65714286 0.08222082 0.95834958 0.12425124 0.50000000 1.00000000
> ecodist::mantel(xdist ~ ydist + zdist, nperm = 99999, mrnk=TRUE)
mantelr      pval1      pval2      pval3  llim.2.5% ulim.97.5%
0.6628403 0.1249512 0.8750588 0.2089321 -0.5215130 0.9954481
```

Figure 9.8.
Simple
and partial
Spearman's
correlations
among the
matrices of
Table 9.3 using
the *ecodist*
package

We may alternatively use the *vegan* package. In this case we can use the commands:

```
library(vegan)

xdist <- read.table("xdist.txt", head=0, sep="\t")
ydist <- read.table("ydist.txt", head=0, sep="\t")
zdist <- read.table("zdist.txt", head=0, sep="\t")

xdist <- as.dist(xdist)
ydist <- as.dist(ydist)
zdist <- as.dist(zdist)

vegan::mantel(xdist, ydist, method="pearson",
permutations=9999)

vegan::mantel.partial(xdist, ydist, zdist,
method="pearson", permutations=9999)

vegan::mantel(xdist, ydist, method="spearman",
permutations=9999)

vegan::mantel.partial(xdist, ydist, zdist,
method="spearman", permutations=9999)
```

The results are shown in Figures 9.9 and 9.10, and are in complete agreement with the results from the *ecodist* package: Pearson's simple Mantel p-value = 0.042, partial Mantel p-value = 0.042, whereas Spearman's simple Mantel p-value = 0.083, partial Mantel p-value = 0.125. It is seen that irrespective of the function used, Pearson and Spearman tests do not agree in what concerns the significance of the correlations. In this case, the researcher should choose the test she/he trusts based on her/his knowledge of the archaeological context.

Figure 9.9.
Simple
and partial
Pearson's
correlations
among the
matrices of
Table 9.3 using
the *vegan*
package

```
> vegan::mantel(xdist, ydist, method="pearson", permutations = 9999)
'nperm' >= set of all permutations: complete enumeration.
Set of permutations < 'minperm'. Generating entire set.

Mantel statistic based on Pearson's product-moment correlation

Call:
vegan::mantel(xdis = xdist, ydis = ydist, method = "pearson",      permutations = 9999)

Mantel statistic r: 0.8847
Significance: 0.041667

Upper quantiles of permutations (null model):
 90%  95% 97.5%  99%
0.605 0.652 0.699 0.731
Permutation: free
Number of permutations: 23

> vegan::mantel.partial(xdist, ydist, zdist, method="pearson", permutations = 9999)
'nperm' >= set of all permutations: complete enumeration.
Set of permutations < 'minperm'. Generating entire set.

Partial Mantel statistic based on Pearson's product-moment correlation

Call:
vegan::mantel.partial(xdis = xdist, ydis = ydist, zdis = zdist,      method = "pearson")

Mantel statistic r: 0.8624
Significance: 0.041667

Upper quantiles of permutations (null model):
 90%  95% 97.5%  99%
0.502 0.528 0.620 0.685
Permutation: free
Number of permutations: 23
```

```

> vegan::mantel(xdist, ydist, method="spearman", permutations = 9999)
'nperm' >= set of all permutations: complete enumeration.
Set of permutations < 'minperm'. Generating entire set.

Mantel statistic based on Spearman's rank correlation rho

Call:
vegan::mantel(xdis = xdist, ydis = ydist, method = "spearman",      permutations = 9999)

Mantel statistic r: 0.6571
Significance: 0.083333

Upper quantiles of permutations (null model):
 90%   95%  97.5%   99%
0.474 0.486 0.614 0.709
Permutation: free
Number of permutations: 23

> vegan::mantel.partial(xdist, ydist, zdist, method="spearman", permutations = 9999)
'nperm' >= set of all permutations: complete enumeration.
Set of permutations < 'minperm'. Generating entire set.

Partial Mantel statistic based on Spearman's rank correlation rho

Call:
vegan::mantel.partial(xdis = xdist, ydis = ydist, zdis = zdist,      method = "spearman")

Mantel statistic r: 0.6628
Significance: 0.125

Upper quantiles of permutations (null model):
 90%   95%  97.5%   99%
0.640 0.716 0.756 0.779
Permutation: free
Number of permutations: 23

```

Figure 9.10.
Simple
and partial
Spearman's
correlations
among the
matrices of
Table 9.3 using
the *vegan*
package

10.

REGRESSION

In statistics, *regression* is any statistical process that can be used to estimate the relationship between a *dependent* variable and one or more *independent* variables. The dependent variable is also called *outcome* or *response* variable, whereas the independent variable is called *explanatory* variable or *predictor*. When there is one predictor, we use *simple regression* to estimate the relationship between the dependent and independent variable, whereas when we have several predictor variables, we use *multiple regression*.

In simple regression analysis, given a data set (y_i, x_i) , where $i = 1, 2, \dots, N$, we are searching for the best straight line, $y = a + bx$ that passes through the (y_i, x_i) points. This means that first we make the scatterplot of the dataset. *Note that the independent variable always goes on the x axis and the dependent variable on the y axis.* Subsequently, we attempt to determine the best straight line that passes through the data points and describes the relationship between the two variables. This is called a *regression line*. The criterion that is usually adopted to determine the best line is the *least squares criterion*. According to this criterion, the best straight line is the one that passes through the data and for which the *sum of the squared residuals is minimum*. A *residual* is the difference between the experimental and the estimated value y at a certain value x and it is denoted as d .

Figure 10.1.
Scatterplot of data and regression line.
The vertical lines are the residuals

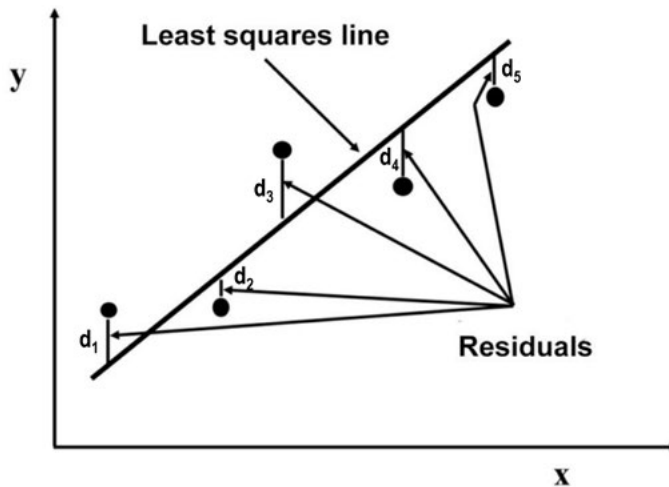


Figure 10.1 shows schematically the scatterplot of a dependent (y) versus an independent (x) variable. If we draw in this plot multiple straight lines that pass through the data points and for each line calculate the sum of the squared residuals, $S = d_1^2 + d_2^2 + d_3^2 + d_4^2 + d_5^2$, the best straight line, *regression line* or *least squares line*, is the one that corresponds to the minimum S value.

The least squares criterion can be applied not only to straight lines but to any type of line. Thus, the best curve, *regression* or *least squares curve*, is the curve for which the sum of the squared residuals is minimum.

10.1 BASIC DEFINITIONS

Regression models

The first step in regression analysis is to build the *regression model*. That is, to select via the least squares technique the mathematical relationship that represents the experimental data. The simplest regression model is the straight line with a constant or not term:

$$y = a + bx$$

or:

$$y = bx$$

If the relationship between two variables is not linear, we examine whether a *quadratic* or a *cubic* model can be used. The mathematical expression for the quadratic model is:

$$y = a + bx + cx^2$$

while for the cubic model it is:

$$y = a + bx + cx^2 + dx^3$$

Note that the above models are limiting cases of a polynomial of degree p : $y = a + b_1x + b_2x^2 + \dots + b_px^p$

In all these models, parameters $a, b, c, \dots, b_1, b_2, \dots, b_p$ are called *model coefficients* or *regression parameters*, they are initially unknown, and they are determined based on the least squares criterion. The procedure of the determination of the adjustable parameters of a model is called *polynomial fitting*. A generalization is the *multilinear regression* or *multilinear fitting*, in which we have one dependent and many independent variables. Therefore, the model has the form:

$$y = b_1 + b_2x_2 + b_3x_3 + \dots + b_px_p$$

The determination of the adjustable parameters is again done by means of the least squares criterion, but it may be useful to assess which terms are significant and which ones can be eliminated, as discussed below.

All the above models belong to the category of *linear models*. Apart from this category, there are *non-linear models* and *generalized linear models*. In the non-linear models, y is expressed by a function, which is a nonlinear combination of the model parameters. For example:

$$y = \frac{ax}{b + x}$$

In the generalized linear models, instead of y , a proper function of y , $f(y)$, is expressed as a linear combination of the independent variables:

$$f(y) = b_1 + b_2x_2 + b_3x_3 + \dots + b_px_p$$

Determination of the regression model

The determination of a regression model involves the estimation of the model coefficients based on the least squares criterion. When the model belongs to the category of linear models, this estimation is straightforward. For the model $y = b_1 + b_2x_2 + b_3x_3 + \dots + b_px_p$, the sum of the squared residuals S is a function of the regression parameters, $b_1, b_2, b_3, \dots, b_p$, i.e. $S = f(b_1, b_2, b_3, \dots, b_p)$. The minimization of this function yields a system of p linear equations, from which we obtain the mathematical expression of the regression parameters in terms of the experimental data. That is, for each regression parameter, we have a mathematical equation of the general form: $b_j = f_j(y, x_1, x_2, \dots, x_p)$, which can be used to estimate b_j .

An alternative approach for the estimation of the regression parameters, often adopted in generalized linear models, is the *maximum-likelihood estimation* (MLE). The basic idea with MLE is to estimate the parameters of a statistical model that "best" fit the data. Thus, depending on the regression model, a likelihood function L is defined, and the regression parameters are selected by maximizing L .

In non-linear regression, the estimation of the regression parameters is performed by minimizing the sum of the squared residuals. However, in this case we should use numerical methods to solve the problem.

Note that in all cases the estimation of the regression parameters is associated with the assessment of their statistical significance. Thus, for each regression parameter b_k , its standard deviation is calculated. In addition, the p-value that corresponds to the null hypothesis $H_0: b_k = 0$ with alternative $H_1: b_k \neq 0$ is computed.

Assessing the model

The assessment of a regression model is a necessary step in regression analysis since it allows the researcher to test the accuracy of the predictions made by the model as well as any information obtained about the contribution of the predictors to the dependent variable. The first measure we can use is the correlation coefficient R defined from:

$$R^2 = 1 - \frac{\sum_{i=1}^N (y_i(\text{model}) - y_i(\text{exp}))^2}{\sum_{i=1}^N (y_i(\text{exp}) - \bar{y})^2} = 1 - \frac{SSR}{\sum_{i=1}^N (y_i(\text{exp}) - \bar{y})^2}$$

where N is the number of x-y values, $\bar{y}$ is the average value of the experimental values $y_i(\text{exp})$, $y_i(\text{model})$ is the y values estimated by the model, and SSR is the sum of the squared residuals. When the experimental points fall exactly on the regression curve, $y_i(\text{model})$ becomes identical to $y_i(\text{exp})$ and, therefore, the correlation coefficient R becomes equal to 1. However, as the deviations of the experimental points from the regression curve increase, i.e. the deviations of $y_i(\text{exp})$ from $y_i(\text{model})$, the R value deviates from the unit and tends to zero. Therefore, the correlation coefficient R is a measure indicating whether the regression curve describes the experimental points satisfactorily or not.

A serious problem with R^2 is that when we add more predictors to the model, its value increases, tending to 1. However, when we increase the number of predictors, overestimation problems are likely to appear and, due to these problems, it is very risky to use the model for prediction.

When two or more models have been determined by means of the least squares method, the optimum model may be selected based on the *standard error of estimate* (SEE) or s_y . This measure is defined from:

$$SEE = \sqrt{\frac{\sum_{i=1}^N (y_i(\text{model}) - y_i(\text{exp}))^2}{N - p}} = \sqrt{\frac{SSR}{N - p}}$$

where p is the number of adjustable parameters.

In fact, SEE gives the average scatter of the experimental points around the regression curve and, therefore, the best model corresponds to the minimum SEE value. Consequently, the smaller the SEE, the better the regression model.

An alternative criterion for the selection of the best model is the Akaike Information Criterion (AIC), defined from:

$$AIC = N \ln \frac{(SEE)^2 (N - p)}{N} + 2p + \text{const}$$

where *const* is a constant. As SEE, AIC is also a measure of the relative quality of statistical models for a given set of data. The best model corresponds to the minimum AIC value.

It should be stressed that different criteria may lead to different results in choosing the best model. In such cases the final decision is left to the investigator.

Number of adjustable parameters

In general, the least squares method allows the estimation of the adjustable parameters $b_1, b_2, \dots, b_p$ when the model is given from the relationship:

$$y = b_1 + b_2 f_2(x) + b_3 f_3(x) + \dots + b_p f_p(x)$$

When the model is known, the number p of the adjustable parameters is also known. However, there are cases where the model is an empirical relationship or a polynomial of degree p . In such cases, it is necessary to determine the optimum regression model or the optimum polynomial degree that should be used. This is the main issue of *stepwise regression*.

This issue can be easily solved if the p -values of the adjustable parameters are known. Consider that the p -value of parameter b_k is known. This value corresponds to the null hypothesis $H_0: b_k = 0$ with alternative $H_1: b_k \neq 0$. Therefore, when $p\text{-value} > 0.05$, the null hypothesis cannot be rejected and, therefore, parameter b_k may be eliminated from the model. Based on this observation, when we know the model but we want to remove the non-significant parameters, we calculate the p -values of all parameters and remove the parameter with the greatest p -value provided that $p\text{-value} > 0.05$. Then we re-apply the least-squares method, without the eliminated parameter in the model. We check if there are parameters with $p\text{-value} > 0.05$ and repeat the above process until all the adjustable parameters of the model are statistically significant, i.e. $p\text{-values} < 0.05$. This is the *backward elimination method*. Other methods are the *forward selection* and *stepwise methods*. In forward selection predictors are added one at a time until a stopping criterion is reached, whereas stepwise methods combine aspects of forward selection and backward elimination.

If we want to determine the optimum degree of the polynomial, we usually start from a polynomial of a small degree, for example $y = a + bx$, and increase the degree of the polynomial by one at a time, that is, $y = a + bx + cx^2$, $y = a + bx + cx^2 + dx^3$, ... In every step we check if the adjustable parameters are statistically significant ($p\text{-value} < 0.05$). Once we find a polynomial in which at least one of the adjustable parameters is non-significant, we stop the above process and select as optimum the previous polynomial. This process can be done in reverse order. That is, we start from a high-degree polynomial, at least 10, and reduce the degree of the polynomial by one at a time. At every step we check whether at least one of the adjustable parameters is non-significant and when this is detected, we stop the above process and select this polynomial as the optimum one.

Usually the selection of the optimum regression model is not based on p -values but on the AIC. Thus, in the *backward method* we begin by placing all predictors in the model and then we look for the predictor the elimination of which will cause the greatest reduction in AIC. This predictor is eliminated and the process continues until the removal of any predictor increases the AIC.

Regression assumptions

Normally, regression is based on very few assumptions. If we are interested only in modeling the experimental data, the basic prerequisite is the absence of outliers. However, if we are additionally interested in statistical inference, for example if we want to identify statistically non-significant terms in the model, to estimate confidence intervals for the predicted values or to determine the optimum regression model, then several other assumptions should be met.

For most linear models, these assumptions are:

- The relationship between the response variable and the explanatory variables should be linear.

- The errors/residuals should have constant variance (homoscedasticity).
- The errors/residuals should follow the normal distribution, although linear regression is robust against violations of the normality assumption so long as the sample is large enough.
- The errors in the response variable values (residuals) should be uncorrelated.
- The explanatory variables should be uncorrelated with one another; strong correlations between the explanatory variables is called *multicollinearity*.
- There should be no outliers.

All these assumptions and potential problems can be checked by producing diagnostic plots visualizing the residual errors, as presented in section 10.2.

10.2. LINEAR REGRESSION

Linear regression is used for finding a linear relationship between the response variable and one or more predictors of the general form:

$$y = b_0 + b_1 f_1(x) + b_2 f_2(x) + \dots + b_p f_p(x)$$

For example, if $p = 1$ and $f_1(x) = x$, the above relationship is reduced to the simple linear model $y = b_0 + b_1 x$ or $y = a + bx$; if $f_1(x) = x$, $f_2(x) = x^2$, ..., $f_p(x) = x^p$, we obtain the polynomial model, $y = b_0 + b_1 x + b_2 x^2 + \dots + b_p x^p$; whereas when $f_1(x) = x_1$, $f_2(x) = x_2$, ..., $f_p(x) = x_p$, we obtain the multiple linear model $y = b_0 + b_1 x_1 + b_2 x_2 + \dots + b_p x_p$.

In R the basic function to perform linear regression is the `lm()` function, where *lm* stands for linear model. In this section, we will use this function and examine simple linear, polynomial and multiple linear models.

Simple linear model

Assume that we want to estimate the age of two infants from classical Greece. All we have left of them is their humeri with length 50 and 55 mm, respectively. To do this estimation, we need to find an equation that will allow the calculation of the age-at-death based on humeral length. For this purpose, we have collected the data shown in **Table 10.1** using a modern sample of infants. Based on these modern data, we will find an equation that allows the prediction of age-at-death based on humeral length and then apply it to the unknown infants.

Table 10.1.
Humeral
length and age
data from a
modern infant
sample

Humeral length (mm)	Age (weeks)
42	28
45	27
58	32
59	34
59	35
61	35
64	36
65	37
65	38
68	40
70	40
70	40
72	41
75	45

We input the data in the R Console using the vectors:

```
length <- c(42, 45, 58, 59, 59, 61, 64, 65, 65, 68, 70, 70, 72, 75)
```

```
age <- c(28, 27, 32, 34, 35, 35, 36, 37, 38, 40, 40, 40, 41, 45)
```

and run the regression analysis using the `lm()` function:

```
fit <- lm(age ~ length)
summary(fit)
```

The results are shown in Figure 10.2. It is seen that the regression model is:

$\text{age} = 4.420 + 0.511 \cdot \text{length}$

The R^2 is 0.946, which suggests that the model is a very good fit for the data.

```
> length <- c(42, 45, 58, 59, 59, 61, 64, 65, 65, 68, 70, 70, 72, 75)
> age <- c(28, 27, 32, 34, 35, 35, 36, 37, 38, 40, 40, 40, 41, 45)
> fit <- lm(age ~ length)
> summary(fit)

Call:
lm(formula = age ~ length)

Residuals:
    Min       1Q   Median       3Q      Max
-2.0592 -0.5867 -0.2023  0.4133  2.2536

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  4.42042     2.21497   1.996  0.0692 .
length       0.51101     0.03514  14.541 5.54e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.206 on 12 degrees of freedom
Multiple R-squared:  0.9463,    Adjusted R-squared:  0.9418
F-statistic: 211.4 on 1 and 12 DF,  p-value: 5.542e-09
```

Figure 10.2.
Results
of linear
regression

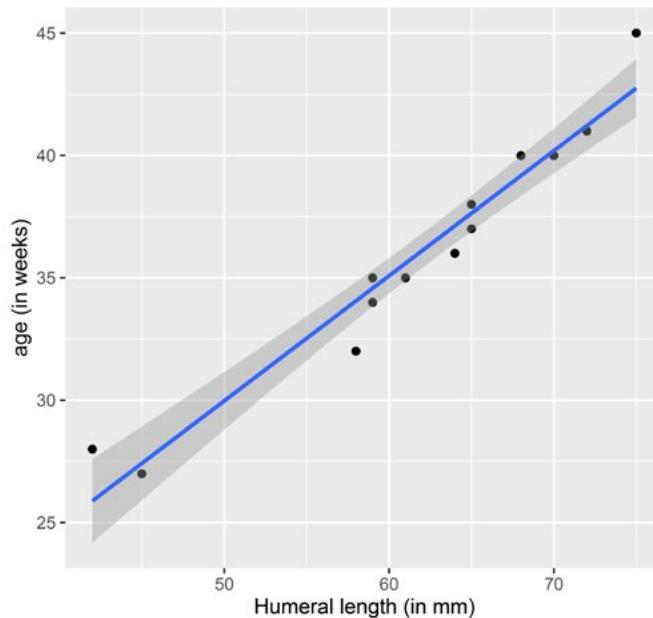
An alternative way to assess the fit of the regression model is to create the scatterplot of the two variables, age vs. length, and add the regression line to this plot.

Using the `ggplot()` function, we can type the following commands:

```
library(ggplot2)
data <- data.frame(cbind(age,length))
ggplot(data, aes(length,age)) + geom_point() +
  geom_smooth(method="lm") +
  xlab("Humeral length (in mm)") + ylab("age (in weeks)")
```

We obtain Figure 10.3, which shows that the regression model describes the experimental data very satisfactorily. Therefore, it can be used for the prediction of the age of infants based on their humeral length.

Figure 10.3.
Plot of age
versus humeral
length, the
least squares
line, and the
corresponding
95%
confidence
interval



For this purpose, we can use the commands:

```
new <- data.frame(length=c(50, 55))
predict(fit, new, se.fit=TRUE)
```

which yield the results of Figure 10.4. We observe that the function `predict()` computes the predicted values, which are:

First infant: 29.97 ± 0.54 weeks

Second infant: 32.53 ± 0.41 weeks

Figure 10.4.
Predicted
values and
their standard
deviations

```
> new <- data.frame(length=c(50, 55))
> predict(fit, new, se.fit=TRUE)
$fit
      1      2
29.97106 32.52612

$se.fit
      1      2
0.5407714 0.4131544
```

All the above, and in particular the statistical information concerning confidence intervals and p-values (Figure 10.3,) as well as predicted standard deviations (Figure 10.4), are reliable under the pre-condition that the regression assumptions hold.

A simple way to examine this is via the function `plot(fit)`:

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

Note that the command `par(mfrow=c(2, 2))` puts the plots in 2 rows and 2 columns. The function `plot(fit)` creates 4 diagnostic plots, which show residuals in four different ways:

1. Residuals vs. fitted values
2. Q-Q plot
3. Scale-Location plot
4. Cook's distance plot

The first plot is a scatterplot between residuals and predicted values, and it should look more or less random. The second plot (Q-Q plot) is a normal probability plot. All the points should fall approximately along the reference line in order to assume normality. The third plot (Scale-Location plot) should look random with no patterns. It is used to check the homogeneity of variance of the residuals (homoscedasticity). Finally, the last plot (Cook's distance plot) is used to identify outliers, i.e. extreme values that might influence the regression results. In this plot the extreme values fall outside Cook's distance lines, which are shown with a red dashed line.

In our example, Figure 10.5 shows that there are some deviations from the expected patterns when all regression assumptions are tested. In particular, there are two points, point 1 and 14, that create problems. These points fall outside Cook's distance, as outliers, and similarly affect the normality.

However, we should stress that these plots give useful information only for relatively large samples. Small samples with traits that exhibit great variability are likely to yield misleading diagnostic plots and, therefore, we should interpret them carefully.

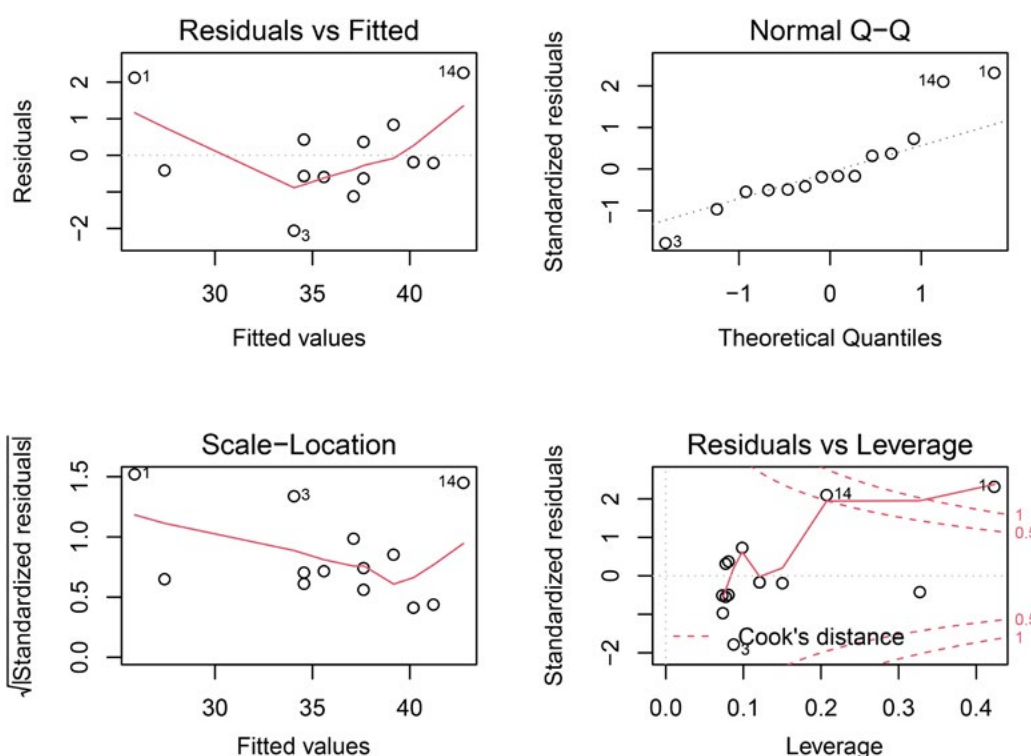


Figure 10.5.
Diagnostic
plots for
regression
assumptions
test

Note that we can obtain the residuals using the function `resid(fit)`. Therefore, we can test their normality using the Shapiro-Wilk test through the command:

The results are given in Figure 10.6 and it is seen that the normality assumption is not violated.

Figure 10.6.
Normality
test for the
residuals

```
> shapiro.test(resid(fit))

      Shapiro-Wilk normality test

data:  resid(fit)
W = 0.92545, p-value = 0.2631
```

Polynomial regression

In more complex cases, the relationship between two variables may not be simple linear but rather described by a quadratic or cubic model. The mathematical expression for the quadratic model is:

$y = a + bx + cx^2$

while for the cubic model, it is:

$y = a + bx + cx^2 + dx^3$

Consider **Table 10.2**, which shows how human dental size has changed over time. As a proxy for dental size, we have measured the total masticatory area of the maxillary and mandibular teeth. Draw a graph to depict the table values and assess the age of two samples with dental area 1150 and 1250 mm².

To address the problem, we first draw a scatterplot to determine if the relationship between the two variables is linear, quadratic or more complicated. Since we are asked to assess the age based on the dental area, age will be the dependent (y) and dental area the independent variable (x).

We can use the following commands:

```
x <- c(1025, 1055, 1060, 1065, 1070, 1060, 1095, 1160,
1170, 1185, 1200, 1310, 1360)

y <- c(0, 1, 1.5, 2, 2.3, 5, 5.5, 6, 12, 15, 20, 35, 55)

data <- data.frame(cbind(x,y))

library(ggplot2)

ggplot(data=data, aes(x,y)) + geom_point() +
xlab("Dental area") + ylab("age")
```

Table 10.2.
Change in
dental area
over the years

Age (thousands years ago)	Dental area (mm ²)
0	1025
1	1055
1.5	1060
2	1065
2.3	1070
5	1060
5.5	1095
6	1160
12	1170
15	1185
20	1200
35	1310
55	1360

The plot is shown in Figure 10.7 and it can be seen that the relationship between age and dental area is not linear; rather it may be described by a quadratic model, $y = a + bx + cx^2$, or a cubic model, $y = a + bx + cx^2 + dx^3$.

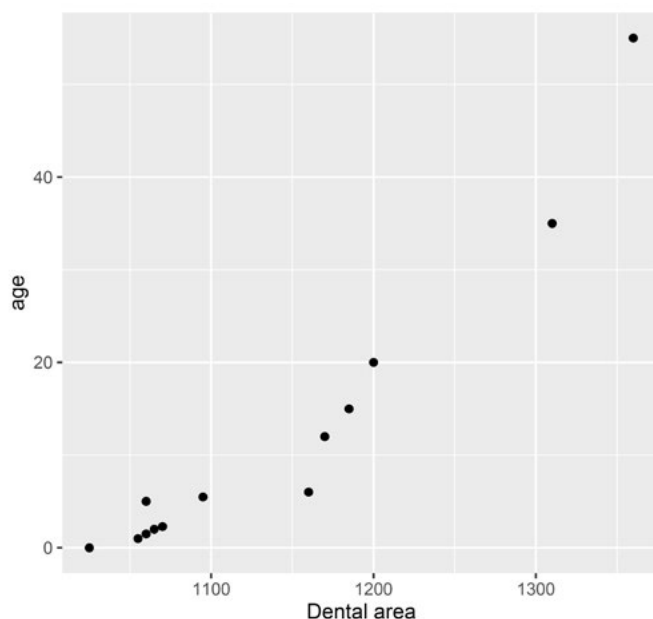


Figure 10.7.
Scatterplot for
age vs. dental
area

In R to build a polynomial regression model of degree k , we can use the function `lm(y~poly(x,k,row=T))`. Therefore, in order to estimate the optimum k , we may use, for example, the commands:

```
AIC(lm(y~poly(x,1,row=T)))
AIC(lm(y~poly(x,2,row=T)))
AIC(lm(y~poly(x,3,row=T)))
```

and examine where the AIC function takes its minimum value. From Figure 10.8 we obtain that this happens when $k=2$. Additionally, we can use the `sigma(lm(y~poly(x,k,row=T)))` function, which computes the SEE value.

```
> AIC(lm(y~poly(x,1,row=T)))
[1] 82.82038
> AIC(lm(y~poly(x,2,row=T)))
[1] 66.75317
> AIC(lm(y~poly(x,3,row=T)))
[1] 67.40173
```

Figure 10.8.
AIC values
when $k = 1$,
2 and 3

A simple code that can be used to create a table of AIC and SEE values as a function of the polynomial degree, k , is the following:

```
AIC=c(); SEE=c()
for(j in 1:7) {AIC[j]= AIC(lm(y~poly(x,j,row=T)));
SEE[j]=sigma(lm(y~poly(x,j,row=T)))}
d <- data.frame(cbind(degree=c(1:7),AIC,SEE))
d
```

Figure 10.9.
Table of AIC
and SEE
values as a
function of
the polynomial
degree

```
> AIC=c(); SEE=c()
> for(j in 1:7) {AIC[j]= AIC(lm(y~poly(x,j,raw=T))); SEE[j]=sigma(lm(y~poly(x,j,raw=T)))}
> d <- data.frame(cbind(degree=c(1:7),AIC,SEE))
> d
```

	degree	AIC	SEE
1	1	82.82038	5.049142
2	2	66.75317	2.643175
3	3	67.40173	2.645031
4	4	68.24084	2.682970
5	5	63.60190	2.221864
6	6	63.60190	2.221864
7	7	57.35337	1.747473

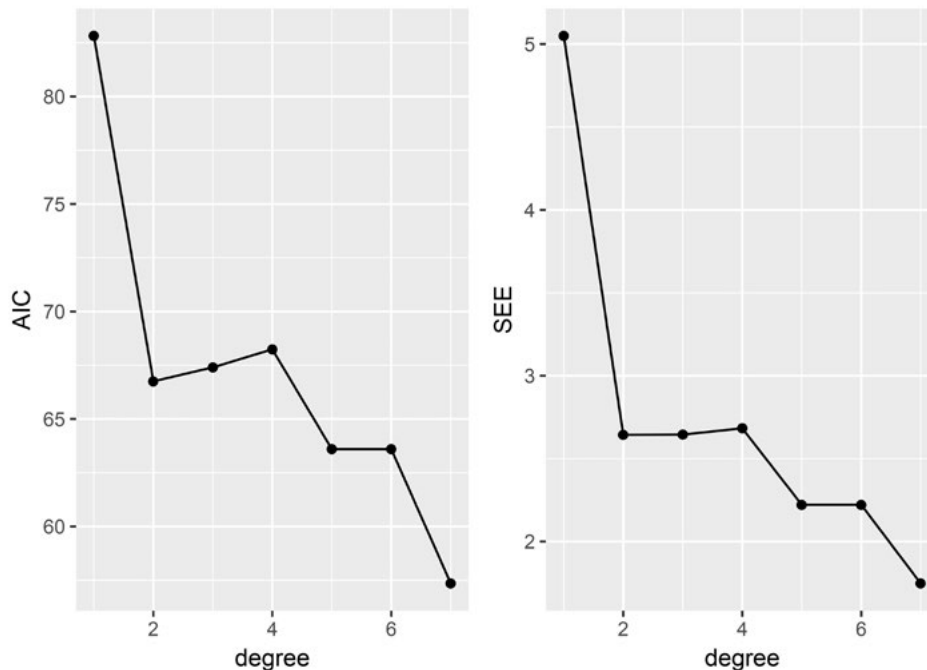
This code yields the table of Figure 10.9, which shows that as the polynomial degree increases, both AIC and SEE decrease.

To obtain a visual picture of the variation, we can use the following commands, where the *patchwork* package is used to display the two plots in one layout:

```
library(patchwork)
p1 <- ggplot(data=d, aes(x=degree,y=AIC)) +
  geom_point() + geom_line()
p2 <- ggplot(data=d, aes(x=degree,y=SEE)) +
  geom_point() + geom_line()
p1 + p2
```

We obtain Figure 10.10, which shows that in both plots AIC and SEE exhibit a sudden fall from $k=1$ to $k=2$, then these quantities show a continuous slight increase from $k=2$ to $k=4$, whereas after $k=4$, there is again a significant drop in their values. In polynomial regression, we are looking for the best polynomial among those with a small k value. For this reason, we select as optimum k value the value of 2, i.e. a quadratic model.

Figure 10.10.
Plots of AIC
and SEE vs.
polynomial
degree



Based on this analysis, we use the following commands to obtain the regression model:

```
fit <- lm(y~poly(x, 2, raw=TRUE))
summary(fit)
```

The output is given in Figure 10.11. It is seen that the model is:

$$\text{age} = 435.1 - 0.8586 * (\text{dental area}) + 0.000425 * (\text{dental area})^2$$

Note that all terms of this model are statistically significant: p-value = 0.002426 for the constant, 0.000872 for dental area, and 0.000265 for squared dental area. In addition, the R^2 is 0.9778, which suggests that the model fits very satisfactorily the data.

```
> fit <- lm(y~poly(x, 2, raw=TRUE))
> summary(fit)

Call:
lm(formula = y ~ poly(x, 2, raw = TRUE))

Residuals:
    Min       1Q   Median       3Q      Max
-4.4854 -0.8885 -0.2434  1.4405  3.7769

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    4.351e+02  1.082e+02   4.023  0.002426 **
poly(x, 2, raw = TRUE)1 -8.586e-01  1.836e-01  -4.677  0.000872 ***
poly(x, 2, raw = TRUE)2  4.246e-04  7.734e-05   5.490  0.000265 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.643 on 10 degrees of freedom
Multiple R-squared:  0.9778,    Adjusted R-squared:  0.9734
F-statistic: 220.2 on 2 and 10 DF,  p-value: 5.399e-09
```

Figure 10.11.
Output of
polynomial
regression

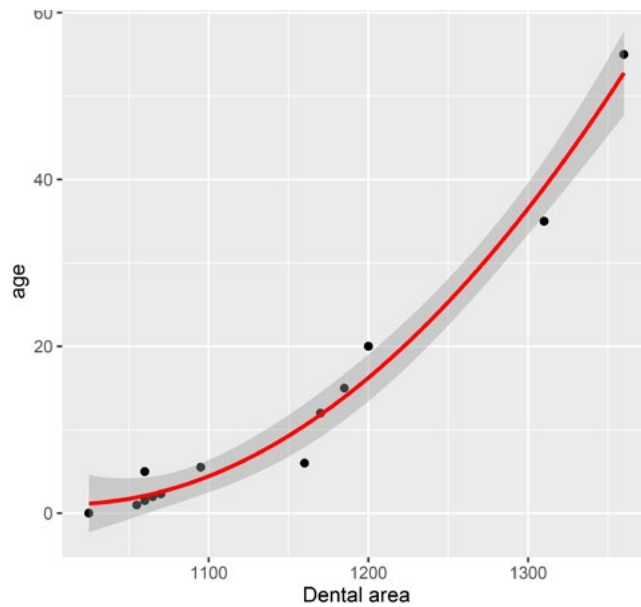
SCIENTIFIC NOTATION IN R

In the R output scientific notation is used when there are four or more leading zeros so that .0000123 is represented as 1.23e-05, and when there are more than 12 digits before the decimal so that 1234567890000 becomes 1.234568e+12.

To create the scatterplot of the two variables, age vs. dental area, and add the regression curve to this plot, we can again use the `ggplot()` function:

```
g <- ggplot(data=data, aes(x,y)) + geom_point() +
  xlab("Dental area") + ylab("age")
g <- g+stat_smooth(method="lm", se=TRUE,
  formula=y ~ poly(x,2,raw=T), colour="red")
g
```

Figure 10.12.
Scatterplot
of age vs.
dental area,
least squares
curve and
corresponding
confidence
interval



We obtain Figure 10.12, which shows that the regression model describes the experimental data very satisfactorily and, therefore, it can be used for prediction:

```
new <- data.frame(x=c(1150, 1250))
predict(fit, new, se.fit=TRUE)
```

From Figure 10.13 we obtain:

First sample: 9.3 ± 1.1 thousands years ago; Second sample: 25.3 ± 1.3 thousands years ago

Alternatively, we can estimate the age via the regression model:

$\text{age}_1 = 435.1 - 0.8586 \cdot 1150 + 0.000425 \cdot 1150^2 = 9.2435$ thousands years ago

$\text{age}_2 = 435.1 - 0.8586 \cdot 1250 + 0.000425 \cdot 1250^2 = 25.2875$ thousands years ago

We note that there is some difference between the predicted values from the `predict()` function and the regression equations. This shows that the number of digits for the coefficients in Figure 10.11 is inadequate for a precise prediction.

To increase the number of digits, we can use the `coef()` function (Figure 10.14):

Figure 10.13.
Predicted
values and
their standard
deviations

```
> new <- data.frame(x=c(1150, 1250))
> predict(fit, new, se.fit=TRUE)
$fit
      1      2
 9.26325 25.30591

$se.fit
      1      2
1.117176 1.256381
```

```
> coef(fit)
      (Intercept) poly(x, 2, raw = TRUE)1 poly(x, 2, raw = TRUE)2
      4.351370e+02      -8.586165e-01      4.246013e-04
```

Figure 10.14.
Model
coefficients

If we use the values of Figure 10.14, we obtain:

$\text{age1} = 435.137 - 0.8586165 \cdot 1250 + 0.0004246 \cdot 1250^2 = 9.26324$ thousands years ago

$\text{age2} = 435.137 - 0.8586165 \cdot 1250 + 0.0004246 \cdot 1250^2 = 25.30591$ thousands years ago

which are identical to those obtained from the `predict()` function.

For the diagnostic plots, we can use the commands:

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

which create the plots of Figure 10.15. We again see that there are some deviations from the expected patterns, which should be attributed to the small sample size and the great variability of the variables used in this example.

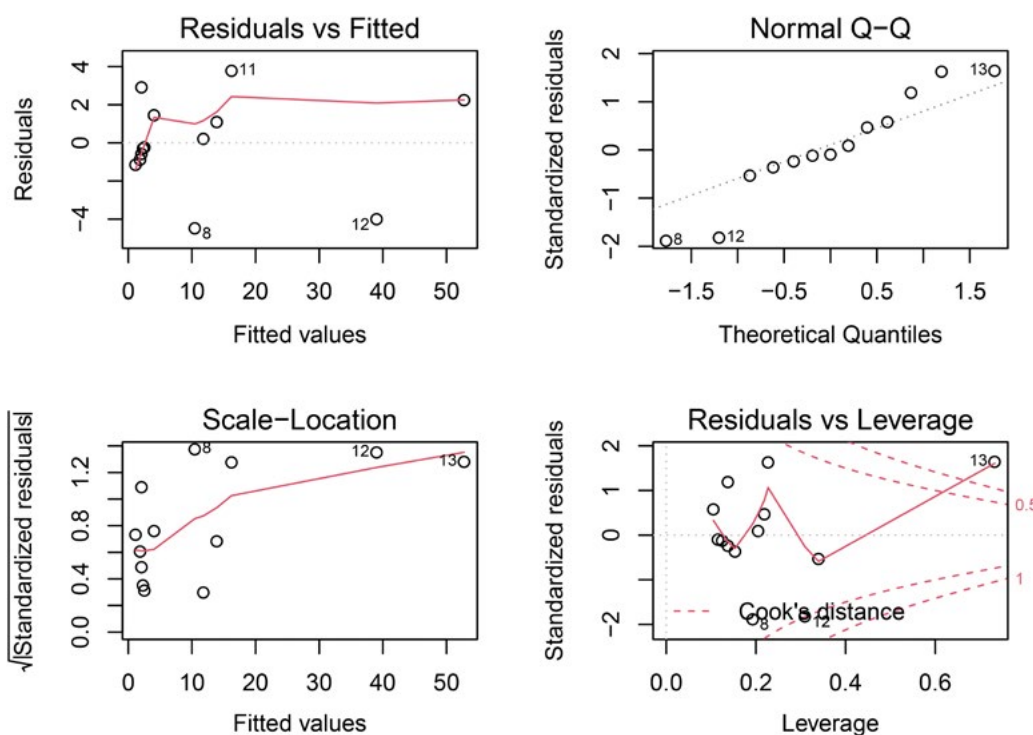


Figure 10.15.
Diagnostic
plots for
regression
assumptions
test

Multiple regression

In multiple regression there are several predictors. Thus, the outcome variable is predicted from a combination of all the variables multiplied by their respective coefficients:

$$y = b_0 + b_1x_1 + b_2x_2 + \dots + b_nx_n$$

Here, y is the outcome variable, b_1 is the coefficient of the first predictor (x_1), b_2 is the coefficient of the second predictor (x_2), ..., b_n is the coefficient of the n^{th} predictor (x_n).

Often many of the predictors may not have a significant effect on the outcome. For this reason, it is advisable to use *stepwise regression* in R to determine the regression model with the fewest predictors, which still describes the experimental data well.

As an example, consider the dataset given in **Table 10.3**, which shows in a contemporary population the dependence of the height (h) of adult males upon maximum femoral length (MFL), maximum tibial length (MTL), maximum humeral length (MHL), maximum ulnar length (MUL), and maximum radial length (MRL). We want to determine the optimum linear model, i.e. to find an equation to predict height based on the minimum number of predictors.

Table 10.3.
Dependence
of the height
(h) of adult
males upon the
maximum length
of the upper
and lower limb
long bones. All
measurements
are in cm

h	MFL	MTL	MHL	MUL	MRL
165.1	45.2	37	36.9	28.2	24.3
160.3	43.2	37.9	33.9	21.1	25.7
163.8	42.7	37.7	34.5	27.5	27.9
167.5	43.6	37	34.4	24	26.3
165.9	45.9	36.1	32.5	24.6	26.4
161.2	43.6	34.1	33.2	27	18.8
161	42.5	36.3	34.5	22.2	19.8
162.1	41.9	36.7	33.6	21	19.7
171	44.4	40.1	33.2	27.3	27.9
149	39.5	33.5	25	21.9	13.6
149.9	37.6	33.3	25.5	22.5	16.4
172	45.8	40.7	34.2	30.3	25.6
174.6	47.9	39	40.2	26.2	29.6
150.7	40.6	33.5	23.7	19.4	15.8
162.7	43.3	37.6	34.5	26.3	20.4
161.2	41.4	36.5	34.9	24.7	20.2
166.9	45.2	35.1	37.1	23.1	24.7
162.3	42	35.2	30.4	22.8	27
152.9	41.5	35.1	29.9	25.3	23
160.7	43.2	34.5	30.4	25.2	20.1
159.2	43.6	35.8	29.4	27.1	24.6
167.8	46.9	39.7	31.9	30.2	22.2
170.2	47	36.9	37.3	28.3	24.5
166.9	43.7	37	32.3	26.8	24.4
151.4	41	34.5	26.9	20	16.3
151.1	38.6	34	28.3	23.2	21.1
148.6	38.1	35.1	29.4	18.5	17.5
157.7	42.5	36.2	29.7	22.8	22.5

We save the data of **Table 10.3** in a tab delimited .txt file, the *Height.txt* file in the *R_Files* folder. Then we input this file to R and run multiple regression analysis through the commands:

```
data <- read.table("Height.txt", head=1, sep="\t")
fit <- lm(h ~ ., data=data)
summary(fit)
```

The results are given in [Figure 10.16](#) and show that the full model is:

```
> data <- read.table("Height.txt", head=1, sep="\t")
> fit <- lm(h ~ ., data=data)
> summary(fit)

Call:
lm(formula = h ~ ., data = data)

Residuals:
    Min       1Q   Median       3Q      Max
-4.975 -1.346  0.030  1.452  3.233

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  57.8966    11.2518   5.146 3.71e-05 ***
MFL           1.2189     0.3250   3.750 0.00111 **
MTL           0.7032     0.3539   1.987 0.05951 .
MHL           0.4766     0.1872   2.546 0.01841 *
MUL           0.1671     0.2091   0.799 0.43279
MRL           0.2711     0.1671   1.622 0.11896
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.334 on 22 degrees of freedom
Multiple R-squared:  0.9201,    Adjusted R-squared:  0.9019
F-statistic: 50.67 on 5 and 22 DF,  p-value: 2.438e-11
```

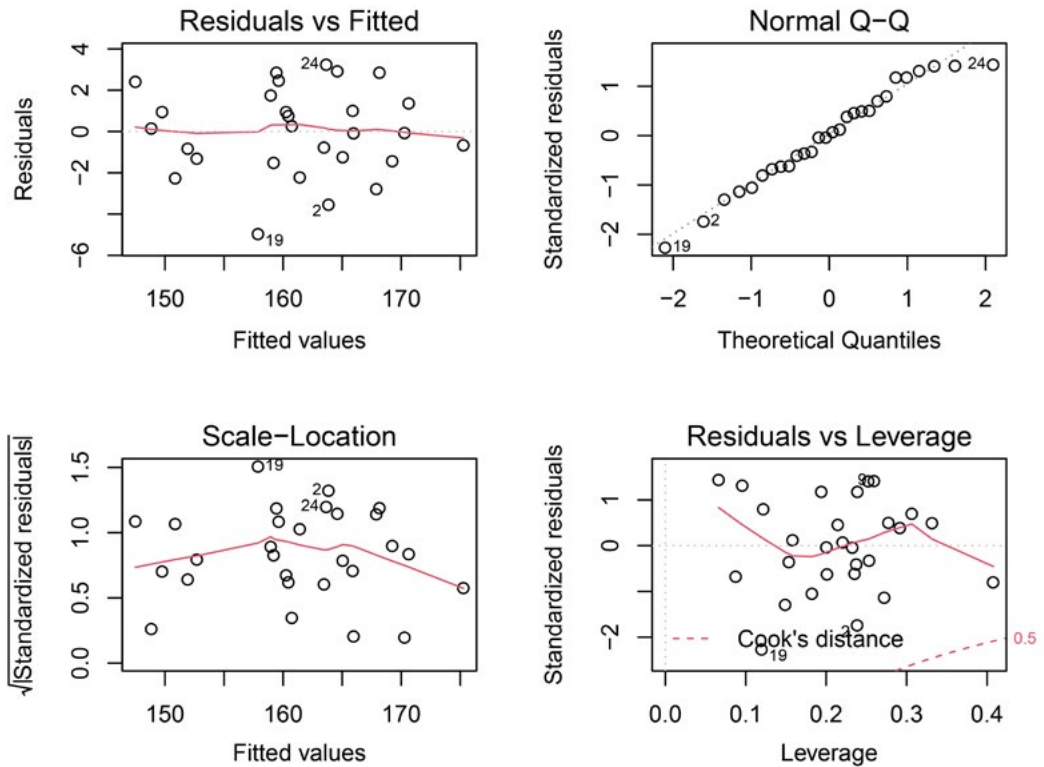
Figure 10.16.
Multiple regression results when all predictors are in the model

$$h = 57.8966 + 1.2189 \cdot \text{MFL} + 0.7032 \cdot \text{MTL} + 0.4766 \cdot \text{MHL} + 0.1671 \cdot \text{MUL} + 0.2711 \cdot \text{MRL}$$

From the model coefficients, only the constant and the coefficients of MFL and MHL are statistically significant ($p\text{-value} < 0.05$).

To examine whether the model fulfils the regression assumptions, we create the diagnostic plots of [Figure 10.17](#) using again the `par(mfrow=c(2, 2))` and `plot(fit)` functions. We observe that in this example all regression assumptions hold. The first plot, the scatterplot between residuals and predicted values, looks more or less random. In the second plot (Q-Q plot) all points fall approximately along the reference line. The third plot (Scale-Location plot) looks random and, therefore, the homoscedasticity assumption is not violated. Finally, the last plot (Cook's distance plot), used to identify outliers, shows that the data do not present any extreme points. The upper Cook's distance line is not shown on the Residuals vs. Leverage plot because all points are well inside the Cook's distance lines.

Figure 10.17.
Diagnostic plots for regression assumptions test



A final test is the multicollinearity assumption. It can be tested using the `vif(fit)` function of the *car* package:

```
library(car)
vif(fit)
```

We obtain the results of Figure 10.18. As a rule of thumb, we should be concerned if the largest VIF is greater than 10. In our example, the VIF score for all predictors is much lower than 10 and, therefore, there seems to be no problem with multicollinearity.

Figure 10.18.
Results of the multicollinearity test

```
> library(car)
> vif(fit)
      MFL      MTL      MHL      MUL      MRL
3.625539 2.483570 2.705275 2.155402 2.405689
```

It is seen that the model fulfils all the regression assumptions but, as shown above, the model includes predictors the contribution of which is not statistically significant. In order to remove the non-significant predictors, we can use backward elimination. For this purpose, we need to use the `stepAIC()` function from the *MASS* package.

Thus, first we install the *MASS* package and then we use the commands:

```
library(MASS)
step <- stepAIC(fit, direction="backward", k=2,
               trace=F)
summary(step)
```

The output is given in Figure 10.19, from which we obtain that the reduced model is:

$$h = 54.771 + 1.3341 \cdot \text{MFL} + 0.7787 \cdot \text{MTL} + 0.4587 \cdot \text{MHL} + 0.2761 \cdot \text{MRL}$$

Note that even this reduced model contains statistically non-significant predictors since MRL has $p\text{-value} = 0.109 > 0.05$. The reason is that the backward elimination technique applied by the `stepAIC()` function searches for the model with the lowest AIC value and not the model with only statistically significant predictors. The AIC value is not given when using the `summary()` function. To obtain this value, we can use the `AIC()` function. These values are given in Figure 10.20. The value 134.1674 corresponds to the initial model and the value 132.9686 to the reduced model. As expected, the AIC value of the reduced model is smaller than that of the initial model and, for this reason, the reduced model is better than the initial one.

```
> library(MASS)
> step <- stepAIC(fit, direction="backward", k=2, trace=F)
> summary(step)

Call:
lm(formula = h ~ MFL + MTL + MHL + MRL, data = data)

Residuals:
    Min       1Q   Median       3Q      Max
-4.6359 -1.3580  0.0425  1.7409  3.4620

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   54.7710     10.4668   5.233 2.62e-05 ***
MFL             1.3341       0.2890   4.616 0.000121 ***
MTL             0.7787       0.3384   2.302 0.030760 *
MHL             0.4587       0.1844   2.488 0.020529 *
MRL             0.2761       0.1656   1.667 0.109107
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.315 on 23 degrees of freedom
Multiple R-squared:  0.9178,    Adjusted R-squared:  0.9035
F-statistic: 64.18 on 4 and 23 DF,  p-value: 3.847e-12
```

Figure 10.19.
Model
parameters
after stepwise
regression

```
> AIC(fit)
[1] 134.1674
> AIC(step)
[1] 132.9686
```

Figure 10.20.
AIC values of
the initial and
the reduced
models

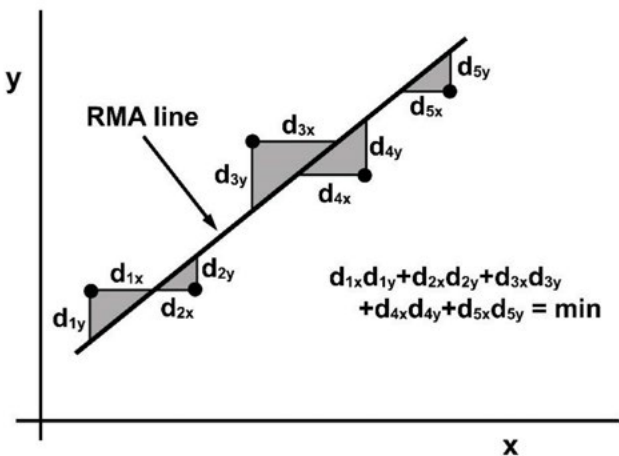
Finally, we should point out that we can perform diagnostic tests for the regression assumptions of the reduced model, precisely as in the initial model, i.e. using the function `plot(step)`.

Reduced Major Axis regression

When the best model is obtained by minimizing the sum of the squared residuals, as described in the above types of regression, the procedure is called *ordinary least squares fitting (OLS)*. The underlying assumption of this procedure is that we control the independent variable x and measure the response variable y , so that measurement errors concern exclusively the variable y . When both variables, x and y , are subject to experimental errors, the regression procedure may minimize the errors in both variables.

There are several ways to do this. In reduced major axis regression (RMA), also called standard major axis (SMA) regression, we minimize the areas of the triangles formed by a point and the regression line (Figure 10.21).

Figure 10.21.
Definition of
the RMA line



For example, Table 10.4 gives the stature and femoral physiological length for 15 males. We want to determine the equation that predicts stature (y) based upon femoral physiological length (FPHL) using both OLS and RMA. Note that variables such as age, stature, body mass, long bone length, etc. are subject not only to measurement error, as any quantitative variable, but also to natural variability, which should be added to the measurement error. For this reason, certain researchers suggest the use of RMA in biological systems (Ruff et al. 2012¹⁰); however, Smith (2009¹¹) has criticised this approach.

Table 10.4.
Data on stature
and femoral
length. All
measurements
in cm

Stature	Femoral Physiological Length (FPHL)	Stature	Femoral Physiological Length (FPHL)
175	48	189	53
170	43	183	53
182	50	175	44
187	52	172	43
177	48	178	45
168	43	180	48
170	42	167	42
192	55		

¹⁰ Ruff CB, Holt BM, Niskanen M, Slad k V, et al. 2012. Stature and body mass estimation from skeletal remains in the European Holocene. *American Journal of Physical Anthropology* 148: 601-617

¹¹ Smith RJ. 2009. Use and misuse of the reduced major axis for line-fitting. *American Journal of Physical Anthropology* 140: 476-486

The RMA method may be implemented in R using the `lmodel2()` function of the *lmodel2* package. In particular, we first install the *lmodel2* package and then we use the commands:

```
library(lmodel2)
rma.data <- read.table("RMA.txt", head=1,
  sep="\t")
fit <- lmodel2(Stature ~ FPL, data=rma.data,
  "relative", "relative", 999)
fit
```

where the arguments *"relative"*, *"relative"* indicate that variables y and x have a true zero (relative-scale variables). If they are interval-scale variables, we should replace *"relative"* by *"interval"*. Finally, the number 999 shows the number of permutations for the estimation of the confidence intervals of the model coefficients.

The results obtained are shown in Figure 10.22.

```
> library(lmodel2)
> rma.data <- read.table("RMA.txt", head=1, sep="\t")
> fit <- lmodel2(Stature ~ FPL, data=rma.data, "relative", "relative", 999)
> fit
```

Model II regression

Call: `lmodel2(formula = Stature ~ FPL, data = rma.data, range.y = "relative", range.x = "relative", nperm = 999)`

n = 15 r = 0.9494537 r-square = 0.9014624
 Parametric P-values: 2-tailed = 6.523001e-08 1-tailed = 3.261501e-08
 Angle between the two OLS regression lines = 2.575742 degrees

Permutation tests of OLS, MA, RMA slopes: 1-tailed, tail corresponding to sign
 A permutation test of r is equivalent to a permutation test of the OLS slope
 P-perm for SMA = NA because the SMA slope cannot be tested

Regression results

	Method	Intercept	Slope	Angle (degrees)	P-perm (1-tailed)
1	OLS	100.09543	1.641140	58.64470	0.001
2	MA	93.78343	1.774681	60.59954	0.001
3	SMA	95.96575	1.728510	59.94921	NA
4	RMA	98.53545	1.674144	59.14928	0.001

Confidence intervals

	Method	2.5%-Intercept	97.5%-Intercept	2.5%-Slope	97.5%-Slope
1	OLS	84.66387	115.5270	1.316031	1.966250
2	MA	74.14238	108.2740	1.468110	2.190218
3	SMA	79.16634	109.9000	1.433710	2.083928
4	RMA	82.20594	113.6951	1.353419	2.019620

Eigenvalues: 79.06976 1.520714

H statistic used for computing C.I. of MA: 0.007178232

Figure 10.22.

Results obtained from the application of the `lmodel2()` function

BE CAREFUL

In the *lmodel2* package RMA is indicated as SMA since RMA is also called Standard Major Axis (SMA). The RMA in Figure 10.22 stands for Ranged Major Axis, which is different from the reduced major axis.

From the results of the RMA fitting presented in Figure 10.22, we obtain that the RMA fitting model may be expressed as:

$$y = 95.966 + 1.729 \cdot \text{FPL}$$

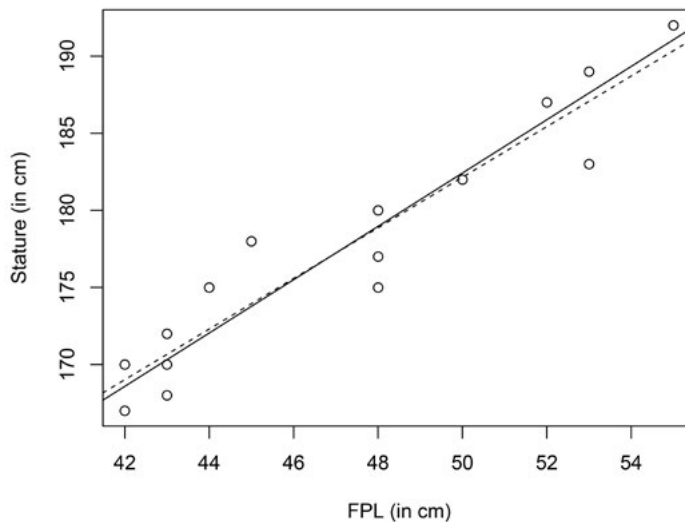
whereas the corresponding OLS model is $y = 100.095 + 1.641 \cdot \text{FPL}$

The plot may be created using:

```
plot(rma.data$FPL, rma.data$Stature,
     xlab="FPL (in cm)", ylab="Stature (in cm)")
abline(95.966, 1.729)
abline(100.095, 1.641, lty=2)
```

where the command `abline(a, b)` plots the straight line $y = a + bx$. The plot is given in Figure 10.23, which shows that the fitting lines of the two models, RMA and OLS, are slightly different.

Figure 10.23.
Plot of stature
vs FPHL and
the RMA (—)
and OLS (---)
fitting straight
lines



10.3. LOGISTIC REGRESSION

Logistic regression is multiple regression but with a response variable that is binary categorical and predictor variables that are continuous or categorical. To understand why we need to develop a new technique to handle a binary dependent variable instead of not simply apply linear regression analysis to a binary outcome, we should take into account the aim of regression as a family of statistical analyses. This issue is discussed below.

The meaning of prediction in regression analysis

In statistics we are interested in the population properties; the sample is the means to get information for the population. Consider that a researcher wants to predict the stature of an individual from femoral length. To do that, the researcher takes a random sample from the population under study consisting of pairs of values (x = femoral length, y = stature) and, using simple linear regression, establishes the linear model:

$$y = a + bx$$

In this model, at a certain x (=femoral length) value corresponds just one y (=stature) value. However, in the population used in the analysis, due to natural variability, many different statures are expected to correspond to each femoral length. Thus, the y value in the linear model that is calculated at

a certain x value is, in fact, the mean value of all y values (say statures) in the population that correspond to a certain x value (say femoral length). From this point of view, the linear model may be written as:

$$\mu_y = a + bx$$

These arguments can be extended to multiple regression, in which there are several predictors.

The linear regression model

$$y = b_0 + b_1x_1 + b_2x_2 + \dots + b_nx_n$$

estimates the mean value of all y values in the population that correspond to a certain set of $x_1, x_2, \dots, x_n$ values. Therefore, in this case the linear model can be written as:

$$\mu_y = b_0 + b_1x_1 + b_2x_2 + \dots + b_nx_n$$

The origin of logistic regression

When the response variable y is a binary variable that takes the values 0 and 1, the mean of y is equal to the probability P that variable y will take the value 1. Therefore, the simple linear model is expressed as:

$$P = a + bx$$

However, here we have to solve the following problem. The variable P takes values in the range from 0 to 1, whereas the quantity $a + bx$ may vary from $-\infty$ to $+\infty$. Therefore, the simple linear model cannot be applied when the dependent variable is a binary variable. To solve this problem, P should be replaced by a function of P that can take any value from $-\infty$ to $+\infty$. Such a function is the *logit function*: $\text{logit}(P) = \ln(P/(1-P))$. Therefore, in its simplest form, when there is only one predictor variable x , the logistic regression equation is given by:

$$\ln \frac{P}{1-P} = a + bx$$

which yields:

$$P = \frac{1}{1 + e^{-(a+bx)}}$$

where e is the base of natural logarithms. When there are several predictors, the above equation is extended to:

$$P = \frac{1}{1 + e^{-(b_0 + b_1x_1 + b_2x_2 + \dots + b_nx_n)}}$$

Model determination and evaluation

From the above expressions of the logistic regression models, we note that just like linear regression, each predictor variable in the logistic regression model has its own coefficient. However, there is a distinct difference between linear and logistic regression. In linear regression the model coefficients are estimated using algebraic equations derived from the minimization of the sum of the squared residuals. In logistic regression, the determination of the model, i.e. the estimation of the predictor coefficients, is carried out using *maximum-likelihood estimation*.

In specific, the predictor coefficients are estimated by maximizing the log-likelihood:

$$\ln L = \sum_{i=1}^N \{y_i \ln P_i + (1 - y_i) \ln (1 - P_i)\}$$

where N is the number of the experimental points, y_i are the values of the binary response variable and P_i is given from:

$$P_i = \frac{1}{1 + e^{-(a+bx_i)}}$$

or:

$$P_i = \frac{1}{1 + e^{-(b_0 + b_1 x_{1i} + b_2 x_{2i} + \dots + b_n x_{ni})}}$$

depending on the number of predictors.

After the determination of the model parameters, we should assess the model. For this purpose, we can use the observed and predicted values. The measure we use is again the log-likelihood, $\ln L$. Small values of the log-likelihood statistic indicate a large amount of unexplained observations, hence poorly fitting statistical models.

The *deviance* is a quantity very closely related to the log-likelihood; it is given by:

$$\text{deviance} = -2 \times \ln L$$

The deviance is related to the *Akaike information criterion (AIC)*, which is often used to judge model fit. It is given by:

$$\text{AIC} = -2 \times \ln L + 2n$$

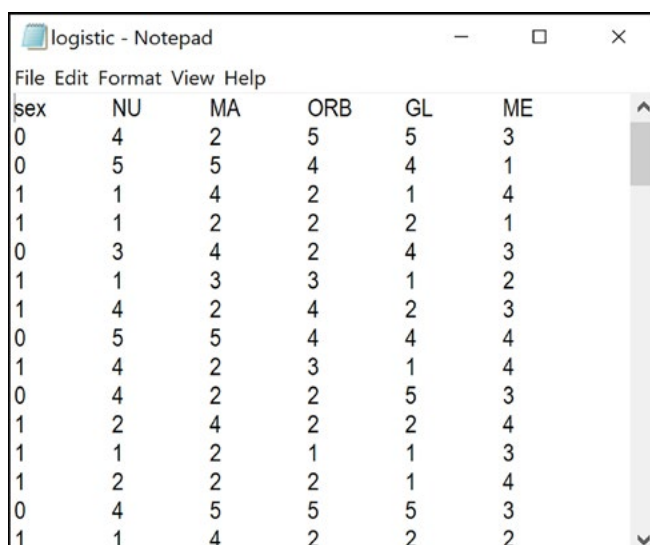
where n is the number of predictors in the model. Given a set of models, the preferred model is the one with the minimum AIC value.

Assumptions of logistic regression

- **Linearity:** In ordinary regression we assume that the relationship between the outcome variable and the predictors is linear. In logistic regression this assumption is violated since the outcome variable is categorical. For this reason, the assumption is that there is a linear relationship between the continuous predictors and the logit of the outcome variable.
- **Independence of errors:** This assumption is the same as for ordinary regression.
- **Multicollinearity:** This assumption is the same as for ordinary regression.

Logistic regression in R

Assume that we want to develop an equation that will allow us to assess if a skeleton is male or female using as predictors the cranial traits mental eminence (ME), supraorbital margin (ORB), supraorbital ridge/glabella (GL), nuchal crest (NU), and mastoid process (MA). The outcome (dependent variable) is sex and it is a categorical variable with two categories, 0 for male and 1 for female. The predictors are ordinal variables recorded using a 5-grade scheme. The data are given in a tab delimited file, part of which is shown in [Figure 10.24](#).



sex	NU	MA	ORB	GL	ME
0	4	2	5	5	3
0	5	5	4	4	1
1	1	4	2	1	4
1	1	2	2	2	1
0	3	4	2	4	3
1	1	3	3	1	2
1	4	2	4	2	3
0	5	5	4	4	4
1	4	2	3	1	4
0	4	2	2	5	3
1	2	4	2	2	4
1	1	2	1	1	3
1	2	2	2	1	4
0	4	5	5	5	3
1	1	4	2	2	2

Figure 10.24.
Part of the
dataset
logistic

In R logistic regression can be performed using the basic function `glm()`, whereas stepwise model selection based on the AIC criterion can be performed using the `stepAIC()` function of the *MASS* package.

Thus, we load the *MASS* package, input the file *logistic.txt* to R and, since it contains missing values, we remove them using the commands:

```
library(MASS)
x <- read.table("logistic.txt", head=1, sep="\t")
xnew <- na.omit(x)
```

Now, we can perform simple logistic regression using the commands:

```
fit <- glm(sex ~ ., family=binomial, data=xnew)
summary(fit)$coefficients
fit$aic
```

where the last two commands give the model coefficients and the AIC value (Figure 10.25).

```
> library(MASS)
> x <- read.table("logistic.txt", head=1, sep="\t")
> xnew <- na.omit(x)
> fit <- glm(sex ~ ., family=binomial, data=xnew)
> summary(fit)$coefficients
              Estimate Std. Error z value Pr(>|z|)
(Intercept) 18.5162744   7.7108342  2.4013322 0.01633550
NU          -1.8263731   0.9917990 -1.8414751 0.06555197
MA           -0.7620358   0.7336747 -1.0386562 0.29896465
ORB          -0.2635446   0.7787876 -0.3384037 0.73505901
GL           -4.1841745   1.6467708 -2.5408360 0.01105878
ME            0.3067214   0.7113711  0.4311693 0.66634528
> fit$aic
[1] 29.82527
```

Figure 10.25.
Model
coefficients
and AIC value

IMPORTANT NOTE

Although in logistic regression the independent variables are discrete and/or continuous in most cases, the discrete independent (explanatory) variables are treated as covariates, i.e. as continuous variables. If we wanted to use say the first three variables and treat them as ordinal, we could use the command:

```
fit <- glm(sex ~ factor(NU)+factor(MA)+factor(ORB), family=binomial, data=xnew)
```

From the results, we obtain the following model:

$$P = \frac{1}{1 + e^{-y}}$$

where P is the probability of being female and

$$y = 18.5163 - 1.8264 * NU - 0.762 * MA - 0.2635 * ORB - 4.1842 * GL + 0.3067 * ME$$

The use of this equation to predict the sex of an unknown individual is straightforward. For example, consider an individual with the following cranial traits: ORB=2, GL=3, ME=2, MA=3, NU=1.

To estimate the individual's sex, we use the commands:

```
ORB=2; GL=3; ME=2; MA=3; NU=1
```

```
y <- 18.5163-1.8264*NU-0.762*MA-0.2635*ORB-4.1842*GL+0.3067*ME
```

```
P <- 1/(1+exp(-y))
```

```
P
```

We obtain the probability P=0.874, which shows that the individual is very likely to be female (Figure 10.26). Note that a P value less than 0.5 corresponds to males.

Figure 10.26.
Prediction
probability P
for an unknown
individual

```
> ORB=2; GL=3; ME=2; MA=3; NU=1
> y<-18.5163-1.8264*NU-0.762*MA-0.2635*ORB-4.1842*GL+0.3067*ME
> P<-1/(1+exp(-y))
> P
[1] 0.8740992
```

However, a prediction from a model is valid only if the accuracy of the model is high enough. To assess model accuracy, we need to compare the initial to the predicted values of the response variable. In our example the initial response variable is the *xnew\$sex* variable and takes the values 0 and 1. The predicted response is given from the variable *fit\$fitted.values*, which gives the probability P. Thus, we should transform P to 0 or 1. This transformation may be achieved via the function *ifelse(test, x, y)*, where *test* is a logical expression that may include a vector, which is filled with either x or y values depending on whether the element of test is TRUE or FALSE.

Thus, to obtain a vector with the predicted sexes, we can use the command:

```
yp <- ifelse(fit$fitted.values>=0.5, 1, 0)
```

Now the comparison between predicted and initial values can be done using the function *table(yp, xnew\$sex)*, which creates the contingency table of Figure 10.27.

Figure 10.27.
Contingency
table

```
> yp<-ifelse(fit$fitted.values >=0.5,1,0)
> t <- table(yp, xnew$sex)
> t

yp    0    1
0  40    2
1   2  44
```

In this table the diagonal elements 40 and 44 give the number of pairs 0-0 and 1-1 and, therefore, the correct predictions for males and females, respectively. The value 2 in the second row is the number of 0-1 pairs and, therefore, the false prediction for females, and the value 2 in the last row is the number of 1-0 pairs and, therefore, the false prediction for males.

Based on these observations, we can compute the accuracy of the model predictions using the following commands that give the results presented in Figure 10.28.

```
t<-table(yp, xnew$sex)
sum(diag(t))/sum(t) # Prediction accuracy (TOTAL)
t[1,1]/(t[1,1]+t[2,1]) # Prediction accuracy (MALES)
t[2,2]/(t[1,2]+t[2,2]) # Prediction accuracy (FEMALES)
```

```
> sum(diag(t))/sum(t) # Prediction accuracy (TOTAL)
[1] 0.9545455
> t[1,1]/(t[1,1]+t[2,1]) # Prediction accuracy (MALES)
[1] 0.952381
> t[2,2]/(t[1,2]+t[2,2]) # Prediction accuracy (FEMALES)
[1] 0.9565217
```

Figure 10.28.
Prediction
accuracy for the
initial model

We observe that the accuracy of the model is high. However, especially in cases of many predictors, the accuracy may be the outcome of *overestimation*. For this reason, we proceed and calculate the *cross-validated accuracy*, adopting the *leave-one-out classification* or *Jack-knife technique*. According to this technique, one case at a time is removed from the analysis, the model is trained on the remaining cases, whereas the removed case is used for prediction and validation.

In R, we can use the `CVbinary()` function of the *DAAG* package for cross-validation. In particular, we can use the commands:

```
library(DAAG)
n <- nrow(xnew)
yy <- CVbinary(fit, rand=c(1:n), nfolds=n,
print.details=F)
yp <- ifelse(yy$cvhat>=0.5,1,0)
t <- table(yp, xnew$sex)
sum(diag(t))/sum(t) # CV accuracy (TOTAL)
t[1,1]/(t[1,1]+t[2,1]) # CV accuracy (MALES)
t[2,2]/(t[1,2]+t[2,2]) # CV accuracy (FEMALES)
```

In these commands, `nrow(xnew)` calculates the number of cases in the *xnew* table and the variable `yy$cvhat` gives the predicted P values from cross-validation. All the other commands are the same as those used previously for the calculation of the model prediction. The results obtained are shown in Figure 10.29; the cross-validated accuracy is less than the prediction accuracy but is still satisfactory.

Figure 10.29.

Cross-validated accuracy for the initial model

```
> library(DAAG)
> n <- nrow(xnew)
> yy <- CVbinary(fit, rand=c(1:n), nfolds=n, print.details=F)
Warning message:
glm.fit: fitted probabilities numerically 0 or 1 occurred
> yp <- ifelse(yy$cvhat>=0.5,1,0)
> t <- table(yp, xnew$sex)
> sum(diag(t))/sum(t) # CV accuracy (TOTAL)
[1] 0.9090909
> t[1,1]/(t[1,1]+t[2,1]) # CV accuracy (MALES)
[1] 0.9285714
> t[2,2]/(t[1,2]+t[2,2]) # CV accuracy (FEMALES)
[1] 0.8913043
```

Finally, we should examine whether a reduced model, i.e. a model with fewer predictors, performs equally well or better than the initial model.

This can be done using the `stepAIC()` function of the *MASS* package:

```
step <- stepAIC(fit, direction="backward",
trace=FALSE)

summary(step)$coefficients

step$aic
```

The results we obtain are given in Figure 10.30. It is seen that we obtain a model with only two predictors:

$$P = \frac{1}{1 + e^{-(16.82202.197 \cdot NU - 4.132 \cdot GL)}}$$

This model has a lower AIC = 25.7 value than the original model, AIC = 29.8, with 5 predictors. Therefore, this model is better than the original model.

Figure 10.30.

Coefficients and AIC value for the reduced model

```
> step <- stepAIC(fit, direction="backward", trace=FALSE)
> summary(step)$coefficients
      Estimate Std. Error  z value    Pr(>|z|)
(Intercept) 16.822154   5.722572   2.939612 0.003286239
NU          -2.197375   0.9422643 -2.332015 0.019699887
GL          -4.132424   1.3314411 -3.103722 0.001911025
> step$aic
[1] 25.70412
```

For the prediction and the cross-validation accuracy, we can use the following commands, which are similar or the same as those used above.

```
yp <- ifelse(step$fitted.values>=0.5,1,0)
t <- table(yp, xnew$sex)
sum(diag(t))/sum(t) # Prediction accuracy (TOTAL)
t[1,1]/(t[1,1]+t[2,1]) # Prediction accuracy (MALES)
t[2,2]/(t[1,2]+t[2,2]) # Prediction accuracy (FEMALES)
```

```

yy <- CVbinary(step, rand=c(1:n), nfolds=n,
print.details=F)

yp <- ifelse(yy$cvhat>=0.5,1,0)

t <- table(yp, xnew$sex)

sum(diag(t))/sum(t) # CV accuracy (TOTAL)
t[1,1]/(t[1,1]+t[2,1]) # CV accuracy (MALES)
t[2,2]/(t[1,2]+t[2,2]) # CV accuracy (FEMALES)

```

The obtained results are shown in Figure 10.31 and they are very satisfactory, especially in what concerns the cross-validated results.

```

> yp <- ifelse(step$fitted.values>=0.5,1,0)
> t <- table(yp, xnew$sex)
> sum(diag(t))/sum(t) # Prediction accuracy (TOTAL)
[1] 0.9545455
> t[1,1]/(t[1,1]+t[2,1]) # Prediction accuracy (MALES)
[1] 0.952381
> t[2,2]/(t[1,2]+t[2,2]) # Prediction accuracy (FEMALES)
[1] 0.9565217
>
> yy <- CVbinary(step, rand=c(1:n), nfolds=n, print.details=F)

Warning message:
glm.fit: fitted probabilities numerically 0 or 1 occurred
> yp <- ifelse(yy$cvhat>=0.5,1,0)
> t <- table(yp, xnew$sex)
> sum(diag(t))/sum(t) # CV accuracy (TOTAL)
[1] 0.9204545
> t[1,1]/(t[1,1]+t[2,1]) # CV accuracy (MALES)
[1] 0.9047619
> t[2,2]/(t[1,2]+t[2,2]) # CV accuracy (FEMALES)
[1] 0.9347826

```

Figure 10.31.
Prediction
and cross-
validated
accuracy for
the reduced
model

10.4. GENERALIZED LINEAR MODELS

Generalized Linear Models (GLM) extend ordinary linear regression to encompass response variables that may have non-normal distributions. In particular, GLM are applied when the dependent variable y follows an exponential distribution, e.g. normal, binomial, Poisson, etc., and it is recorded as a function of n explanatory variables, $x_1, x_2, \dots, x_n$, which can be continuous or/and categorical. The logistic regression examined above is a typical case of GLM.

The mathematical expression of GLM may be written as:

$$\eta = b_0 + b_1x_1 + b_2x_2 + \dots + x_nx_n$$

where $\eta = g(\mu)$ and g is the link function of the mean (μ) of the distribution function of the response variable y in the population (see section 10.2). The linear combination of explanatory variables, $b_0 + b_1x_1 + b_2x_2 + \dots + b_nx_n$, is called the *linear predictor*.

Thus, GLM consist of three elements:

1. a probability distribution for the response variable y ,
2. a linear predictor, and
3. a link function that provides the relationship between the linear predictor and the mean of the distribution function.

There are several options for the distribution function of y and, therefore, for the nature of the response variable y . Based on this feature, the most important GLM are the following:

1. y is a continuous variable following the normal distribution, the link function is the identity function and, therefore, $\eta = \mu$, where μ is the predicted by the model y value, and the predictors are continuous or/and categorical variables. This is the General Linear Model:

$$\eta = \mu_y = b_0 + b_1x_1 + b_2x_2 + \dots + x_px_p$$

This model is reduced to the ANCOVA model (Analysis of Covariance) when the dependent variable varies linearly with each covariate at each level of the categorical variable, and the lines of the above relationships have the same slope.

2. y is a binary variable following the binomial distribution, the link function is the logit function and the predictors are continuous or/and categorical variables. This is, in fact, the *Logistic Regression Model*:

$$\eta = \text{logit}(P) = \ln \frac{P}{1-P} = b_0 + b_1x_1 + b_2x_2 + \dots + b_nx_n$$

However, we should clarify that in logistic regression all the predictors that are ordinal variables are treated as continuous variables, whereas when the above model is analyzed as GLM, the ordinal predictors are treated as factors.

3. y expresses counts/frequencies following the Poisson distribution, the link function is the log function and the predictors are continuous or/and categorical variables. This is the *Log-Linear Regression Model*:

$$\eta = \ln \mu_y = b_0 + b_1x_1 + b_2x_2 + \dots + b_nx_n$$

4. y is an ordinal variable taking the values 1, 2, ..., J , the link function η is a *cumulative logit function* and the predictors are continuous or/and categorical variables. In this case the GLM are expressed via $J-1$ models as follows. Consider that the probabilities of the response variable y to take the values 1, 2, ..., J are $\pi_1, \pi_2, \dots, \pi_J$. We define for the outcome category j ($j = 1, 2, \dots, J$) the cumulative probability from

$$P(y \leq j) = \pi_1 + \pi_2 + \dots + \pi_j$$

Now the GLM for ordinal responses are defined from:

$$\text{logit}(P(y = 1)) = \ln \frac{P(y = 1)}{1 - P(y = 1)} = a_1 + b_1x_1 + b_2x_2 + \dots + b_nx_n$$

$$\text{logit}(P(y \leq 2)) = \ln \frac{P(y \leq 2)}{1 - P(y \leq 2)} = a_2 + b_1x_1 + b_2x_2 + \dots + b_nx_n$$

.....

$$\text{logit}(P(y \leq J-1)) = \ln \frac{P(y \leq J-1)}{1 - P(y \leq J-1)} = a_{J-1} + b_1x_1 + b_2x_2 + \dots + b_nx_n$$

or in a more compact expression:

$$\text{logit}(P(y \leq j)) = a_j + b_1x_1 + b_2x_2 + \dots + b_nx_n, \quad (j = 1, 2, \dots, J-1)$$

where $\text{logit}(P(y \leq j))$ is the *cumulative logit function*.

The use of GLM offers additional information compared to conventional methods. We can assess the impact of multiple factors simultaneously, examine the effect of interactions, and model the experimental data. When GLM is used to model experimental data, the best model is usually determined as follows. We start from the *saturated model* or from any model that includes several interactions, and then we gradually remove the non-significant factors and interactions one by one. The statistically optimum GLM corresponds to the smallest *Akaike information criterion (AIC)*.

GLM in R

In R the basic function for GLM is the `glm()` function provided that the response y is binary or continuous or counts (frequencies) **but not ordinal**. We have already used this function in the logistic regression above. Apart from this function, useful functions are the `Anova()` function of the *car* package to test model effects, i.e. for testing the significance of the various independent variables of the model, and the `predict()` function if we use GLM for prediction. The `ifelse()` and `table()` functions, useful for binary responses, have already been discussed in the logistic regression.

When the response variable y is ordinal, we can use the `polr()` function of the *MASS* package. This function has a similar structure as `glm()` but we do not specify the *family* argument.

In this section, we will examine three examples, the first with a binary response, the second with an ordinal response, and the third with counts (frequencies). For the first example, consider the data of **Table 10.5**, which show the dependence of enthesal changes (EC), i.e. alterations on the bone surface at the sites of muscle/tendon attachment, upon population and age. Enthesal changes (y) have been recorded as a binary variable (0 = absent, 1 = present), the population (pop) is a categorical variable, and age is a continuous variable. We will use GLM to analyze these data.

pop	age	y	pop	age	y	pop	age	y
1	75	1	2	80	1	3	33	1
1	72	1	2	75	0	3	41	0
1	28	1	2	45	0	3	50	0
1	81	1	2	24	0	3	60	1
1	79	1	2	57	0	3	72	1
1	45	0	2	49	0	3	78	1
1	73	1	2	55	0	3	65	1
1	37	0	2	79	1	3	65	0
1	25	0	2	65	0	3	55	1
1	52	1	2	60	0	3	48	0
1	29	0	2	38	0	3	28	0
1	38	0	2	72	1	3	28	0
1	45	1	2	65	0	3	45	1
2	61	0	2	60	0	3	66	1

Table 10.5.
Enthesal change expression for *tibialis anterior* (y) in three population samples (pop) and corresponding age data

First, we organize these data in a .txt tab delimited file, such as the *GLM.txt* file in the *R_Files* folder, and then we use the following commands:

```
GLM <- read.table("GLM.txt", head=1, sep="\t")
fit <- glm(y~factor(pop) + age, data=GLM,
family=binomial, na.action=na.omit)
summary(fit)
library(car)
Anova(fit)
```

These commands yield the table of parameter estimates and model effects (Figure 10.32).

From the table with the parameter estimates, we obtain that the optimum fitting model is:

$$\eta = \ln \frac{p}{1-p} = -5.201 - 4.212 * pop2 - 0.815 * pop3 + 0.123 * age$$

Figure 10.32.
Parameter estimates and model effects table for the model $\eta = b_1 pop_2 + b_2 pop_3 + b_3 age$

```
> GLM <- read.table("GLM.txt", head=1, sep="\t")
> fit <- glm(y~factor(pop)+age, data=GLM, family=binomial, na.action=na.omit)
> summary(fit)

Call:
glm(formula = y ~ factor(pop) + age, family = binomial, data = GLM,
     na.action = na.omit)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0623  -0.5823  -0.1683   0.4968   2.0395

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  -5.20105    1.85608  -2.802  0.00508 **
factor(pop)2  -4.21178    1.57469  -2.675  0.00748 **
factor(pop)3  -0.81466    1.14102  -0.714  0.47524
age           0.12331    0.04073   3.027  0.00247 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 57.843  on 41  degrees of freedom
Residual deviance: 33.065  on 38  degrees of freedom
AIC: 41.065

Number of Fisher Scoring iterations: 6

> library(car)
> Anova(fit)
Analysis of Deviance Table (Type II tests)

Response: y
              LR Chisq Df Pr(>Chisq)
factor(pop)   14.484  2  0.0007159 ***
age           18.392  1  1.798e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note that the variables *pop2* (*pop*=2) and *pop3* (*pop*=3) take only the values 0 and 1. Therefore, when *pop2*=1, then *pop3*=0 and the above equation describes the data that correspond to the second sample; when *pop3*=1, then *pop2*=0 and the equation describes the data of the third sample; and when *pop2*=0 and *pop3*=0, the equation describes the data of the first sample.

From Figure 10.32, and in particular from the output of the command *Anova(fit)*, we observe that both *pop* and *age* have a statistically significant effect on EC. The statistically significant impact

of *pop* means that between the classes (categories) of this variable exist statistically significant differences in EC for the entheses under study. To interpret the results for this variable, which exhibits three classes, we should examine the parameter estimates table. In this table the p-values = 0.00748 and 0.47524 that correspond to the lines *factor (pop) 2* and *factor (pop) 3* refer to the p-values that test the significance of the difference in EC between populations 1-2 and 1-3, respectively, since in this table population 1 acts as a reference population. Thus, we observe that there is a significant difference in EC expression between groups 1-2 but not 1-3 when controlling for the effect of age.

Note that in both tables of parameter estimates and model effects, the impact of each variable on the EC under study is estimated while controlling for the effect of all other variables, i.e. by excluding the effect of all other variables on EC.

In order to obtain the p-value for the pair of groups 2 - 3, we should use the commands:

```
pop <- factor(GLM$pop, levels=c(2,3,1))
fit2 <- glm(GLM$y~pop + GLM$age,
family=binomial, na.action=na.omit)
summary(fit2)
```

where the `factor()` function places group 2 first so that it acts as the reference group. Now, we obtain the p-value 0.00791 for the pair of populations 2-3 (Figure 10.33), which shows a significant difference in EC expression between groups 2-3.

At this point we should stress that since we have multiple comparisons, the p-values between population groups should be corrected using the Holm-Bonferroni correction.

```
> pop<- factor(GLM$pop, levels = c(2,3,1))
> fit2<-glm(GLM$y~pop+GLM$age,family=binomial, na.action=na.omit)
> summary(fit2)

Call:
glm(formula = GLM$y ~ pop + GLM$age, family = binomial, na.action = na.omit)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0623  -0.5823  -0.1683   0.4968   2.0395

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -9.41284    2.90565  -3.239  0.00120 **
pop3           3.39712    1.27916   2.656  0.00791 **
pop1           4.21178    1.57469   2.675  0.00748 **
GLM$age        0.12331    0.04073   3.027  0.00247 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 57.843  on 41  degrees of freedom
Residual deviance: 33.065  on 38  degrees of freedom
AIC: 41.065

Number of Fisher Scoring iterations: 6
```

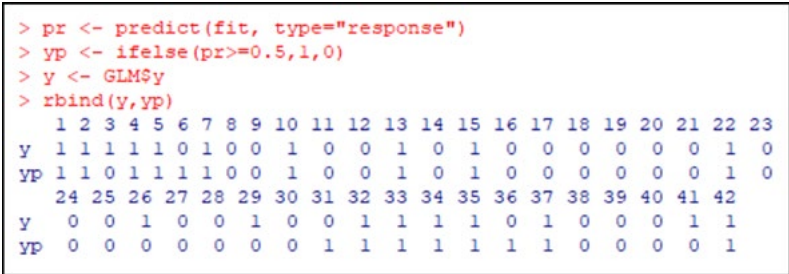
Figure 10.33.
Parameter estimates table for the model $\eta = b_1\text{pop}_2 + b_2\text{pop}_3 + b_3\text{age}$

To assess the prediction performance of the model, we can use the following commands that allow the direct comparison between original (y) and predicted (yp) values.

```
pr <- predict(fit, type="response")
yp <- ifelse(pr>=0.5,1,0)
y <- GLM$y
rbind(y,yp)
```

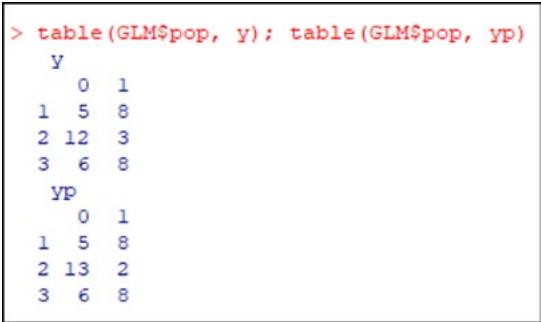
Figure 10.34 shows that this comparison is very satisfactory.

Figure 10.34. Comparison between original y and predicted yp values



Finally, based on the original y and predicted yp values, we have created the contingency tables of Figure 10.35, where it is seen that the model describes the experimental data very satisfactorily.

Figure 10.35. Contingency tables of original y and predicted yp values



The contingency tables are created using the `table(GLM$pop, y); table(GLM$pop, yp)` commands:

As a second example, consider the dataset of Table 10.6. This dataset is the dataset of Table 10.5 but here the enthesal changes have been recorded in an ordinal scale. Therefore, we should use an ordinal GLM. In R for ordinal GLM we can use the `polr()` function of the MASS package. This function has a similar structure as `glm()` but we do not specify the *family* argument.

pop	age	y	pop	age	y	pop	age	y
1	75	3	2	80	3	3	33	2
1	72	3	2	75	1	3	41	1
1	28	2	2	45	1	3	50	1
1	81	3	2	24	1	3	60	3
1	79	2	2	57	2	3	72	2
1	45	1	2	49	1	3	78	3
1	73	2	2	55	1	3	65	3
1	37	1	2	79	2	3	65	1
1	25	1	2	65	1	3	55	2
1	52	2	2	60	1	3	48	1
1	29	1	2	38	1	3	28	1
1	38	1	2	72	3	3	28	1
1	45	2	2	65	3	3	45	2
2	61	1	2	60	1	3	66	3

Table 10.6.
Enthesal change
expression
for *tibialis
anterior*
(y) in three
population
samples
(pop) and
corresponding
age data

The analysis is very similar to that of the previous example with `polr()` in place of `glm()`.

Thus, the following commands give the parameter estimates and model effects table for the ordinal GLM of Figure 10.36.

```
library(MASS)
GLMord <- read.table("GLM-ord.txt", head=1,
  sep="\t")
fit <- polr(factor(y)~factor(pop) + age,
  data=GLMord, na.action=na.omit)
summary(fit)
library(car)
Anova(fit)
```

Note that the `polr()` function does not give p-values for the parameter coefficients but t-values. According to this statistic, a coefficient is statistically significant when the absolute value of the t-value is greater than 2.

In order to obtain the t-value for the pair of groups 2 - 3, we should use the following commands, which yield Figure 10.37.

```
pop <- factor(GLMord$pop, levels=c(2,3,1))
fit2 <- polr(factor(GLMord$y)~pop +
  GLMord$age, na.action=na.omit)
summary(fit2)
```

Figure 10.36.
Parameter estimates and model effects table for the ordinal GLM when group 1 is the reference group

```
> library(MASS)
> GLMord <- read.table("GLM-ord.txt", head=1, sep="\t")
> fit <- polr(factor(y) ~ factor(pop) + age, data=GLMord, na.action=na.omit)
> summary(fit)
```

Re-fitting to get Hessian

Call:
polr(formula = factor(y) ~ factor(pop) + age, data = GLMord, na.action = na.omit)

Coefficients:

	Value	Std. Error	t value
factor(pop)2	-1.77066	0.88807	-1.9938
factor(pop)3	0.11119	0.80923	0.1374
age	0.09617	0.02543	3.7817

Intercepts:

	Value	Std. Error	t value
1 2	4.6109	1.4412	3.1994
2 3	6.4784	1.6615	3.8992

Residual Deviance: 65.04306
AIC: 75.04306

```
> library(car)
> Anova(fit)
```

Analysis of Deviance Table (Type II tests)

Response: factor(y)

	LR	Chisq	Df	Pr(>Chisq)
factor(pop)	6.1993	2	0.04506	*
age	20.3598	1	6.416e-06	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Figure 10.37.
Parameter estimates table for the ordinal GLM when group 2 is the reference group

```
> pop <- factor(GLMord$pop, levels=c(2,3,1))
> fit2 <- polr(factor(GLMord$y) ~ pop + GLMord$age, na.action=na.omit)
> summary(fit2)
```

Re-fitting to get Hessian

Call:
polr(formula = factor(GLMord\$y) ~ pop + GLMord\$age, na.action = na.omit)

Coefficients:

	Value	Std. Error	t value
pop3	1.88186	0.87490	2.151
pop1	1.77064	0.88807	1.994
GLMord\$age	0.09617	0.02543	3.782

Intercepts:

	Value	Std. Error	t value
1 2	6.3815	1.6909	3.7741
2 3	8.2491	1.9372	4.2582

Residual Deviance: 65.04306
AIC: 75.04306

From Figures 10.36 and 10.37 we observe that again both *pop* and *age* have a statistically significant effect on EC. The t-values = -1.99 and 0.137 that correspond to the lines *factor (pop) 2* and *factor (pop) 3* in Figure 10.36 refer to t-values that test the significance of the difference in EC between populations 1-2 and 1-3, respectively, since in this table population 1 is used as the reference population. Thus, we observe that the differences in EC expression between groups 1-2 and 1-3 when controlling for the effect of age are not statistically significant. In contrast, the t-value 2.151 for the pair of populations 2-3 (Figure 10.37) shows a significant difference in EC expression between groups 2-3.

Finally, for the comparison between original (*y*) and predicted (*ypr*) values, we can use the following commands:

```
ypr <- predict(fit, type="class")
y <- GLMord$y
data.frame(rbind(y,ypr))
table(GLMord$pop, y);
table(GLMord$pop, ypr)
```

These commands give the results presented in Figures 10.38 and 10.39, which show a rather unsatisfactory prediction accuracy for the model.

```
> ypr <- predict(fit, type="class")
> y <- GLMord$y
> data.frame(rbind(y,ypr))
  X1 X2 X3 X4 X5 X6 X7 X8 X9 X10 X11 X12 X13 X14 X15 X16 X17 X18 X19 X20 X21 X22 X23
y   3  3  2  3  2  1  2  1  1  2  1  1  2  1  3  1  1  1  2  1  1  2  1
ypr 3  3  1  3  3  1  3  1  1  2  1  1  1  1  2  2  1  1  1  1  1  2  1
  X24 X25 X26 X27 X28 X29 X30 X31 X32 X33 X34 X35 X36 X37 X38 X39 X40 X41 X42
y   1  1  3  3  1  2  1  1  3  2  3  3  1  2  1  1  1  2  3
ypr 1  1  2  1  1  1  1  1  2  3  3  3  3  2  1  1  1  1  3
```

Figure 10.38.
Comparison
between
original *y* and
predicted *ypr*
values

```
> table(GLMord$pop, y); table(GLMord$pop, ypr)
  y
  1 2 3
1  5 5 3
2 10 2 3
3  6 4 4
 ypr
  1 2 3
1  7 1 5
2 11 4 0
3  7 2 5
```

Figure 10.39.
Contingency
tables of
original *y* and
predicted *ypr*
values

Note that an ordinal GLM is usually expressed as:

$$\text{logit}(P(y \leq j)) = a_j + b_1x_1 + b_2x_2 + \dots + b_nx_n, \quad (j = 1, 2, \dots, J - 1)$$

However, when using the `polr()` function, the ordinal GLM is defined from:

$$\text{logit}(P(y \leq j)) = a_j - (b_1x_1 + b_2x_2 + \dots + b_nx_n), \quad (j = 1, 2, \dots, J - 1)$$

That is, the model coefficients $b_1, b_2, \dots$ are determined with the opposite sign. Therefore, the ordinal GLM of this example is expressed by the following two equations, depending on the value of a_j (Figure 10.36):

$$\eta_j = \text{logit}(P(y \leq j)) = a_j - (-1.771 * \text{pop2} + 0.111 * \text{pop3} + 0.096 * \text{age}) \quad (j = 1, 2)$$

where $a_1 = 4.611$ and $a_2 = 6.478$. This expression seems complicated but things are simplified if we take into account that predictions can be made via the `predict()` function.

If, though, we want to apply the above model, we should work as follows. First, we need to compute the inverse logit in order to calculate the probabilities $P(y \leq j)$ from η_j . This can be achieved using the `inv.logit()` function of the `boot` package. Then, we should define the values of the independent variable. For example, if `pop=1` and `age=79`, then we should use `pop2=0`, `pop3=0` and `age=79`.

For this example, the estimated probabilities of the three categories of the response variables by the ordinal GLM can be computed using the commands:

```
library(boot)
pop2=0; pop3=0; age=79
n1 <- 4.611 -(-1.771*pop2 + 0.111*pop3 +
0.096*age)
n2 <- 6.478 -(-1.771*pop2 + 0.111*pop3 +
0.096*age)
p1 <- inv.logit(n1)
p2 <- inv.logit(n2)
p1 # probability of first category of y
p2-p1 # probability of second category of y
1-p2 # probability of third category of y
```

We obtain the results presented in Figure 10.40, which show that the greatest probability is that of the third category and, therefore, the predicted value of y is 3.

Figure 10.40.
Model
predictions

```
> library(boot)
> pop2 <- 0; pop3 <- 0; age <- 79
> n1 <- 4.611 -(-1.771*pop2 + 0.111*pop3 + 0.096*age)
> n2 <- 6.478 -(-1.771*pop2 + 0.111*pop3 + 0.096*age)
> p1 <- inv.logit(n1)
> p2 <- inv.logit(n2)
> p1 # probability of first category of y
[1] 0.04866066
> p2-p1 # probability of second category of y
[1] 0.1999567
> 1-p2 # probability of third category of y
[1] 0.7513826
```

In order to improve the model, we can include interactions via the commands:

```
fit3 <- polr(factor(y)~factor(pop) + age +
factor(pop)*age, data=GLMord,
na.action=na.omit)
summary(fit3)
library(car)
Anova(fit3)
```

However, the results we obtain are still unsatisfactory (Figure 10.41): The AIC value of the new (saturated) model, 77.95, is greater than the AIC value of the model with main effects, 75.04, and

the interaction terms are not statistically significant. Thus, in this dataset we cannot improve the model performance.

```
> fit3<-polr(factor(y)~factor(pop)+age+factor(pop)*age, data=GLMord,
> summary(fit3)

Re-fitting to get Hessian

Call:
polr(formula = factor(y) ~ factor(pop) + age + factor(pop) *
      age, data = GLMord, na.action = na.omit)

Coefficients:
                Value Std. Error t value
factor(pop)2    -6.34247    4.81662  -1.3168
factor(pop)3    -0.77411    2.81837  -0.2747
age              0.08110    0.03155   2.5706
factor(pop)2:age  0.07197    0.07291   0.9871
factor(pop)3:age  0.01611    0.04991   0.3227

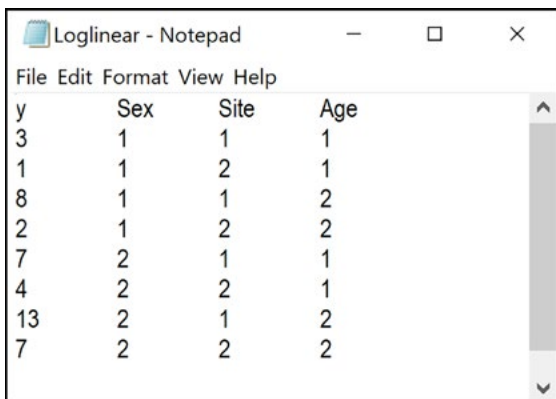
Intercepts:
      Value  Std. Error t value
1|2  3.7684   1.7572    2.1446
2|3  5.6648   1.9316    2.9326

Residual Deviance: 63.95253
AIC: 77.95253
> library(car)
> Anova(fit3)
Analysis of Deviance Table (Type II tests)

Response: factor(y)
              LR Chisq Df Pr(>Chisq)
factor(pop)      6.1993  2   0.04506 *
age             20.3598  1  6.416e-06 ***
factor(pop):age   1.0905  2   0.57969
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 10.41.
Parameter estimates and model effects table for the full ordinal GLM

As a third example, assume that we want to examine the association between sex, age, archaeological site and dental caries based on the dataset given in Figure 10.42. In this Figure, y is the frequency (number of individuals with dental caries), Sex takes the values 1 for males and 2 for females, Age is 1 for young adults and 2 for old adults, and Site represents the archaeological site and takes the values 1 for site A and 2 for site B.



y	Sex	Site	Age
3	1	1	1
1	1	2	1
8	1	1	2
2	1	2	2
7	2	1	1
4	2	2	1
13	2	1	2
7	2	2	2

Figure 10.42.
Table showing how many individuals exhibit dental caries per sex, age group and archaeological site

In order to model the data of Figure 10.42 and examine the association between sex, age, archaeological site and dental caries, we should take into account the following. First, the dependent

variable expresses counts/frequencies and, therefore, we should use a Poisson GLM. Second, with three variables, Sex, Site, Age, in the saturated model we have:

1. Main effects: Sex + Site + Age
2. Two-way interactions:
 - a. Sex $\times$ Site
 - b. Age $\times$ Site
 - c. Sex $\times$ Age
3. Three-way interaction: Sex $\times$ Site $\times$ Age

Therefore, in order to determine the saturated Poisson GLM, we can use the commands:

```
pois <- read.table("Loglinear.txt",
  head=1, sep="\t")

fit <- glm(y ~ Sex + Age + Site + Sex*Site +
  Age*Site + Sex*Age + Sex*Site*Age,
  data=pois, family=poisson, na.action=na.omit)

summary(fit)
```

Note that, for simplicity, all categorical independent variables are treated as continuous variables. Figure 10.43 presents the estimated model parameters and the AIC value for the saturated Poisson GLM. It is interesting to note that none of the parameters is statistically significant. This result is usually obtained when an excessive number of independent variables is used. In this case, more reliable information about the significance of the model parameters is obtained from the table of the model effects.

To obtain this table, we can use the commands:

```
library(car)

Anova(fit)
```

Figure 10.43.
Model
parameters
and AIC
value for the
saturated
Poisson GLM

```
> pois <- read.table("Loglinear.txt", head=1, sep="\t")
> fit <- glm(y ~ Sex + Age + Site + Sex*Site + Age*Site + Sex*Age +
> summary(fit)

Call:
glm(formula = y ~ Sex + Age + Site + Sex * Site + Age * Site +
  Sex * Age + Sex * Site * Age, family = poisson, data = pois,
  na.action = na.omit)

Deviance Residuals:
[1] 0 0 0 0 0 0 0 0

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.03036     6.71338   0.005   0.996
Sex            0.89835     3.75094   0.239   0.811
Age           1.85856     3.82168   0.486   0.627
Site          -1.12167     5.06205  -0.222   0.825
Sex:Site       0.31074     2.78379   0.112   0.911
Age:Site      -0.51594     2.90619  -0.178   0.859
Sex:Age       -0.59005     2.14600  -0.275   0.783
Sex:Age:Site   0.22826     1.60342   0.142   0.887

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 1.9450e+01  on 7  degrees of freedom
Residual deviance: 1.7764e-15  on 0  degrees of freedom
AIC: 42.84

Number of Fisher Scoring iterations: 3
```

The model effects obtained for the saturated Poisson GLM are given in Figure 10.44 and show that Sex, Site, and Age have a significant contribution to the outcome, whereas the contribution of the interactions is not statistically significant and, therefore, these interactions can be eliminated from the model. However, a better approach to determine the optimum GLM is via the AIC value, as presented below.

```
> library(car)
> Anova(fit)
Analysis of Deviance Table (Type II tests)

Response: y
      LR Chisq Df Pr(>Chisq)
Sex      6.5844  1  0.01029 *
Age      5.0970  1  0.02397 *
Site      6.5844  1  0.01029 *
Sex:Site   0.9020  1  0.34225
Age:Site   0.0272  1  0.86912
Sex:Age    0.1861  1  0.66616
Sex:Age:Site 0.0201  1  0.88723
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 10.44.
Model effects
table for the
saturated
Poisson GLM

At this point, it is interesting to see that the saturated model gives a precise prediction of the frequencies. Thus, if we use the following commands, we obtain the results of Figure 10.45, which show that the predicted frequencies coincide with the original ones.

```
ypr <- predict(fit, data=pois[, -1],
type="response")
y <- pois$y
cbind(ypr, y)
```

```
> ypr <- predict(fit, data=pois[, -1], type="response")
> y <- pois$y
> cbind(ypr, y)
  ypr  y
1   3  3
2   1  1
3   8  8
4   2  2
5   7  7
6   4  4
7  13 13
8   7  7
```

Figure 10.45.
Comparison
of original
y and
predicted ypr
frequencies

Now, in order to determine the optimum Poisson GLM based on the AIC value, we can use the `stepAIC()` function of the *MASS* package:

```
library(MASS)
step <- stepAIC(fit, trace=F)
ystep <- predict(step, data=pois[, -1],
type="response")
summary(step)
cbind(ystep, y)
```

Figure 10.46.
Results
obtained from
the Poisson
GLM

```
> library(MASS)
> step <- stepAIC(fit, trace=F)
> ystep <- predict(step, data=pois[-1], type="response")
> summary(step)

Call:
glm(formula = y ~ Sex + Age + Site, family = poisson, data = pois,
     na.action = na.omit)

Deviance Residuals:
    1      2      3      4      5      6      7      8 
-0.12118 -0.39752  0.59637 -0.56218 -0.04455  0.42169 -0.33278  0.22173 

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.4746    0.8798   0.539   0.5896
Sex            0.7949    0.3220   2.469   0.0136 *
Age            0.6931    0.3162   2.192   0.0284 *
Site          -0.7949    0.3220  -2.469   0.0136 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 19.4500  on 7  degrees of freedom
Residual deviance:  1.1841  on 4  degrees of freedom
AIC: 36.024

Number of Fisher Scoring iterations: 4
```

The results obtained are given in Figures 10.46 and 10.47 and show that the reduced model:

$$\ln(\text{Frequency}) = 0.4746 + 0.7949 * \text{Sex} + 0.6931 * \text{Age} - 0.7949 * \text{Site}$$

$$\text{or } \text{Frequency} = e^{0.4746+0.7949*\text{Sex}+0.6931*\text{Age}-0.7949*\text{Site}}$$

describes satisfactorily the dataset of Figure 10.42.

Figure 10.47.
Comparison
between
original and
predicted y
values

```
> cbind(ystep, y)
      ystep y
1  3.214815 3
2  1.451852 1
3  6.429630 8
4  2.903704 2
5  7.118519 7
6  3.214815 4
7 14.237037 13
8  6.429630 7
```

10.5. GENERALIZED ESTIMATING EQUATIONS

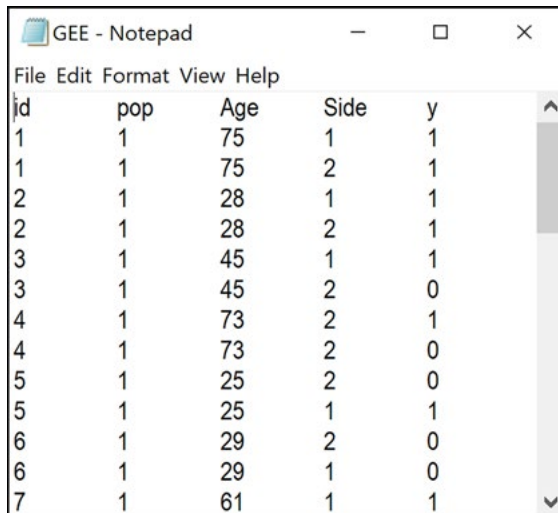
A basic prerequisite for the application of GLM is the lack of any correlation between outcomes (the values of the dependent variable). If this condition is not met, we should apply *Generalized Estimating Equations* (GEE). That is, GEE is a version of GLM that are appropriate when a possible correlation exists between outcomes, i.e. when there are more than one measurements for each case under study. A typical example is the study of bilateral asymmetry in skeletal remains. Note that bilateral asymmetry may also be treated using the traditional approach based on *paired samples t-test* and *Wilcoxon test*.

To apply GEE, data should be properly arranged. Each variable should be given in a column and each individual/object in a row. Note that a necessary parameter for applying GEE is the *id*, i.e. an integer denoting the individual/object to whom the data of each case belong. In the GLM/GEE terminology this variable is called *subject variable*. For example, **Table 10.7** shows the data arrangement in a dataset where we examine enthesal changes (EC). The first variable, *id*, denotes the skeleton number, the second variable, *pop*, denotes the population (group) to which the individual belongs, then we have the *Age* of the individual, the *Side* of the body (right or left) on which we examine the enthesal changes, and finally *y*, which is the response variable, i.e. the variable that depends on all other variables and which, in the present example, is the enthesal changes recorded as present/absent. Note that **Table 10.7** is similar to **Table 10.6** but now we also have the side variable, where 1 = right and 2 = left. Because we want to examine bilateral asymmetry, we need to use GEE, and we need to include the *id* variable, as explained above.

id	pop	Age	Side	y	id	pop	Age	Side	y
1	1	75	1	1	12	2	79	1	1
1	1	75	2	1	12	2	79	2	0
2	1	28	1	1	13	2	38	1	0
2	1	28	2	1	13	2	38	2	0
3	1	45	1	1	14	2	72	1	1
3	1	45	2	0	14	2	72	2	0
4	1	73	2	1	15	3	33	1	0
4	1	73	2	0	15	3	33	2	1
5	1	25	2	0	16	3	41	1	0
5	1	25	1	1	16	3	41	2	0
6	1	29	2	0	17	3	72	1	1
6	1	29	1	0	17	3	72	2	1
7	1	61	1	1	18	3	78	1	1
7	1	61	2	0	18	3	78	2	1
8	2	80	1	1	19	3	55	1	0
8	2	80	2	0	19	3	55	2	1
9	2	24	1	0	20	3	48	1	0
9	2	24	2	0	20	3	48	2	0
10	2	57	1	0	21	3	28	1	0
10	2	57	2	0	21	3	28	2	1
11	2	55	1	0	22	3	66	1	1
11	2	55	2	0	22	3	66	2	0

Table 10.7.
Enthesal change expression for *tibialis anterior* (y) in three population samples (pop) and corresponding age and side data

Figure 10.48.
Part of the
GEE.txt file



id	pop	Age	Side	y
1	1	75	1	1
1	1	75	2	1
2	1	28	1	1
2	1	28	2	1
3	1	45	1	1
3	1	45	2	0
4	1	73	2	1
4	1	73	2	0
5	1	25	2	0
5	1	25	1	1
6	1	29	2	0
6	1	29	1	0
7	1	61	1	1

The `glm()` function cannot perform GEE. Instead we use the `geeglm()` function of the *geepack* package. To run GEE, we create the tab delimited *GEE.txt* file (Figure 10.48) in the *R_Files* folder and use the commands:

```
library(geepack)
GEE <- read.table("GEE.txt", head=1, sep="\t")
fit <- geeglm(y~factor(pop) + factor(Side) +
Age, data=GEE, id = GEE$id, family = binomial)
summary(fit)
anova(fit)
```

We obtain the results shown in Figures 10.49 and 10.50. From these results we observe that the effect of side does not appear to be statistically significant, i.e. statistically significant differences in EC between the right and left side are not demonstrated ($p\text{-value}=0.1787 > 0.05$). This is the only information that we need to obtain from the application of GEE. The effect of the other variables, *pop*, *age*, may also be obtained from the Model Effects table and the corresponding Parameter Estimates table but these effects have already been explored by means of GLM, as described above. Note that, as shown in Nikita (2014)¹², for the study of bilateral asymmetry and paired samples in general, GEE is recommended, whereas in all other analyses, GLM is more appropriate.

¹² Nikita E. 2014. The use of generalized linear models and generalized estimating equations in bioarchaeological studies. *American Journal of Physical Anthropology* 153: 473-483

```

> library(geepack)
> GEE <- read.table("GEE.txt",head=1,sep="\t")
> fit <- geeglm(y~factor(pop) + factor(Side) + Age, data=GEE, id = GEE$id,
> summary(fit)

Call:
geeglm(formula = y ~ factor(pop) + factor(Side) + Age, family = binomial,
       data = GEE, id = GEE$id)

Coefficients:
              Estimate Std. err   Wald Pr(>|W|)
(Intercept)    -1.5001   1.1811    1.61  0.20406
factor(pop)2    -2.6376   0.7424   12.62  0.00038 ***
factor(pop)3    -0.6968   0.8046    0.75  0.38650
factor(Side)2   -1.1800   0.8413    1.97  0.16073
Age              0.0530   0.0202    6.86  0.00883 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Correlation structure = independence
Estimated Scale Parameters:

              Estimate Std. err
(Intercept)     0.95    0.552
Number of clusters: 22 Maximum cluster size: 2

```

Figure 10.49.
Model
parameters of
the GEE

```

> anova(fit)
Analysis of 'Wald statistic' Table
Model: binomial, link: logit
Response: y
Terms added sequentially (first to last)

              Df   X2 P(>|Chi|)
factor(pop)    2  5.00    0.0819 .
factor(Side)   1  1.81    0.1787
Age            1  6.86    0.0088 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

Figure 10.50.
Model effects
of the GEE

11.

CATEGORICAL DATA

The tests we examined in the previous chapters deal with quantitative data. However, archaeological material is often described using qualitative variables. For example, such variables are the colour of the pottery, the type of temper used, the raw material from which different stone tools were made, the presence or absence of a certain disease in the skeletal remains of a population, or the existence of different animals in an assemblage. These are qualitative variables.

An effective approach to treat categorical data is to calculate frequencies and based on these frequencies, to construct contingency tables and associated graphs: Clustered and stacked bar charts and pie charts. This issue has been discussed in section 3.5. However, from a chart it is difficult to assess whether the observed patterns are statistically significant or not. For this purpose, we should run hypotheses tests to the contingency table. The most important of these tests are the chi-squared test of independence and the Fisher exact test.

11.1. CHI-SQUARED TEST

The chi-squared test of independence is used to determine if there is a significant relationship between two nominal (categorical) variables. The null hypothesis and its alternative are:

H_0 : The variables under examination are independent

H_1 : The variables under examination depend upon each other

The test statistic for the test of independence is denoted as χ^2 , and is computed from Pearson's chi-square:

$$\chi^2 = \sum_{i=1} \sum_{j=1} \frac{(\text{observed}_{ij} - \text{expected}_{ij})^2}{\text{expected}_{ij}}$$

where i represents the rows in the contingency table and j represents the columns. Thus, observed_{ij} is the observed cell counts in the i^{th} row and j^{th} column of the contingency table and expected_{ij} is the expected cell counts in the i^{th} row and j^{th} column of the table, computed from:

$$\text{expected}_{ij} = \frac{(\text{sum of } i \text{ row}) \times (\text{sum of } j \text{ column})}{\text{total sum}}$$

The test statistic follows the chi-square distribution with $(R-1)(C-1)$ degrees of freedom, where R , C are the number of rows and columns, respectively, of the contingency table. Two important assumptions underline the chi-squared test: The sample size should be sufficiently large, and there should be adequate expected cell counts. In a 2x2 table a common rule is 5 or more expected counts in each cell, whereas in larger tables the expected counts should be 5 or more in at least 80% of the cells and no cell should have an expected count of less than one.

In R we can use the functions `chisq.test(x)` and `chisq.test(x, simulate.p.value=TRUE)`, where x is the contingency matrix and `simulate.p.value` is a logical indicating whether to compute p-values by Monte-Carlo simulation or not. As an example, consider that in a cemetery we want to examine if there is an association between the sex of the deceased and the presence of dental caries. It has been postulated that females exhibit dental caries more often than males due to their stronger

involvement in food processing, as well as due to various hormonal factors. We want to test if this general pattern also applies to our dataset. In this example, there are two categorical variables: *sex* (male or female) and *dental caries* (present or absent). First, we count how many individuals fall within each category, and the results are given in the contingency **Table 11.1**.

		Sex		Total
		Males	Females	
Dental Caries	Yes	28	43	71
	No	19	21	40
	Total	47	64	111

Table 11.1.
Contingency table showing how many individuals exhibit dental caries per sex

To apply the `chisq.test(x)` function, we can use the commands:

```
r1 <- c(28, 43)
r2 <- c(19, 21)
x <- rbind(r1, r2)
chisq.test(x)
chisq.test(x, simulate.p.value=TRUE)
```

The output is given in **Figure 11.1** and it is seen that the p-value is 0.5317 from the original test and 0.421 from Monte-Carlo simulations. Thus, even though females exhibit more often caries than males, there does not appear to be a statistically significant association between sex and dental caries in this sample.

```
> r1 <- c(28, 43)
> r2 <- c(19, 21)
> x <- rbind(r1, r2)
> chisq.test(x)

Pearson's Chi-squared test with Yates' continuity correction

data: x
X-squared = 0.39113, df = 1, p-value = 0.5317

> chisq.test(x, simulate.p.value=TRUE)

Pearson's Chi-squared test with simulated p-value (based on
2000 replicates)

data: x
X-squared = 0.68139, df = NA, p-value = 0.4213
```

Figure 11.1.
Results of the chi-squared test

11.2. FISHER'S EXACT TEST

As stated above, the chi-squared test is used when in each cell of the contingency table we have an expected frequency of at least 5. If this criterion is not met, we apply Fisher's exact test. Note that Fisher's exact test is often performed in 2x2 contingency tables, that is, when we examine the association between two variables and each variable has two categories. In R it is applied using the `fisher.test(x)` function.

Assume that we want to examine again the association between sex and dental caries; however, this time using the data given in contingency **Table 11.2**.

Table 11.2.
Contingency
table showing
how many
individuals
exhibit dental
caries per sex

		Sex		Total
		Males	Females	
Dental Caries	Yes	11	20	31
	No	3	11	14
	Total	14	31	45

We input the data and run the test by typing in the R Console:

```
r1 <- c(11, 20)
r2 <- c(3, 11)
x <- rbind(r1, r2)
fisher.test(x)
```

We obtain the output shown in Figure 11.2, where it is seen that the p-value is 0.4921, thus in this case too we cannot identify a significant difference between males and females in the frequency of dental caries.

Figure 11.2.
Results of the
Fisher Exact
test

```
> r1 <- c(11, 20)
> r2 <- c(3, 11)
> x <- rbind(r1, r2)
> fisher.test(x)

Fisher's Exact Test for Count Data

data: x
p-value = 0.4921
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
 0.3988899 13.4503446
sample estimates:
odds ratio
 1.986953
```

11.3. CORRESPONDENCE ANALYSIS

The χ^2 test of independence and the Fisher exact test are effective when the contingency table has few rows and columns. In contrast, if the contingency table has many rows and columns, the above tests do not offer any particular information since there will definitely be correlations among the different variables. In such cases a useful tool is *Correspondence Analysis*, which provides a graphic method of exploring the relationship between the variables of a contingency table.

There are several packages in R that can be used to perform correspondence analysis. Here, we will use the *ca* package and, based on this package, we will examine the following example. Consider the contingency Table 11.3, which shows the inter-relationship between tomb typology and age of the deceased.

To assess whether there is an association between tomb typology and the age of the deceased, we apply correspondence analysis using the commands:

```
library(ca)
R1 <- c(4, 1, 3, 15)
R2 <- c(4, 4, 8, 2)
R3 <- c(3, 15, 3, 1)
X <- rbind(R1, R2, R3)
rownames(X) <- c("cist", "tholos", "pit")
colnames(X) <- c("SA", "YA", "MA", "OA")
fit <- ca(X)
plot(fit)
```

	Subadult (SA)	Young adult (YA)	Middle adult (MA)	Old adult (OA)
Cist	4	1	3	15
Tholos	4	4	8	2
Pit	4	15	3	1

Table 11.3.
Contingency table showing the inter-relationship between tomb typology and age of the deceased

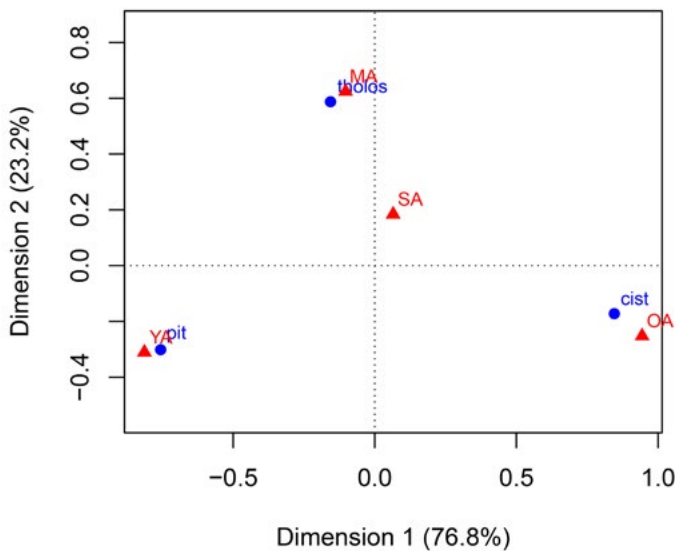


Figure 11.3.
Plot of the correspondence analysis for the data of Table 11.3

The output is given in Figure 11.3, where we observe that there is indeed some differentiation in tomb typology according to the age of the deceased, with the exception of subadults. Old adults are mostly buried in cist tombs, young adults in pits, and middle adults in tholos tombs.

12.

MULTIVARIATE ANALYSES

Multivariate Data Analysis refers to any statistical technique for analyzing multivariate datasets. A *multivariate dataset* consists of measurements or observations on two or more variables recorded for each case. The primary target of Multivariate Analysis is to identify patterns and relationships between and within sets of variables. For example, we may want to assess if the pottery from two archaeological sites is significantly different based simultaneously on the height, base diameter, lip diameter, and other dimensions of the pots.

Multivariate analyses consist of many different techniques. In this chapter we will examine *Principal Component Analysis* (PCA), *Hierarchical Cluster Analysis* (HCA), *Linear Discriminant Analysis* (LDA), *Multidimensional Scaling* (MDS), and *Multivariate Analysis of Variance* (MANOVA). LDA and MANOVA are quite sensitive to outliers and the size of the smallest group must be larger than the number of predictor variables. In addition, they require the dependent variables to follow the multivariate normal distribution within each group of the independent variables, and the covariance matrices of each group to be equal. In contrast, PCA and HCA are free from assumptions concerning the distribution of the data.

12.1. MULTIVARIATE NORMALITY TEST AND DETECTION OF OUTLIERS

Tests for multivariate normality are available in special packages of R, such as the *MVN* package. For example, we can use the function `mvn(x, mvnTest)` of this package, where `x` is a numeric matrix or data frame with multivariate data, and `mvnTest` selects one of the MVN tests. In specific, we often use `mvnTest = "mardia"` for Mardia's test, `mvnTest = "hz"` for Henze-Zirkler's test, or `mvnTest = "royston"` for Royston's test.

Mardia's test involves the calculation of Mardia's multivariate skewness and kurtosis coefficients as well as their corresponding statistical significance. Henze-Zirkler's multivariate normality test is based on the Mahalanobis distance and is generally considered quite powerful. Finally, Royston's test is a multivariate extension of the univariate Shapiro-Wilk test.

Consider that we want to examine the normality of the multivariate dataset of **Table 12.1**, which presents the cranial dimensions of three contemporary Cypriot populations (groups).

Table 12.1.
Cranial
dimensions
(in mm)

group	maximum breadth	basi-bregmatic height	basi-alveolar length	nasal height
1	130	138	90	49
1	126	132	91	52
1	132	133	100	50
1	117	132	96	47
1	133	145	99	55
1	137	137	89	56
1	140	133	106	49
1	122	139	93	45
1	131	134	105	55
1	133	134	97	51

group	maximum breadth	basi-bregmatic height	basi-alveolar length	nasal height
2	137	140	96	53
2	128	135	93	49
2	130	139	89	46
2	131	134	107	50
2	134	134	96	48
2	144	132	95	50
2	139	138	98	43
2	137	146	99	55
2	136	131	92	46
2	126	137	95	58
3	138	126	92	46
3	132	129	88	59
3	143	128	84	54
3	134	124	94	50
3	132	127	97	52
3	138	125	88	57
3	129	130	84	53
3	140	135	95	58
3	147	129	88	42
3	138	127	87	50

We save the data of **Table 12.1** in a tab delimited .txt file titled *cranialmeasurements.txt* and install the *MVN* package. Then we input the data in R and load the package *MVN*:

```
cranialdata <-  
read.table("cranialmeasurements.txt", head=1, sep="\t")  
library(MVN)
```

In order to perform the test per group and select only the columns with cranial measurements, excluding the grouping variable, we use the command:

```
group <- subset(cranialdata, group=="1")[,2:5]
```

and perform Henze-Zirkler's multivariate normality test using:

```
x <- group  
mvn(x, mvnTest="hz")
```

Part of the results obtained is shown in **Figure 12.1**, which shows that deviations from multivariate normality are not demonstrated for the first group ($p\text{-value}=0.541>0.05$). Note that the function also tests for (univariate) normality all variables using the Shapiro-Wilk test.

Figure 12.1.
Part of the
results when
performing
Henze-
Zirkler's
multivariate
normality test

```
> cranialdata<-read.table("cranialmeasurements.txt",head=1,sep="\t")
> library(MVN)
> group <- subset(cranialdata, group=="1")[,2:5]
> x <- group
> mvn(x, mvnTest = "hz")
$multivariateNormality
      Test      HZ  p value MVN
1 Henze-Zirkler 0.6061627 0.541075 YES

$univariateNormality
      Test      Variable Statistic  p value Normality
1 Shapiro-Wilk    max_breadth    0.9540    0.7164    YES
2 Shapiro-Wilk basi_bregmatic_height 0.8431    0.0480    NO
3 Shapiro-Wilk basi_alveolar_length 0.9384    0.5353    YES
4 Shapiro-Wilk    nasal_height    0.9516    0.6879    YES
```

To test all three groups, we repeat the above procedure by altering in the command `group = subset(mvdata, group=="1")` the number 1 to 2 and, finally, to 3. In all cases we find that the data under consideration do not show deviations from multivariate normality.

For the detection of outliers, we may calculate the Mahalanobis distance of each case from the centroid of the dataset. If this distance for a certain case is statistically significant, i.e., if its p-value is smaller than 0.05, the case may be considered to be an outlier. Note that the MD of a point from the centroid of points follows the χ^2 distribution with degrees of freedom equal to the number of variables included in the calculation.

In R the Mahalanobis distances of each case from the centroid of the dataset can be calculated using the `outlier(x)` function of the *psych* package, where *x* is a numeric matrix or dataframe with multivariate data, excluding a grouping variable. In addition, the p-value can be estimated using the `pchisq(d, df)` function, where *d* is the value of the Mahalanobis distance and *df* the degrees of freedom.

Thus, we can use the commands:

```
library(psych)
group <- subset(cranialdata, group=="1")[,2:5]
x <- group
mah <- outlier(x)
pval <- 1-pchisq(mah, ncol(x))
y <- cbind(mah,pval)
y
plot(y[,1],y[,2], xlab="Mahalanobis distance",
     ylab="p-value")
abline(h=0.05, col="blue")
```

Note that *y* is a table with Mahalanobis distances vs. p-values and the two last commands give the plot of p-value vs. Mahalanobis distance with a horizontal blue line at p-value = 0.05. Therefore, the outliers in this plot are shown as points below the blue line.

The results we obtain from the above code are given in Figures 12.2 and 12.3. It is seen that the data are free from outliers, as all p-values are above 0.17 (thus > 0.05), and this is why the blue horizontal line is not visible in the graph of Figure 12.3. This is in line with the finding that these data do not show deviations from multivariate normality.

```

> library(psych)
> group <- subset(cranialdata, group=="1")[,2:5]
> x <- group
> mah <- outlier(x)
> pval <- 1-pchisq(mah, ncol(x))
> y <- cbind(mah,pval)
> y

```

	mah	pval
1	2.1737868	0.7038311
2	2.7982918	0.5921276
3	0.7894802	0.9398528
4	4.2991786	0.3670280
5	6.4132167	0.1703412
6	4.6753475	0.3222590
7	5.6708798	0.2251123
8	4.2511261	0.3730851
9	4.4469737	0.3488814
10	0.4817188	0.9752561

Figure 12.2.
Table of
Mahalanobis
distances vs.
p-values

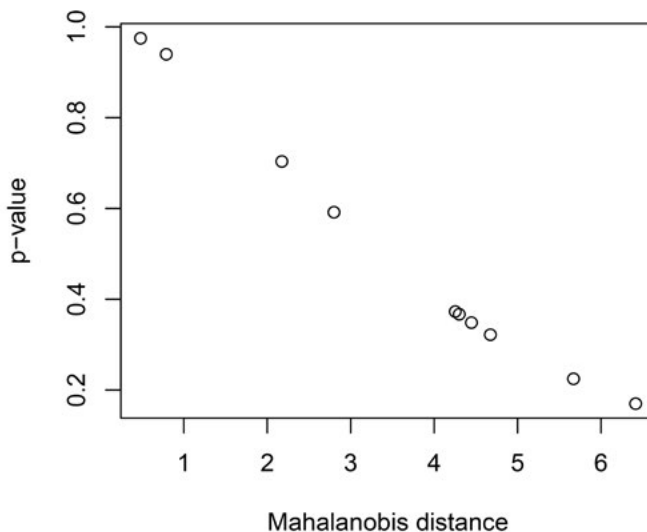


Figure 12.3.
Plot of
p-values vs.
Mahalanobis
distances to
detect outliers

As a second example, consider the data of the *outlier.txt* file. Note that in this file there is no grouping variable. To test for outliers, we use the following commands and obtain the results given in Figures 12.4 and 12.5:

```

data <- read.table("outlier.txt", head=1,
sep="\t")
x <- data
mah <- outlier(x)
pval <- 1-pchisq(mah, ncol(x))
y <- cbind(mah, pval)
y
plot(y[,1],y[,2], xlab="Mahalanobis distance",
ylab="p-value")
abline(h=0.05, col="blue")

```

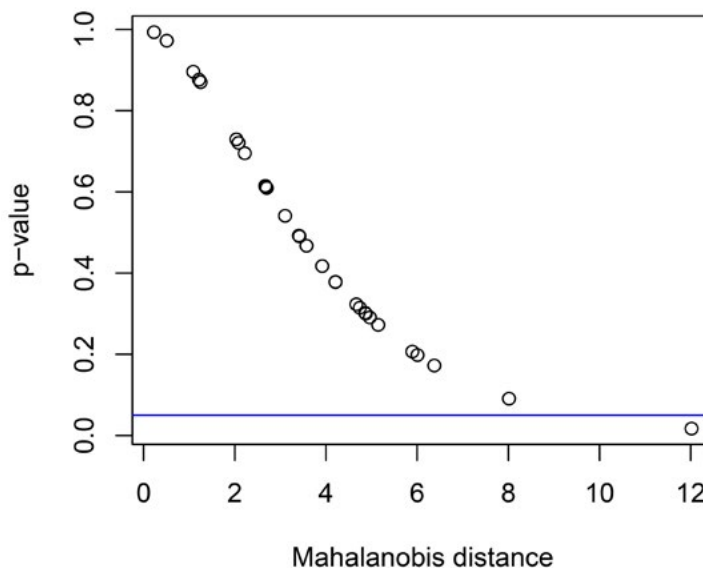
It is seen that in this dataset the fifth case, (147, 124, 81, 40), can be considered to be an outlier.

Figure 12.4.
Table of
Mahalanobis
distances vs.
p-values

```
> data<-read.table("outlier.txt",head=1,sep="\t")
> x <- data
> mah <- outlier(x)
> pval <- 1-pchisq(mah, ncol(x))
> y <- cbind(mah,pval)
> y
```

	mah	pval
1	2.2179711	0.69574052
2	2.0317613	0.72991696
3	1.2097804	0.87648572
4	8.0115251	0.09115693
5	12.0165287	0.01722878
6	4.6657895	0.32333914
7	3.0998051	0.54126440
8	5.8940819	0.20719917
9	6.0062916	0.19867891
10	4.2063869	0.37879439
11	0.2259838	0.99407750

Figure 12.5.
Plot of
p-values vs.
Mahalanobis
distances to
detect outliers



12.2. PRINCIPAL COMPONENT ANALYSIS

Principal Component Analysis (PCA) is a multivariate technique used to reduce multidimensional data to lower dimensions, i.e. to reduce the number of variables, while retaining most of the information about the existence of clusters in the dataset. Principal component analysis (PCA) finds hypothetical variables (principal components – PCs) accounting for as much as possible of the variance in a multivariate data table. PCA may be used for the reduction of the dataset to only two variables (the first two components) for plotting purposes and subsequently for identifying *clusters* (groups of similar objects) in a data table.

In order to visualize how this is done, let us examine the simple dataset presented in **Table 12.2**. The graphic display of the data presented in this table is given in **Figure 12.6**, where we observe that there are two clusters of data points. It is interesting to note that the values shown in **Table 12.2** are, in fact, the coordinates of the data points in a 2D scatterplot. Therefore, in general, the values of K variables in a multivariate dataset are also coordinates of data points in a K dimensional scatterplot.

However, we can draw scatterplots only for $K = 1, 2$ or 3 and, therefore, there is need to reduce the number of variables up to three, while retaining most of the information.

	var 1	var 2
case 1	2	2
case 2	3	3
case 3	3	2
case 4	7	4.5
case 5	7	4
case 6	8	5.5

Table 12.2.
Example data
table

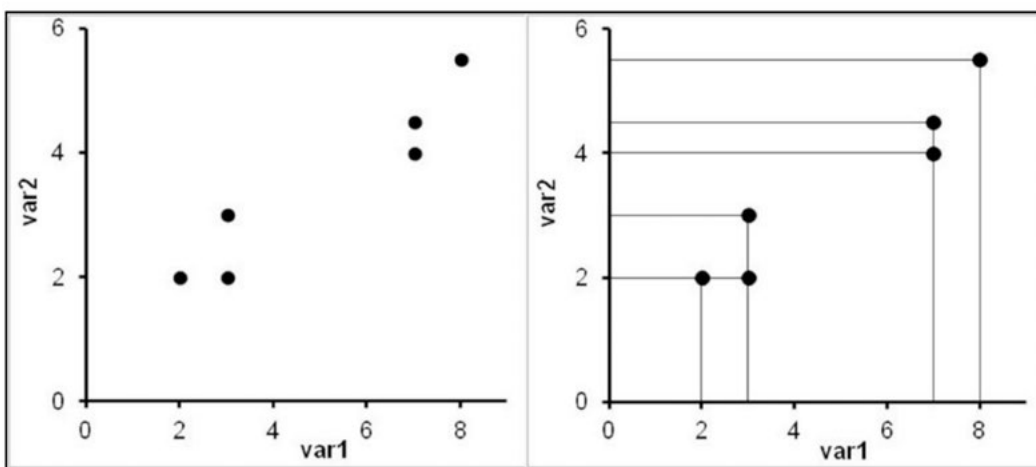


Figure 12.6.
Scatterplots
of the data
points of Table
12.2

In order to apply PCA and reduce the variables of **Table 12.2** from 2 to 1, we first center the variables, i.e. we subtract from the values of each variable the mean variable value. The scatterplot of the centered data is the same as that of the raw values but the origin of the coordinate system is located in the centroid of the data points (**Figure 12.7**). Now, we may transform this coordinate system to a new one as follows:

We draw an axis, called PC1, through the data points of **Figure 12.7** along the greatest dispersal of these points. This axis may be the least-squares straight line. Subsequently, we draw a second axis, PC2, perpendicular to PC1. These two axes form a new coordinate system, which is basically the original coordinate system of **Figure 12.7** rotated by a certain angle.

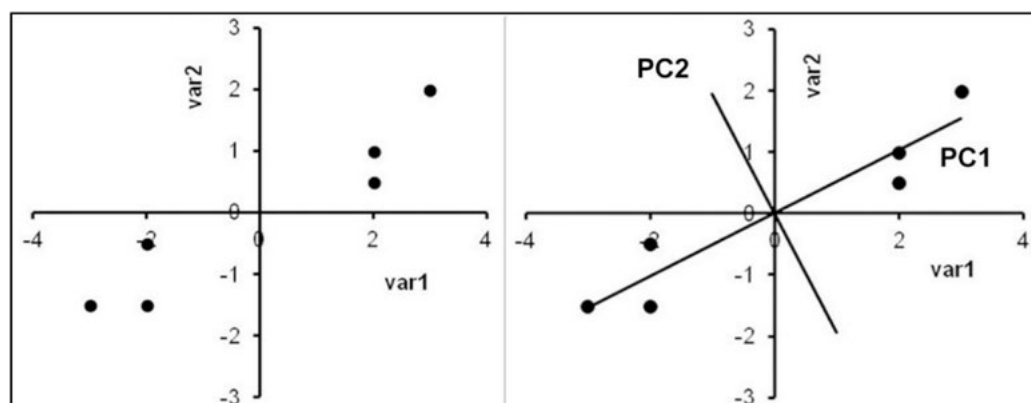
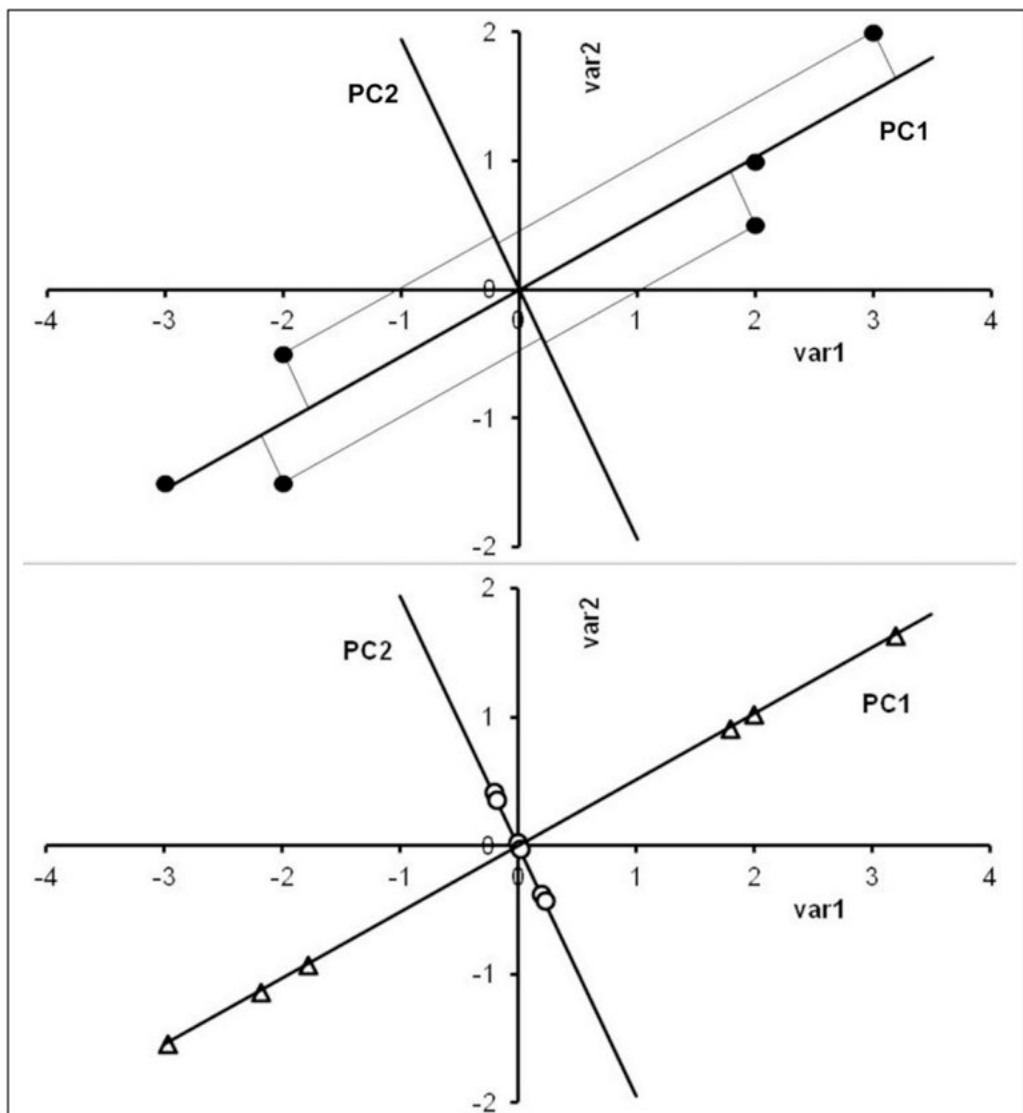


Figure 12.7.
Scatterplots of
the centered
data points of
Table 12.2 and
the axes PC1
and PC2

If we project all data points on the new axes, we obtain the coordinates of these points in the new coordinate system of PC1 - PC2 axes. These coordinates are called *scores* and form a set of data with number of variables equal to the original dataset. However, it is clear from Figure 12.8 that the PC1 axis describes very satisfactorily the data variation, since we observe that the coordinates on this axis, i.e. the values of variable PC1, form two distinct clusters, the same clusters shown in the original scatterplot of Figure 12.6. In contrast, the PC2 axis (in this example) contributes almost no information about the existence of clusters of data. Therefore, variable PC2 may be deleted without losing much information. Thus, the original two dimensional dataset of Table 12.2 is reduced to one-dimensional data, while retaining most of the information. Clearly, transforming a two-dimensional table to a one-dimensional dataset has no practical value. PCA is useful in cases when the data table contains many columns (variables).

Figure 12.8.
Definition
of PC scores
as the
coordinates of
the centered
data points on
the PC1 and
PC2 axes



A measure that shows how spread out the data are on a PC axis is the *eigenvalue* of this axis. PC axes with small eigenvalues, usually smaller than 1, provide almost no information about the existence of clusters of data and they are deleted. The eigenvalues are the variances of the principal components. Since each successive component captures less variance, the first eigenvalue is the largest and each successive eigenvalue is less than the preceding one. A plot of the decreasing eigenvalues is called a *scree plot*. Another interesting characteristic of PCs are *eigenvectors*, also referred to as *principal component loadings*. The loadings show which of the original variables are mostly represented in each component.

In statistical software packages the above process of reducing the dimensions of a table is performed in a strict mathematical way. With this method, a *score matrix* is produced, which contains the obtained principal components (PC1, PC2 ...) as variables. In this matrix we keep the factors that correspond to eigenvalues greater than 1 and delete the rest. This new table can substitute the original one without losing any substantial information regarding possible clusters, that is, uncorrelated data. Such clusters may be visually identified when we draw the scatterplot of PC1 vs. PC2, and in some cases the 3D plot of PC1 vs. PC2 vs. PC3. Note that for the effective application of PCA, the swarm of the data points should have a specific shape in some direction.

Principal components can be extracted from a covariance matrix or a correlation matrix. The covariance matrix should be used when the variables are measured in the same unit and have comparable variances. The correlation matrix should be used when the variables are measured in different units or have very different variances. The main drawback to using the correlation matrix is that the various tests of significance used for principal component analysis apply only when the covariance matrix is used, but we are usually more interested in visualizing the data than in testing the statistical significance of the principal components.

The number of principal components to use is not fixed. Many scholars select all components with eigenvalues greater than 1. Alternatively, we may look for a change in the slope of the scree plot.

Rotation changes the orientation of the principal components. It is used in *factor analysis* in order to make the components easier to interpret; however, in principal components analysis it is not an important step and results in the loss of information regarding the direction of the greatest amount of variance in the original data. There are various rotation methods that give somewhat different results.

FACTOR ANALYSIS

Principal component analysis and factor analysis are related methods for analysing the structure of multivariate data. While principal component analysis focuses on summarizing multivariate data, factor analysis assumes that the data are the product of unobserved (latent) factors and aims at discovering these latent factors through the correlation between variables.

PCA in R

There are several packages in R that have functions to perform PCA. Here, we will use the function `princomp()` that comes with the default *stats* package, which means that there is no need to install anything.

As an example, consider again the data given in **Table 12.1**. Given that cranial shape is largely determined genetically, we will compare the three groups/populations in respect to their cranial dimensions in order to draw conclusions concerning their biological affinity. Note that these data are saved in the tab delimited .txt file titled *cranialmeasurements.txt*.

To run PCA, we can use the commands:

```
x <- read.table("cranialmeasurements.txt", head=1,
sep="\t")
xn <- data.matrix(x[,2:5])
#from our dataset we select only the data in columns 2 to 5
pca <- princomp(xn, cor=T)
#cor=T suggests that we use a correlation matrix
summary(pca)
```

The output is shown in Figure 12.9 and gives the standard deviation of each principal component and the (cumulative) proportion of variance explained by them. We observe that the Cumulative Proportion for Comp. 1 is 0.365 and for Comp. 1 and Comp. 2 jointly it is 0.618. Thus, the first two PCs explain 61.8% of the original variance in the data. This is a rather small percentage and, therefore, any conclusions from the basic PCA plot, the PC1 versus PC2 plot, should be treated with caution since a lot of the variance in the dataset is not explained by PC1 and PC2.

The PCA score and loading matrices are obtained using the following commands and they are given in Figures 12.10 and 12.11:

Figure 12.9.
PCA output

```
> x <- read.table("cranialmeasurements.txt", head=1, sep="\t")
> xn <- data.matrix(x[,2:5]) #from our dataset we select only the data in columns 2 to 5
> pca <- princomp(xn, cor=T) #cor=T suggests that we use a correlation matrix
> summary(pca)
Importance of components:
              Comp.1      Comp.2      Comp.3      Comp.4
Standard deviation  1.2086875  1.0062713  0.9537409  0.7854112
Proportion of Variance 0.3652314  0.2531455  0.2274054  0.1542177
Cumulative Proportion 0.3652314  0.6183768  0.8457823  1.0000000
```

Figure 12.10.
Score matrix

```
> pca$sco
              Comp.1      Comp.2      Comp.3      Comp.4
[1,] -0.36253162  0.17904915 -0.74063925  1.030426810
[2,] -0.03951693 -0.40890785 -1.31738250 -0.134287350
[3,] -0.67076822  0.41837770  0.13405829 -0.722201340
[4,] -1.09025530  0.79494958 -2.36096811 -0.752674187
[5,] -2.06718106 -0.84665958  0.73453174  0.838147403
[6,]  0.22689681 -1.31795709  0.34619484  0.933013964
[7,] -0.74476241  0.94194318  1.64591592 -1.096390239
[8,] -1.27634477  1.08484648 -1.72733205  0.756339009
[9,] -1.46523207 -0.50864233  0.54534641 -1.368551086
[10,] -0.42907844  0.08211123  0.11864518 -0.268271506
[11,] -0.83127825 -0.42702870  0.85240095  0.693330374
[12,] -0.43918014  0.30971565 -0.89534983  0.234926416
[13,] -0.33436739  0.78750949 -0.87572862  1.397213191
[14,] -1.59557607  0.66486735  0.53294572 -1.384700404
[15,] -0.20945829  0.71152651  0.09080982 -0.006401658
[16,]  0.78530022  0.35129912  1.40524154  0.055124606
[17,] -0.49952529  1.88548778  0.89749577  0.670460274
[18,] -1.92191438 -0.82079710  1.32066811  1.094922335
[19,]  0.74829725  1.05104114 -0.07813491  0.168245177
[20,] -1.17415432 -1.63055597 -0.68262347 -0.140104225
[21,]  1.50354328  1.12772538  0.04210092 -0.447452524
[22,]  0.94031891 -1.96850740 -0.57668646 -0.313386702
[23,]  2.30346339 -0.90964616  0.47111211  0.527049046
[24,]  1.20951079  0.31218955 -0.30523765 -1.221267092
[25,]  0.35651982 -0.06553145 -0.20859884 -1.286283127
[26,]  1.87250567 -1.42712190  0.06688073 -0.591064557
[27,]  1.12825434 -0.84449846 -1.43893534  0.413044405
[28,]  0.01689176 -1.47628428  1.19602147  0.014522871
[29,]  2.21971610  1.90207445  0.96964210  0.832770256
[30,]  1.83990660  0.04742453 -0.16239460  0.073499860
```

```
> pca$1

Loadings:

               Comp.1 Comp.2 Comp.3 Comp.4
max_breadth      0.428      0.881  0.193
basi_bregmatic_height -0.665      0.168  0.725
basi_alveolar_length -0.608  0.222  0.420 -0.636
nasal_height      -0.971  0.139 -0.180

               Comp.1 Comp.2 Comp.3 Comp.4
SS loadings      1.00   1.00   1.00   1.00
Proportion Var    0.25   0.25   0.25   0.25
Cumulative Var    0.25   0.50   0.75   1.00
```

Figure 12.11.
Loading
matrix

In what concerns the basic PCA plots, the *biplot* and the *screeplot* are created using the commands:

```
biplot(pca)
screeplot(pca, type="lines")
```

We obtain the plots given in Figures 12.12 and 12.13. Figure 12.12 is the biplot, which shows how the samples are differentiated according to different variables. For example, individuals coded as 6, 20, 22, and 28 are differentiated from the others mostly with regard to nasal height. In addition, the biplot also provides information regarding the relationship between variables. In specific, the angle between each pair of axes is proportional to the correlation coefficient of the variables that correspond to these axes. In our example, we observe that there is high positive correlation between maximum breadth and basi-bregmatic height.

The screeplot in Figure 12.13 is a line plot of the eigenvalues of the principal components and it is used to determine the number of the most significant PCs, usually the PCs with values greater than 1. Note that the eigenvalues are the variances that correspond to the standard deviations presented in Figure 12.9.

Therefore, they can be obtained using the command:

```
pca$sdev^2
```

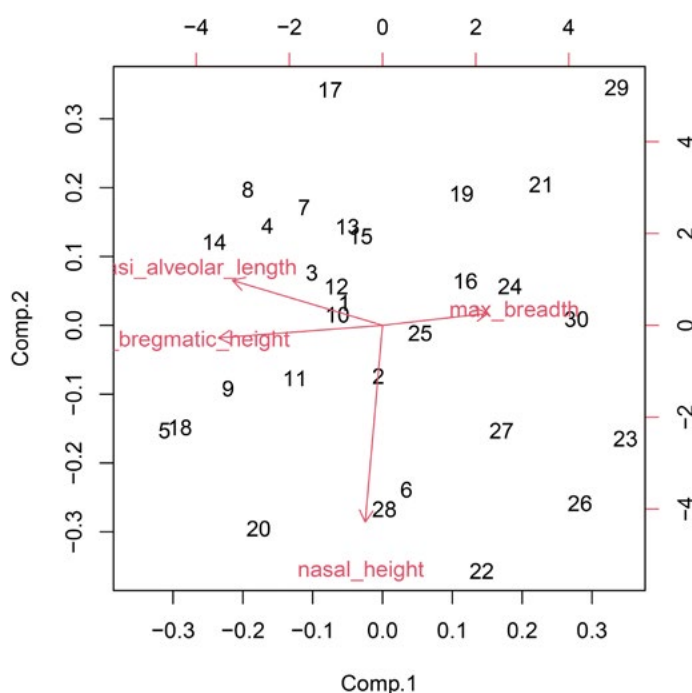
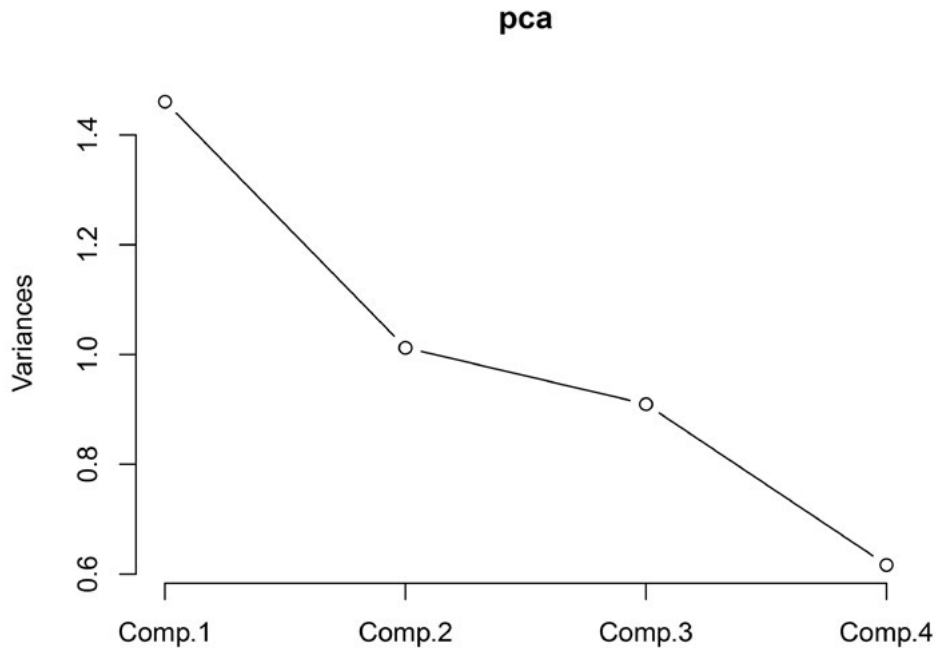


Figure 12.12.
PCA biplot

Figure 12.13.
Screeplot



Finally, the PC1 vs. PC2 plot may be created using the `ggplot()` function:

```
library(ggplot2)

p <- ggplot(data.frame(pca$sco),
  aes(x=pca$sco[,1], y=pca$sco[,2],
  color=factor(x[,1]))))

p <- p + geom_point()

p <- p + stat_ellipse(aes(x=pca$sco[,1],
  y=pca$sco[,2], color=factor(x[,1]), group=
  factor(x[,1])), type = "norm", level=0.95)

p <- p + xlab("PCA1") + ylab("PCA2")

p <- p + theme_classic() +
  theme(legend.title=element_blank())

p
```

The plot is given in [Figure 12.14](#) and shows that Group 3 is partly differentiated from the other two, whereas Groups 1 and 2 largely overlap.

Note that if we want to rotate the principal components during PCA, instead of the `princomp()` function, we use the `principal()` function in the *psych* package.

In this case, we install and load the *psych* package and use the command:

```
pca <- principal(xn, nfactors=2,
  rotate="varimax", scores=TRUE)
```

where *nfactors* is an integer showing the number of principal components extracted, the argument *rotate* can take the values "none", "varimax", "quatimax", "promax", "oblimin", "simplimax", or "cluster", which are possible rotation techniques.

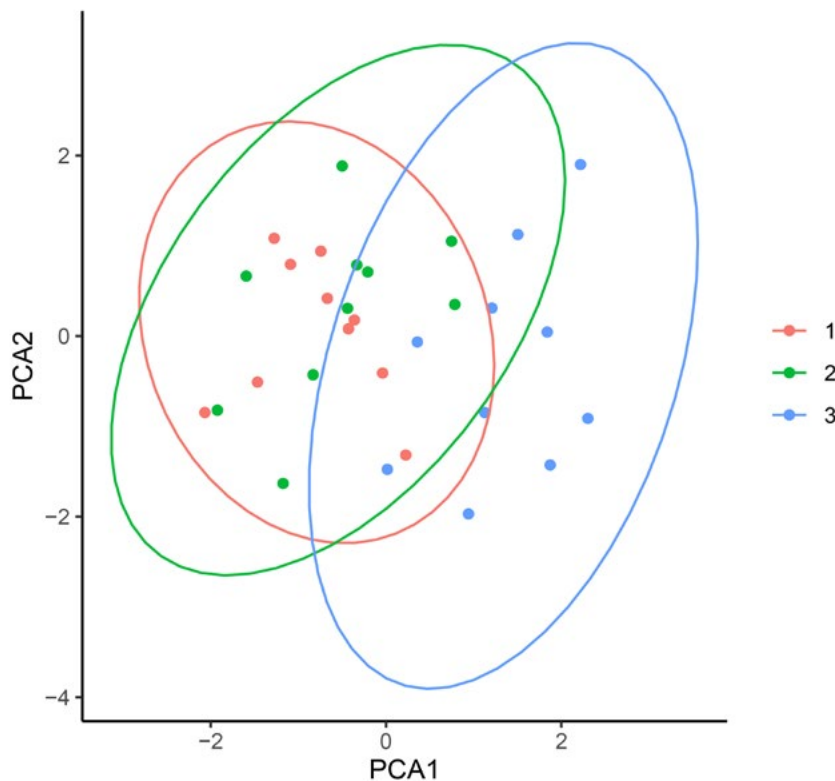


Figure 12.14.
PC1 vs PC2
plot

Subsequently, we can obtain the scores and loadings using the commands:

```
pca$sco
pca$li
```

We can also draw the plot PC1 vs PC2 using the same commands as above for obtaining Figure 12.13.

12.3. METRIC AND NON-METRIC MULTIDIMENSIONAL SCALING

Multidimensional scaling (MDS) comprises a group of techniques that create a map visualizing the relative position of a number of objects based on the matrix of distances between them. The map may consist of one, two, or three dimensions. There are two broad categories in MDS: *Metric* or *Classical Multidimensional Scaling* and *Non-Metric Multidimensional Scaling*.

The goal of metric MDS is to produce a configuration of points, i.e. a map, for which the Euclidean distances between the points are as close as possible to the distances used to produce the map. Metric MDS is also known as *Principal Coordinates Analysis*. The objective of non-metric MDS is to find a map that reproduces not the input distances but the ranks of these distances. The distances themselves are not reproduced.

In general, the input data to MDS is a *proximity* matrix, i.e. a symmetric square matrix, the values of which quantify how “close” or “distant” two objects are. Therefore, its values may be either *dissimilarity* or *similarity* measures. The dissimilarity matrix is in fact a distance matrix. The dissimilarity is a measure of distinction; the greater the dissimilarity of two objects, the greater the value of this measure. Therefore, the diagonal elements of a *dissimilarity* matrix are equal to zero, since the distinction between an object and itself is zero. The *similarity* is the opposite concept to the dissimilarity; the more two objects resemble one another, the larger the similarity of these objects is. For example, the correlation matrix may often be considered as a similarity matrix. Note that in this case the diagonal elements of the similarity matrix are equal to one.

MDS is similar to other data reduction techniques, such as PCA. Thus, MDS attempts to reduce the dimensions (columns) of the proximity matrix, while preserving most of the information about inter-point distances. The optimum number of dimensions in the MDS model is selected by means of the *stress function*, i.e. a function that measures the difference between the actual distances and their predicted values or between the ranks of actual distances and their predicted values.

R has a number of packages to perform metric and non-metric MDS. The following list shows several functions to perform metric MDS with the corresponding packages in parentheses:

- `cmdscale()` (stats)
- `mds()` (smacof)
- `wcmdscale()` (vegan)
- `pco()` (ecodist)
- `pco()` (labdsv)
- `pcoa()` (ape)
- `dudi.pco()` (ade4)

For non-metric MDS, we may use:

- `isoMDS()` (MASS)
- `mds()` (smacof)
- `metaMDS()` (vegan)
- `nmds()` (ecodist)

All these functions require a distance matrix as the main argument to the function. Therefore, we may input the data in R in dissimilarity matrix format. Alternatively, we should calculate the distance matrix using R via the function `dist()`.

Assume that we have estimated the percentage frequency of different animals in six archaeological sites and the data is given in **Table 12.3**. The aim is to construct the map that visualizes the differentiation of the six sites based on the percentage frequency of domesticated animals using Manhattan distances (see distances in section 12.4).

Table 12.3.
Percentage
frequency
of different
domesticated
animals in six
archaeological
sites

Site	Sheep	Pig	Cow	Goat
1	22.3	15.7	34.9	27.1
2	15.2	16.8	20.3	47.7
3	33.3	34.1	27.2	5.4
4	12.4	17.6	29.9	40.1
5	38.5	26.1	19.2	16.2
6	25	24.2	23.9	26.9

The data of **Table 12.3** is in the file *MDS.txt* in the *R_Files* folder.

We input these data to R and then we compute the distance matrix using Manhattan distances:

```
MDSData <- read.table("MDS.txt", head=1,
sep="\t")
MDS.dist <- dist(MDSData[,2:5], method=
"manhattan") #we calculate the Manhattan distance
print(MDS.dist, digits=3)
#we print the distance matrix in the R Console
```

We obtain the output of Figure 12.15:

```
> print(MDS.dist, digits=3)
      1      2      3      4      5
2 43.4
3 58.8 84.6
4 29.8 20.8 74.8
5 53.2 65.2 32.0 69.2
6 22.4 41.6 43.0 38.4 30.8
```

Figure 12.15.
Manhattan
distance
matrix based
on animal
frequencies

Now we can run classical Multidimensional scaling using the function `cmdscale()` and plot the first two coordinates:

```
g <- cmdscale(MDS.dist)
g # view results (the first two coordinates)
plot(g[,1], g[,2], xlab="Coordinate 1",
     ylab="Coordinate 2")
text(g[,1], g[,2], pos=4,
     labels=rownames(MDSData), cex=1)
```

Note that MDS reduces the dimensions (columns) of the proximity matrix. By default, the `cmdscale()` function reduces the dimensions to two. If we want to increase the dimensions, say to three, this is done via the `k` argument of the function.

Thus, we could use:

```
g <- cmdscale(MDS.dist, k=3)
```

The results are shown in Figures 12.16 and 12.17. Figure 12.16 shows the first two coordinates obtained from the MDS, and Figure 12.17 is the corresponding MDS plot, which shows the map of the animal sites distribution using Manhattan distances. It is seen that the distribution of sites is rather random.

To examine whether the correlation between the original distances and the distances computed from the coordinates that `cmdscale()` estimated is satisfactory, we can use the commands:

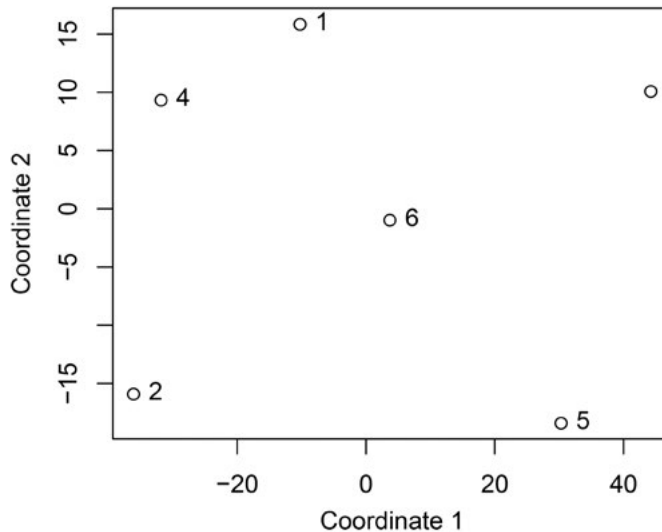
```
MDS.pco <- cmdscale(MDS.dist)
cor(MDS.dist, dist(MDS.pco))
```

We obtain the value 0.991, which shows a very high correlation.

```
> g <- cmdscale(MDS.dist)
> g # view results (the first two coordinates)
      [,1]      [,2]
[1,] -10.26137  15.8503217
[2,] -36.10497 -15.9023407
[3,]  44.21618  10.0862601
[4,] -31.83020   9.3319326
[5,]  30.27657 -18.3916749
[6,]   3.70378 -0.9744988
```

Figure 12.16.
MDS output

Figure 12.17.
MDS plot



We can repeat the analysis using Kruskal's non-metric multidimensional scaling using the `isoMDS()` function in the *MASS* package. Non-metric scaling involves iteration from a starting configuration to minimize stress, a measure of the deviation of the distances computed from the estimated points from the observed distances. The `isoMDS()` function uses the results of `cmdscale()` as the starting configuration.

To apply `isoMDS()`, we use the commands:

```
library(MASS)
gn <- isoMDS(MDS.dist)
gn # view results
plot(gn$points[,1], gn$points[,2],
     xlab="Coordinate 1", ylab="Coordinate 2")
text(gn$points[,1], gn$points[,2],
     pos=4, labels=rownames(MDSData), cex=1)
```

The results obtained are shown in Figures 12.18 and 12.19 and they are practically the same as those obtained from metric MDS.

Figure 12.18.
Non-metric
MDS output

```
> library(MASS)
> gn <- isoMDS(MDS.dist)
initial value 1.554871
iter 5 value 0.021524
iter 5 value 0.007498
iter 5 value 0.002074
final value 0.002074
converged
> gn # view results
$points
      [,1]      [,2]
[1,] -9.809687 18.341453
[2,] -35.695424 -14.336794
[3,] 43.946781 10.016545
[4,] -32.911622 6.857177
[5,] 30.294960 -18.418612
[6,] 4.174992 -2.459768

$stress
[1] 0.002074302
```

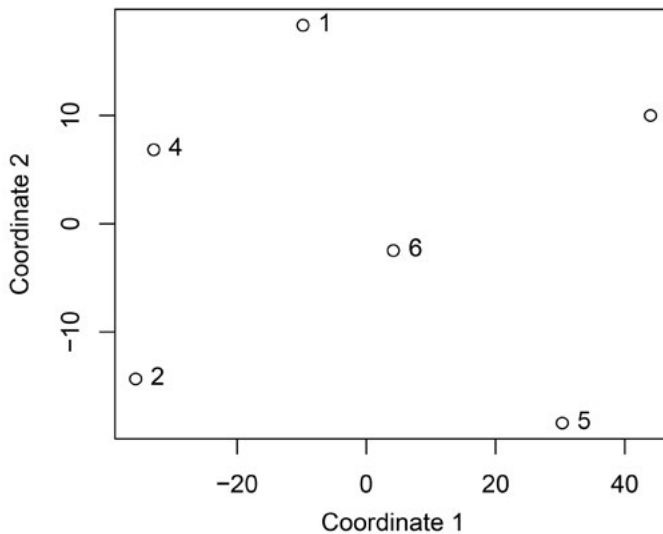


Figure 12.19.
Non-metric
MDS plot

As a second example, consider the squared Mahalanobis distances between all pairs of the four groups presented in Figure 7.28. These distances are given in tabular format in Figure 12.20. In this case MDS can be used to detect clusters among these groups.

g1	g2	g3	g4
0	0.631	8.056	8.466
0.631	0	8.082	6.43
8.056	8.082	0	12.071
8.466	6.43	12.071	0

Figure 12.20.
Squared
Mahalanobis
distances in
a distance
matrix format

Note that the initial dataset is a distance matrix, so there is no need to recalculate distances. Therefore, to perform metric MDS, we can use the commands:

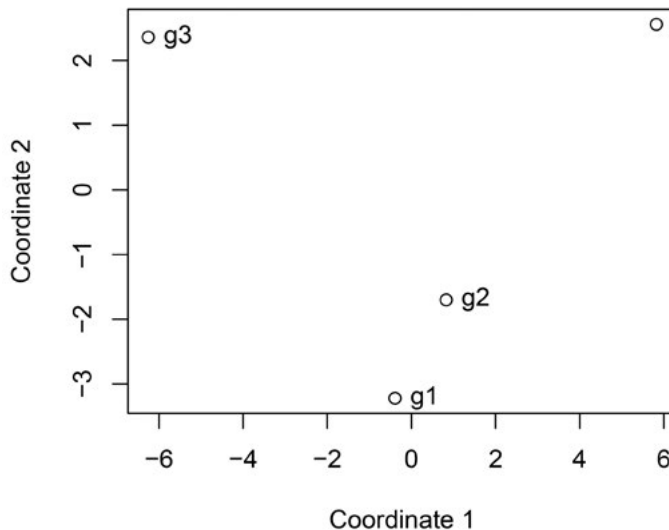
```
d <- read.table("MD.txt", head=1, sep="\t")
g <- cmdscale(as.dist(d))
g
plot(g[,1], g[,2], xlab="Coordinate 1",
      ylab="Coordinate 2")
text(g[,1], g[,2], pos=4, labels=colnames(d),
      cex=1)
```

The obtained results are given in Figures 12.21 and 12.22. It is interesting to point out that the MDS plot shows that groups 1 and 2 tend to form a cluster, whereas 3 and 4 stand apart. This result remains the same if we repeat the treatment using non-metric MDS.

```
> d <- read.table("MD.txt", head=1, sep="\t")
> g <- cmdscale(as.dist(d))
> g
      [,1]      [,2]
g1 -0.3923537 -3.219511
g2  0.8247154 -1.697233
g3 -6.2517424  2.359007
g4  5.8193807  2.557737
```

Figure 12.21.
Metric MDS
output for the
distances of
Figure 12.20

Figure 12.22.
Metric MDS
plot for the
distances of
Figure 12.20



12.4. CLUSTER ANALYSIS

Cluster analysis includes techniques for grouping cases based on one or more variables. The methods most used in archaeology are k-means cluster analysis and hierarchical cluster analysis.

k-means Cluster Analysis

In *k-means cluster analysis* all cases are examined simultaneously. This method is used when we want to identify k clusters in the dataset under examination and it presupposes that we know the number of clusters (k). A simple way to do that may use the following steps:

1. k initial cases are randomly selected and k clusters are created based on the distances of all cases from the k selected cases. In particular, if the distance of a certain case from the j^{th} initial case is minimum, then this case is attached to the j^{th} cluster.
2. The centroid of each of the k clusters is calculated, the distances of all cases from the centroids are recalculated, and their position to the clusters is reconsidered based on these distances, as in step 1.
3. Step 2 is repeated until convergence has been reached.

These steps are schematically shown in Figure 12.23 when $k = 2$. Thus, if we have the points 1 to 5, we select randomly two of them, say point 1 and 3. The distances of the remaining points are closer to point 3 than to point 1. Therefore, we start by forming two clusters, one is the cluster of point 1 alone and the other includes the points 2, 3, 4, 5. Now we calculate the centroids of these clusters, shown in Figure 12.23 by red crosses. We observe that points 2, 3 are closer to the centroid at point 1, whereas 4, 5 are closer to the other centroid. Thus, in the next step we form two new clusters, one including points 1, 2, 3 and the other points 4 and 5. At this step the process stops because convergence has been reached.

In R k-means clustering can be easily performed using the `kmeans()` function. As an example, we will apply this function to the data given in the file `cranmeasurements_nogroup.txt`.

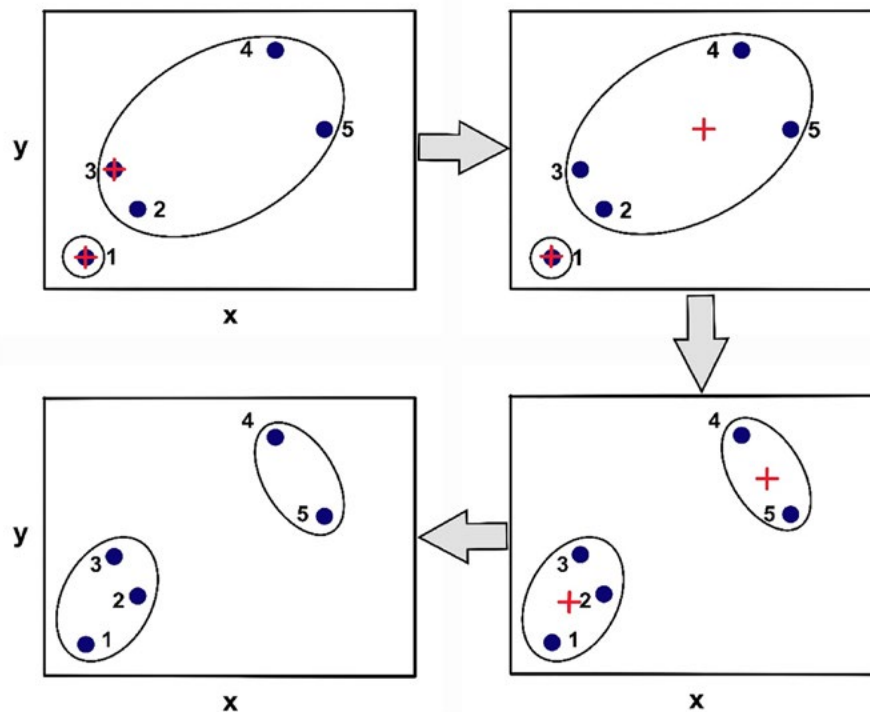


Figure 12.23.
Schematic
presentation
of k-means
cluster
analysis when
 $k = 2$

The commands we should use are the following:

```
data <- read.table("cranmeasurements_nogroup.txt", head=1, sep="\t")
ca <- kmeans(data, 3) #apply k-means clustering
ca$cluster # obtain the cluster to which each cranium belongs
ca$centers # obtain the centroid per cluster
```

We obtain the results of Figure 12.24. Note that when we run the R commands repeatedly, we may obtain different results. This is because in the first step of the analysis, the `kmeans()` function selects the initial cases in a random way.

In this Figure, the top row of the integers shows the categorization of cases based on the `kmeans()` function, whereas the table below gives the predicted centroids per cluster. It should be noted that since the `kmeans()` function does not know how the groups are represented in the dataset, it uses integers 1, 2, 3, ... without committing to any order.

```
> data <- read.table("cranmeasurements_nogroup.txt", head=1, sep="\t")
> ca <- kmeans(data, 3) #apply k-means clustering
> ca$cluster #we obtain the cluster to which each cranium belongs
[1] 3 3 2 3 2 1 2 3 2 2 2 3 3 2 2 1 2 2 1 3 1 1 1 1 1 3 2 1 1
> ca$centers #we obtain the centroid per cluster
```

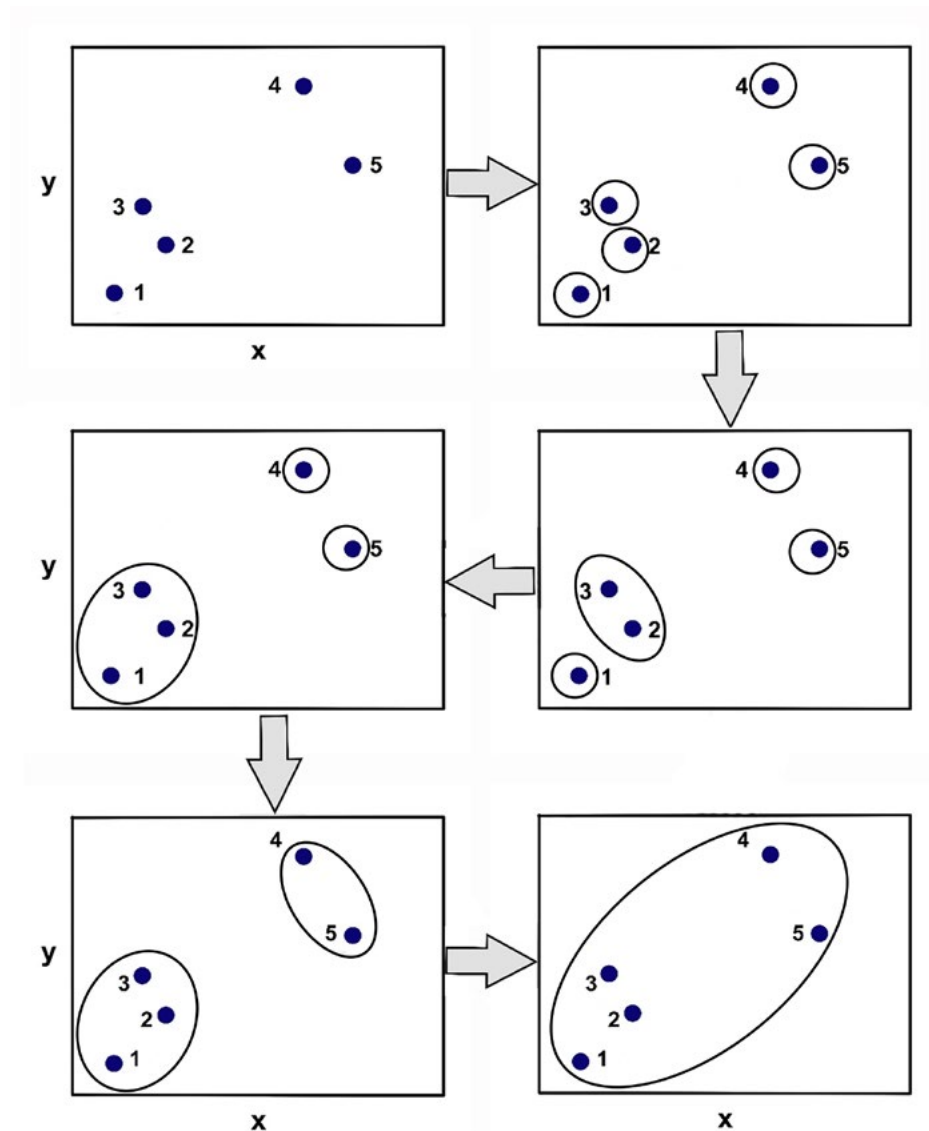
	max_breadth	basi_bregmatic_height	basi_alveolar_length	nasal_height
1	138.0909	128.6364	90.36364	51.09091
2	135.1818	136.9091	99.81818	51.54545
3	126.0000	135.2500	91.37500	49.87500

Figure 12.24.
Cluster
Analysis
output

Hierarchical Cluster Analysis

Hierarchical Cluster Analysis (HCA) separates cases into clusters with similar properties. The clusters are formed by adding to the cluster one case at a time. The main outcome is a *dendrogram*, which is used to visualize how clusters are formed.

Figure 12.25.
Steps followed
in clustering



The steps we follow in order to construct a dendrogram are schematically presented in [Figure 12.25](#). First, we assume that each sample is a separate cluster. Then, we form an initial cluster between samples 2 and 3, which have the smallest distance. Subsequently, we observe that the second smallest distance is between samples 1 and 2. Since sample 2 already belongs to the first cluster, we connect samples 1, 2 and 3 in a broader cluster. The next smallest distance is between samples 1 and 3 but these already belong to the same cluster. Thus, we proceed and form a cluster for samples 4 and 5. Finally, we group all samples in a single cluster.

The steps shown in [Figure 12.25](#) can also be seen in the dendrogram of [Figure 12.26](#). We observe again that samples 2 and 3 are the first to be grouped together. This cluster then includes sample 1. This broad cluster is differentiated from another cluster that contains samples 4 and 5. The vertical

axis of the dendrogram may depict the distance d_{ij} between samples i and j , but it may also express the similarity between these samples, defined as:

$$(\text{similarity})_{ij} = 100(1 - d_{ij}/d_{\max})$$

where $d_{\max}$ is the maximum distance (that is, the maximum d_{ij}).

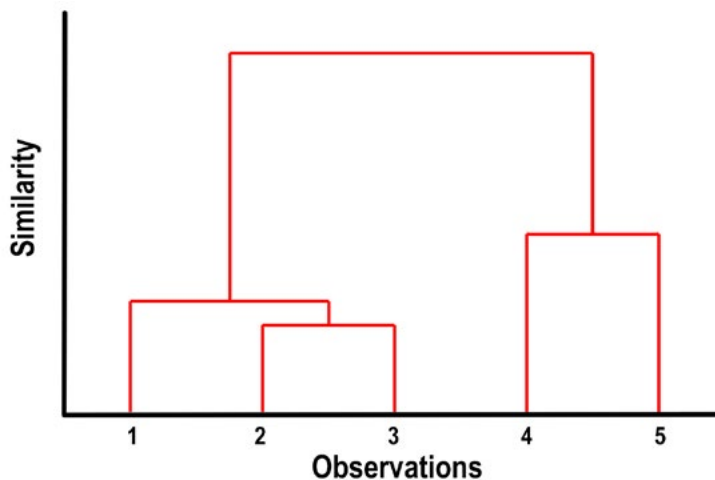


Figure 12.26.
Dendrogram
depicting the
clusters of
Figure 12.25

As an example, we will examine again the biological affinity between the three populations of **Table 12.1** but this time by means of HCA. In R we can use the `hclust(d, method)` function, where d is a distance matrix and *method* is the agglomeration method to be used.

In specific, we can use the commands:

```
x <- read.table("cranialmeasurements.txt", head=1, sep="\t")
y <- data.matrix(x[,2:5])
# we select the data in columns 2 to 5

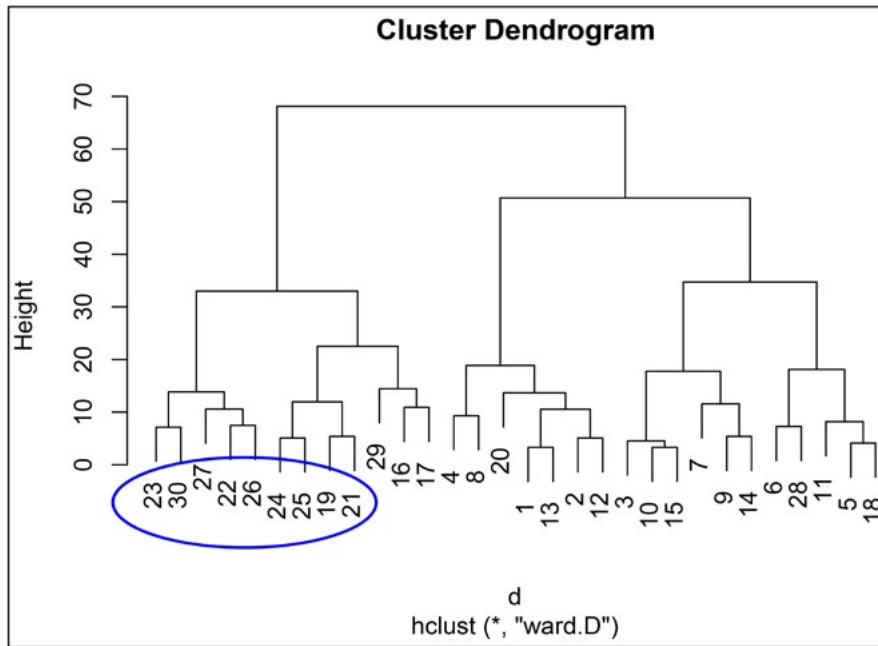
d <- dist(y, method="euclidean")
# we calculate the Euclidean distance between cases

fit <- hclust(d, method="ward.D")
# we perform HCA using the Ward method

plot(fit)
# we plot a dendrogram
```

In the dendrogram of [Figure 12.27](#) we observe that each branch has a number, which reflects its row in the dataset. Remember that crania belonging to the first group were in the first 10 rows, crania belonging to the second group were arranged in rows 11 to 20, while crania in the third group were in rows 21 to 30. It can be seen that there is no clear division among the three Cypriot groups, that is, we do not see the individuals from each of the three groups clustering together; however, most of the individuals of group 3 do form a cluster, which is in agreement to the PCA results.

Figure 12.27.
Dendrogram
based on
the cranial
dimensions of
three Cypriot
populations



DISTANCES

In the above example we used the Euclidean distance. However, there are many different distances that we could have used instead, such as Manhattan, Gower, Bray, Canberra, Mahalanobis and others. If x_{ij} are the elements of the original data table, some of the distances between these elements d_{jk} are estimated as follows:

$$\text{Euclidean} \quad d_{jk} = \sqrt{\sum_i (x_{ij} - x_{ik})^2}$$

$$\text{Manhattan} \quad d_{jk} = \sum_i |x_{ij} - x_{ik}|$$

$$\text{Gower} \quad d_{jk} = \frac{1}{M} \sum_i \frac{|x_{ij} - x_{ik}|}{R} \quad , \quad R = \max(x_i) - \min(x_i), \quad M = \text{number of cases}$$

$$\text{Canberra} \quad d_{jk} = \sum_i \frac{|x_{ij} - x_{ik}|}{|x_{ij}| + |x_{ik}|}$$

$$\text{Bray} \quad d_{jk} = \frac{\sum_i |x_{ij} - x_{ik}|}{\sum_i (x_{ij} + x_{ik})}$$

R has many different functions for the estimation of distances. Some of these functions are:

- `dist()` in the *basic* package
- `vegdist()` in the *vegan* package
- `distance()` in the *ecodist* package
- `Dist()` in the *amap* package

`dist()` estimates the euclidean, maximum, manhattan, canberra, binary, and minkowski distances. Its syntax is `dist(x, method="euclidean")`, where `x` is a numeric matrix or a dataframe. If we want to estimate the minkowski distance, we also need to define the power of this distance, usually $p = 2$.

`vegdist()` estimates the manhattan, euclidean, canberra, bray, kulczynski, jaccard, gower, alt-Gower, morisita, horn, mountford, raup, binomial, and chao distances. Its syntax is `vegdist(x, method="euclidean")`.

`distance()` estimates the manhattan, euclidean, mahalanobis, bray, and jaccard distances. Its syntax is `distance(x, method="euclidean")`.

`Dist()` estimates the euclidean, maximum, manhattan, canberra, binary, pearson, correlation, spearman, and kendall distances. Its syntax is `Dist(x, method="euclidean")`.

How to combine observations into clusters and clusters into bigger clusters

(from Carlson 2017¹³)

There are seven choices in `hclust()`:

1. **Ward's method** creates clusters by minimizing the within cluster sum of squares. There are two approaches in `hclust()`: `method="ward.D"` and `method="ward.D2"`, depending on whether the distance adopted is the Euclidean or squared Euclidean.
2. **Single linkage**, `method="single"`, adds a case to a cluster if the distance from the case to the nearest member of the cluster is smaller than the distance to the nearest member of any other cluster.
3. **Complete linkage**, `method="complete"`, adds a case to a cluster if the distance from the case to the farthest member of the cluster is smaller than the distance to the farthest member of any other cluster. This is the default method used in `hclust()`.
4. **Average linkage**, `method="average"`, combines clusters if the average of the distances between the members of that cluster is smaller than for any other cluster (also called unweighted pair group method with arithmetic mean [UPGMA]).
5. **McQuitty's method**, `method="mcquitty"`, is similar to average linkage but now the distance from a case to a cluster is the weighted average of the distances to the clusters that make up the cluster (also called weighted pair group method with arithmetic mean [WPGMA]).
6. **Centroid**, `method="centroid"`, adds cases to a cluster if the distance to the centroid of the cluster is smaller than for any other cluster.
7. **Median**, `method="median"`, is similar to the centroid method, but the centroid of the joined clusters is halfway between the centroids of the joined clusters.

¹³ Carlson DL. 2017. *Quantitative Methods in Archaeology using R*. Cambridge University Press

12.5. LINEAR DISCRIMINANT ANALYSIS

Linear Discriminant Analysis (LDA) is used when we want to explore how effectively two or more samples are discriminated, and it subsequently allows us to classify one or more unknown cases in one out of several predetermined samples (categories). In other words, Discriminant Analysis serves both a descriptive and a predictive function. As mentioned above, a basic precondition is that samples follow the multivariate distribution.

To apply LDA, we find the centroid of each cluster and then the Mahalanobis distance of each case from these centroids. Each case is allocated to the group from which it exhibits the smallest distance. The same is valid for new (unknown) cases. If the data follows the multivariate normal distribution, we may estimate the probability of a certain case to belong to a certain group from:

$$P = ce^{-d_{MD}^2/2}$$

where d_{MD} is the Mahalanobis distance between the case and the centroid of the group and c is a constant determined from the condition that the sum of all P values should be equal to 1. For cross-validation, the *leave-one-out classification* or *Jack-knife technique* is usually adopted.

LDA is a parametric multivariate technique and, therefore, it is subject to several assumptions: The size of the smallest group should exceed the number of explanatory variables, the variables should follow the multivariate normal distribution, the variance/covariance matrices of variables should be homogenous across groups, whereas multi-collinearity among variables should be excluded. From these assumptions, the most important is the first; there should exist at least five more observations in each group than the number of variables. For the other assumptions, Tabachnick and Fidell (2013)¹⁴ point out that if classification is the primary goal, these requirements do not affect the classification performance, provided that there are no outliers.

In R LDA can be implemented using the `lda()` function of the MASS package. The details of the application of this function are given in the following example:

Consider again the dataset of Table 12.1. Apply LDA in order to examine how effectively the samples of this dataset are discriminated and subsequently confirm the conclusions drawn from PCA. In addition, examine whether the crania of Table 12.4 can be effectively discriminated.

Table 12.4.
Measurements
from individuals
of unknown
Cypriot group
affiliation

Maximum breadth	Basi-bregmatic height	Basi-alveolar length	Nasal height
126	129	109	51
132	136	100	50
128	126	91	57
130	125	85	55

¹⁴ Tabachnick BG, Fidell LS. 2013. *Using Multivariate Statistics*, 6th Edition. Pearson

First, we examine whether the groups of **Table 12.1** can be satisfactorily discriminated using LDA. For this purpose, we can use the following commands:

```
library(MASS)

xdata <- read.table("cranialmeasurements.txt",
  head=1, sep="\t")

fit <- lda(group ~ ., data=xdata,
  na.action="na.omit", CV=TRUE)

xdata$group

fit$class

round(fit$posterior,3)

ct <- table(xdata$group, fit$class)

diag(prop.table(ct, 1))

sum(diag(prop.table(ct)))
```

In the above commands, the argument *na.action = "na.omit"* uses a listwise deletion of missing data, the argument *CV=TRUE* generates jackknifed (i.e., leave-one-out) predictions, and the formula *group ~ .* specifies that the discriminators are all variables located to the right of the grouping variable. If we wanted to reduce the discriminators to say *max_breadth* and *nasal_height*, we would have to use the formula *group ~ max_breadth + nasal_height*. The command *xdata\$group* gives the original categories (groups) of the cases, and the command *fit\$class* gives the categories predicted by LDA. The command *round(fit\$posterior,3)* presents the probabilities that the cases are distributed among the different categories, while the *diag(prop.table(ct, 1))* and *sum(diag(prop.table(ct)))* commands calculate the percentage of correctly classified cases of each category and the total percentage of fully separated cases, respectively.

The above commands result in the table of results of **Figure 12.28**. At the end of this table we note that only 56.67% of the cases are correctly classified. Note that this percentage is the jackknifed (i.e., leave-one-out classification) prediction. Moreover, from the output of the command *diag(prop.table(ct, 1))* we observe that the correctly classified cases are 50%, 30% and 90% for group 1, group 2 and group 3, respectively. That is, only the last group is classified satisfactorily. This is also shown if we compare the initial classification (values of *xdata\$group* variable) to the predicted classification (values of *fit\$class* variable) as well as the probabilities of a case to be classified into one of the three groups presented below the command *round(fit\$posterior,3)*.

```
> library(MASS)
> xdata<-read.table("cranialmeasurements.txt",head=1,sep="\t")
> fit <- lda(group ~ ., data=xdata, na.action="na.omit", CV=TRUE)
> xdata$group
[1] 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3
> fit$class
[1] 2 1 1 1 1 2 3 2 2 1 1 2 1 1 1 2 3 2 1 3 1 3 3 3 3 3 3 2 3 3
Levels: 1 2 3
> round(fit$posterior,3)
      1      2      3
1  0.381 0.613 0.006
2  0.496 0.281 0.223
3  0.543 0.443 0.013
4  0.789 0.210 0.001
5  0.201 0.799 0.000
6  0.071 0.363 0.565
.....
28 0.312 0.469 0.219
29 0.026 0.379 0.594
30 0.003 0.004 0.993
> ct <- table(xdata$group, fit$class)
> diag(prop.table(ct, 1))
      1      2      3
0.5 0.3 0.9
> sum(diag(prop.table(ct)))
[1] 0.566667
```

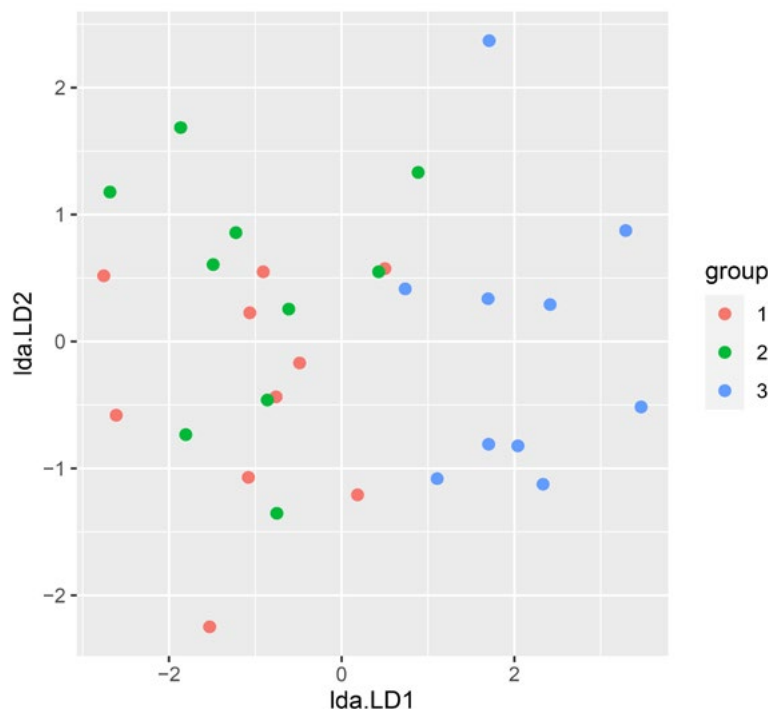
Figure 12.28.
Part of the
LDA output

If we want to construct the LDA plot, we need to re-run the `lda()` function using `CV=FALSE` and use the following commands:

```
fit <- lda(group ~ ., data=xdata, na.action="na.omit",
CV=FALSE)
xp <- predict(fit)
library(ggplot2)
newdata <- data.frame(group=factor(xdata[,1]),
lda=xp$x)
ggplot(newdata) +
geom_point(aes(lda.LD1, lda.LD2, colour=group),
size=2)
```

It is seen that the LDA plot (Figure 12.29) is very similar to the PCA plot (Figure 12.14) and shows that group 3 is satisfactorily differentiated from the other two, whereas groups 1 and 2 largely overlap.

Figure 12.29.
Scatter plot
using the
first two
discriminant
dimensions of
LDA



Now, in order to predict the group to which the crania of **Table 12.4** belong, we place the values of the four crania of unknown origin at the bottom of the dataset of Cypriot crania and arbitrarily assign them to group 3.

We save the new dataset as *LDAcranialmeasurements.txt* and use the commands:

```
pdata <- read.table("LDAcranialmeasurements.txt",
head=1, sep="\t")
train <- c(1:30)
fitp <- lda(group ~ ., data=xdata, subset=train,
na.action="na.omit", CV=FALSE)
predict(fitp, pdata[-train, ])$class
round(predict(fitp, pdata[-train, ])$posterior,3)
```

In these commands, the *subset = train* argument, where *train=c(1:30)*, indicates that only the first 30 rows of the input data table will be used to discriminate the three data groups. The command *predict(fitp, pdata[-train,])\$class* returns the group to which the unknown cases belong. Finally, the *round(predict(fitp, pdata[-train,])\$posterior,3)* command gives with accuracy of three decimals the probability that each unknown skull belongs to one of the three groups. The obtained results are shown in the table of Figure 12.30.

```
> pdata <- read.table("LDAcranialmeasurements.txt", head=1, sep="\t")
> train <- c(1:30)
> fitp <- lda(group ~ ., data=xdata, subset=train, na.action="na.omit", CV=FALSE)
> predict(fitp, pdata[-train,])$class
[1] 1 1 3 3
Levels: 1 2 3
> round(predict(fitp, pdata[-train,])$posterior,3)
      1      2      3
31 0.834 0.164 0.002
32 0.532 0.466 0.001
33 0.010 0.003 0.987
34 0.001 0.000 0.999
```

Figure 12.30.
Discrimination
results for
the unknown
crania

We observe that the first two unknown skulls belong to group 1 with probabilities 83.4% and 53.2%, respectively. However, taking into account that the probability of correct classification for group 1 is merely 50%, the above prediction cannot be considered accurate. In contrast, the prediction for the last two skulls is that they belong to Group 3 with probabilities 98.7% and 99.9%, respectively. Since the probability of correct classification for group 3 is 90%, we can be confident that these individuals belong to this group.

12.6. MULTIVARIATE ANALYSIS OF VARIANCE

Multivariate Analysis of Variance (MANOVA) is similar to ANOVA but now there are multiple dependent variables and one or several independent variables. Like ANOVA, MANOVA is a parametric test based on a test statistic. In fact, many test statistics have been proposed, and subsequently, different tests, the most powerful of which are *Pillai's Trace* and *Wilks' Lambda*. In the case where the null hypothesis is rejected and we want to identify the samples between which significant differences are traced, we may run MANOVA between pairs of groups and adjust the p-values using the Holm-Bonferroni correction.

MANOVA, as a parametric test, presupposes that the variables follow the multinomial normal distribution and there is homogeneity of variance and equality of the covariance matrices of the dependent variables across groups. If these assumptions are not fulfilled, it is simpler to assess the distance among the various groups through their Mahalanobis distances. The same procedure may also be followed in the case where the null hypothesis is rejected and we want to identify the samples between which significant differences are traced. However, this interesting approach is beyond the current guide, which offers only a basic introduction to multivariate statistics.

To examine an application of MANOVA in R, consider again the dataset of Table 12.1, which shows the variation in cranial shape among Cypriot groups 1, 2 and 3. Using PCA and HCA, we have seen that group 3 is differentiated from the other two, whereas groups 1 and 2 largely overlap.

To apply MANOVA, we should first check if the key MANOVA assumptions are met. In section 12.1 we tested the multivariate normality assumption and found that it is not violated.

Now, we use the `leveneTest()` function from the *car* package to test the homogeneity of variances assumption:

```
library(car)

x <- read.table("cranialmeasurements.txt", head=1,
sep="\t")

leveneTest(x$max_breadth, factor(x$group),
"mean")

leveneTest(x$basi_bregmatic_height,
factor(x$group), "mean")

leveneTest(x$basi_alveolar_length, factor(x$group),
"mean")

leveneTest(x$nasal_height, factor(x$group),
"mean")
```

We obtain the results presented in Figure 12.31, which show that, according to Levene's tests for the homogeneity of variance across groups, the corresponding null hypothesis cannot be rejected.

To test the equality of the covariance matrices of the dependent variables across groups, we may use *Box's M* test. In R this test may be performed by means of the function `boxM()` of the package *biotools*.

We install *biotools* and then we use the commands:

```
library(biotools)

boxM(x[,2:5], factor(x$group))
```

The results are given in Figure 12.32 and it is seen that the null hypothesis that the covariance matrices of the dependent variables across groups are equal cannot be rejected ($p\text{-value} = 0.9732 > 0.05$).

Figure 12.31.
Levene's test
results

```
> library(car)
> x<-read.table("cranialmeasurements.txt", head=1, sep="\t")
> leveneTest(x$max_breadth, factor(x$group), "mean")
Levene's Test for Homogeneity of Variance (center = "mean")
      Df F value Pr(>F)
group  2   0.153 0.8589
      27
> leveneTest(x$basi_bregmatic_height, factor(x$group), "mean")
Levene's Test for Homogeneity of Variance (center = "mean")
      Df F value Pr(>F)
group  2   0.7877 0.4651
      27
> leveneTest(x$basi_alveolar_length, factor(x$group), "mean")
Levene's Test for Homogeneity of Variance (center = "mean")
      Df F value Pr(>F)
group  2   0.735 0.4889
      27
> leveneTest(x$nasal_height, factor(x$group), "mean")
Levene's Test for Homogeneity of Variance (center = "mean")
      Df F value Pr(>F)
group  2   0.5067 0.6081
      27
```

```
> library(biotools)
> boxM(x[,2:5], factor(x$group))

Box's M-test for Homogeneity of Covariance Matrices

data:  x[, 2:5]
Chi-Sq (approx.) = 9.7057, df = 20, p-value = 0.9732
```

Figure 12.32.
Box's Test of
equality of
covariance
matrices
results

Therefore, all assumptions necessary to run MANOVA are valid. To run MANOVA in R, we use the `manova()` function:

```
fit <- manova(as.matrix(x[,2:5]) ~ factor(x$group))
summary(fit, test="Wilks")
```

The MANOVA output is given in Figure 12.33 and it is seen that $p\text{-value} = 5.16 \times 10^{-5} \ll 0.05$; thus, the overall difference among the three groups is statistically significant.

```
> fit <- manova(as.matrix(x[,2:5]) ~ factor(x$group))
> summary(fit, test="Wilks")
              Df    Wilks approx F num Df den Df    Pr(>F)
factor(x$group)  2 0.26983    5.5507     8    48 5.516e-05 ***
Residuals       27
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 12.33.
MANOVA
output for all
three groups

In order to assess between which pair(s) of groups the significant difference is traced, we may use the commands:

```
fit <- manova(as.matrix(x[,2:5]) ~ factor(x$group),
subset=factor(x$group) %in% c("1","2"))
summary(fit, test="Wilks")

fit <- manova(as.matrix(x[,2:5]) ~ factor(x$group),
subset=factor(x$group) %in% c("1","3"))
summary(fit, test="Wilks")

fit <- manova(as.matrix(x[,2:5]) ~ factor(x$group),
subset=factor(x$group) %in% c("2","3"))
summary(fit, test="Wilks")
```

In these commands the argument `subset=factor(mvdatag$group) %in% c("1","2")` restricts the application of MANOVA only to groups 1 and 2. By changing the values "1", "2" we can apply MANOVA between all possible pairs. The raw p-values obtained from the pairwise comparisons are shown in Figure 12.34 and their adjustment using the Holm-Bonferroni correction is given in Figure 12.35.

This adjustment employed the following commands:

```
p<-c(0.582, 0.0002, 0.0002995)
p.adjust(p, "holm")
```

It is seen that, as expected, the difference between groups 1 and 2 is not statistically significant (p -value = 0.582), whereas the difference between groups 2 and 3 as well as the difference between groups 1 and 3 is statistically significant (adjusted p -value = 0.0006 in both cases).

Figure 12.34.
MANOVA
output for
pairs of
groups

```
> fit <- manova(as.matrix(x[,2:5]) ~ factor(x$group), subset=factor(x$group) %in% c("1","2"))
> summary(fit, test="Wilks")
              Df    Wilks approx F num Df den Df Pr(>F)
factor(x$group) 1 0.83604  0.73545      4    15 0.582
Residuals      18
> fit <- manova(as.matrix(x[,2:5]) ~ factor(x$group), subset=factor(x$group) %in% c("1","3"))
> summary(fit, test="Wilks")
              Df    Wilks approx F num Df den Df    Pr(>F)
factor(x$group) 1 0.25001  11.249      4    15 0.0002022 ***
Residuals      18
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> fit <- manova(as.matrix(x[,2:5]) ~ factor(x$group), subset=factor(x$group) %in% c("2","3"))
> summary(fit, test="Wilks")
              Df    Wilks approx F num Df den Df    Pr(>F)
factor(x$group) 1 0.26401  10.454      4    15 0.0002995 ***
Residuals      18
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 12.35.
Raw and
corrected
 p -values for
pairwise
comparisons
in MANOVA
using Holm-
Bonferroni's
method

```
> p <- c(0.582, 0.0002, 0.0002995)
> p.adjust(p, "holm")
[1] 0.5820 0.0006 0.0006
```

SUGGESTED FURTHER READINGS

Baxter M. 2015. *Notes on Quantitative Archaeology and R*

https://www.academia.edu/12545743/Notes_on_Quantitative_Archaeology_and_R

Baxter M. 2016. *Multivariate Analysis of Archaeometric Data: An Introduction*

https://www.academia.edu/24456912/Multivariate_Analysis_of_Archaeometric_Data_An_Introduction

Baxter M, Cool H. 2016. *Basic Statistical Graphics for Archaeology with R: Life Beyond Excel*

https://www.academia.edu/29415587/Basic_Statistical_Graphics_for_Archaeology_with_R_Life_Beyond_Excel

Carlson DL. 2017. *Quantitative Methods in Archaeology Using R*. Cambridge University Press

Field A, Miles J, Field Z. 2012. *Discovering Statistics Using R*. Sage Publishing.

Paradis E. 2005. *R for Beginners*

http://cran.r-project.org/doc/contrib/Paradis-rdebuts_en.pdf

SUGGESTED ONLINE COURSES

Methods and Statistics in Social Sciences Specialization, University of Amsterdam (through Coursera)

<https://www.datacamp.com/>

PROMISE 



RESEARCH
& INNOVATION
FOUNDATION



The preparation of the manuscript was made possible through generous funding from the European Union Horizon 2020 (Promised, Grant Agreement No 811068) and the Research and Innovation Foundation (People in Motion, EXCELLENCE/1216/0023)

This work is distributed under a creative commons licence (CC BY-NC 2.0)



Published by The Cyprus Institute, Nicosia

ISBN 978-9963-2858-6-0

PROMISE



RESEARCH
& INNOVATION
FOUNDATION



Structural Funds
of the European Union in Cyprus

ISBN 978-9963-2858-6-0

