Stephan Haug and Aleksey Min

Statistics for Business Administration

With introduction to R

Summer term 2017

Preface

These lecture notes are based on books [2], [3] and [8].The theoretical part of the lecture notes is similar to the one of [2]. Examples with applications of R programming language were mainly taken from [8]. The part related to R is also based on teaching materials in *Statistics Training* (in German *Statistik-Praktikum*), which was offered in summer semesters 2008 and 2009.

The initial version of the lecture notes was created in the summer semester 2012. All parts of it are included in the version you are reading now. However, a few topics are no longer addressed within the current accompanying course. These topics are marked by * in the table of contents.

The current version may still contain some small errors. We apologize for any inconvenience. We are glad to take into account your suggestions for improvement and encourage you to read the lecture notes critically.

We thank Yevhen Havrylenko and Konstantin Schernstein for their help on translating the originally German lecture notes to English.

Contents

1 Introduction 1

1.1 Statistical units, variables and populations . . . 1

1.2 Scales and types of variables . . . 2

2 Introduction to R 4

2.1 First steps with R . . . 9

2.2 R literature . . . 10

2.3 R as a calculator . . . 11

2.4 Objects . . . 13

2.5 Vectors: Working with data sets . . . 13

2.5.1 Sequences and repititions . . . 16

2.5.2 Accessing the elements of a vector . . . 17

2.5.3 Logical values . . . 19

2.5.4 Missing values . . . 21

2.6 Loading already existing data sets . . . 22

2.7 Help in R . . . 24

2.8 Installing new packages . . . 26

2.9 Further possibilities for reading and outputting data . . . 27

2.10 Vector and matrix computations . . . 29

3 Univariate Data 32

3.1 Tabular and graphical representation of data . . . 32

3.1.1 Frequencies . . . 32

3.1.2 Empirical distribution function . . . 34

3.1.3 Diagrams for graphical representation . . . 36

3.2 Measures of location and dispersion . . . 38

3.2.1 Measures of location for nominal and ordinal data . . . 38

3.2.2 Measures of location for metric data . . . 40

3.2.3 Measures of dispersion . . . 42

3.2.4 Linear transformations of location and dispersion measures . . . 48

3.3 Classified data and histograms . . . 49

3.3.1 Classification . . . 50

3.3.2 Histograms . . . 51

3.4 Boxplots . . . 58

4 Bivariate Data **61**
4.1 Dependence measures . . . 61
4.1.1 Nominal variables . . . 61
4.1.2 Metric variables . . . 73
4.1.3 Ordinal variables . . . 82
4.2 Descriptive linear regression . . . 86
4.2.1 Method of least squares and the least squares estimator . . . 87
4.2.2 Evaluating the model fit . . . 91

5 Probability theory **102**
5.1 Discrete probability distributions . . . 104
5.2 Probability measure with Riemann density . . . 107
5.2.1 Examples of continuous probability distributions . . . 110
5.3 Conditional probabilities . . . 115
5.4 Stochastic independence of events . . . 118

6 Random variables **119**
6.1 Distribution function und quantile function . . . 121
6.2 Random vectors and multidimensional distributions . . . 125
6.3 Stochastic independence of random variables . . . 129
6.4 Expectation, variance, covariance and correlation . . . 131
6.5 Moment-generating function* . . . 139
6.6 Conditional distributions . . . 140
6.7 Central Limit Theorem* . . . 142

7 Introduction to inductive statistics **147**

8 Point estimation **150**
8.1 Maximum likelihood estimation . . . 153

9 Confidence intervals **159**
9.1 Confidence intervals for expectation . . . 160
9.1.1 Normal distribution with known variance . . . 160
9.1.2 Normal distribution with unknown variance . . . 163
9.1.3 Bernoulli distribution . . . 165
9.2 Confidence intervals for the variance of a normal distribution . . . 169
9.3 Confidence interval in the case of two samples . . . 169

10 Hypothesis testing **175**
10.1 One-sample tests for normal distribution . . . 181
10.1.1 Gauss test . . . 181
10.1.2 t-Tests . . . 186
10.2 Two-sample test for normally distributed data . . . 190
10.2.1 Independent samples . . . 190
10.2.2 Dependent samples . . . 194
10.3 One-sample test for binomial distribution . . . 199
10.4 Test of independence: contingency tables . . . 202

11 Multiple linear regression **205**
11.1 Point estimates . . . 206
11.1.1 Estimation of parameters in R . . . 206
11.2 Hypothesis testings . . . 211
11.3 Coefficient of determination . . . 216
11.4 Model selection . . . 219

A Standard plots **229**
A.1 High-level plotting functions . . . 230
A.2 Configurability . . . 233
A.3 Colors . . . 235
A.4 Low-level plotting functions . . . 237
A.5 Labels . . . 238

B ggplot2 **240**
B.1 ggplot2 package vs. graphics package . . . 241
B.2 ggplot2: `aes()` . . . 244
B.3 ggplot2: `geom` . . . 244
B.4 ggplot2: statistical transformations . . . 247
B.5 ggplot2: scales . . . 249
B.6 ggplot2: facets . . . 251
B.7 ggplot2: themes . . . 251

C Combinatorics **254**
C.1 Permutations . . . 254
C.2 Combinations . . . 255

D Distribution tables **256**

Index **257**

Bibliography **261**

Chapter 1

Introduction

In this course, we distinguish between descriptive and inferential statistics. Descriptive statistics aims to compress the collected data so that properties of the data set are summarized. An important aspect is also the graphical demonstration of the data set. Conclusions on units that are not part of the data set can not be drawn in this filed of statistics since no underlying probabilistic models are assumed. Without a probabilistic model for the population from which the data are collected, the framework is missing, in which properties of the observed data can be assessed. In order to be able to construct such models, we need an introduction to the basics of probability theory. This knowledge makes it possible to establish theoretical probabilistic models in inferential statistics, so that conclusions about the population can be drawn.

1.1 Statistical units, variables and populations

Let us start by defining some important terms. In the data collection process or survey, we are interested in certain quantities which we call *variables* $X, Y, Z, \ldots$. These variables are *characteristics* of some observed objects (e.g. grade of students, length of cars, price of gasoline at different days). The possible values, that a variable can take, form the set $\mathcal{X}$, which is called the *sample space*. Thus, the sample space consists of all (different) *values of the variable*. These values are recorded on fixed objects, which are also referred to as *statistical units* ω_i. The observed or recorded value of the variable is also called an *outcome*. The *population* consists of the set of all relevant statistical units and is denoted by Ω. If the investigation is limited to a subset of the population, one speaks of a *partial population*. Since it is often impossible to carry out a *full survey* (examination of all statistical units of the population), the survey is limited to a *sample*. This is, therefore, the actual examined subset of the population. The list of all observed values of the variables is referred to as a *data set*. Data sets usually have a matrix form. Each row of a data set represents an observed value of all considered variables measured on a statistical unit and each column represents observed values of a given variable for all considered statistical units.

Example 1.1.1. In 2011, x high school students have taken their Baccalaureate in mathematics. We consider a sample of 10 students. For each statistical unit (student), the variable "Grade in Mathematics" is observed. This variable can take the values $0, 1, \ldots, 15$. For the sample, which we want to examine, the following data set was obtained:

$$10, 11, 8, 13, 9, 15, 12, 11, 15, 12.$$

Remark 1.1.2. If the 10 high school students in the example above are all from the same class, we can barely hope that the distribution of the grades in the sample is approximately the same as that of the (underlying) population. In this case, the selection of the sample was too systematic. Therefore, the elements of the sample are usually selected by a random procedure. This is to ensure that theoretically every statistical unit of the population can become part of the sample. There are specially designed procedures for this, which are out of the scope of this first course in statistics. The interested reader is referred to [6].

1.2 Scales and types of variables

The methods by which the data in the original list can now be evaluated depend strongly on how the respective variable has been recorded, that is, on which scale it was measured. Different scale / variable types are distinguished. For our application, **only** the classification into nominal, ordinal, and metric types of variables will be important.

Nominal and ordinal variables are so-called *qualitative variables*, which only characterize belonging to a group or category. Thus, the values of *nominal variables* can only be compared with respect to their equality (or inequality). A ranking of the values is not possible. On the other hand, *ordinal variables* take values that satisfy a ranking, that means it can be compared whether one value is smaller, larger or equal to another value. However, no meaning can be attributed to the distances between the values. If nominal and ordinal variables are even coded with the help of numbers, calculations with these numbers are generally not interpretable and thus not meaningful.

Example 1.2.1. The eye color of a person represents a nominal variable. It takes one of the values *brown, green, blue*. The grade in mathematics in the year report for the fourth class of this person, on the other hand, is an ordinal variable. It takes (in Germany) the values *very good, good, satisfying, sufficient, deficient, inadequate*. Even if the variable values are encoded by the numbers 1 to 6, the distance between two grades cannot be interpreted meaningfully.

Metric (quantitative) variables take numerical values, whose differences can be meaningfully and consistently interpreted. With respect to the number of possible values, they are classified as discrete (at most countable infinite) or continuous (uncountable infinite) variables. Additionally, they can be interval-scaled, ratio-scaled or absolutely-scaled. This further subdivision is of no further relevance to us. Any metric variable is an *interval-scaled* variable, i.e. possible values can be represented by numbers and the distances between values (numbers) can be interpreted meaningfully. If an interval-scaled variable additionally has the property that the scales that are possible for measurement (for a variable, several scales are possible, e.g. degrees Celsius, Fahrenheit and Kelvin are possible scales for the variable temperature) have a common natural zero point value, we call it a *ratio-scaled* variable. The natural zero point guarantees that ratios of corresponding values on different scales are nevertheless always the same. If the variable also has a natural unit, then one speaks of an *absolutely-scaled* variable.

Example 1.2.2. A calendar divides the variable TIME in days (weeks, months, years). The distance between two points in time can be interpreted meaningfully. It is therefore an interval-scaled variable; but not a ratio-scaled variable as before, because a common natural zero point value is missing. For example, the Jewish calendar records the year 3761 BC of the Gregorian calendar as year 0. On the other hand, the study duration in the Bachelor program TUM-BWL is a ratio-scaled variable. It has possible values 6 (perhaps even less), 7, 8, 9 (perhaps also more) semesters and is therefore interval-scaled. But since the statement that a study duration of 9 semesters is 1.5 times a study duration of 6 semesters makes sense, the variable is also ratio-scaled.

Remark 1.2.3. (a) A metric variable is called continuous, if it can take any value from some interval (this is an uncountable number of values). However, many variables with discrete values are considered as continuous in practice, since they are discretely measured due to measurement inaccuracies (e.g. temperature measured with a thermometer used for household use) or due to their measurement unit (e.g. prices in euro and cent). Values of these variables usually have fine gradation.

(b) The same variable may have different variable types in different situations. For example, in a study on retirees in Germany, only a subdivision into *younger than 65 years/at least 65 years of age* could be important for the variable age and thus the age would be an ordinal variable. If, however, the age of each examined statistical unit is measured in days then the age is a metric variable.

As it is already mentioned, only the distinction in nominal, ordinal and metric is important for our application. Note that values of a metric variable fulfill all properties of an ordinal variable and any metric variable can therefore be considered as an ordinal one. Values of an ordinal variable fulfill all properties of a nominal variable and ordinal variables can similarly be treated as nominal ones. In this hierarchy one also speaks of a higher measurement level, i.e. metric data have, for example, a higher measurement level than ordinal data. The higher the measurement level, the more complex statistical methods can be used to evaluate the data.

For nominal data, only the frequency of the individual expressions can be used to describe the data. Because of the existing order relationship for ordinal data, the concept of an average value can already be used here and monotonic relationships between variables can be analyzed. Since the distances between the expressions are interpreted for metric data, dispersion measures can be introduced, which characterize the variability of the data.

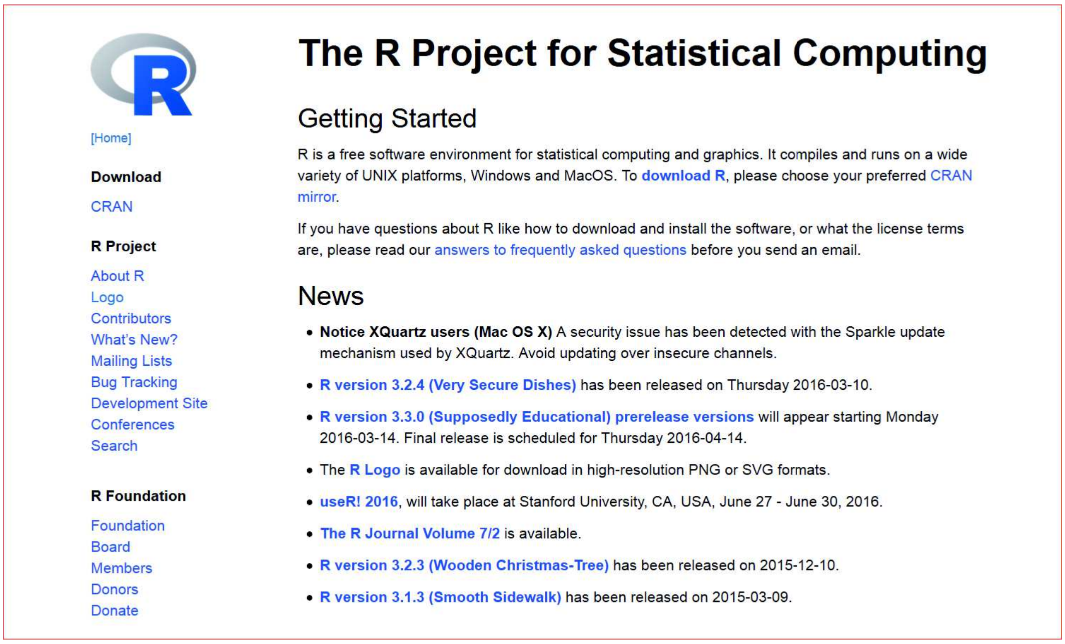
Before starting with the basics of descriptive statistics, we start with a brief introduction to the statistical programming language R.

Chapter 2

Introduction to R

What is R ?

- R is a statistical programming language for numerical and graphical data analysis
- R is based on the programming language **S** and was initially developed by Ross Ithaka and Robert Gentleman
- R is freely available under the GNU GPL license
- R is actively developed (2 releases per year)
- Official R Homepage: `http://www.R-project.org`



The R Project for Statistical Computing

Getting Started

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To **download R**, please choose your preferred CRAN mirror.

If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

News

- **Notice XQuartz users (Mac OS X)** A security issue has been detected with the Sparkle update mechanism used by XQuartz. Avoid updating over insecure channels.
- **R version 3.2.4 (Very Secure Dishes)** has been released on Thursday 2016-03-10.
- **R version 3.3.0 (Supposedly Educational) prerelease versions** will appear starting Monday 2016-03-14. Final release is scheduled for Thursday 2016-04-14.
- The **R Logo** is available for download in high-resolution PNG or SVG formats.
- **useR! 2016**, will take place at Stanford University, CA, USA, June 27 - June 30, 2016.
- **The R Journal Volume 7/2** is available.
- **R version 3.2.3 (Wooden Christmas-Tree)** has been released on 2015-12-10.
- **R version 3.1.3 (Smooth Sidewalk)** has been released on 2015-03-09.

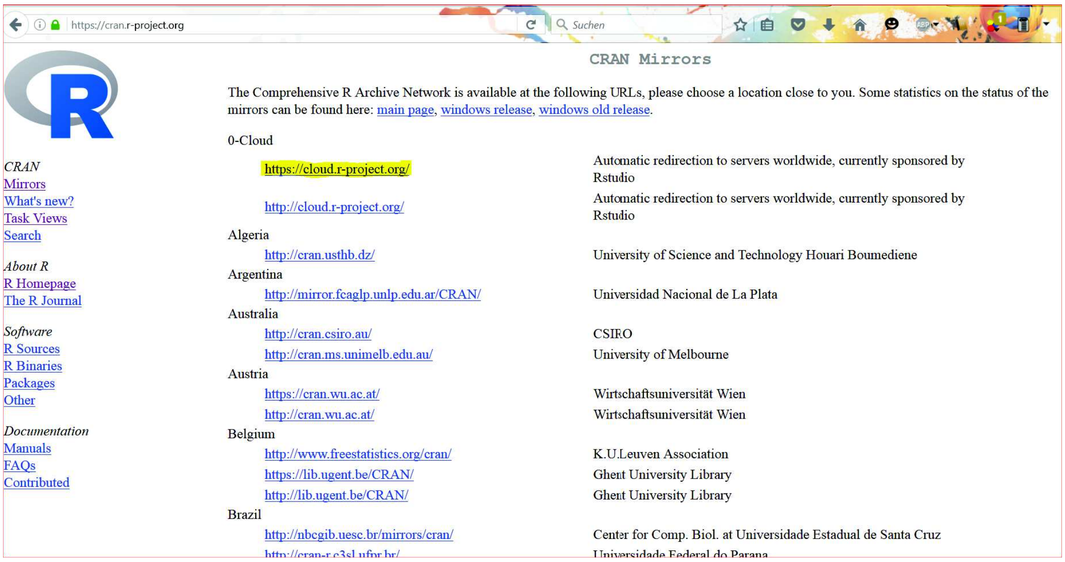
Figure 2.1: Homepage of the R-project for Statistical Computing.

Where can I get R ?

R consists of a base distribution (`base`) and a variety of additional packages (`packages`). In the course we will use the basic distribution and a few additional packages (later more).

Everything is available on the Comprehensive R Archive Network (CRAN).

`http://CRAN.R-project.org`



CRAN Mirrors

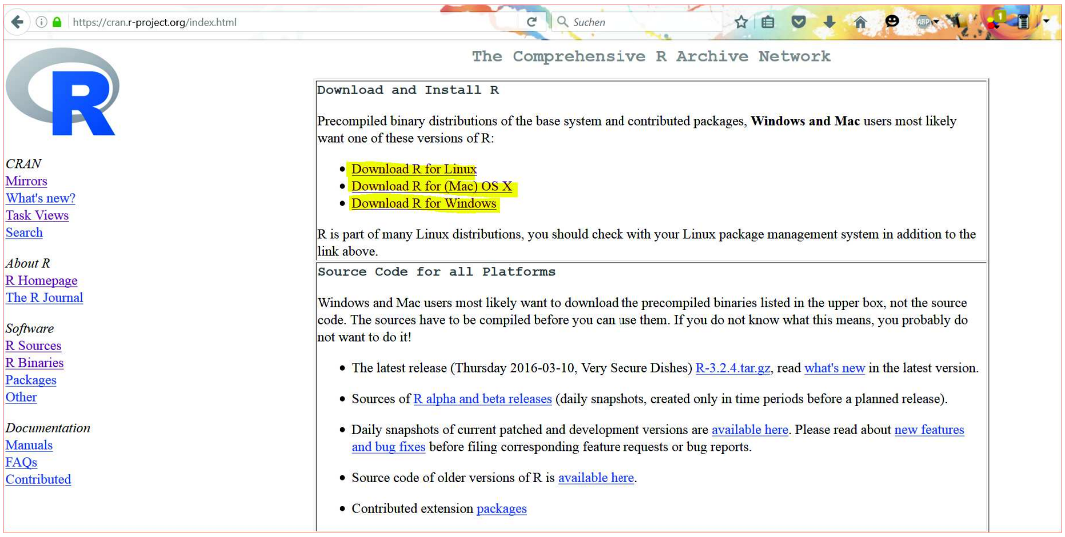
The Comprehensive R Archive Network is available at the following URLs, please choose a location close to you. Some statistics on the status of the mirrors can be found here: main page, windows release, windows old release.

0-Cloud	
https://cloud.r-project.org/	Automatic redirection to servers worldwide, currently sponsored by Rstudio
http://cloud.r-project.org/	Automatic redirection to servers worldwide, currently sponsored by Rstudio
Algeria	
http://cran.usthb.dz/	University of Science and Technology Houari Boumediene
Argentina	
http://mirror.fcaglp.unlp.edu.ar/CRAN/	Universidad Nacional de La Plata
Australia	
http://cran.csiro.au/	CSIRO
http://cran.ms.unimelb.edu.au/	University of Melbourne
Austria	
https://cran.wu.ac.at/	Wirtschaftsuniversität Wien
http://cran.wu.ac.at/	Wirtschaftsuniversität Wien
Belgium	
http://www.freestatistics.org/cran/	K.U.Leuven Association
https://lib.ugent.be/CRAN/	Ghent University Library
http://lib.ugent.be/CRAN/	Ghent University Library
Brazil	
http://nbcgib.uesc.br/mirrors/cran/	Center for Comp. Biol. at Universidade Estadual de Santa Cruz
http://cran-r.c3sl.ufpr.br/	Universidade Federal do Parana

Figure 2.2: CRAN Mirrors.

There you can find R distributions for

- Windows: `http://cran.r-project.org/bin/windows/base/`
- MacOS X: `http://cran.r-project.org/bin/macosx/`
- Linux: `http://cran.r-project.org/bin/linux/`



The Comprehensive R Archive Network

CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Other

Documentation
Manuals
FAQs
Contributed

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux
- Download R for (Mac) OS X
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

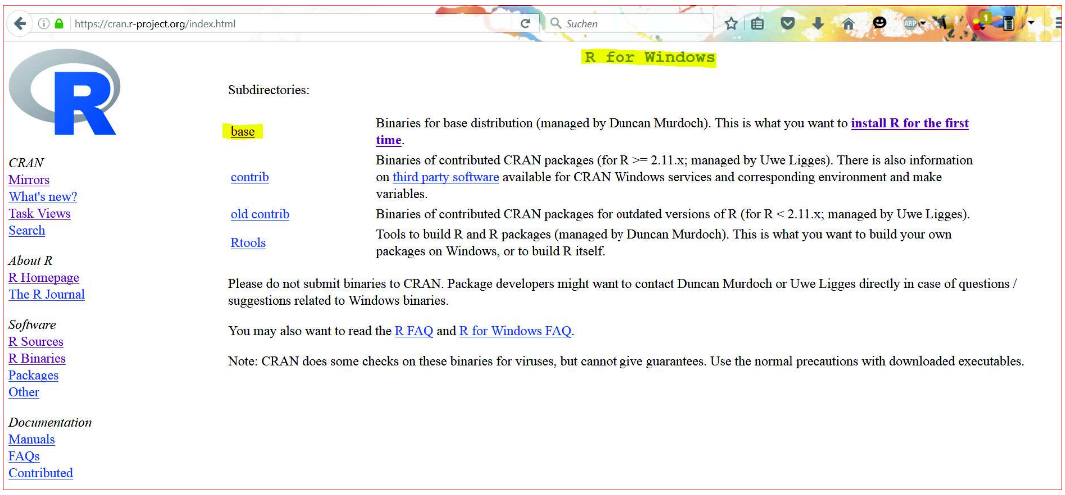
Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (Thursday 2016-03-10, Very Secure Dishes) R-3.2.4.tar.gz, read what's new in the latest version.
- Sources of R alpha and beta releases (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are available here. Please read about new features and bug fixes before filing corresponding feature requests or bug reports.
- Source code of older versions of R is available here.
- Contributed extension packages

Figure 2.3: Choice of the Windows, Mac oder Linux Distribution.

If working with a Windows operating system, the base distribution (base) is to be selected in the next step. The download of the latest R version can then be started.



R for Windows

CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Other

Documentation
Manuals
FAQs
Contributed

Subdirectories:

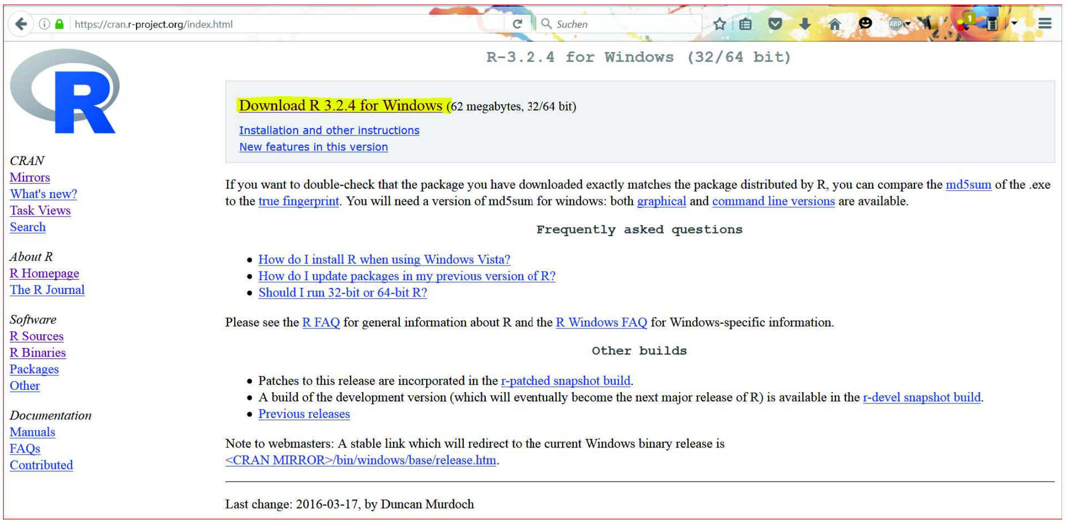
base	Binaries for base distribution (managed by Duncan Murdoch). This is what you want to **install R for the first time**.
contrib	Binaries of contributed CRAN packages (for R >= 2.11.x; managed by Uwe Ligges). There is also information on third party software available for CRAN Windows services and corresponding environment and make variables.
old contrib	Binaries of contributed CRAN packages for outdated versions of R (for R < 2.11.x; managed by Uwe Ligges).
Rtools	Tools to build R and R packages (managed by Duncan Murdoch). This is what you want to build your own packages on Windows, or to build R itself.

Please do not submit binaries to CRAN. Package developers might want to contact Duncan Murdoch or Uwe Ligges directly in case of questions / suggestions related to Windows binaries.

You may also want to read the R FAQ and R for Windows FAQ.

Note: CRAN does some checks on these binaries for viruses, but cannot give guarantees. Use the normal precautions with downloaded executables.

Figure 2.4: Selection of the Windows base distribution.



R-3.2.4 for Windows (32/64 bit)

Download R 3.2.4 for Windows (62 megabytes, 32/64 bit)

Installation and other instructions
New features in this version

If you want to double-check that the package you have downloaded exactly matches the package distributed by R, you can compare the md5sum of the .exe to the true fingerprint. You will need a version of md5sum for windows: both graphical and command line versions are available.

Frequently asked questions

- How do I install R when using Windows Vista?
- How do I update packages in my previous version of R?
- Should I run 32-bit or 64-bit R?

Please see the R FAQ for general information about R and the R Windows FAQ for Windows-specific information.

Other builds

- Patches to this release are incorporated in the r-patched snapshot build.
- A build of the development version (which will eventually become the next major release of R) is available in the r-devel snapshot build.
- Previous releases

Note to webmasters: A stable link which will redirect to the current Windows binary release is
<CRAN MIRROR>/bin/windows/base/release.htm.

Last change: 2016-03-17, by Duncan Murdoch

CRAN
Mirrors
What's new?
Task Views
Search

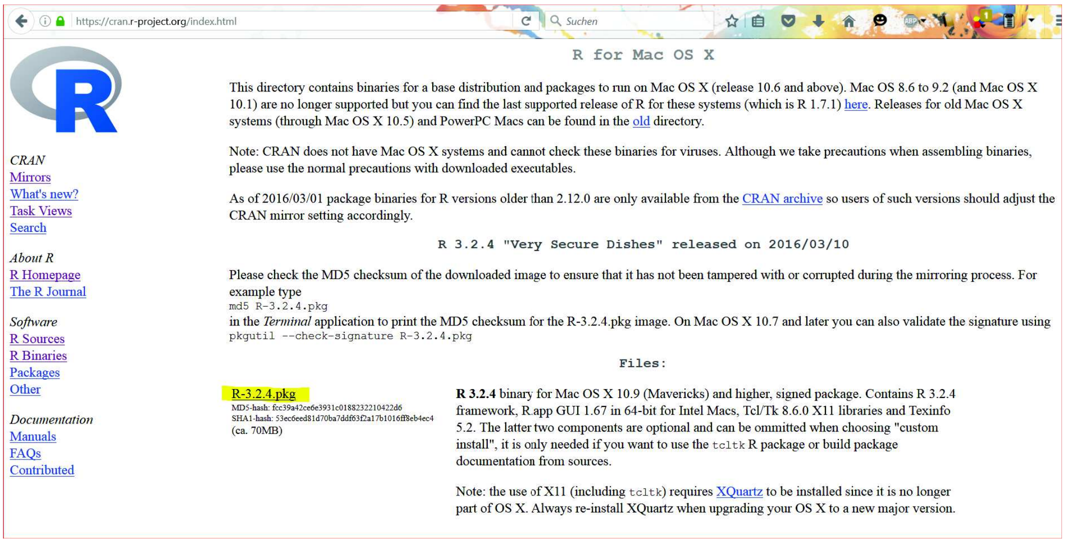
About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Other

Documentation
Manuals
FAQs
Contributed

Figure 2.5: Download of the current (as of April 01, 2016) Windows base distribution.

If you have decided to download R for (Mac) OS X (see Figure 2.3), you get directly to the corresponding download page.



R for Mac OS X

This directory contains binaries for a base distribution and packages to run on Mac OS X (release 10.6 and above). Mac OS 8.6 to 9.2 (and Mac OS X 10.1) are no longer supported but you can find the last supported release of R for these systems (which is R 1.7.1) here. Releases for old Mac OS X systems (through Mac OS X 10.5) and PowerPC Macs can be found in the old directory.

Note: CRAN does not have Mac OS X systems and cannot check these binaries for viruses. Although we take precautions when assembling binaries, please use the normal precautions with downloaded executables.

As of 2016/03/01 package binaries for R versions older than 2.12.0 are only available from the CRAN archive so users of such versions should adjust the CRAN mirror setting accordingly.

R 3.2.4 "Very Secure Dishes" released on 2016/03/10

Please check the MD5 checksum of the downloaded image to ensure that it has not been tampered with or corrupted during the mirroring process. For example type
`md5 R-3.2.4.pkg`
in the *Terminal* application to print the MD5 checksum for the R-3.2.4.pkg image. On Mac OS X 10.7 and later you can also validate the signature using
`pkgutil --check-signature R-3.2.4.pkg`

Files:

R-3.2.4.pkg
MD5-hash: fcc39a42ce6e3931c0188232210422d6
SHA1-hash: 53ec6eed81d70ba7ddf63f2a17b1016ff8eb4ec4
(ca. 70MB)

R 3.2.4 binary for Mac OS X 10.9 (Mavericks) and higher, signed package. Contains R 3.2.4 framework, R.app GUI 1.67 in 64-bit for Intel Macs, Tcl/Tk 8.6.0 X11 libraries and Texinfo 5.2. The latter two components are optional and can be ommitted when choosing "custom install", it is only needed if you want to use the `tcltk` R package or build package documentation from sources.

Note: the use of X11 (including `tcltk`) requires XQuartz to be installed since it is no longer part of OS X. Always re-install XQuartz when upgrading your OS X to a new major version.

CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

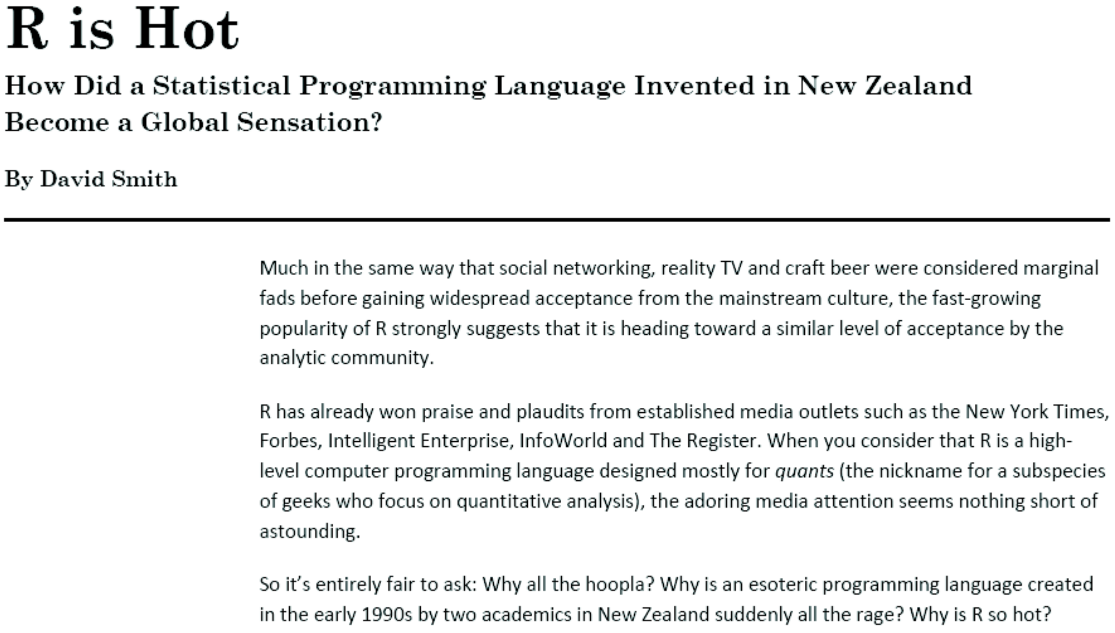
Software
R Sources
R Binaries
Packages
Other

Documentation
Manuals
FAQs
Contributed

Figure 2.6: Download of the current (as of April 01, 2016) Mac base distribution.

Who uses R?

- University:
  - Courses
  - Theses, Project studies
  - Research (leading Software in the field of statistics)
- Economy:
  - Banks and Insurance companies: AXA, Munich Re, Credit Suisse, Bank of America, Swiss Re etc.
  - others: Google, Facebook, Pfizer, Merck, The New York Times etc.



R is Hot

How Did a Statistical Programming Language Invented in New Zealand Become a Global Sensation?

By David Smith

Much in the same way that social networking, reality TV and craft beer were considered marginal fads before gaining widespread acceptance from the mainstream culture, the fast-growing popularity of R strongly suggests that it is heading toward a similar level of acceptance by the analytic community.

R has already won praise and plaudits from established media outlets such as the New York Times, Forbes, Intelligent Enterprise, InfoWorld and The Register. When you consider that R is a high-level computer programming language designed mostly for *quants* (the nickname for a subspecies of geeks who focus on quantitative analysis), the adoring media attention seems nothing short of astounding.

So it's entirely fair to ask: Why all the hoopla? Why is an esoteric programming language created in the early 1990s by two academics in New Zealand suddenly all the rage? Why is R so hot?

`http://www.revolutionanalytics.com/`

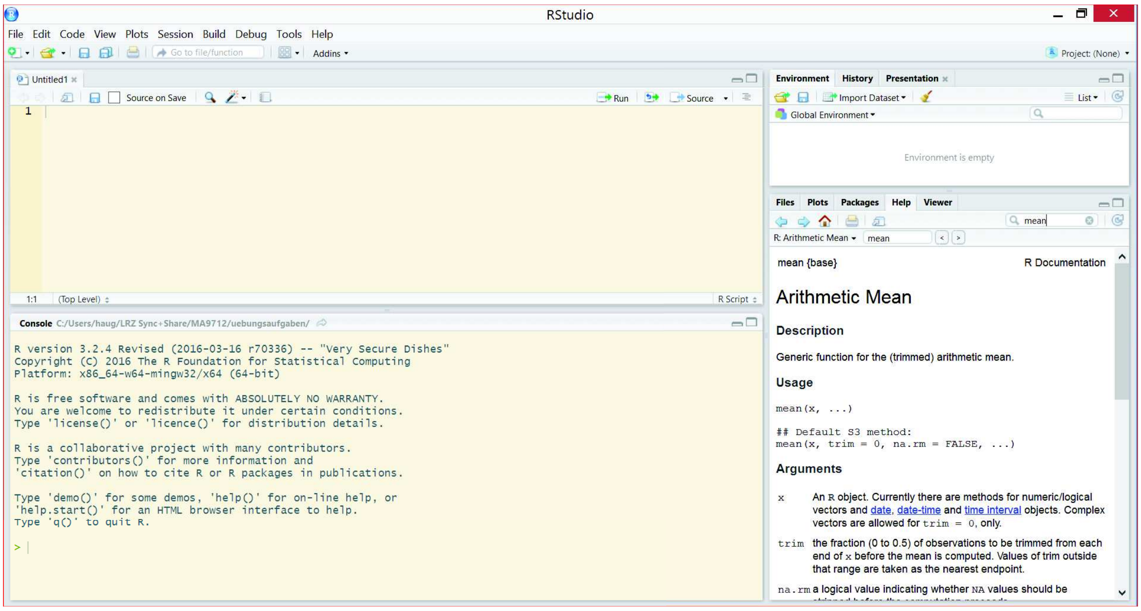
RStudio

RStudio is a development environment for R. In the course we will work with RStudio and therefore recommend to use it.

In addition to RStudio, there are other user interfaces, such as: The RCommander, but we will not go into details on it in the course.

2.1 First steps with R

After the installation of R and RStudio (the software automatically recognizes the existing R installation) and a start of RStudio you get a screen as shown in the figure below.



Console C:/Users/haug/LRZ Sync+Share/MA9712/uebungsaufgaben/

```
R version 3.2.4 Revised (2016-03-16 r70336) -- "Very Secure Dishes"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

mean {base} R Documentation

Arithmetic Mean

Description

Generic function for the (trimmed) arithmetic mean.

Usage

```
mean(x, ...)

## Default S3 method:
mean(x, trim = 0, na.rm = FALSE, ...)
```

Arguments

x	An R object. Currently there are methods for numeric/logical vectors and date, date-time and time interval objects. Complex vectors are allowed for trim = 0, only.
trim	the fraction (0 to 0.5) of observations to be trimmed from each end of x before the mean is computed. Values of trim outside that range are taken as the nearest endpoint.
na.rm	a logical value indicating whether NA values should be

Communication with R is typically done via the command line.

- Commands are entered at the ”Prompt” >
- Continuation lines are marked with a +
- ENTER sends the command to the interpreter
- Assignments are made via = or <-

```
> 2 + 2

## [1] 4
```

All objects (generally everything is an object generated in R) that are generated during an R session will be saved in the *workspace* .

- `ls()` shows the objects in the current workspace
- `rm(Objektname)` deletes `Objektname` from the current workspace
- The current working directory can be obtained via `getwd()` and changed via `setwd()`

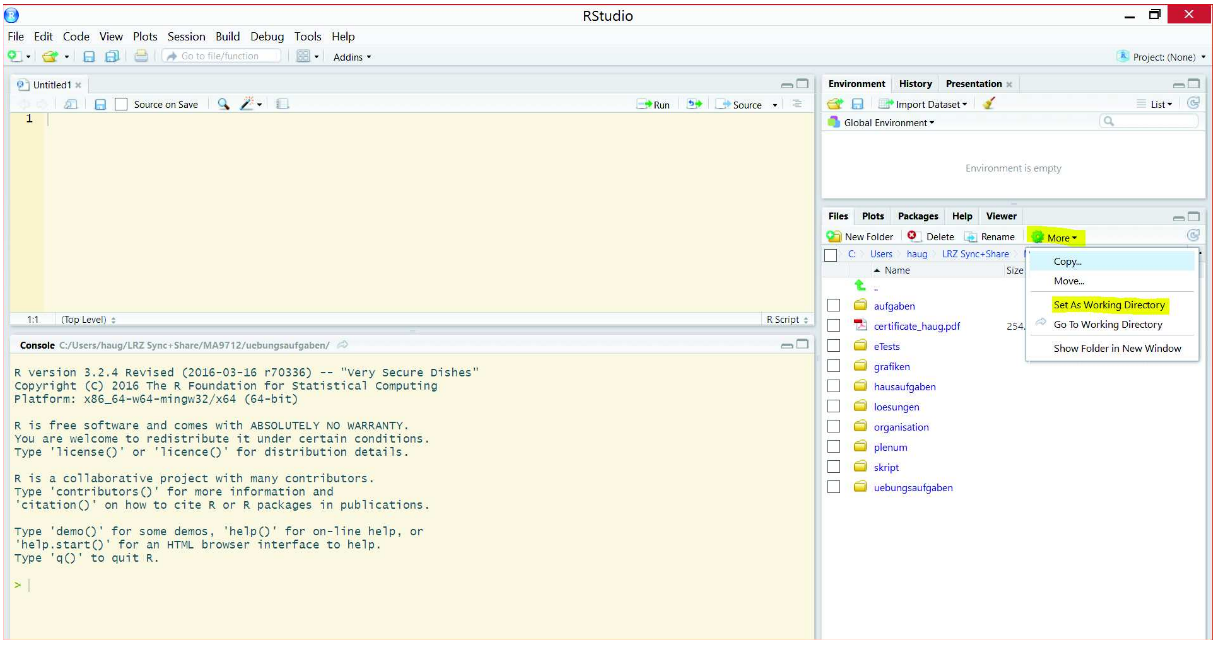
```
> getwd()

## [1] "Z:/lehre/statistik_fuer_bwl_MA9712/skript/englische_version"
```

```
> setwd("..")
> getwd()

## [1] "Z:/lehre/statistik_fuer_bwl_MA9712/skript"
```

These actions can of course be carried out using RStudio. The current workspace is also permanently displayed in an extra window. Alternatively, you can display the progress of the executed R commands in this window. Instead of an overview of the files in the current working directory, you can display the current plot, the installed additional packages (more on this later) or the help page of an R function (also later on) in the window at the bottom right.



Console C:/Users/haug/LRZ Sync+Share/MA9712/uebungsaufgaben/

```
R version 3.2.4 Revised (2016-03-16 r70336) -- "Very Secure Dishes"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |
```

R is terminated by q(). You can save the workspace together with commands. The commands entered are stored in the .Rhistory file and the workspace in .Rdata.

2.2 R literature

There are a variety of books on R:

http://www.r-project.org/doc/bib/R-books.html

For the course it is enough to focus on one book. A suggestion would be

Verzani, J. (2005). *Using R for Introductory Statistics*, Chapman& Hall.

This book is also available as an eBook. Access to this and other eBooks of the TUM University Library is available on the site https://eaccess.ub.tum.de/login. A preprint

version of this book can be found on this page
`http://www.math.csi.cuny.edu/Statistics/R/simpleR/printable/simpleR.pdf`.

Further interesting books on R and statistics with R are:

- Crawley, M. (2005). *Statistics: An Introduction using R*, Wiley.
- Crawley, M. (2007). *The R Book*, Wiley.
- Field, A., Miles, J., and Field, Z. (2012). *Discovering Statistics Using R*, Sage.
- Ohri, A. (2012). *R for Business Analytics*, Springer.

2.3 R as a calculator

R can be used as a calculator. The notation for four basic arithmetic operations is as follows: `+`, `-`, `*` and `/`. One can calculate the (nth-) power via `^`.

```
> 2 + 2

## [1] 4

> 3^2

## [1] 9

> (1 - 2) * 4

## [1] -4

> 1 - 2 * 4

## [1] -7
```

There are a variety of mathematical and statistical functions in R . Functions are called by their names followed by parentheses. Functional arguments can be passed within brackets.

```
> sqrt(2)

## [1] 1.414214

> cos(pi)

## [1] -1

> sin(pi)

## [1] 1.224606e-16

> exp(1)
```

```
## [1] 2.718282

> log(10)

## [1] 2.302585
```

Many functions in R can have several arguments, e.g.:

```
> log(10, 10)

## [1] 1

> log(10, base = 10)

## [1] 1

> log

## function (x, base = exp(1))  .Primitive("log")

> log(10, base = exp(1))

## [1] 2.302585
```

If you do not use functions correctly, you receive an error message such as:

```
> squareroot(2)

## Error in squareroot(2):  konnte Funktion "squareroot" nicht finden
```

```
> sqrt(-2)

## Warning in sqrt(-2):  NaNs wurden erzeugt

## [1] NaN
```

In the last example, the function `squareroot()` in R was not known. We can, however, define a function with this name.

```
> squareroot <- function(x, n = 2) {
+     squareroot <- x^(1/n)
+ }
```

The last object in the function definition (here `root`) is returned by the function.

```
> squareroot(2)
> sqrt(2)

## [1] 1.414214

> squareroot(2, 3)
```

2.4 Objects

One reason to deal with R is to work with data sets. These usually consist of several observations. An example would be the number of stranded whales in Texas in the nineties:

```
74 122 235 111 292 111 211 133 156 79
```

To store this data in R , we use a data vector. It can be created, for example, using the command `c()`.

```
> wale <- c(74, 122, 235, 111, 292, 111, 211, 133, 156, 79)
```

For the elements of a data vector, R distinguishes four data types:

Data type	Description	Example
logical	logical values	`FALSE`
numeric	integers and real numbers	`3.54`
complex	complex numbers	`5+4i`
character	characters and strings	`"statprak"`

The variable `wale` is now an object. Rughly speaking, anything in R is an object. Thus, the numerical data vector `wale` as well as the function `log()` is an object. Both belong to different object classes. The respective class of an object can be found with the function `class()`

```
> class(wale)

## [1] "numeric"

> class(log)

## [1] "function"
```

2.5 Vectors: Working with data sets

Once a data set is stored as a vector, we can always access these values

```
> wale

##  [1]  74 122 235 111 292 111 211 133 156  79
```

From the last section we already know the function `c()` This combines not only individual values but also several data vectors.

```
> x <- c(74, 122, 235, 111, 292)
> y <- c(111, 211, 133, 156, 79)
> c(x, y)

##  [1]  74 122 235 111 292 111 211 133 156  79
```

Data vectors, however, have the restriction that all elements must be of the same type (`numeric`, `character`, `logical`, `complex`). For instance, a `character` vector is defined by:

```
> simpsons <- c("Homer", "Marge", "Bart", "Lisa", "Maggie")
```

The command

```
> simpsons1 <- c(1, simpsons)
```

does not give an error message, but 1 is now of type `character`.

```
> simpsons1[1]

## [1] "1"

> simpsons1[1] + 1

## Error in simpsons1[1] + 1:  nicht-numerisches Argument für binären Operator
```

You can assign names to the entries of a data vector using the function `names()`.

```
> names(simpsons) <- c("father", "mother", "son", "daughter1", "daughter2")
> simpsons

##    father    mother       son daughter1 daughter2
##   "Homer"   "Marge"    "Bart"    "Lisa"  "Maggie"
```

The R function `names()` is one of the few functions that are used on the left of the assignment sign.

Some R functions can be applied directly to vectors and provide the desired result, for example:

```
> sum(wale)

## [1] 1524

> length(wale)

## [1] 10

> sum(wale)/length(wale)  #arithmetic mean

## [1] 152.4

> mean(wale)

## [1] 152.4
```

Further examples are:

```
> sort(wale)

##  [1]  74  79 111 111 122 133 156 211 235 292

> min(wale)

## [1] 74

> max(wale)

## [1] 292

> range(wale)

## [1]  74 292

> diff(wale)  #difference of successive values

## [1]   48  113 -124  181 -181  100  -78   23  -77

> cumsum(wale)  #cumulative sum

##  [1]   74  196  431  542  834  945 1156 1289 1445 1524
```

Some R functions work on every single entry of a vector. Consider, for example, the number of stranded whales in Florida and Texas during the 1990s.

```
> wale_florida <- c(89, 254, 306, 292, 274, 233, 294, 204, 204, 90)
> wale_texas <- wale
```

```
> wale_texas + wale_florida

##  [1] 163 376 541 403 566 344 505 337 360 169

> wale_texas - wale_florida

##  [1]  -15 -132  -71 -181   18 -122  -83  -71  -48  -11

> wale_texas - mean(wale_texas)

##  [1] -78.4 -30.4  82.6 -41.4 139.6 -41.4  58.6 -19.4   3.6 -73.4
```

Other functions such as `sin()`, `cos()`, `exp()`, `log()`,`^` oder `sqrt()` work elementwise as well.

```
> sqrt(wale_florida)

##  [1]  9.433981 15.937377 17.492856 17.088007 16.552945 15.264338
##  [7] 17.146428 14.282857 14.282857  9.486833
```

If you have already entered data in R , you can edit it with the `data.entry()` or `edit()` function. `data.entry()` opens a spreadsheet window, while `edit()` opens a text editor.

2.5.1 Sequences and repititions

The input of successive data, such as the numbers 1 to 99, is very easy to perform in R .

```
> 1:10

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

> rev(1:10)

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

> 10:1

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

A sequence of the form $a, a+h, a+2h, \ldots, a+(n-1)h$ can be generated in R with

```
> a <- 1
> h <- 4
> n <- 5
> a + h * (0:(n - 1))
```

```
## [1]  1  5  9 13 17
```

When you create a sequence, you specify the beginning and end, the step size, or the length of the sequence. You can use the function `seq()`

```
> seq(1, 17, by = 4)

## [1]  1  5  9 13 17

> seq(1, 18, by = 4)

## [1]  1  5  9 13 17

> seq(1, 17, length = 5)

## [1]  1  5  9 13 17
```

The `rep()` function is used to repeat individual values or whole vectors.

```
> rep(1, 10)

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

> rep(1:3, 4)

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

To specify for each element the number of times it should be repeated, use the following command

```
> rep(c("one", "two"), c(1, 2))

## [1] "one" "two" "two"
```

2.5.2 Accessing the elements of a vector

An index is assigned to each element of a vector `x`. The command `x[Index]` returns this element.

```
> price <- c(88.8, 88.3, 90.2, 93.5, 95.2, 94.7, 99.2, 99.4, 101.6)
> price[1]

## [1] 88.8

> price[9]

## [1] 101.6
```

The last element (without knowing the exact length of the vector) is obtained by the command

```
> price[length(price)]

## [1] 101.6
```

If you want to get more elements, you use a vector of positive indexes as an index.

```
> price[1:4]

## [1] 88.8 88.3 90.2 93.5
```

The indexes do not have to be consecutive. We can also access the 1st, 5th and 6th element.

```
> price[c(1, 5, 6)]

## [1] 88.8 95.2 94.7
```

If a vector contains `n` elements, a negative number between `-n` and `-1` can also be used as an index. The following result is obtained

```
> price[-1]

## [1]  88.3  90.2  93.5  95.2  94.7  99.2  99.4 101.6
```

So we get all entries except for the first element.

```
> price[-(1:7)]

## [1]  99.4 101.6
```

If elements of a vector have names, then they can also be accessed via their names.

```
> simpsons["father"]

##  father
## "Homer"
```

Individual elements of a vector can be assigned new values with the `[]` notation.

```
> price[1] <- 88
> price

## [1]  88.0  88.3  90.2  93.5  95.2  94.7  99.2  99.4 101.6
```

You can also add new elements in this way.

```
> price[10:13] <- c(100, 98.5, 97.9, 99.3)
> price

##  [1]  88.0  88.3  90.2  93.5  95.2  94.7  99.2  99.4 101.6 100.0
## [11]  98.5  97.9  99.3
```

Here one has to be careful. The command

```
> price[10:13] <- c(100, 98.5)
> price

##  [1]  88.0  88.3  90.2  93.5  95.2  94.7  99.2  99.4 101.6 100.0
## [11]  98.5 100.0  98.5
```

does not provide any error, even if it was not desired in that way.

2.5.3 Logical values

R also provides answers (`TRUE` or `FALSE`) for questions such as ”Is `price` greater than `95`?” (the answer is given for each element in this case). `TRUE` and `FALSE` are called logical values.

```
> price

##  [1]  88.0  88.3  90.2  93.5  95.2  94.7  99.2  99.4 101.6 100.0
## [11]  98.5 100.0  98.5

> price > 95  #Question: price bigger 95

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

Which observed prices are greater than 95? These values are obtained, for example, with the command

```
> price[price > 95]

## [1]  95.2  99.2  99.4 101.6 100.0  98.5 100.0  98.5
```

To get the indexes of price observations greater than 95 we use

```
> which(price > 95)

## [1]  5  7  8  9 10 11 12 13
```

One gets a detailed description of the function `which()` with `?which`.

```
> price[c(5, 7, 8, 9, 10, 11, 12, 13)]

## [1]  95.2  99.2  99.4 101.6 100.0  98.5 100.0  98.5
```

One can do calculations with logical values. `TRUE` is interpreted as 1 and `FALSE` as 0.

```
> FALSE + TRUE + FALSE

## [1] 1

> TRUE * FALSE

## [1] 0

> TRUE - TRUE

## [1] 0
```

```
> x <- c(-2, 4, -6, 1)
> x < 0

## [1]  TRUE FALSE  TRUE FALSE

> sum(x < 0)  #Number of observation values less than 0

## [1] 2
```

`Logical` vectors can also be compared with one another. The `&` (and) and`|` (or) operators compare element by element, while `&&` and `||` compare all elements from left to right until the result is `TRUE` or `FALSE` respectively.

```
> x <- 1:5
> x < 5

## [1]  TRUE  TRUE  TRUE  TRUE FALSE

> x > 1

## [1] FALSE  TRUE  TRUE  TRUE  TRUE

> x > 1 & x < 5

## [1] FALSE  TRUE  TRUE  TRUE FALSE

> x > 1 && x < 5

## [1] FALSE
```

```
> x > 1 | x < 5

## [1] TRUE TRUE TRUE TRUE TRUE

> x > 1 || x < 5

## [1] TRUE

> x == 3

## [1] FALSE FALSE  TRUE FALSE FALSE

> x != 3

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

The comparison with several values does not work as one would guess intuitively:

```
> x == c(2, 4)

## Warning in x == c(2, 4):  Länge des längeren Objektes
##  ist kein Vielfaches der Länge des kürzeren Objektes

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

Instead, one uses the `%in%` operator:

```
> x %in% c(2, 4)

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

2.5.4 Missing values

If there are missing values in a data set, this is indicated in R by `NA` (not available). Entries with `NA` can be found using the `is.na()` function.

```
> (y <- c(0, 1, 0, NA, 0, 0))

## [1]  0  1  0 NA  0  0

> y > 0

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

The following R -command is not appropriate for checking whether a vector has a missing value:

```
> y == NA

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

One should apply the function `is.na()` to check that:

```
> is.na(y)

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

If one is interested in the arithmetic (or empirical) mean of a data vector that has missing values, one does not obtain the desired result using

```
> mean(y)

## [1] NA
```

The above outcome makes sense, since we tried to calculate the arithmetic mean of all entries, including those indicated as NA. Therefore, the result is also NA, since at least one entry in not known (missing). In some functions it is possible to use the option `na.rm` to indicate that missing values should be ignored while computation.

```
> mean(y, na.rm = TRUE)

## [1] 0.2
```

2.6 Loading already existing data sets

A series of data sets is already contained in R . You can also load further packages in R using `library()`, which contain further new data sets.

```
> range(lynx)

## [1]   39 6991
```

Thus, the data set `lynx` is already contained in R .

```
> survey

## Error in eval(expr, envir, enclos):  Objekt 'survey' nicht gefunden
```

Hence, the data set `survey` is not found in the first place. It is not contained directly in R . The data set `survey` is a part of the `MASS` package. This add-on package, as well as a few additional packages, is already included in the basic distribution. However, these packages must be explicitly loaded with the `library()` function before they are first used in an R session. An overview of the existing packages can be obtained with the command

```
> library()
```

An output of the installed packages was suppressed here. Now lets go back to the data set `survey`. Since it is included in the `MASS` package, we load this package.

```
> library(MASS)
> dim(survey)

## [1] 237  12
```

Larger records, such as `survey` (containing 12 columns at 237 rows), are usually stored as `data.frame` (later more). In this format, for example, each column can be given a name (equivalent to a variable). We can use this name to access the individual columns directly.

```
> names(geyser)  #also included in the MASS package

## [1] "waiting"  "duration"
```

So the data set `geyser` has two columns `waiting` and `duration`. The former concerns waiting time between two eruptions of the Old Faithful Geyser (`https://en.wikipedia.org/wiki/Old_Faithful`). The latter contains information about the duration of an eruption.

```
> waiting

## Error in eval(expr, envir, enclos):  Objekt 'waiting' nicht gefunden
```

However, direct access to the variable (column) `waiting` does not appear to be possible. The $ operator

```
> geyser$waiting[1:5]

## [1] 80 71 57 80 75
```

allows access to the variables of a data set. It is also possible to integrate the data set into the search path with `attach()`. Then it works

```
> attach(geyser)
> waiting[1:5]

## [1] 80 71 57 80 75
```

With `detach()`, variables can be removed from the search path. However, we do **not** recommend using `attach()` It is often advantageous to specify the reference to the data record explicitly. Namely, e.g. for two attached data sets, both containing a variable with the same name, only the variable that is contained in the data set that was last attached can be accessed.

```
> new_dataset <- data.frame(waiting = 1:5)
> attach(new_dataset)

## The following object is masked from geyser:
##
##     waiting

> waiting[1:5]

## [1] 1 2 3 4 5

> detach(geyser)
> detach(new_dataset)
```

The `with()` function allows the application of a function to a variable in a data set without having previously loaded it to the search path. The syntax for this is

`with( data.frame, function )`

```
> names(Sitka)

## [1] "size"  "Time"  "tree"  "treat"
```

These is growth data of a tree species (also included in the `MASS` package).

```
> size

## Error in eval(expr, envir, enclos):  Objekt 'size' nicht gefunden

> length(Sitka$size)

## [1] 395

> with(Sitka, range(size))

## [1] 2.23 6.63
```

2.7 Help in R

The help in R is generally started with `help()` Looking for specific help on a particular function, e.g. `var()`, you can use `help(var)` or `?var`
If you do not know the function `var()` yet, you can display all the help pages that contain the character `var` with `help.search("var")`.
If there is an Internet connection, you can use `help.start()` to display an HTML help page where there is also a search function.

If, for example, we want to learn more about the `summary()` function, we can call the corresponding help function with the command

```
> `?`(summary)
```

```
summary                 package:base                 R Documentation

Object Summaries

Description:

     ‘summary’ is a generic function used to produce result summaries
     of the results of various model fitting functions.  The function
     invokes particular ‘methods’ which depend on the ‘class’ of the
     first argument.

Usage:

     summary(object, ...)

     ## Default S3 method:
     summary(object, ..., digits = max(3, getOption("digits")-3))
     ## S3 method for class 'data.frame':
     summary(object, maxsum = 7,
            digits = max(3, getOption("digits")-3), ...)

     ## S3 method for class 'factor':
     summary(object, maxsum = 100, ...)

     ## S3 method for class 'matrix':
     summary(object, ...)
...
```

In Chapter 3.2.2, we will learn more about the output of the `summary()` function.

The `help` function can be used not only to get help information about another function. It can also be applied to a data set that is contained in R . In such a case, the function provides a brief description of the existing variables in the corresponding data set. For example, `?lynx` provides the following information

```
> `?`(lynx)
```

```
lynx                 package:datasets                 R Documentation

Annual Canadian Lynx trappings 1821-1934

Description:

     Annual numbers of lynx trappings for 1821-1934 in Canada. Taken
     from Brockwell & Davis (1991), this appears to be the series
     considered by Campbell & Walker (1977).

Usage:

     lynx

...
```

2.8 Installing new packages

We have already loaded some individual packages. But not all packages are included in the basic version of R . New packages are installed with the command

```
 install.packages( "PackageName" )
```

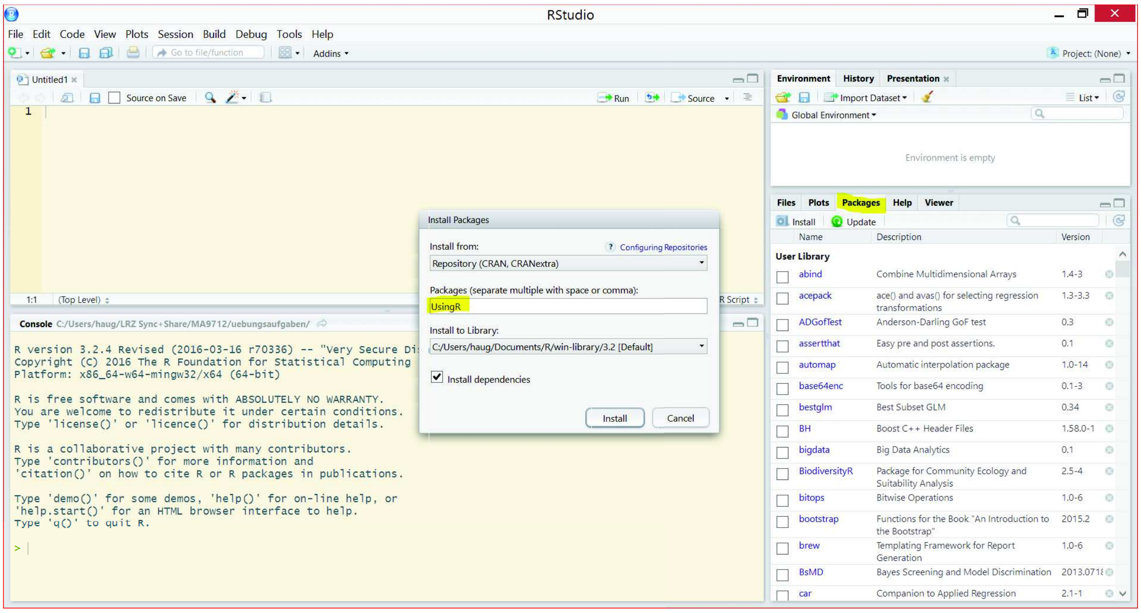
from one of the CRAN websites. For example, to install the package UsingR (package to [8]), use the command

```
> install.packages("UsingR")
```

Homework: Install the UsingR package. In the homework we will always use data sets from this package.

An alphabetical list of currently available packages (more than 8000, as of 01.04.2016) can be found on the website http://lib.stat.cmu.edu/R/CRAN/web/packages/available_packages_by_name.html.

In RStudio there is a possibility to install packages via the menu bar in the lower right window. To do this, select *Install Packages* and then enter the package name under *Packages* in the window that opens.



Console C:/Users/haug/LRZ Sync+Share/MA9712/uebungsaufgaben/

```
R version 3.2.4 Revised (2016-03-16 r70336) -- "Very Secure Di
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |
```

Install Packages

Install from: ? Configuring Repositories

Repository (CRAN, CRANextra)

Packages (separate multiple with space or comma):

UsingR

Install to Library:

C:/Users/haug/Documents/R/win-library/3.2 [Default]

☑ Install dependencies

Install Cancel

	Name	Description	Version
	User Library		
□	abind	Combine Multidimensional Arrays	1.4-3
□	acepack	ace() and avas() for selecting regression transformations	1.3-3.3
□	ADGofTest	Anderson-Darling GoF test	0.3
□	assertthat	Easy pre and post assertions.	0.1
□	automap	Automatic interpolation package	1.0-14
□	base64enc	Tools for base64 encoding	0.1-3
□	bestglm	Best Subset GLM	0.34
□	BH	Boost C++ Header Files	1.58.0-1
□	bigdata	Big Data Analytics	0.1
□	BiodiversityR	Package for Community Ecology and Suitability Analysis	2.5-4
□	bitops	Bitwise Operations	1.0-6
□	bootstrap	Functions for the Book "An Introduction to the Bootstrap"	2015.2
□	brew	Templating Framework for Report Generation	1.0-6
□	BsMD	Bayes Screening and Model Discrimination	2013.071
□	car	Companion to Applied Regression	2.1-1

2.9 Further possibilities for reading and outputting data

We already know the function `c()` for entering data. If the data is stored in a table form, the `read.table()` function (and its variants, see below) is suitable. The file `"geysershort.txt"` is, for example, of the form (separated by a tab stop)

```
   waiting duration
1       80 4.016667
2       71 2.150000
3       57 4.000000
4       80 4.000000
5       75 4.000000
6       77 2.000000
7       60 4.383333
8       86 4.283333
9       77 2.033333
10      56 4.833333
```

Then the data can be read (imported) with the command

```
> read.table("geysershort.txt", header = TRUE, sep = "\t")

##        waiting.duration
## 1  1        80  4.016667
## 2  2        71  2.150000
## 3  3        57 4.000000
## 4  4        80 4.000000
## 5  5        75 4.000000
## 6  6        77 2.000000
## 7  7        60 4.383333
## 8  8        86 4.283333
## 9  9        77 2.033333
## 10 10       56 4.833333
```

The `read.delim()` function exists for importing data records whose columns are separated by tab stop. Thus, the command

```
> read.delim("geysershort.txt")

##       waiting.duration
## 1  1        80 4.016667
## 2  2        71 2.150000
## 3  3        57 4.000000
## 4  4        80 4.000000
## 5  5        75 4.000000
## 6  6        77 2.000000
## 7  7        60 4.383333
## 8  8        86 4.283333
## 9  9        77 2.033333
## 10 10       56 4.833333
```

gives the same result. Note that in this case, the `header=TRUE` option is omitted since this is already the default setting of the `read.delim()` function. If you use a comma as a decimal separator in contrast to the example above, you can either use the option `sep=','` in the `read.delim()` function or use the `read.delim2()` function.

If you want to read a data set whose columns are separated by commas, use either the `read.table(file="..", sep=",")` command or the `read.csv()` function. You can use the `read.csv2()` function to import records whose columns are separated by semicolons without having to specify additional options. In this case, by default, the comma is assumed to be a decimal separator. Most spreadsheet programs can export csv files.

If the file is not located in the current work directory (`getwd()`), the complete path must be specified to read the file.

To export tables easily, one uses `write.table()`.

```
> write.table(wale, file = "wale.txt")
```

The function `write.csv()` also exists analogously to `read.csv()`.

2.10 Vector and matrix computations

Now consider arithmetic operations with vectors and matrices. By default, vectors in R are column vectors. A row vector is obtained by transposing the column vector.

```
> t(2:4)

##      [,1] [,2] [,3]
## [1,]    2    3    4
```

The scalar product of two vectors is not calculated using the operator * This leads, as we have already seen, an element-wise multiplication. The scalar product is obtained by means of the operator %*%.

```
> t(2:4) %*% 1:3

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

Surprisingly, with the command (scalar product of two column vectors)

```
> 2:4 %*% 1:3

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

we get no errors but the same result as before.

In this case, R assumes that we want to calculate the scalar product of a row vector and a column vector. This is convenient if we really want that and dangerous (there is no error message) if that was not our intention. Maybe we wanted to calculate the following matrix.

```
> 2:4 %*% t(1:3)

##      [,1] [,2] [,3]
## [1,]    2    4    6
## [2,]    3    6    9
## [3,]    4    8   12
```

Matrices can be also created directly without previous computation. You can use the commands `matrix()`, `rbind()` or `cbind()` for this.

```
> (Z <- matrix(c(4, 6, 2, 1, 8, 9), ncol = 3))

##      [,1] [,2] [,3]
## [1,]    4    2    8
## [2,]    6    1    9

> (Y <- matrix(c(4, 6, 2, 1, 8, 9), nrow = 3))
```

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

> (M <- rbind(1:3, 2:4))

##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    2    3    4

> (A <- cbind(c(4, 7), c(0, 5)))

##      [,1] [,2]
## [1,]    4    0
## [2,]    7    5
```

The inverse of the matrix A is obtained using the function `solve()` (solving the linear system $Ax = I_2$).

```
> (A_inv <- solve(A))

##       [,1] [,2]
## [1,]  0.25  0.0
## [2,] -0.35  0.2

> A_inv %*% A

##      [,1] [,2]
## [1,]    1    0
## [2,]    0    1
```

We can also access elements of a matrix using the `[]` notation. However, the line and column are required. However, the specification of a row and a column is required. If you specify only a row or a column, you get the entire row or the entire column. For example, the first row of the inverse of `A` is obtained with the command

```
> A_inv[1, ]

## [1] 0.25 0.00
```

and the second column with the command

```
> A_inv[, 2]

## [1] 0.0 0.2
```

The second element in the first column can be accessed with the command

```
> A_inv[2, 1]

## [1] -0.35
```

R summary

In this section, we learned a few feaures of R . In addition to basic arithmetic operators, we learned several elementary functions (`log()`, `exp()`, `sin()`, `cos()`, `sum()`,...). After that we addressed the topic, how vectors can be created in R (`c()`, `seq()`, or `rep()`). We also discussed how to perform component-wise operations with vectors. After explaining how to access individual elements of a vector (`[]` notation), we addressed the topic of logical operators in R (`&`, `&&`, `|`, `||`). It was shown how additional packages can be installed with `install.packages()` and packages can be loaded to an R session with the help of the function `library()`. The most important functions for inputting and outputting data were discussed, e.g. `read.table()` and `write.table()`. At the end of the section, we discussed how to create matrices, multiply them (`%*%`) and access their elements.

Chapter 3

Univariate Data

In the following, graphical as well as numerical methods for an analysis of univariate data are presented. Depending on the variable type (nominal, ordinal, metric) different statistical methods will be used.

3.1 Tabular and graphical representation of data

Before more detailed statistical analyzes can be carried out, the collected univariate data should usually be prepared and brought in a vector form (i.e. should be represented as a vector). Therefore, we assume that we are given a data set $x_1, \ldots, x_n$, which are observed values or **observations** of the variable X with values $u_1, \ldots, u_m$.

As a first step, we can compute the frequency of each value u_i in the given data set $x_1, \ldots, x_n$ and display the resulted frequencies either in tabular or graphical form. This step makes sense primarily for nominal/ordinal and discrete metric variables. In the case of continuous variables, a large number of possible values are generally present. Therefore, their values are often categorized in several classes and the frequency distribution of these classes is further treated. We discuss this topic in Section 3.3.1 in more detail.

3.1.1 Frequencies

We distinguish between absolute and relative frequencies.

Definition 3.1.1. *Let $u_1, \ldots, u_m$ be the possible values of the variable X and $x_1, \ldots, x_n$ be the given data set. Denote by $|\{\cdots\}|$ the number of elements in $\{\cdots\}$. We call*

$$n_j = \left|\{\ i \in \{1, \ldots, n\} \mid x_i = u_j\ \}\right|$$

the absolute frequency of value u_j, $j \in \{1, \ldots, m\}$. Thus, the absolute frequency n_j represents the number of appearances of value u_j among $x_1, \ldots, x_n$.

For a set $A \subseteq \mathbb{R}$ and $x \in \mathbb{R}$ we define by

$$\mathbb{1}_A(x) = \begin{cases} 1, & x \in A \\ 0, & x \notin A \end{cases}$$

the indicator function of the set A. Using the indicator function, the absolute frequency n_j of value u_j can be expressed as

$$n_j = \sum_{i=1}^{n} \mathbb{1}_{\{u_j\}}(x_i).$$

wenn x_i den wert u_j hat
wird += 1 addiert

Example 3.1.2. Consider the following data:
25 students expressed their preference for beer with a classification into the following four categories.

(1) local beer, can

(2) local beer, bottle

(3) beer from a small brewery

(4) imported beer.

The variable "beer preference" is therefore a nominal variable, since no order of the four categories can be specified. The results of the survey are recorded in the vector `beer`.

```
> beer <- c(3, 4, 1, 1, 3, 4, 3, 3, 1, 3, 2, 2, 2, 1, 4, 3, 2, 3, 2,
+     1, 2, 3, 2, 4, 2)
```

Hence the absolute frequency n_3 of the value "beer from a small brewery" is

```
> sum(beer == 3)

## [1] 8
```

A tabular overview of all values is as follows:

```
> table(beer)

## beer
## 1 2 3 4
## 5 8 8 4
```

Definition 3.1.3. *Let n_j, $j \in \{1, \ldots, m\}$ be the absolute frequency of the value u_j in the data set $x_1, \ldots, x_n$. The quotient*

$$f_j := \frac{n_j}{n}$$

is called the relative frequency of the value u_j, $j \in \{1, \ldots, m\}$.

Example 3.1.4. Once again consider the data from example 3.1.2. The relative frequency of the value "local beer, can" is given by

```
> sum(beer == 1)/length(beer)

## [1] 0.2
```

A tabular representation of all relative frequencies is as follows:

```
> prop.table(table(beer))

## beer
##    1    2    3    4
## 0.20 0.32 0.32 0.16
```

The approach shown in the above examples is not very helpful for continuous data because many values of continuous variables are often observed only once (or at least not very often). Therefore more advanced techniques and concepts are needed to analyze continuous data.

3.1.2 Empirical distribution function

For a metric variable, it makes sense to consider the cumulative relative frequencies. Thus, statements of the form ”x% of the data are less than or equal to y” can be made. If all these cumulative relative frequencies are presented in a graph, the empirical distribution function is plotted.

Definition 3.1.5. *For observations of a metric variable X the empirical distribution function $F_n : \mathbb{R} \to [0, 1]$ is defined by*

$$F_n(x) := \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{(-\infty,x]}(x_i), \qquad x \in \mathbb{R}.$$

Thus, for $x \in \mathbb{R}$, the value $F_n(x)$ describes the proportion of the observations whose value is less than or equal to x.

Example 3.1.6. We consider the data set `exec.pay` (`UsingR`), which contains severance payments (in 10000 $) for directors of 199 American companies. First, we want to calculate the proportion of directors who have received a compensation equal to or less than 100000 US dollars.

```
> library(UsingR)
> data(exec.pay)
> sum(exec.pay <= 10)/length(exec.pay)

## [1] 0.1457286
```

For this we can also use the `ecdf()` function. Applied to a data set, it defines the complete empirical distribution function, which can then be evaluated at the desired values.

```
> Fn <- ecdf(exec.pay)
> Fn(10)

## [1] 0.1457286
```

Now we are only interested in the distribution of the ”big” terminal bonuses, more precisely, bonuses of more than one million US dollars. We therefore plot the empirical distribution function of the severance payments, which were greater than one million US dollars.

```
> exec.pay_new <- exec.pay[exec.pay>100]
> plot(ecdf(exec.pay_new),
+      main = "Emp. distr. function of the  severance compensations > 1 million US dollars"
```

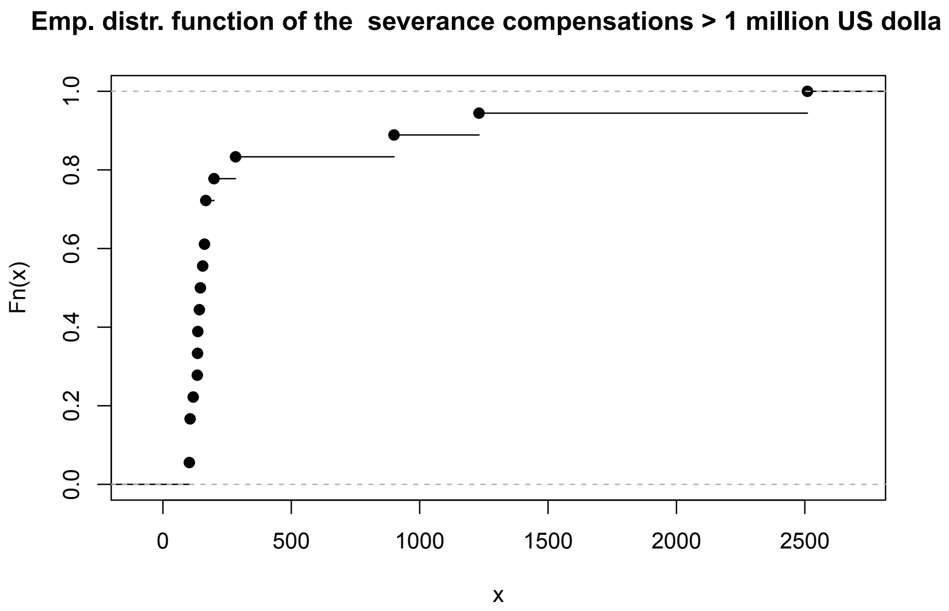


Figure 3.1: Empirical distribution function of the severance compensations greater than 1 million US dollars

From Figure 3.1, it becomes clear that the empirical distribution function is a monotone increasing step (or staircase) function with jumps at the observed variable values. In areas of many different observations, the empirical distribution function increases very fast (e.g., in the interval $(100, 300)$ in Figure 3.1) and is constant in intervals without observations.

In order to characterize the properties of the empirical distribution function, we need the concept of the order statistics, which is introduced in the next definition.

Definition 3.1.7. *For observations $x_1, \ldots, x_n$ of an ordinal or metric variable, we call the ascending sequence of the observed values*

$$x_{(1)} \leq x_{(2)} \leq \cdots \leq x_{(n)}$$

order statistics. The observed value $x_{(j)}$ is denoted as the j-th order statistic and it is the $j-$th smallest observation. $x_{(1)}$ is called minimum and $x_{(n)}$ maximum.
If an observation x_j occurs exactly once, its position in the sequence of order statistics is called the rank of x_j and is denoted by $Rg(x_j)$. If an observation x_j occurs several times (s times), that is, it holds

$$x_{(r-1)} < \underbrace{x_{(r)} = x_{(r+1)} = \cdots = x_{(r+s-1)}}_{=x_j} < x_{(r+s)},$$

we denote the rank of x_j as the arithmetic mean of all positions of the order statistics with value x_j, i.e.

$$Rg(x_j) = \frac{r + (r+1) + \cdots + (r+s-1)}{s} = r + \frac{s-1}{2}.$$

r = 1? oder Platz?

Properties of the empirical distribution function are now summarized in the following remark.

Remark 3.1.8. Let $u_{(1)} < \ldots < u_{(n)}$ be the ascending sequence of possible values of the variable X. The empirical distribution function F_n has the following properties:

(i) F_n is a monotone increasing and right-continuous step function.

(ii) F_n may jump only at $u_{(j)}$ and the jump height at the point $u_{(j)}$ is equal to the relative frequency $f_{(j)}$ of $u_{(j)}$.

(iii) By definition we have

$$F_n(x) = 0 \quad \text{for} \quad x < u_{(1)} \qquad \text{and} \qquad F_n(x) = 1 \quad \text{for} \quad x \geq u_{(m)}.$$

(iv) $F_n(x)$ describes the proportion of the observations from the interval $(-\infty, x]$.

(v) $1 - F_n(x)$ describes the proportion of the observations from the interval (x, ∞).

(vi) $F_n(y) - F_n(x)$ describes the proportion of the observation values from the interval $(x, y]$ with $x < y$.

3.1.3 Diagrams for graphical representation

In the last section, we have seen how the frequencies (absolute / relative) of the observed values of a data set can be calculated and tabulated. A good overview is often obtained by means of a graphical representation. We distinguish the following types of diagrams:

- bar chart (or bar plot)
- pie chart
- dot chart (or point diagram)

The generation of such diagrams in R is presented in the next two examples.

Example 3.1.9. We consider again the data from Example 3.1.2. A bar chart is created by the function `barplot()`.

```
> par(mfrow = c(1, 2))
> barplot(beer)
> barplot(table(beer), xlab = "beer - category", ylab = "frequency")
```

The command `barplot(beer)` did not return the desired result. The function `barplot()` expects as input the height of the bars, i.e. the corresponding frequencies. Here, the observations were wrongly interpreted as frequencies. If you first determine the frequencies with the command `table(beer)`, you get the desired result.

Another possibility of the representation would be a pie chart.

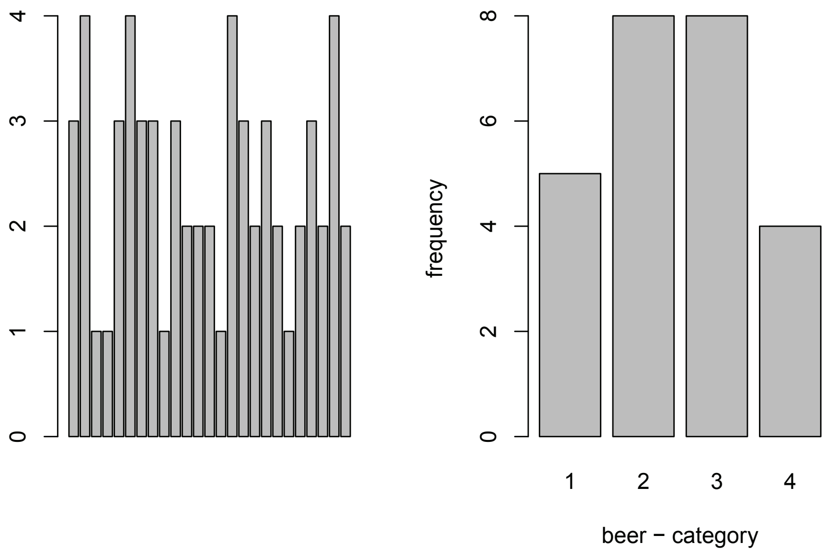


Figure 3.2: Bar charts of the beer data. On the left, the observations are wrongly used as frequencies. On the right, the actual absolute frequencies of the four values are shown.

```
> pie(table(beer), col = rainbow(4))
```

A pie chart is problematic for data with relative frequencies of almost the same order since it is difficult to assess the difference between relative frequencies.

Example 3.1.10. We consider an artificial example given by the frequency (number) of days in the months January, February and April.

```
> days <- c(31, 28, 30)
> names(days) <- c("January", "February", "April")
```

For this example, it is preferable to use a dot chart instead of a pie chart. One can plot a dot chart in R with the function `dotchart()`.

```
> par(mfrow = c(1, 2))
> pie(days)
> dotchart(days, xlab = "days per month")
```

Looking at the pie chart, it is difficult to distinguish any difference between the number of days in January, February and April. If we look at the dot chart then we can clearly see the existing differences in the number of days.

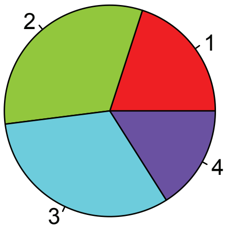


Figure 3.3: Pie chart for the beer data

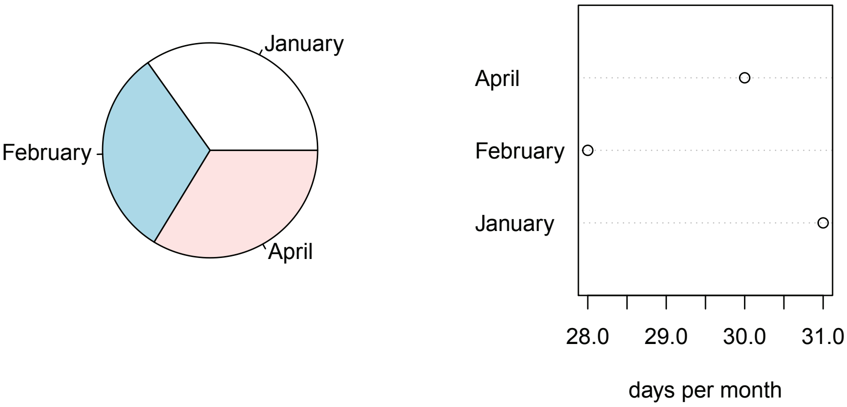


Figure 3.4: Pie chart and dot chart for the number of days in January, February and April.

3.2 Measures of location and dispersion

The description of a data set using observed frequencies does not constitute a great reduction yet. The data reduction is achieved by concentrating only on parameters such as the location of the center or the variability (or dispersion) of the data. If one considers the location and dispersion parameters then a simple comparison of two data sets is often possible. Location and dispersion parameters can be determined in the dependence of the variable type. We begin with measures of location for nominal and ordinal data.

3.2.1 Measures of location for nominal and ordinal data

The mode is a measure of location for the description of nominal data sets. The value(s) that occur(s) most frequently in the data set is (are) called the mode.

Definition 3.2.1. *Assume that values $u_1, \ldots, u_m$ of a variable X are observed and represented in the data set $x_1, \ldots, x_n$. Let u_j have the absolute frequency n_j resp. the relative frequency f_j, $j \in \{1, \ldots, m\}$. Each value u_{j^*} with*

$$n_{j^*} = \max\{n_1, \ldots, n_m\}$$

resp.

$$f_{j^*} = \max\{f_1, \ldots, f_m\}$$

is called the mode. We denote the mode by x_{mod}.

The mode is easily read from a graphical representation of the data. In a bar chart, the mode is (are) the observation(s) with the highest bar.

Due to the order structure of ordinal data, additional measures of location can be used for this variable type. These measures are based on the order statistics of the data set and are referred to as p-quantiles.

Definition 3.2.2. *Let $x_{(1)} \leq \cdots \leq x_{(n)}$ be the order statistics of an ordinal data set $x_1, \ldots, x_n$. For $p \in (0, 1)$ a p-quantile $\tilde{x}_p$ is an observation with the property*

$$\begin{aligned} \tilde{x}_p &:= x_{(k)}, && \text{if } np < k < np + 1, np \notin \mathbb{N}, \\ \tilde{x}_p &\in \{x_{(k)}, x_{(k+1)}\}, && \text{if } k = np, np \in \mathbb{N}. \end{aligned}$$

The $p-$quantile for $p = 0.5$ has its own name and is called the median. Specifically, this means that an observation $\tilde{x}_{0.5}$ with the properties

$$\begin{aligned} \tilde{x}_{0.5} &= x_{\left(\frac{n+1}{2}\right)}, && \text{if } n \text{ odd}, \\ \tilde{x}_{0.5} &\in \{x_{\left(\frac{n}{2}\right)}, x_{\left(\frac{n}{2}+1\right)}\}, && \text{if } n \text{ even} \end{aligned}$$

is called the median.

Every p-quantile $\tilde{x}_p$ satisfies the following condition:

At least $p \cdot 100\%$ of all observations are smaller or equal than $\tilde{x}_p$ and at least $(1-p) \cdot 100\%$ of all observations are greater or equal than $\tilde{x}_p$.

A p quantile is called

- median for $p = 0.5$
- lower quartile for $p = 0.25$
- upper quartile for $p = 0.75$
- k-th decile for $p = \frac{k}{10}$, $k = 1, \ldots, 9$
- k-th percentile for $p = \frac{k}{100}$, $k = 1, \ldots, 99$

Example 3.2.3. Let us consider the grades in a mathematics exam of a fictional school class with 12 students. Since the school class is fictional, we first generate the grades

```
> (marks <- sample(1:6, 12, replace = TRUE))

##  [1] 2 5 3 2 4 4 1 2 4 4 4 4
```

Since $n = 12$ is even, there are two candidates for the median - 6th and 7th order statistic.

```
> sort(marks)[6:7]

## [1] 4 4
```

Since these are identical, 4 is the unique median. The same result is obtained with the command

```
> median(marks)

## [1] 4
```

4 is the most frequent value of the variable and thus the mode as seen here:

```
> table(marks)

## marks
## 1 2 3 4 5
## 1 3 1 6 1
```

3.2.2 Measures of location for metric data

The median for metric data is defined analogous to the ordinal case besides a small modification. In the case of an odd sample size, the definition is identical to the original case. If the sample size is event the same, theoretically any value from the interval $[x_{(\frac{n}{2})}, x_{(\frac{n}{2}+1)}]$ is a candidate for the median aAll these values are now allowed values since we have a metric scale). Often, the arithmetic mean from the boundary points is defined as the median. This is what we want to do. However, as before we give the generic definition of a p quantile.

Definition 3.2.4. *Let $x_{(1)} \leq \cdots \leq x_{(n)}$ be the order statistics of a metric data set $x_1, \ldots, x_n$. For $p \in (0,1)$, a p-quantile $\tilde{x}_p$ is an observation with the property*

$$\tilde{x}_p := \begin{cases} x_{(k)}, & \text{if } np < k < np+1, np \notin \mathbb{N}, \\ \frac{1}{2}(x_{(k)} + x_{(k+1)}), & \text{if } k = np, np \in \mathbb{N}. \end{cases}$$

The special case of the median $x_{0.5}$ is given by

$$\tilde{x}_{0.5} := \begin{cases} x_{(\frac{n+1}{2})}, & \text{if } n \text{ odd}\,, \\ \frac{1}{2}(x_{(\frac{n}{2})} + x_{(\frac{n}{2}+1)}), & \text{if } n \text{ even}\,. \end{cases}$$

Example 3.2.5. We consider the annual income (in thousands of €) of five managing directors.

```
> income <- c(50, 60, 100, 75, 200)
```

Since n is odd, the median is given by the third order statistic 75. The 0.2-quantile should be calculated as the arithmetic mean of the first and second order statistic. In R we get the following results

```
> median(income)

## [1] 75

> quantile(income, prob = 0.2)

## 20%
##  58
```

The number 58 is not the arithmetic mean from 50 and 60. Let us investigate how this result comes up. For this, we look at the help of the function `quantile()`:

```
> `?`(quantile)
```

```
...
Usage
quantile(x, ...)

## Default S3 method:
quantile(x, probs = seq(0, 1, 0.25), na.rm = FALSE,
         names = TRUE, type = 7, ...)
...
Types
quantile returns estimates of underlying distribution quantiles based on one or two
order statistics from the supplied elements in x at probabilities in probs. One of
the nine quantile algorithms discussed in Hyndman and Fan (1996), selected by type,
is employed.
```

Thus, we can read in the help that 9 different methods are implemented to compute a p-quantile. This is also consistent with our approach, since we have established that every point in the interval $[x_{(k)}, x_{(k+1)}]$ for $k = np \in \mathbb{N}$ is an acceptable quantile value. For example, with the `type=2` option, we obtain the p-quantile according to Definition 3.2.4:

```
> quantile(income, prob = 0.2, type = 2)

## 20%
##  55
```

Probably the best-known measure of location for metric data is the arithmetic mean, which we also refer to as the empirical mean value and will introduce in the following definition.

Definition 3.2.6. *Let $x_1, \ldots, x_n$ be observations of some metric variable X. The the empirical mean (arithmetic mean) $\overline{x}_n$ is then defined by*

$$\overline{x}_n := \frac{1}{n} \sum_{i=1}^{n} x_i \,.$$

For simplicity we refer to the empirical mean by $\overline{x}$ if there is no confusion about the number n.

Example 3.2.7. We look again at the annual income of the five managing directors. The empirical mean is obtained as

```
> mean(income)

## [1] 97
```

The value is larger than the median, but it still describes the center of the observed feature expressions. Let us include the sixth managing director, Bill G., in our sample:

```
> income.with.bill <- c(income, 50000)
```

We obtain the empirical mean

```
> mean(income.with.bill)

## [1] 8414.167
```

This value describes no longer the center of the data as we all would hopefully agree. The empirical mean of the extended data is greater than 5 out of the 6 observations. Thus, we recognize that the empirical mean is very sensitive with respect to few very large (or very small) observations. In contrast, the median behaves much more robust in such situations, as it can be seen here:

```
> median(income)

## [1] 75

> median(income.with.bill)

## [1] 87.5
```

It is well known that the median behaves more robust against outliers (very large / small observations compared to the rest of the data).

3.2.3 Measures of dispersion

In most cases, location measures are not sufficient to describe a given data set. Two different data sets with an identical center (described by some measure of location) can have different variability. In Figure 3.5, two data sets of sample size 500 are displayed. They have a similar empirical mean, but a significant difference in data variability.

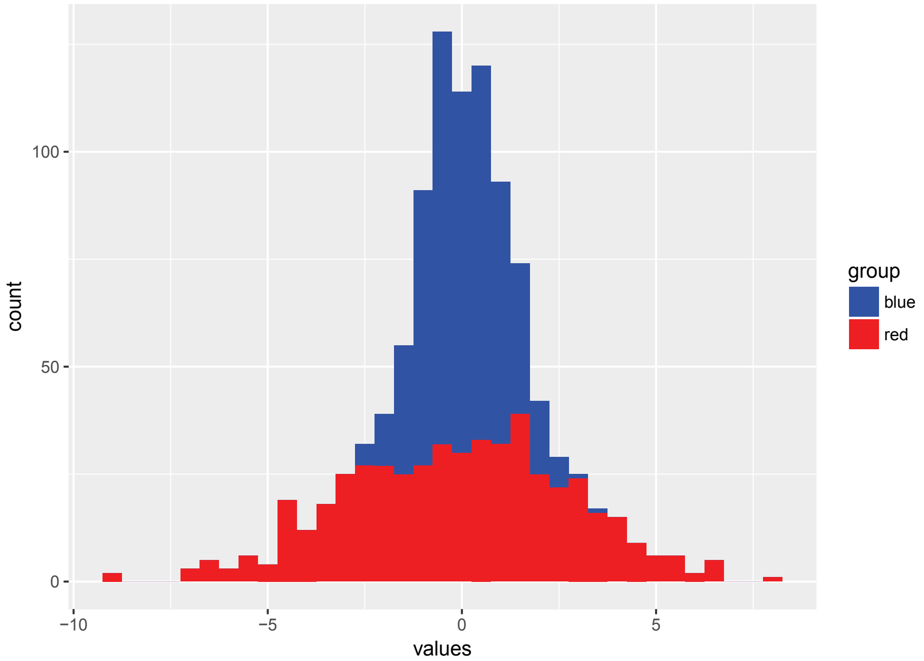


Figure 3.5: Histograms of two data sets with similar empirical means but a significant difference in data variability.

```
> head(dataset)

##        values group
## 1  0.08541773  blue
## 2  1.11661021  blue
## 3 -1.21885742  blue
## 4  1.26736872  blue
## 5 -0.74478160  blue
## 6 -1.13121857  blue

> tail(dataset)

##         values group
## 995   1.325408   red
## 996  -3.417801   red
## 997  -2.918534   red
## 998  -6.639505   red
## 999   1.779918   red
## 1000 -2.073043   red
```

```
> ggplot(dataset, aes(x = values, fill = group)) + geom_histogram(binwidth = 0.5) +
+     scale_fill_manual(values = c("blue", "red"))
```

The data set colored red has obviously a larger variability than the blue one.

We can quantify the above graphical observation using measures of dispersion. In the sequel, we consider several measures of dispersion. Before we introduce a set of different measures of dispersion, consider the following remark.

Remark 3.2.8. Each measure of dispersion uses the distance between values of the variables in one way or another. Since this distance can not be interpreted meaningfully for both nominal and ordinal features, we can not specify any measures of dispersion for these variable types.

A first measure of the data variability would be the difference between the largest and the smallest observation.

Definition 3.2.9. *Let $x_1, \ldots, x_n$ be observations of a metric variable X. Then the span (*or the range*) of the data set $x_1, \ldots, x_n$ is defined as the difference between the maximum $x_{(n)}$ and the minimum $x_{(1)}$*

$$Sp := x_{(n)} - x_{(1)}.$$

R calculates the minimum and maximum values of the data using the function `range()`. The span is then obtained by computing the difference of the output of `range()` using the function `diff()`.

```
> diff(range(income))

## [1] 150
```

The span is defined through the maximal and minimal observations. Therefore, by definition, it is very sensitive to outliers. A dispersion measure, which is more robust against outliers, is the interquartile distance.

Definition 3.2.10. *Let $x_1, \ldots, x_n$ be a data set of metric observations. The interquartile range is defined as the distance between the lower and upper quartiles as follows*

$$IQR = \tilde{x}_{0.75} - \tilde{x}_{0.25}.$$

The interquartile distance does not usually change if the smallest and largest observations vary. It describes the span of the 50% data from the middle region of the data range.

Example 3.2.11. Let us consider the data set `exec.pay` from the package `UsingR`, which contains terminal bonuses (in $\$10,000$) for managing directors of 199 American companies.

```
> IQR(exec.pay)

## [1] 27.5
```

Thus, the middle 50% of the severance compensations vary over a range of length $275000. If we calculate the interquartile distance ”by hand”, we get the same result.

```
> (q.exec.pay <- quantile(exec.pay))

##     0%    25%    50%    75%   100%
##    0.0   14.0   27.0   41.5 2510.0

> q.exec.pay[4] - q.exec.pay[2]

##  75%
## 27.5
```

The above calculated quantiles are also given in a short summary of the data set, which can be generated using the function `summary()`. However, please keep in mind that quantiles computed with the function `quantile()` and the option `type=2` are only consistent with Definition 3.2.4.

```
> (s <- summary(exec.pay))

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   14.00   27.00   59.89   41.50 2510.00
```

From this summary we also recognize that the empirical mean and the median are quite different. Let us recall that the value of the empirical distribution function (see Definition 3.1.5) at a point x is the fraction of the observations that are less than or equal to x. Using the empirical distribution function, we can now compute the proportion of the paid compensations, which are less than or equal to the empirical mean.

```
> vf <- ecdf(exec.pay)
> vf(s[4])  # The object s contains in the fourth place the empirical mean value

## [1] 0.839196
```

More than 83% of the observations are thus smaller than the empirical mean. Thus, the empirical mean does not describe the center of the data in this example, as we all would hopefully agree. Let us now evaluate the empirical distribution function at all points from the summary. We get

```
> vf(s)

## [1] 0.01507538 0.26633166 0.51256281 0.83919598 0.74874372
## [6] 1.00000000
```

Surprisingly, the values of the empirical distribution function at the p-quantiles for $p = 0, 0.25, 0.5, 0.75, 1$ are not always equal to p as it would be expected. This is explained by the fact that quantiles are not uniquely determined in general and therefore they do not always separate an univariate data set in prespecified proportions. Therefore, the quantile function approximates the inverse of the empirical distribution function.

The previous measures of dispersion were defined by distances between different observations (minimum, maximum, quartiles). Now we introduce a measure of dispersion, which takes into account distances of all observations to some location measure. More specifically, the squared deviations from the empirical mean will be considered. Squaring makes very small deviations hardly significant and large deviations very important.

Definition 3.2.12. *Let $x_1, \ldots, x_n$ be observations of a metric variable X with the associated empirical mean $\overline{x}_n$. The empirical variance s_n^2 is then defined by*

$$s_n^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \overline{x}_n)^2 \,.$$

The empirical standard deviation s_n is defined as the square root of the empirical variance, i.e.

$$s_n = \sqrt{s_n^2}.$$

Remark 3.2.13. (i) In Definition 3.2.12, the sum of the quadratic deviations is divided by $n-1$ instead of n. This seems surprising at first sight, since the data set has n observations. Here the term *degrees of freedom* comes into play. Consider the following example. We would like to have a sample of the size (length) $n = 5$. Each combination of 5 statistical units from the population will constitute the desired sample. So we have *freedom* to choose the 5 statistical units. Using 5 measured observations, we can now compute the empirical mean. Therefore, we divide here by n, i.e. 5, because the effective sample size is 5. However, if we want to calculate the empirical variance, we must determine the empirical mean first. Let us assume $\overline{x}_n = 10$. If we now want to replace the 5 statistical units by randomly choosing new ones from the population (the new sample should have the already known properties of the old sample, i.e. the same empirical mean), then we can choose only 4 statistical units arbitrarily, since $n\overline{x}_n = 50$. Thus, there are only $df = 4$ degrees of freedom to choose statistical units. The effective sample size is only $df = n-1$. Therefore, it makes sense to divide the sum of the squared deviations by the effective sample size $n-1$. We will come back to the concept of degrees of freedom in the section on inferential statistics.

(ii) If we divide by n instead of $n-1$ then we obtain the average of quadratic deviations (empirical dispersion). This distinction is no longer of great importance for large sample sizes.

In R , the empirical variance and the empirical standard deviation are calculated by the functions `var()` and `sd()`, respectively. Let's look at the following data sets

```
> test.a <- c(80, 85, 75, 77, 87, 82, 88)
> test.b <- c(100, 90, 50, 57, 82, 100, 86)
> mean(test.a)

## [1] 82

> mean(test.b)

## [1] 80.71429
```

Samples `test.a` and `test.b` have similar empirical means.

```
> var(test.a)

## [1] 24.66667

> sd(test.a)

## [1] 4.966555

> var(test.b)

## [1] 394.2381

> sd(test.b)

## [1] 19.85543
```

The empirical variances (standard deviations) are, however, very different, since the observations of the sample `test.b` are scattered much more (symmetrically) around the center of the data than the observations of the sample `test.a`.

The measures of dispersion introduced so far do not connect the data variability with the location of the data. It is obvious that a data set with an empirical mean, which is 10000 times the empirical mean of another data set, naturally has a greater variability than the second data set. As an example, let us consider the prices of a particular auto brand in different car dealerships and on the other hand the prices of one kilo of flour in various supermarkets. The prices for the car certainly vary by several hundred euros, while the prices for one kilo of flour certainly differ by only a few cents. Nevertheless, the variation of the flour data with regard to the average price for one kilo of flour could be significantly greater than for the car prices. A dispersion measure, which compares data variability in relation to the population mean, is the coefficient of variation.

Definition 3.2.14. *Let $x_1, \ldots, x_n$ be observations of a metric variable X with non-zero empirical mean $\overline{x}_n \neq 0$ and empirical standard deviation s_n. The coefficient of variation V_n is then defined by*

$$V_n = \frac{s_n}{\overline{x}_n}.$$

The empirical standard deviation and the empirical mean have the same unit. The coefficient of variation is the quotient of these two quantities and does not have any measurement unit. Therefore, it is also suitable for comparing the variability of two data sets measured in different units.

Example 3.2.15. We generate a data set for our previous example on flour and car prices. These are generated in such a way that the car prices have a significantly greater empirical standard deviation than the flour prices but have a smaller coefficient of variation. In the basic version of R , there is no function for calculating the coefficient of variation (since it is a simple function of `sd()` and `mean()`). Therefore, we define our own function.

```
> var_koef <- function( x ){
+   # Function for calculating the coefficient of variation
+   # We should first check whether the empirical mean value is not equal to 0
+   if(mean(x) != 0)
+     v <- sd(x) / mean(x)
+   else
+     v <- "coefficient of variation not defined, since mean(x)=0"

+   v # Last call is returned
+ }
```

Now we generate the data and calculate the empirical standard deviation as well as the coefficient of variation.

```
> preis_auto <- sample(c(30000, 30100, 29800, 30150, 29950), size = 50,
+                         replace = TRUE)
> preis_mehl <- sample( seq(2.5, 3.5, by = 0.1), size = 50, replace = TRUE)
> # Since the data was generated randomly with sample(), the results are different each tim
> sd(preis_auto)

## [1] 123.7839

> sd(preis_mehl)

## [1] 0.2749471

> var_koef(preis_auto)

## [1] 0.004129157

> var_koef(preis_mehl)

## [1] 0.08739578
```

3.2.4 Linear transformations of location and dispersion measures

Transformations (or functions) play an important role in statistical analyses. First, we can use transformations to model a relationship between different variables. Secondly, one may improve the accuracy of statistical analyses using transformed data. The simplest transformation is the linear function $f(x) = a + bx$ with parameters a and b.

Definition 3.2.16. *For any numbers $a, b \in \mathbb{R}$ we call the map or the function*

$$y = a + bx, \qquad x \in \mathbb{R},$$

a linear transformation. An application of the linear transformation $y = a + bx$ to the metric data $x_1, \ldots, x_n$ yields the linearly transformed data set $y_1, \ldots, y_n$ with

$$y_i = a + bx_i, \qquad i \in \{1, \ldots, n\}.$$

Many location and dispersion measures of linearly transformed data can be computed using the corresponding location and dispersion measures of the original data and the given linear transformation. Namely, the following lemma holds.

Lemma 3.2.17. *Let $a, b \in \mathbb{R}$ and $y = a + bx$ be the linear transformation. Further, let $y_1, \dots, y_n$ be the linear transformed data set obtained from the metric data set $x_1, \dots, x_n$. Then*

(i) $\tilde{y}_{0.5} = a + b\tilde{x}_{0.5}$,

(ii) $\overline{y}_n = a + b\overline{x}_n$,

(iii) $s^2_{y,n} = b^2 s^2_{x,n}$,

(iv) $s_{y,n} = |b| \cdot s_{x,n}$,

where $s^2_{y,n}, s^2_{x,n}, s_{y,n}, s_{x,n}$ denote the empirical variance and standard deviation of the corresponding data set, respectively.

Proof Exercise.

If one wants to compare observations of different data sets then it is often useful to standardize the given observations.

Definition 3.2.18. *Suppose we have observations $x_1, \dots, x_n$ with positive empirical standard deviation $s_{x,n} > 0$ and empirical mean $\overline{x}_n$. The linear transformation*

$$z_i = \frac{x_i - \overline{x}_n}{s_{x,n}}, \qquad i \in \{1, \dots, n\},$$

is called the standardization. The transformed data elements $z_1, \dots, z_n$ are called standardized. If one applies the linear transformation

$$y_i = x_i - \overline{x}_n, \qquad i \in \{1, \dots, n\},$$

then we speak of centering the data.

After standardization, different data sets have the same empirical mean 0 and the same empirical standard deviation 1.

3.3 Classified data and histograms

In this section, the histogram will be presented as a method for graphical demonstration of quantitative data. To draw a histogram, classification of the original data set is first necessary, i.e the observed data must be grouped into classes.

3.3.1 Classification

The goal of the data classification is to split the observations $x_1, \ldots, x_n$ into classes $K_1, \ldots, K_M$. The resulting data is then called classified. This data allocation enables a meaningful graphical demonstration. We assume that observations belong to an interval $[a, b]$, where $a = -\infty$ (then the interval is open to the left) and / or $b = \infty$ (then the interval is open to the right) are also permitted. We consider non-overlapping intervals and each interval is now assigned to one unique class K_i. Therefore, each observation x_j belongs to one unique class K_i. Let us formalize the partition of the interval $[a, b]$ in the following definition.

Definition 3.3.1. *Let $a < b$ with $a, b \in \mathbb{R} \cup \{-\infty, \infty\}$. A segmentation of $[a, b]$ into intervals*

$$K_1 = [v_0, v_1], K_2 = (v_1, v_2], \ldots, K_M = (v_{M-1}, v_M]$$

with $a = v_0 < v_1 < \cdots < v_{M-1} < v_M = b$ is called partition of $[a, b]$. For $a = -\infty$ and $b = \infty$, one chooses the intervals $K_1 = (-\infty, v_1]$ and $K_M = (v_{M-1}, \infty)$, respectively.

The classified data can now be considered as observations of an ordinal variable. In order to construct a histogram using these classified data, the absolute (relative) frequencies of the individual classes should be determined.

Definition 3.3.2. *Let $x_1, \ldots, x_n \in [a, b]$ be observations of a variable X with values $u_1, \ldots, u_m$. Further, let $u_1, \ldots, u_m$ have absolute frequencies $n_1, \ldots, n_m$ and relative frequencies $f_1, \ldots, f_m$. The absolute frequencies of the classes $K_1, \ldots, K_m$ are given by*

$$n(K_j) = \sum_{k \in \{1,\ldots,m\}: u_k \in K_j} n_k, \qquad j \in \{1, \ldots, M\}.$$

The relative frequencies of the classes $K_1, \ldots, K_m$ are given by

$$f(K_j) = \frac{n(K_j)}{n} = \sum_{k \in \{1,\ldots,m\}: u_k \in K_j} f_k, \qquad j \in \{1, \ldots, M\}.$$

For the classified data, absolute and relative frequencies can thus be calculated as in Section 3.1.1.

Example 3.3.3. We consider again the data set `exec.pay` from the package `UsingR`. A simple way to create classified data in R is to use the function `cut()`. Suppose that we want to create the partition

$$\begin{aligned} &K_1 = [0, 10], K_2 = (10, 20], \ldots, K_{10} = (90, 100], K_{11} = (100, 200], K_{12} = (200, 500], \\ &K_{13} = (500, 1000], K_{14} = (1000, 2510] \end{aligned}$$

of the observation range $[0, 2510]$ (in 10000\$). From the help (`?cut`) to the function `cut()`, we deduce that the smallest value is included in K_1 if we set the option `include.lowest` to `TRUE`. Note that K_1 is by default an open interval on the left.

```
> exec.pay_kl <- cut(exec.pay, breaks = c(seq(0, 100, by = 10), 200,
+     500, 1000, 2510), include.lowest = TRUE)
```

Using the function `table()`, we can now determine absolute frequencies of all classes, which have just been calculated.

```
> table(exec.pay_kl)

## exec.pay_kl
##           [0,10]          (10,20]          (20,30]
##               29               47               41
##          (30,40]          (40,50]          (50,60]
##               29               16                7
##          (60,70]          (70,80]          (80,90]
##                3                3                3
##         (90,100]        (100,200]        (200,500]
##                3               14                1
##      (500,1e+03] (1e+03,2.51e+03]
##                1                2
```

We can also plot a bar chart of the classified data.

```
> ggplot(data.frame(classified_data = exec.pay_kl), aes(x = classified_data)) +
+   geom_bar(fill = "darkorchid3")
```

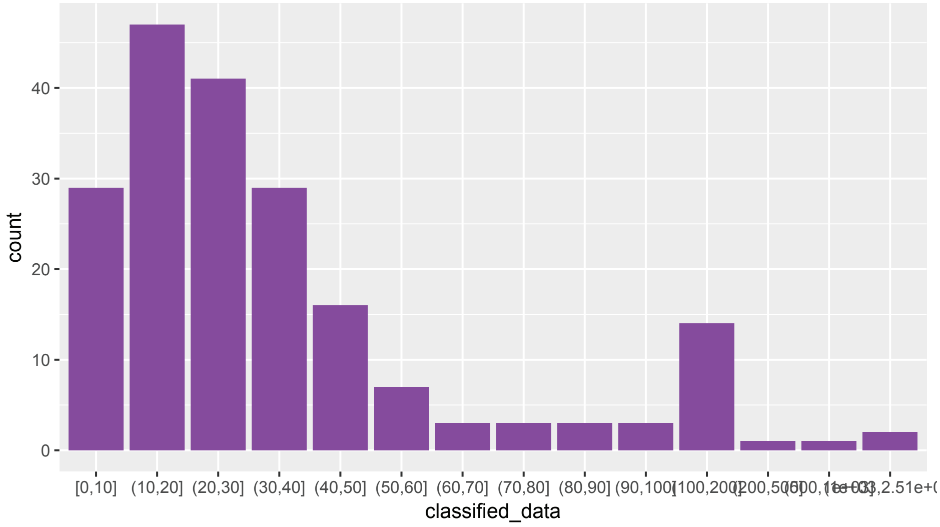


Figure 3.6: Bar plot of the classified data on severance payments.

3.3.2 Histograms

As it is already explained in the last section, many different observations for metric variables are usually present. Therefore, a graphical representation of such original data with a bar, pie or dot chart is often inappropriate. Similarly, a frequency table does not lead to a desired compressed presentation of data. The classification of the data introduced in Section 3.3.1 provides a solution for graphical representation of metric data. The frequencies of the individual classes can alternatively be used for graphical representation. In particular, a histogram can be created with the classified data.

Definition 3.3.4. *Let $x_1, \dots, x_n \in [a, b]$ be a metric data set and*

$$K_1 = [v_0, v_1], K_2 = (v_1, v_2], \dots, K_M = (v_{M-1}, v_M]$$

be a partition of $[a, b]$ with interval widths (bin width or bandwidth) $b_1 = v_1 - v_0, \dots, b_M = v_M - v_{M-1}$. Furthermore, let $f(K_1), \dots, f(K_M)$ be the relative frequencies. A diagram is called a histogram when it is constructed in the following manner. On the horizontal axis, the class boundaries v_0 to v_M are labeled. Further, a rectangle is drawn above each interval K_j in such a way that its width is equal to b_j and its height h_j is calculated according to the formula

$$h_j = \frac{f(K_j)}{b_j}.$$

In a histogram, the area of a rectangle is (by default) equal to the relative frequency of the corresponding class:

$$\text{Rectangle area of class } K_j = b_j h_j = f(K_j).$$

For a better visualization, a proportionality factor $c > 0$ can also be used for the height of the rectangles. In R, the default setting is $c = n$. Therefore, the area of rectangles is now equal to the absolute frequency and not to the relative one. No matter which proportionality factor is used, the histogram is in any case an area diagram, i.e. the histogram visualizes the frequencies proportionally to the area. In contrast, in the bar chart, the height and the area of the bars are proportional to the frequency since the bars all have the same width.

In the case of $c = 1$, the total area of the histogram rectangles is equal to one. If $c = n$ then the total area of the histogram rectangles is equal to the number of observations.

Example 3.3.5. Let us consider again the waiting times between two eruptions of Old Faithful. First we would like to draw a histogram for the following partition:

$$K_1 = [40, 60], K_2 = (60, 80], K_3 = [80, 100].$$

To do this, we do not have to create a new classified data set in R . We can just use the function `hist()` (default graphical function) with the option `breaks` to specify the interval boundaries.

```
> hist(faithful$waiting, breaks = c(40, 60, 80, 100))
```

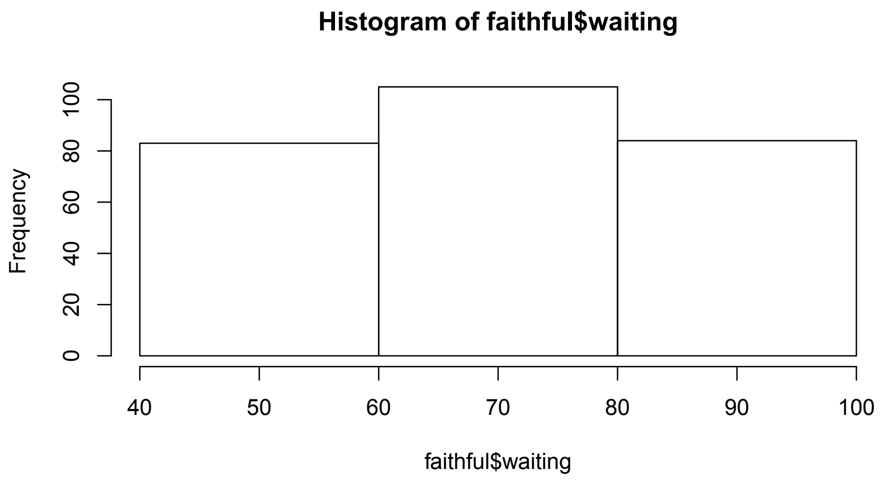


Figure 3.7: Histogram of waiting times separated into three classes

As already noted, we see the standard use of $c = n$, since absolute frequencies are displayed. Furthermore, it can be seen from the graph that the class with the highest absolute frequency is K_2 and the frequency distribution is rather symmetrical. Since the class division is very coarse, we next want to create a histogram with the default setting for `breaks`.

```
> h <- hist(faithful$waiting)
```

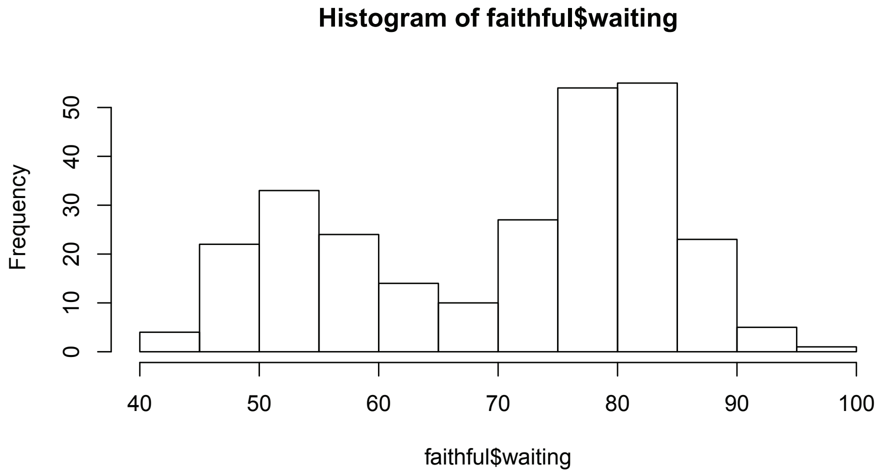


Figure 3.8: Histogram of waiting times divided into 12 classes.

```
> length(h$breaks)  #Number of interval boundaries

## [1] 13

> h$breaks

##  [1]  40  45  50  55  60  65  70  75  80  85  90  95 100
```

By default, R uses the *Sturges* algorithm to calculate the interval boundaries as seen in the help (`?hist`). For the considered data set, the algorithm provides 12 classes. In Figure 3.8 , one also recognizes that the frequency distribution is probably not symmetric, since two local maximums are clearly visible. This observation cannot be made in Figure 3.7 due to the use of a too small number of classes. A rule of thumb for determining an appropriate number of classes is that the number of classes should not exceed the square root of the number of observations. In our case, we should not use more than 16 classes (number of observations is 272), but also not significantly less. The 12 classes from the default setting are certainly fine, but we still want to generate a histogram with 16 classes. For this, we compute boundaries of 16 equally long intervals.

```
> (cut <- seq(min(faithful$waiting), max(faithful$waiting),
+             by = (max(faithful$waiting) - min(faithful$waiting)) / 16))

##  [1] 43.0000 46.3125 49.6250 52.9375 56.2500 59.5625 62.8750
##  [8] 66.1875 69.5000 72.8125 76.1250 79.4375 82.7500 86.0625
## [15] 89.3750 92.6875 96.0000

> hist(faithful$waiting, breaks = cut)
```

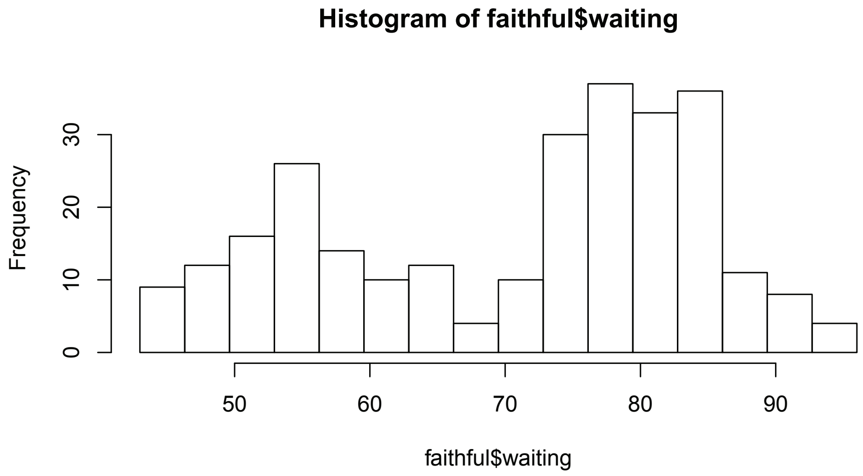


Figure 3.9: Histogram of the waiting times separated into 16 classes.

In the following example we illustrate further options of the function `hist()`.

Example 3.3.6. Let us consider the data sets OBP (UsingR), which contains on base percentage of the 438 baseball players in 2002 major league baseball season. The on base percentage, OBP, is a measure of how often a player gets on base.

```
> par(mfrow = c(2, 2))
> hist(OBP, prob = TRUE, col = 5)
> hist(OBP, prob = TRUE, breaks = 4, col = colors()[433])
> hist(OBP, freq = FALSE, breaks = "scott", col = "palevioletred2")
> hist(OBP, freq = TRUE, breaks = seq(0.2, 0.7, by = 0.05))
```

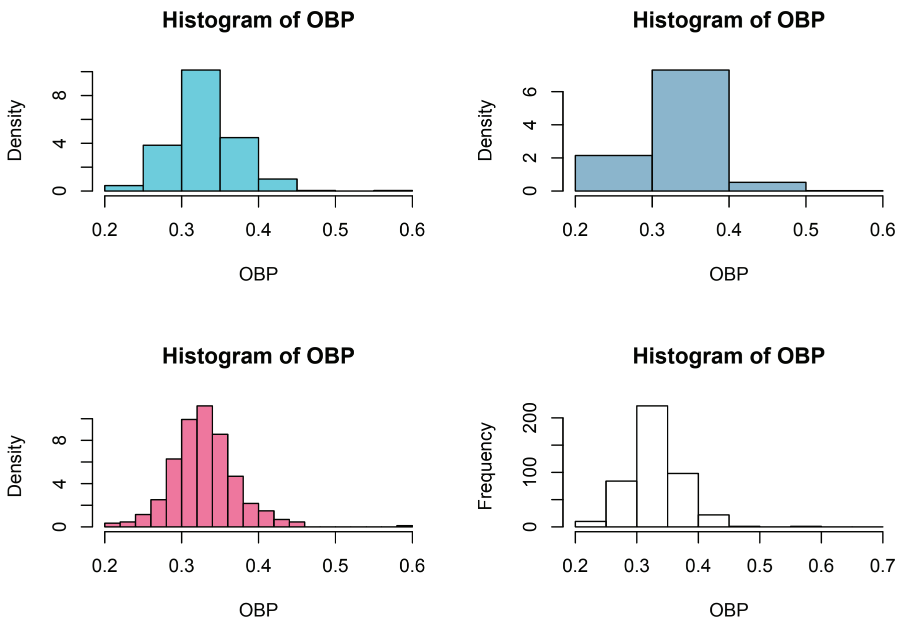


Figure 3.10: Four histograms of the extttOBP data, each with different options of the function exttthist().

Now we know a method to represent graphically the frequency distribution of a (classified) metric data set. Since frequency distributions of various data sets can have different forms, frequency distributions have been systemized depending on their shape. They are first differentiated with respect to the number of their local maxima. If there is only one local maximum (it is also a global maximum), we talk about a *unimodal* frequency distribution. On the other hand, i.e. in the presence of several local maxima, one speaks of a *multimodal* frequency distribution. If exactly two maxima are present then the multimodal frequency distribution is further specified as a *bimodal* one.

```
> par(mfrow = c(1, 3))
> hist(OBP, main = "unimodal", prob = TRUE, ylim = c(0, 12))
> hist(faithful$waiting, main = "bimodal", prob = TRUE)
> library(MASS)
> cut <- seq(from = 5000, to = 35000, by = 2500)
> hist(galaxies, main = "multimodal", breaks = cut, prob = TRUE, ylim = c(0,
+     0.00015))
```

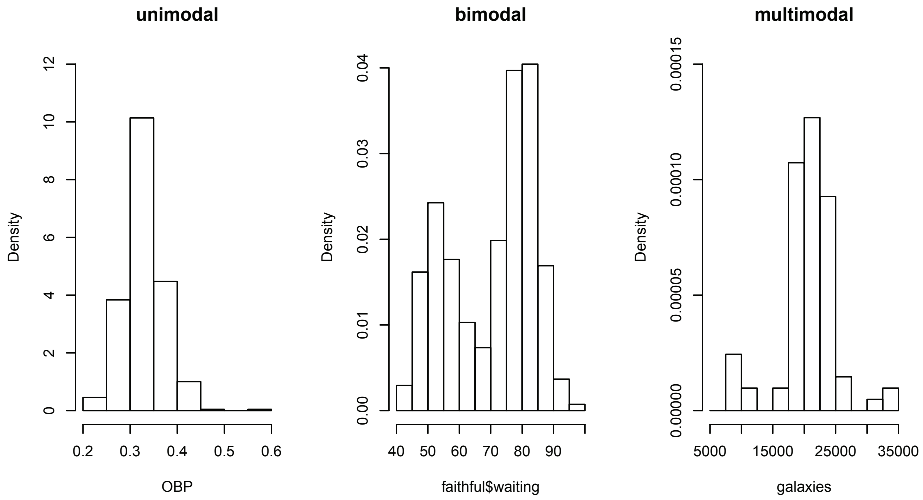


Figure 3.11: Histogram of a unimodal, bimodal and multimodal frequency distribution.

When dealing with a multimodal distribution, one has to carefully interpret location measures. These are, in fact, designed to describe the center (the (!) center) of the data. For bimodal frequency distributions, it is well possible that the observations are densely concentrated around two peaks in the histogram, which are then located on the left and right of a chosen location measure, let say e.g. empirical mean. Therefore, location measures for bimodal frequency distributions will not usually describe the center of a considered data set.

Unimodal distributions can further be distinguished with regard to their shape. If the frequency distribution is approximately mirror-symmetric to some perpendicular straight line, then one speaks of a *symmetric* distribution. If, on the other hand, a large part of the observations are concentrated on the left or right side of the observations range then the distribution is referred to as *skewed*. It is called *positively skewed* (or right skewed), if the frequencies have their maximum on the left side and fall off to the right. In the opposite case, the distribution is *negatively skewed* (or left skewed).

We consider the two data sets `OBP` and `cfb` from the package `UsingR`. The `cfb` data set is a comprehensive survey of consumer finances sponsored by the United States Federal Reserve. Variable `VEHIC` contains a value of all vehicles (includes cars, motor homes, RVs, airplanes, boats) of an interviewed family.

```
> par(mfrow = c(1, 2))
> hist(OBP[-which.max(OBP)], xlab = "OBP without Barry Bonds",
+      main = "Symmetric Distribution")
```

```
> hist(cfb$VEHIC, xlab = "Equity from vehicles",
+      main = "Positively Skewed Distribution")
```

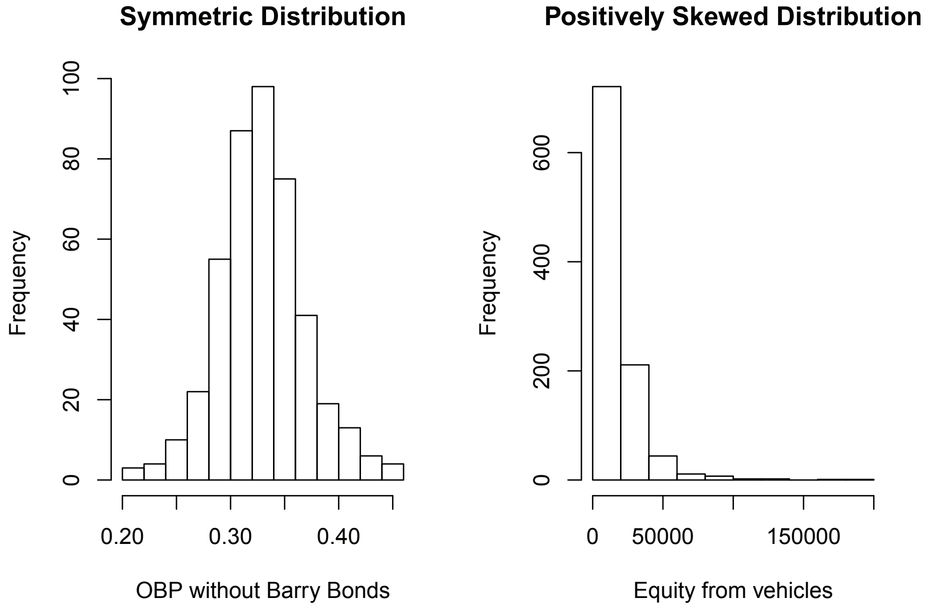


Figure 3.12: Histogram of a symmetric and a positively skewed frequency distribution.

In Example 3.2.7, we have already seen that the empirical mean is sensitive to outliers. The median behaves more robust here. In the case of a positively skewed distribution, we also have few large observations. Therefore, we want to use the function `summary` to compare these two location measures.

```
> summary(cfb$VEHIC)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0    3875   11000   15398   21300  188000
```

Thus, we observe a large difference between the empirical mean and the median. Therefore, a significant difference between empirical mean and median is an indication for a skewed distribution. In a histogram, it can then be identified whether the distribution is really skewed or whether there is an approximately symmetric distribution with a few very large (or very small) outliers.

In general, we can establish the following empirical rules for the location measures $\overline{x}_n, \tilde{x}_{0.5}$ and x_{mod} (provided we deal with classified data).

- For symmetric unimodal distributions, it holds that

  $$\overline{x}_n \approx \tilde{x}_{0.5} \approx x_{mod},$$

  where $\approx$ means approximately equal.

- For positively skewed unimodal distributions, it holds that

$$\overline{x}_n > \tilde{x}_{0.5} > x_{mod}.$$

- For negatively skewed unimodal distributions, it holds that

$$\overline{x}_n < \tilde{x}_{0.5} < x_{mod}.$$

3.4 Boxplots

The box plot offers a further possibility to visualize univariate metric data sets.

Definition 3.4.1. *A boxplot visualizes the five-point summary of a data set $x_1, \dots, x_n$ consisting of the minimum, the median of the lower half of the data set $\tilde{x}_{0.5,u}$ (also called lower hinge), the median, the median of the upper half of the data set $\tilde{x}_{0.5,o}$ (called also upper hinge) and the maximum. The lower and upper hinge define a box, which contains the median as a separating line. In addition, whiskers are drawn from the hinges as straight lines. The theoretical length of the whiskers is calculated as* 1.5 *times the length of the box (approximately the interquartile distance). However, each of the whiskers is shortened to the observation directly preceding the theoretical end of the whisker. Thus, the actual lower and upper end of the whiskers are given by*

$$x_{uG} = \min_{i \in \{1,\dots,n\}} \{x_i \mid x_i \geq \tilde{x}_{0.5,u} - 1.5(\tilde{x}_{0.5,o} - \tilde{x}_{0.5,u})\}$$
$$x_{oG} = \max_{i \in \{1,\dots,n\}} \{x_i \mid x_i \leq \tilde{x}_{0.5,o} + 1.5(\tilde{x}_{0.5,o} - \tilde{x}_{0.5,u})\}.$$

Observations, which are outside of the box and whiskers, are considered as outliers and they are displayed as dots. This means that the minimal and maximal observations are either marked by a dot or indicate the end of the whiskers.

Definition 3.4.1 refers to the implementation in R . If sample size n is even then the lower and the upper half of a data set can trivially be determined by considering the sequence of the order statistics. The first $n/2$ order statistics constitute now the lower half of the data set and the remaining part is the upper half of the data set. If the sample size is odd then the median is a part of the data. In this case, we extend the data set by one observation, which is equal to the median. Now the extended data set has an even sample size and we can proceed as it is described above.

In other definitions, the median of the lower (upper) half of the data set may be replaced by the lower (upper) quartile. In the case of an even sample size, the median of the lower (upper) half of the data set always matches the lower (upper) quartile. For an odd sample size, the cases $n = 4k + 3$ and $n = 4k + 1$ must be distinguished. In the first case, the lower (upper) quartile and the median of the lower (upper) half of the data set differ, while in the second case they are the same again.

Example 3.4.2. Let us consider the data set `alltime.movies` (`UsingR`), which contains the sales figures of the 79 most successful films in the US up to 2003. Gross receipts are recorded in the variable `Gross`

```
> b <- boxplot(alltime.movies$Gross)
> f <- fivenum(alltime.movies$Gross)
> text(rep(1.37, 5), f, labels = c("minimum", "median of the lower half",
+      "median", "median of the upper half", "maximum"), cex = 0.7)
```

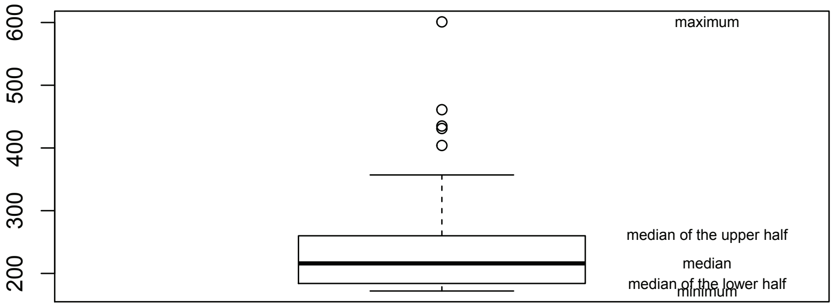


Figure 3.13: Boxplot of the sales figures of the 79 most successful films in the USA (up to 2003) in a vertical orientation.

Alternatively, we can also align the box plot horizontally. To do this, we have to set the `horizontal` option to `TRUE` in the `boxplot()` function. In addition, we want to use the function `rug()` to plot the individually observed expressions on the x axis.

```
> boxplot(alltime.movies$Gross, horizontal = TRUE, col = "red")
> rug(alltime.movies$Gross)
```

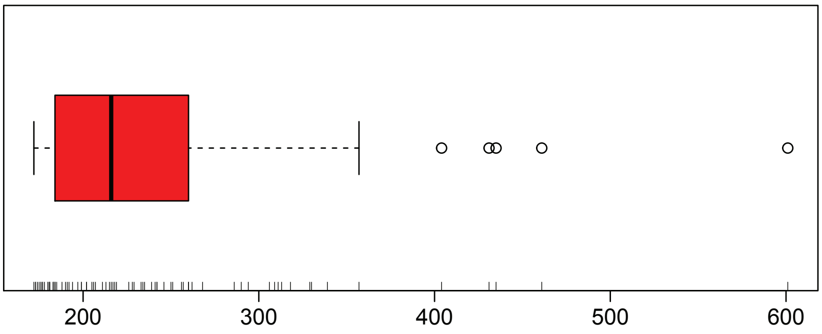


Figure 3.14: Boxplot of the sales figures of the 79 most successful films in the USA (up to 2003) in a horizontal orientation.

Now we also want to know which films are outliers.

```
> Filmtitel <- rownames(alltime.movies)
> Filmtitel[which(alltime.movies$Gross %in% b$out)]

## [1] "Titanic                                  "
## [2] "Star Wars                                "
## [3] "E.T.                                     "
## [4] "Star Wars: The Phantom Menace            "
## [5] "Spider-Man                               "
```

R summary

In this section, we have learned first descriptive methods to analyze univariate data. For an overview of the frequency distribution of categorical data, see `table()`. For graphical analysis of categorical data the functions `barplot()`, `pie()` and texttdotchart() are suitable. For metric data the empirical mean `mean()` and the median `median()` (also for ordinal data) were introduced as a measures of location. As measures of dispersion, we have discussed the span `range()`, the interquartile distance `IQR()` and the empirical variance `var()`. To define the interquartile distance, we extended the concept of the median to general p quantiles (`quantile()`). Using the `summary()` function, you get the minimum, first quartile, median, empirical mean, third quartile, and maximum of a data set. In the end, it was explained which key figures can be read from a box plot (`boxplot()`) and what can be said about the form of the distribution. For the graphical representation of the frequency distribution, we also know the histogram `hist()`.

Chapter 4

Bivariate Data

In Chapter 3, we have always considered only one variable, which characterizes one particular property or feature of considered statistical units. This lead us to a univariate data and a univariate statistical analysis. However, two and more variables can often be associated with statistical units. This challenges us to statistically analyze the interaction or the dependence between different variables. In this chapter we restrict ourselves to a statistical analysis of two variables, whose observations constitute a bivariate data set.

It should be noted that one can also obtain a bivariate (multivariate) data set by considering observations of one variable for two (several) groups of statistical units. One example might be daily returns of two different stock indices.

4.1 Dependence measures

Similarly to the sections on location and dispersion measures, dependence measures are introduced in the next section. These measures quantify the dependence between two variables. Depending on variable type, different dependence measures can be introduced. We assume that both variables are of the same type. If one wants to measure the dependence between two variables of different types then one should consider dependence measures, which are suitable for the variable with lowest measurement level.

4.1.1 Nominal variables

Since values of nominal variables are not ordered, monotonic or functional relationships between two nominal variables cannot be investigated. As previously for location measures, the absolute and relative frequencies can only be used to describe the dependence between two nominal variables. Therefore, dependence measures for nominal variables can only use the information contained in the joint frequency distribution. In order to clearly distinguish between the dependence of nominal variables and the dependence of variables with higher measurement level, we will speak of association measures when we consider nominal variables. Before we define such association measures, we must first consider how the joint frequency distribution can be represented.

Contingency tables

Let us consider two nominal variables X and Y, which constitute the bivariate variable (X, Y). The values of the bivariate variable (X, Y) are specified as pairs (u_i, v_j), $i \in \{1, \ldots, r\}, j \in \{1, \ldots, s\}$. The absolute frequency of the value (u_i, v_j) is denoted by n_{ij}. These frequencies are then represented in a contingency table of dimension $r \times s$. Each value in the contingency table represents the number of times a particular combination of a bivariate variable outcomes occurred. The rows of the contingency table correspond to the values of variable X and its columns correspond to the values of variable Y. Thus, the absolute frequency of the bivariate outcome (u_i, v_j) is found in the i-th row of the j-th column. The frequency distribution of X is often specified in the last column and the frequency distribution of Y in the last line.

Example 4.1.1. Consider the data set `grades` (`UsingR`). It includes students' grades in two consecutive math tests. We would guess that there is a certain association (dependence) between these two variables, namely between the student's grade in the first math test and the student's grade in the second math test.

```
> library(UsingR)
> t <- table(grades$prev, grades$grade)
> t_rand <- addmargins(t)  #Adds the marginal distributions of X and Y
> ftable(t_rand)

##          A    A-   B+   B    B-   C+   C    D    F   Sum
##
##   A      15    3    1    4    0    0    3    2    0  28
##   A-      3    1    1    0    0    0    0    0    0   5
##   B+      0    2    2    1    2    0    0    1    1   9
##   B       0    1    1    4    3    1    3    0    2  15
##   B-      0    1    0    2    0    0    1    0    0   4
##   C+      1    1    0    0    0    0    1    0    0   3
##   C       1    0    0    1    1    3    5    9    7  27
##   D       0    0    0    1    0    0    4    3    1   9
##   F       1    0    0    1    1    1    3    4   11  22
##  Sum     21    9    5   14    7    5   20   19   22 122
```

The above table provides one representation of the joint frequency distribution. Note that the rows correspond to the grades in the first test. A better overview on the joint frequency distribution can probably be obtained graphically. To do this we use the function `mosaicplot()`.

```
> mosaicplot(table(grades), color = TRUE)
```

We can observe a concentration of the frequencies in the upper left and lower right corner of the graph as well as of the table. This is an indication for some association (dependence) between the current and the previous math grade. Note that rectangle areas in the mosaic plot are proportional to the corresponding numbers of observations in the contingency table.

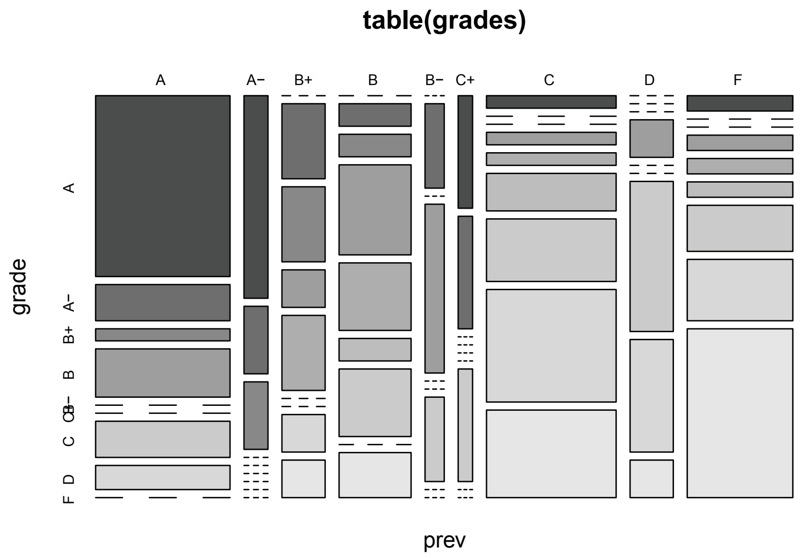


Figure 4.1: Mosaic plot of the joint frequency distribution of the variables extttprev (first test) and extttgrade (second test).

The marginal frequency of the i-th row is denoted by $n_{i\bullet}$ and it is the sum of the frequencies associated with the values $(u_i, v_1), \ldots, (u_i, v_s)$, i.e.

$$n_{i\bullet} = n_{i1} + \cdots + n_{is}.$$

It is called the marginal frequency of the i-th row since we are only interested in the number of bivariate observations, when variable X takes the particular value corresponding to the row i and variable Y can take any value from $v_1, \ldots, v_s$.

In Example 4.1.1 the marginal frequency of the second row is the frequency for the grade `A-` in the first exam

$$n_{2\bullet} = 3 + 1 + 1 + 0 + 0 + 0 + 0 + 0 + 0 = 5.$$

The marginal frequency of column j is denoted by $n_{\bullet j} = n_{1j} + \cdots + n_{rj}$.

Example 4.1.2. If you want to determine the marginal frequencies in R for a given contingency table then you can use the function `margin.table()`.

```
> margin.table(table(grades$prev, grades$grade), 1)

##
##  A    A-   B+   B    B-   C+   C    D    F
##   28    5    9   15    4    3   27    9   22

> margin.table(table(grades$prev, grades$grade), 2)
```

```
##
##   A    A-   B+   B    B-   C+   C    D    F
##   21    9    5   14    7    5   20   19   22
```

One can observe a similar frequency distribution for both math tests. The last command `margin.table(table(prev, grade), 2)` sums all entries in each column of the contingency table. This result is also provided by

```
> apply(table(grades$prev, grades$grade), MARGIN = 2, FUN = sum)

##   A    A-   B+   B    B-   C+   C    D    F
##   21    9    5   14    7    5   20   19   22
```

The R function `apply()` applies (therefore it is called *apply*) a pre-specified function (in this case `sum()`) to margins of a data set (in this case 2 columns). The function `apply()` (as well as its variations `sapply()`, `tapply()`) will reappear later throughout the course.

Data representation in a contingency table is useful, if considered variables can take only few values. Otherwise the overall view is lost quickly. This means, in particular, that a representation of unclassified (ungrouped) bivariate metric variables in a contingency table will make no sense.

Similarly to the joint absolute frequency distribution, the joint relative frequency distribution can also be represented in a contingency table. The relative frequency of the value (u_i, v_j) is defined by $f_{ij} = \frac{n_{ij}}{n}$. Analogously to the absolute marginal distributions, the following relations for marginal relative frequencies $f_{i\bullet}$ and $f_{\bullet j}$ hold:

$$\begin{aligned} f_{i\bullet} &= f_{i1} + \cdots + f_{is}, \qquad & i \in \{1, \ldots, r\} \\ f_{\bullet j} &= f_{1j} + \cdots + f_{rj}, \qquad & j \in \{1, \ldots, s\}. \end{aligned}$$

Example 4.1.3. For the two math grades from the data set `grades`, the following contingency table of the relative frequencies is obtained:

```
> pt <- prop.table(table(grades$prev, grades$grade))
> pt_rand <- addmargins(pt)
> options(digits = 1)  # Show shortened number of decimal places
> ftable(pt_rand)

##          A     A-    B+    B     B-    C+    C     D     F     Sum
##
##   A     0.123 0.025 0.008 0.033 0.000 0.000 0.025 0.016 0.000 0.230
##   A-    0.025 0.008 0.008 0.000 0.000 0.000 0.000 0.000 0.000 0.041
##   B+    0.000 0.016 0.016 0.008 0.016 0.000 0.000 0.008 0.008 0.074
##   B     0.000 0.008 0.008 0.033 0.025 0.008 0.025 0.000 0.016 0.123
##   B-    0.000 0.008 0.000 0.016 0.000 0.000 0.008 0.000 0.000 0.033
##   C+    0.008 0.008 0.000 0.000 0.000 0.000 0.008 0.000 0.000 0.025
##   C     0.008 0.000 0.000 0.008 0.008 0.025 0.041 0.074 0.057 0.221
##   D     0.000 0.000 0.000 0.008 0.000 0.000 0.033 0.025 0.008 0.074
##   F     0.008 0.000 0.000 0.008 0.008 0.008 0.025 0.033 0.090 0.180
## Sum     0.172 0.074 0.041 0.115 0.057 0.041 0.164 0.156 0.180 1.000
```

Conditional frequencies

In the case of two variables X and Y, the following question is certainly of interest. Which frequency distribution does variable Y have under condition that variable X takes the value u_i? To answer this question, one relates the absolute frequencies $n_{i1}, \dots, n_{is}$ of the bivariate values $(u_i, v_1), \dots, (u_i, v_s)$ with the marginal frequency $n_{i\bullet}$ of the value u_i. In this case one speaks of a conditional frequency. Since the relative frequencies are only a scaling of the absolute frequencies by the factor n, they can also be used to define the conditional frequency.

Definition 4.1.4. *(i) Let $n_{i\bullet} > 0$. Then the quotient*

$$f_{Y=v_j|X=u_i} = \frac{n_{ij}}{n_{i\bullet}} = \frac{f_{ij}}{f_{i\bullet}}, \qquad i \in \{1, \dots, r\},$$

is called the conditional frequency (of $Y = v_j$ conditioned on $X = u_i$). The respective frequency distribution

$$f_{Y=v_1|X=u_i}, \dots, f_{Y=v_s|X=u_i}$$

is called the conditional frequency distribution (of Y conditioned on $X = u_i$).

(ii) Let $n_{\bullet j} > 0$. Then

$$f_{X=u_i|Y=v_j} = \frac{n_{ij}}{n_{\bullet j}} = \frac{f_{ij}}{f_{\bullet j}}, \qquad j \in \{1, \dots, s\},$$

is called the conditional frequency (of $X = u_i$ conditioned on $Y = v_j$). The respective frequency distribution

$$f_{X=u_1|Y=v_j}, \dots, f_{X=u_r|Y=v_j}$$

is called the conditional frequency distribution (of X conditioned on $Y = v_j$).

Remark 4.1.5. In Definition 4.1.4 we require $n_{i\bullet} > 0, n_{\bullet j} > 0$. Since we are still in the descriptive part of statistics, this requirement makes sense. If we do not observe the values u_i or v_j then we cannot say how often the values of one variable occur under the condition $X = u_i$ or $Y = v_j$.

Example 4.1.6. Consider again the data set `grades`. Now we are interested in the frequency distribution of grades in the current exam given the grade of the previous one. If we denote the variable `prev` as X and `grade` as Y, then we are interested in

$$f_{Y=v_j|X=u_i} = \frac{n_{ij}}{n_{i\bullet}}, \qquad u_i, v_j \in \{A, A-, \dots, D, F\}.$$

Conditional frequency distributions can be obtained in R again with the function `prop.table()`. However, the option `margin` must specify the margin (or variable) on which we condition. In this example, we want to condition on the rows (that is, `margin=1`).

```
> options(digits = 1)
> pt <- prop.table(table(grades$prev, grades$grade), margin = 1)
> pt_rand <- addmargins(pt)
> ftable(pt_rand)
```

```
##        A    A-   B+   B    B-   C+   C    D    F    Sum
##
##  A    0.54 0.11 0.04 0.14 0.00 0.00 0.11 0.07 0.00 1.00
##  A-   0.60 0.20 0.20 0.00 0.00 0.00 0.00 0.00 0.00 1.00
##  B+   0.00 0.22 0.22 0.11 0.22 0.00 0.00 0.11 0.11 1.00
##  B    0.00 0.07 0.07 0.27 0.20 0.07 0.20 0.00 0.13 1.00
##  B-   0.00 0.25 0.00 0.50 0.00 0.00 0.25 0.00 0.00 1.00
##  C+   0.33 0.33 0.00 0.00 0.00 0.00 0.33 0.00 0.00 1.00
##  C    0.04 0.00 0.00 0.04 0.04 0.11 0.19 0.33 0.26 1.00
##  D    0.00 0.00 0.00 0.11 0.00 0.00 0.44 0.33 0.11 1.00
##  F    0.05 0.00 0.00 0.05 0.05 0.05 0.14 0.18 0.50 1.00
## Sum   1.55 1.18 0.52 1.21 0.50 0.22 1.66 1.03 1.11 9.00
```

Using the function addmargins(), we have added the marginal distributions as before. We see that the conditional frequencies add up to 1 in each row. Therefore, each row i represents a relative frequency distribution, i.e.

$$\sum_{j=1}^{s} f_{Y=v_j|X=u_i} = 1, \qquad s = 9, u_i \in \{A, A-, \ldots, D, F\}.$$

Since we have conditioned on the rows, the conditional frequencies in a chosen column do not form a relative frequency distribution, that means that in general it holds that

$$\sum_{i=1}^{r} f_{Y=v_j|X=u_i} \neq 1, \qquad r = 9, v_j \in \{A, A-, \ldots, D, F\}.$$

The same result could have been generated again with apply(). To do this, one should define a function that calculates the conditional frequencies. This function can then be applied to a contingency table using apply().

```
> cond_frequencies <- function(x) {
+     x/sum(x)
+ }
> t(apply(table(grades$prev, grades$grade), 1, cond_frequencies))

##
##         A    A-   B+   B    B-   C+   C    D    F
##    A   0.54 0.11 0.04 0.14 0.00 0.00  0.1 0.07  0.0
##    A-  0.60 0.20 0.20 0.00 0.00 0.00  0.0 0.00  0.0
##    B+  0.00 0.22 0.22 0.11 0.22 0.00  0.0 0.11  0.1
##    B   0.00 0.07 0.07 0.27 0.20 0.07  0.2 0.00  0.1
##    B-  0.00 0.25 0.00 0.50 0.00 0.00  0.2 0.00  0.0
##    C+  0.33 0.33 0.00 0.00 0.00 0.00  0.3 0.00  0.0
##    C   0.04 0.00 0.00 0.04 0.04 0.11  0.2 0.33  0.3
##    D   0.00 0.00 0.00 0.11 0.00 0.00  0.4 0.33  0.1
##    F   0.05 0.00 0.00 0.05 0.05 0.05  0.1 0.18  0.5

> options(digits = 7)
```

To get the same results as before with `prop.table()`, we have also transposed the resulting matrix of `apply()` with the transpose function `t()`. This is necessary because `apply()` sorts the result columnwise.

χ^2-quantity

So far we have only introduced a description of the joint frequency distribution of two nominal variables. In this section, an association measure will be introduced, which quantifies the dependence between two nominal variables.

Definition 4.1.7. *Let (X, Y) be a bivariate nominal variable with values (u_i, v_j), $i \in \{1, \ldots, r\}, j \in \{1, \ldots, s\}$ and n_{ij} be corresponding absolute frequencies .*

(i) For positive marginal frequencies $n_{i\bullet}, n_{\bullet j}$ the χ^2-quantity is defined by

$$\chi^2 = \sum_{i=1}^{r} \sum_{j=1}^{s} \frac{(n_{ij} - \nu_{ij})^2}{\nu_{ij}}$$

with $\nu_{ij} := \frac{n_{i\bullet} n_{\bullet j}}{n}, i \in \{1, \ldots, r\}, j \in \{1, \ldots, s\}$.

(ii) If $n_{i\bullet} = 0$ for some i or $n_{\bullet j} = 0$ for some j then the χ^2-quantity is defined by

$$\chi^2 = \sum_{i,j:\nu_{ij}>0} \frac{(n_{ij} - \nu_{ij})^2}{\nu_{ij}}.$$

In Case (ii) of Definition 4.1.7, some values of the bivariate nominal variable are not observed in a given data set. Therefore, we cannot make any conclusion about these non-observed values and they are therefore neglected. In the following, it is always assumed that the contingency table contains neither zero rows nor zero columns.

In order to understand the χ^2-quantity, we need the concept of empirical independence.

Definition 4.1.8. *The variables X and Y are called empirically independent if*

$$\frac{n_{ij}}{n} = \frac{n_{i\bullet}}{n} \frac{n_{\bullet j}}{n} \qquad \text{for all } i \in \{1, \ldots, r\} \text{ and for all } j \in \{1, \ldots, s\}.$$

or equivalently

$$f_{ij} = f_{i\bullet} f_{\bullet j} \qquad \text{for all } i \in \{1, \ldots, r\} \text{ and for all } j \in \{1, \ldots, s\}.$$

The joint frequency distribution of two empirically independent variables is thus completely determined by the two marginal frequency distributions. The following example explains the main idea of Definition 4.1.8.

Assume that there is no relationship between the variables X and Y. This would imply that the conditional frequency distribution of X given $Y = v_j$ equals to the (unconditional) frequency distribution of X (X is not influenced by values of Y). This means that for each $j \in \{1, \ldots, s\}$ the equality

$$f_{X=u_i|Y=v_j} = \frac{n_{ij}}{n_{\bullet j}} \overset{*}{=} \frac{n_{i\bullet}}{n} = f_{i\bullet} \qquad \text{for all } i \in \{1, \ldots, r\}$$

holds or equivalently

$$f_{ij} = \frac{n_{ij}}{n} \overset{*}{=} \frac{n_{i\bullet} n_{\bullet j}}{n^2} = f_{i\bullet} f_{\bullet j} \qquad \text{for all } i \in \{1, \dots, r\} \text{ and } j \in \{1, \dots, s\}. \tag{4.1.1}$$

Equation (4.1.1) is precisely the definition of empirical independence. Thus, the empirical independence is a necessary condition for the fact that there is no dependence between two variables X and Y. For reasons of symmetry, the argument that we have just conducted is also valid for influence of X on Y.

With the concept of empirical independence, the definition of the χ^2-quantity is now clear. It compares the actually observed frequencies n_{ij} with the frequencies ν_{ij} that would occur in the case of empirical independence

$$\nu_{ij} = n f_{ij} = n f_{i\bullet} f_{\bullet j} = \frac{n_{i\bullet} n_{\bullet j}}{n}.$$

In the following Lemma, we summarize some properties of the χ^2-quantity.

Lemma 4.1.9. *Let χ^2 be the χ^2-quantity based on a $r \times s$ contingency table of the absolute frequencies of two variables X and Y.*

(i)

$$\chi^2 \geq 0$$

(ii)

$$\chi^2 = 0 \Leftrightarrow X \textit{ and } Y \textit{ are empirically independent.}$$

(iii)

$$\chi^2 \leq n \cdot (\min\{r, s\} - 1).$$

(iv) The equality $\chi^2 = n \cdot (\min\{r, s\} - 1)$ holds if and only if one of the following conditions is satisfied:

(a) We have $r < s$ and in each column the frequencies are concentrated in exactly one field/entry of the contingency table.

(b) We have $r = s$ and in each row and in each column the frequencies are concentrated in exactly one field/entry of the contingency table

(c) We have $r > s$ and in each row the frequencies are concentrated in exactly one field/entry of the contingency table.

Proof Statement (i) follows directly from the definition.

Statement (ii) follows from Definitions 4.1.7 and 4.1.8 as well as Equation (4.1.1).

For the proof of (iii) (and (iv)), we first derive another representation of the χ^2-quantity.

$$\chi^2 = \sum_{i=1}^{r} \sum_{j=1}^{s} \frac{(n_{ij} - \nu_{ij})^2}{\nu_{ij}} = \sum_{i=1}^{r} \sum_{j=1}^{s} \frac{(n_{ij} - \frac{n_{i\bullet} n_{\bullet j}}{n})^2}{\frac{n_{i\bullet} n_{\bullet j}}{n}} = n \left(\sum_{i=1}^{r} \sum_{j=1}^{s} \frac{n_{ij}^2 - 2 n_{ij} \frac{n_{i\bullet} n_{\bullet j}}{n} + (\frac{n_{i\bullet} n_{\bullet j}}{n})^2}{n_{i\bullet} n_{\bullet j}} \right)$$

$$= n \left(\sum_{i=1}^{r} \sum_{j=1}^{s} \frac{n_{ij}^2}{n_{i\bullet} n_{\bullet j}} - 2 \underbrace{\sum_{i=1}^{r} \sum_{j=1}^{s} \frac{n_{ij}}{n}}_{=1} + \underbrace{\sum_{i=1}^{r} \sum_{j=1}^{s} \frac{n_{i\bullet} n_{\bullet j}}{n^2}}_{=1} \right) = n \left(\sum_{i=1}^{r} \sum_{j=1}^{s} \frac{n_{ij}^2}{n_{i\bullet} n_{\bullet j}} \right) - n.$$

With this representation it remains to show that

$$\sum_{i=1}^{r}\sum_{j=1}^{s}\frac{n_{ij}^2}{n_{i\bullet}n_{\bullet j}} \le \min\{r,s\}.$$

Because of $\frac{n_{ij}}{n_{\bullet j}} = \frac{n_{ij}}{\sum_{i=1}^{r} n_{ij}} \le 1$ the double sum above can be estimated by

$$\sum_{i=1}^{r}\sum_{j=1}^{s}\frac{n_{ij}^2}{n_{i\bullet}n_{\bullet j}} \le \sum_{i=1}^{r}\sum_{j=1}^{s}\frac{n_{ij}}{n_{i\bullet}} = \sum_{i=1}^{r} 1 = r\,.$$

Similarly, one obtains the inequality $\sum_{i=1}^{r}\sum_{j=1}^{s}\frac{n_{ij}^2}{n_{i\bullet}n_{\bullet j}} \le s$, which leads to the conclusion. Statement (iii) is proven.

Now we prove Statement (iv). Suppose $r \le s$ (the opposite case can be shown analogously). From the proof of (iii) it is clear that

$$\chi^2 = n \cdot (\min\{r,s\} - 1) \Leftrightarrow \sum_{i=1}^{r}\sum_{j=1}^{s}\frac{n_{ij}^2}{n_{i\bullet}n_{\bullet j}} = r\,.$$

For all $i \in \{1,\dots,r\}, j \in \{1,\dots,s\}$ with $n_{ij} > 0$ it holds that

$$\sum_{i=1}^{r}\sum_{j=1}^{s}\frac{n_{ij}^2}{n_{i\bullet}n_{\bullet j}} = r \Leftrightarrow \frac{n_{ij}}{n_{\bullet j}} = 1\,. \tag{4.1.2}$$

By assumption we have $n_{\bullet j} > 0$. Therefore, there is, in general, at least one i with $n_{ij} > 0$ for each j. For a fixed j, the equation

$$n_{ij} = n_{\bullet j} = n_{1j} + \cdots + n_{rj}$$

follows from (4.1.2).

Thus, all $n_{kj}, k \in \{1,\dots,r\} \setminus \{i\}$, are equal to zero. This means that there is only one non-zero entry in each column. This has the value $n_{\bullet j}$. □

Example 4.1.10. Let us consider the theoretical example of a 4 × 5 contingency table.

	v_1	v_2	v_3	v_4	v_5	$n_{i\bullet}$
u_1	0	5	0	0	0	5
u_2	0	0	0	8	0	8
u_3	1	0	3	0	0	4
u_4	0	0	0	0	7	7
$n_{\bullet j}$	1	5	3	8	7	24

For the χ^2-quantity we obtain

$$\begin{aligned}
\chi^2 &= \sum_{i=1}^{4}\sum_{j=1}^{5}\frac{(n_{ij}-\nu_{ij})^2}{\nu_{ij}} \\
&= \underbrace{\frac{(\frac{5}{24})^2}{\frac{5}{24}} + \frac{(5-\frac{25}{24})^2}{\frac{25}{24}} + \frac{(\frac{15}{24})^2}{\frac{15}{24}} + \frac{(\frac{40}{24})^2}{\frac{40}{24}} + \frac{(\frac{35}{24})^2}{\frac{35}{24}}}_{i=1} + \underbrace{\frac{8}{24} + \frac{40}{24} + \frac{24}{24} + \frac{(8-\frac{64}{24})^2}{\frac{64}{24}} + \frac{56}{24}}_{i=2} \\
&\quad + \underbrace{\frac{(1-\frac{4}{24})^2}{\frac{4}{24}} + \frac{20}{24} + \frac{(3-\frac{12}{24})^2}{\frac{12}{24}} + \frac{32}{24} + \frac{28}{24}}_{i=3} + \underbrace{\frac{7}{24} + \frac{35}{24} + \frac{21}{24} + \frac{56}{24} + \frac{(7-\frac{49}{24})^2}{\frac{49}{24}}}_{i=4} \\
&= \frac{95}{24} + \frac{(\frac{95}{24})^2}{\frac{25}{24}} + \frac{128}{24} + \frac{(\frac{128}{24})^2}{\frac{64}{24}} + \frac{(\frac{20}{24})^2}{\frac{4}{24}} + \frac{80}{24} + \frac{(\frac{60}{24})^2}{\frac{12}{24}} + \frac{119}{24} + \frac{(\frac{119}{24})^2}{\frac{49}{24}} \\
&= \frac{422}{24} + \frac{361}{24} + \frac{256}{24} + \frac{100}{24} + \frac{300}{24} + \frac{289}{24} \\
&= \frac{1728}{24} = 72 = 3 \cdot 24
\end{aligned}$$

Remark 4.1.11. From Lemma 4.1.9, we recognize that for small values of the χ^2-quantity, there is a weak association (dependence) between the respective variables. If, on the other hand, the value is close to the upper limit, a strong dependence between the valriables can be assumed. If the upper bound is attained, one speaks of complete dependence. In the case of $r \leq s$, the assumption of the upper limit means, that in the corresponding contingency table in every column all observations are concentrated in one field. This means that once the value of variable Y is known, one can predict the value of X. The same holds for the case $s \leq r$. The concept of complete dependency is therefore justified.

Example 4.1.12. We look again at the data set `grades`. Given the concentration of the entries in the contingency table along the diagonals, we already have the impression that there is a rather strong relation (dependence) between the two variables. Now let's see if this conjecture is confirmed by a high value of χ^2-quantity. The χ^2- quantity can be computed with the function `chisq.test()`. This function provides many other additional quantities, which we will discuss in the section on inductive statistics.

```
> chisq.test(table(grades))$statistic

## Warning in chisq.test(table(grades)):  Chi-squared approximation may be incorrect

## X-squared
##  137.2646

> # We can ignore the warning. It refers to the hypothesis test,
> # which was also conducted.
```

In order to judge this value, we need the upper bound of the χ^2-quantity for the corresponding contingency table.

```
> (n <- sum(table(grades)))

## [1] 122

> # upper bound:
> n * (min(c(nlevels(grades$prev), nlevels(grades$grade)) - 1))

## [1] 976
```

The upper bound is 976. The χ^2-quantity is thus closer to 0 than to the upper limit. Nevertheless, it is difficult to judge how strong the dependence between the two variables is. As a guess, we would rather say that the variables are only weakly associated.

Finally, we'll make sure that the value computed by the `chisq.test()` function is actually the same as the χ^2-quantity from Definition 4.1.7. We consider the following simple example of a 3×3 contingency table of two completely dependent variables.

```
> (contingency_tabel <- as.table(matrix(c(3, 0, 0, 0, 0, 4, 0, 1, 0),
+     nrow = 3)))

##   A B C
## A 3 0 0
## B 0 0 1
## C 0 4 0

> chisq.test(contingency_tabel)$statistic

## Warning in chisq.test(contingency_tabel):  Chi-squared approximation may be incorrect

## X-squared
##        16

> 8 * (min(c(3, 3) - 1))

## [1] 16
```

From Lemma 4.1.9, we know that in the case of complete dependence, the upper bound is attained. It is equal to 16 and thus the χ^2-quantity is 16. This value is also obtained with the function `chisq.test()`.

Contingency coefficient

As we saw in Section 4.1.1, the χ^2-quantity can be used to describe the association (dependence) between two nominal variables. Unfortunately, this association measure has one disadvantage since an interpretation of its values is difficult. The reason for this is its unrestricted value range. The upper bound (for a fixed number n of observations) of the χ^2-quantity contains the observation number n and grows with a growing number of observations. Specifically, this means that the upper bound must always be computed and the computed value of the χ^2-quantity should be compared with this computed upper bound.

However, using the χ^2-quantity, it is possible to construct association measures, which are independent of the sample size n and are therefore easier to interpret. The following definition introduces the contingency coefficient and its corrected form. The corrected contingency coefficient, as we shall see, has an advantage that its range of values is independent of the sample size and of the number of different possible values r and s of the considered variables X and Y.

Definition 4.1.13. *Let $(x_1, y_1), \ldots, (x_n, y_n)$ be the observed values of the bivariate nominal variable (X, Y), which can take possible values (u_i, v_j), $i \in \{1, \ldots, r\}, j \in \{1, \ldots, s\}$. Let χ^2 be the χ^2-quantity based on the contingency table of these two variables. The contingency coefficient C (according to Pearson) is defined by*

$$C := \sqrt{\frac{\chi^2}{n + \chi^2}}.$$

The corrected contingency coefficient C_* (according to Pearson) is defined by

$$C_* := C \cdot \sqrt{\frac{\min\{r, s\}}{\min\{r, s\} - 1}}.$$

The value range of the contingency coefficient is no longer dependent on n, but is still dependent on r and s. It holds that

$$0 \leq C \leq \sqrt{\frac{\min\{r, s\} - 1}{\min\{r, s\}}} < 1.$$

A comparison of two data sets with contingency tables of different dimensions is therefore not possible with the contingency coefficient according to Pearson.

By normalizing the contingency coefficient, we get the corrected contingency coefficient C_* and it holds

$$0 \leq C_* \leq 1.$$

Thus, this dependence measure can now be used to compare two data sets with contingency tables of different dimensions.

The behavior of the corrected contingency coefficient is analogous to the χ^2-quantity. For values close to the lower limit, there is an evidence for rather weak dependency and for values close to the upper limit 1, a strong dependence between nominal variables is present. By normalizing the value range of C, an assessment of association between two nominal variables is easier than in the case of the χ^2-quantity. We look at the data set `grades` again and calculate the (corrected) contingency coefficient.

Example 4.1.14. There is no function in the basic version of R for calculating the (corrected) contingency coefficient. Therefore we write our own function for this.

```
> contingency_coefficient <- function(table, corrected = TRUE) {
+     options(warn = -1)  #turn of error warnings

+     chi.sq <- chisq.test(table, correct = FALSE)$statistic
+     # For 2x2 tables, an unwanted correction would be done. We do not
+     # want this, so correct=FALSE

+     options(warn = 0)  #turn on error warnings
+     n <- sum(table)
+     C <- sqrt(chi.sq/(n + chi.sq))
+     names(C) <- "Contingency coefficient"
+     r <- dim(table)[1]
+     s <- dim(table)[2]
+     if (corrected == TRUE) {
+         C <- C * sqrt(min(c(r, s))/(min(c(r, s)) - 1))
+         names(C) <- "corrected contingency coefficient"
+     }
+     C  # return the last call
+ }
```

```
> contingency_coefficient(table(grades), corrected = FALSE)

## Contingency coefficient
##               0.7276251

> contingency_coefficient(table(grades))

## corrected contingency coefficient
##                         0.7717629
```

We get a corrected contingency coefficient C_* of 0.77. This value is significantly closer to the upper limit 1 than to the lower limit 0, confirming our initial guess.

The above-mentioned dependence measures can only describe the strength of dependence. No statement can be made with respect to a dependence/relationship direction: Variable X increases when variable Y decreases. For this purpose, data with a higher measurement level is required and more advanced dependence measures such as the Spearman rank correlation coefficient or the Bravais-Pearson correlation coefficient should be considered. These correlation coefficients are introduced in the next sections. Since the Spearman correlation coefficient can be deduced from the Bravais-Pearson coefficient, we will first deal with metric variables and then with ordinal ones. It should be noted that the Spearman correlation coefficient is already applicable to ordinal data and the Spearman correlation coefficient requires metric variables.

4.1.2 Metric variables

In this section, we deal with a bivariate metric variable (X, Y). Let $(x_1, y_1), \dots, (x_n, y_n)$ be observations of this bivariate variable (X, Y). Thus, we deal here with data sets consisting of n bivariate vectors (x_i, y_i), $i = 1, \dots, n$. Before we consider the measurement of the relationship (dependence) between X and Y, let us briefly discuss the graphical representation of bivariate metric data.

Scatter plot and qq-plot

A scatter plot is a graphical representation of a bivariate data set with metric components. The observations $(x_1, y_1), \ldots, (x_n, y_n)$ are entered into a two-dimensional coordinate system as points, where x_i and y_i determine the position of the point (x_i, y_i) on x- and y-axis, respectively. From the scatter plot, one can already draw first conclusions about the possible relationship between the considered two variables. In R , a scatter plot can be created simply with the function `plot()`.

Example 4.1.15. Consider the first 150 values in the data set `homedata` (`UsingR`). It contains estimated values of houses in New Jersey in 1970 and in 2000.

```
> y1970_150 <- homedata$y1970[1:150]
> y2000_150 <- homedata$y2000[1:150]
> plot(y1970_150, y2000_150)
```

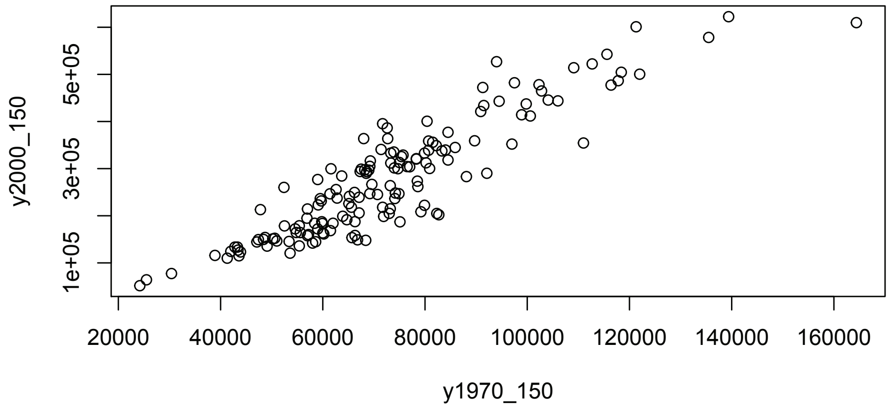


Figure 4.2: Scatter plot of estimated values of houses in 1970 (extitx-axis) and in 2000 (extity-axis).

One clearly observes a linear dependence with an increasing scattering (variability, dispersion) for more expensive houses. A summary of important key figures is obtained using the function `summary()`.

```
> summary(y1970_150)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   24200   59025   69450   72903   81350  164400

> summary(y2000_150)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   51600  178575  257900  276659  340400  622400
```

```
> summary(y2000_150/y1970_150)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.132   2.985   3.737   3.674   4.296   5.605
```

In 1970, the empirical mean and the median are still much closer together than in 2000. The last output also shows that the values of some houses have almost grown by 600% in 30 years.

A scatter plot gives information about a possible relationship between the two components of a bivariate variable (X, Y). Another question which is also of interest in the analysis of bivariate variables is whether they have similar (frequency) distributions. We already know that the p quantiles describe the location of variable values. The p quantiles of two univariate data sets are located relatively similarly, if the distances between successive quantiles are similar in both data sets, i.e. they are proportional with some factor. Thereby, absolute position of quantiles is not important. If p quantiles of two univariate data sets are located relatively similarly then they have a similar frequency distribution. To check similarity of frequency distributions of X and Y, one just plots the points $(x_{(1)}, y_{(1)})$, $(x_{(2)}, y_{(2)}), \ldots, (x_{(n)}, y_{(n)})$, where $x_{(1)} \leq \cdots \leq x_{(n)}$ and $y_{(1)} \leq \cdots \leq y_{(n)}$ are order statistics computed from $(x_1, y_1), \ldots, (x_n, y_n)$. The resulting diagram is called a *QQ-Plot*. Now, if X and Y have similar frequency distributions then the points on the QQ-plot are located along a straight line that is defined by the points $(\tilde{x}_{0.25}, \tilde{y}_{0.25})$ and $(\tilde{x}_{0.75}, \tilde{y}_{0.75})$. In R , you create a QQ-plot using the function `qqplot()`.

Example 4.1.16. We obtained the following qq-plot (see Figure 4.3) for the two univariate data sets from `homedata`.

```
> qqplot(homedata$y1970, homedata$y2000)
> qx <- quantile(homedata$y1970, prob = c(0.25, 0.75))
> qy <- quantile(homedata$y2000, prob = c(0.25, 0.75))
> b <- (qy[2] - qy[1])/(qx[2] - qx[1])  # slope
> a <- qy[1] - b * qx[1]  # intercept
> # draws a straight line with intercept a and slope b
> abline(a = a, b = b, lty = 2)
> # mark the .25 and .9 quantile of both data sets
> abline(v = quantile(homedata$y1970, prob = 0.25), col = "red")
> abline(v = quantile(homedata$y1970, prob = 0.9), col = "red")
> abline(h = quantile(homedata$y2000, prob = 0.25), col = "blue")
> abline(h = quantile(homedata$y2000, prob = 0.9), col = "blue")
```

The points on the QQ-plot are located pretty well along the straight line between the 0.25 and the 0.9 quantile. The frequency distributions are thus relatively similar only in this interval. As we have already seen from the *summary*, there are huge differences outside of this area in Figure 4.3, such that we cannot talk about similar frequency distributions here.

Empirical covariance

At first we define a (dependence) measure with an unlimited value range (cf. Section 4.1.1). We will then normalize this measure so that we obtain a measure with a value range

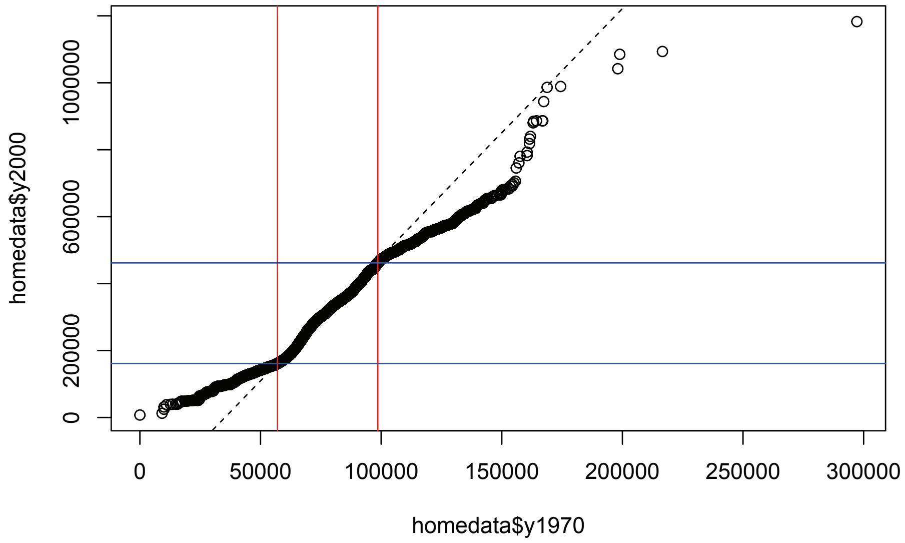


Figure 4.3: QQ-plot of estimated values of houses in 1970 (extitx-axis) and in 2000 (extity-axis).

independent of the underlying data set. A similar procedure has already been applied to the χ^2-quantity to define the corrected contingency coefficient. Let us start with the definition of the empirical covariance.

Definition 4.1.17. *Consider observations $(x_1, y_1), \dots, (x_n, y_n)$ of a bivariate metric variable (X, Y). The empirical covariance of the variables X and Y is then defined by*

$$s_{xy} = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \overline{x}_n)(y_i - \overline{y}_n),$$

where $\overline{x}_n$ and $\overline{y}_n$ are the empirical means of $x_1, \dots, x_n$ and $y_1, \dots, y_n$, respectively.

Before we continue with a further modification of the empirical covariance, let us first summarize some important properties of the empirical covariance in the following lemma.

Lemma 4.1.18. *Let $(x_1, y_1), \dots, (x_n, y_n)$ be observations (a data set) of a bivariate metric variable (X, Y).*

(i) Consider the bivariate variable (X, X) and the corresponding data set $(x_1, x_1), \dots, (x_n, x_n)$. Then it holds that

$$s_{xx} = s_x^2.$$

Thus, the empirical covariance of (X, X) is the empirical variance of X.

(ii) For the linearly transformed data set $(x_1^*, y_1^*), \dots, (x_n^*, y_n^*)$ with

$$x_i^* = a + bx_i, \quad a, b \in \mathbb{R}, \qquad \text{and} \qquad y_i^* = c + dy_i, \quad c, d \in \mathbb{R},$$

for $i \in \{1, \dots, n\}$, we have

$$s_{x^*y^*} = bd \cdot s_{xy}.$$

(iii) For the empirical covariance s_{xy}, it holds

$$s_{xy} = \frac{1}{n-1} \sum_{i=1}^{n} x_i y_i - \frac{n}{n-1} \overline{x}_n \cdot \overline{y}_n.$$

Proof (i) We obtain the following empirical covariance for the bivariate variable (X, X):

$$s_{xx} = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \overline{x}_n)(x_i - \overline{x}_n) = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \overline{x}_n)^2 = s_x^2 \,.$$

(ii) For the linearly transformed bivariate variable (X^*, Y^*), we have

$$\begin{aligned} s_{x^*y^*} &= \frac{1}{n-1} \sum_{i=1}^{n} (x_i^* - \overline{x}_n^*)(y_i^* - \overline{y}_n^*) \\ &= \frac{1}{n-1} \sum_{i=1}^{n} \left(a + bx_i - \frac{1}{n} \sum_{j=1}^{n} (a + bx_j) \right) \left(c + dy_i - \frac{1}{n} \sum_{j=1}^{n} (c + dy_j) \right) \\ &= \frac{1}{n-1} \sum_{i=1}^{n} \left(bx_i - b\frac{1}{n} \sum_{j=1}^{n} x_j \right) \left(dy_i - d\frac{1}{n} \sum_{j=1}^{n} y_j \right) \\ &= \frac{bd}{n-1} \sum_{i=1}^{n} (x_i - \overline{x}_n)\,(y_i - \overline{y}_n) = bd \cdot s_{xy} \,. \end{aligned}$$

(iii) For the empirical covariance of the bivariate variable (X, Y), it holds that

$$\begin{aligned} s_{xy} &= \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \overline{x}_n)(y_i - \overline{y}_n) = \frac{1}{n-1} \sum_{i=1}^{n} (x_i y_i - x_i \overline{y}_n - \overline{x}_n y_i + \overline{x}_n \overline{y}_n) \\ &= \frac{1}{n-1} \sum_{i=1}^{n} x_i y_i - \overline{y}_n \left(\frac{n}{n-1} \cdot \frac{1}{n} \sum_{i=1}^{n} x_i \right) - \overline{x}_n \left(\frac{n}{n-1} \cdot \frac{1}{n} \sum_{i=1}^{n} y_i \right) + \frac{1}{n-1} \sum_{i=1}^{n} (\overline{x}_n \overline{y}_n) \\ &= \frac{1}{n-1} \sum_{i=1}^{n} x_i y_i - \frac{n}{n-1} \overline{y}_n \overline{x}_n - \frac{n}{n-1} \overline{x}_n \overline{y}_n + \frac{n}{n-1} \overline{x}_n \overline{y}_n \\ &= \frac{1}{n-1} \sum_{i=1}^{n} x_i y_i - \frac{n}{n-1} \overline{x}_n \overline{y}_n. \end{aligned}$$

□

Example 4.1.19. Let us again consider the data set `homedata` from Example 4.1.15. Now we consider only the first 50 measurements and compute at first the empirical covariance ”by hand” and afterwards with the function `cov()`.

```
> y1970_new <- homedata$y1970[1:50]
> y2000_new <- homedata$y2000[1:50]
> (emp_cov <- sum((y1970_new - mean(y1970_new)) *
+                   (y2000_new - mean(y2000_new))) / 49)
```

```
## [1] 2383533363

> cov(y1970_new,y2000_new)

## [1] 2383533363
```

We observe the very large empirical covariance. However, it is not surprising since the underlying scale of the covairance is equal to ”price^2”

We also want to use this example to justify, why the empirical covariance is a suitable indicator for linear dependence. Therefore,we consider the scatterplot of the first 50 observations and mark observations 1, 14, 28 and 38 with different colours.

```
> plot(y1970_new, y2000_new, xlab = "price in 1970",
+      ylab = "price in 2000")
> abline(v = mean(y1970_new))
> abline(h = mean(y2000_new))
> points(mean(y1970_new), mean(y2000_new), cex = 1.5, col= "red")
> text(mean(y1970_new) + 5000, mean(y2000_new) + 15000,
+      labels = expression(paste("(",bar(x), ",", bar(y), ")")), col = "red")
> text(y1970_new[c(1, 14, 28, 38)], y2000_new[c(1, 14, 28, 38)],
+      labels = c(1, 14, 28, 38), cex = 0.8, col = "blue", pos = c(4, 2, 4, 4))
```

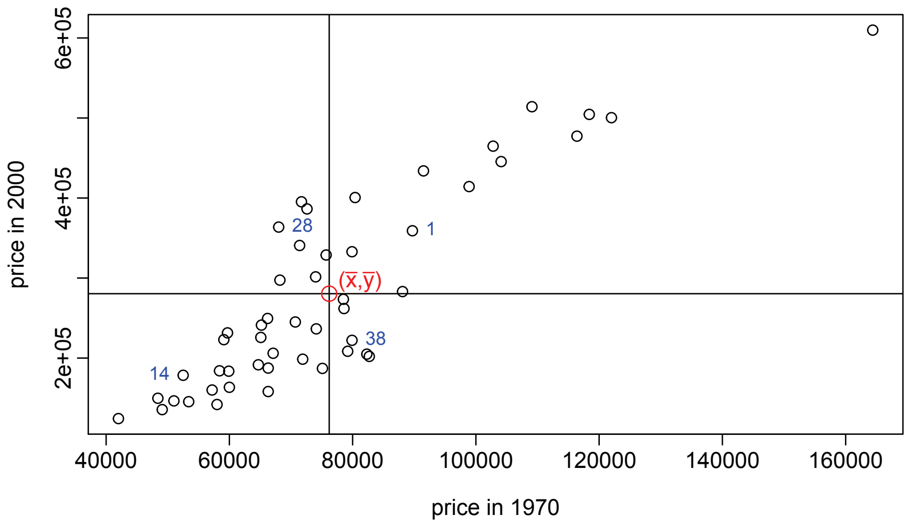


Figure 4.4: Scatter plot of estimated values of 50 houses in 1970 (extitx-axis) and in 2000 (extity-axis).

Consider a new coordinate system with centre $(\overline{x},\overline{y})$. Then the data points in the first quadrant (e.g. 1) and the third quadrant (e.g. 14) have a positive contribution to the empirical covariance. This means that large values of X coincide with large values of Y and the other way around, i.e. small values of X coincide with small values of Y. Data points in quadrants two (e.g. 38) and four (e.g. 28) contribute negatively to the empirical covariance.

Hence, a positive value of the empirical covariance indicates a concordant behaviour of the variables, i.e. growth in X indicates growth in Y. A discordant behaviour is indicated by negative covariances. Values close to zero do not give any hint on dependence between X and Y since a non-linear dependence between the two variables can still exist in this case.

The empirical covariance only indicates the direction of a linear dependence by means of its sign since it depends on the scale of the considered variables. Hence, no statement on the strength of the dependence is possible. We therefore introduce the Bravais-Pearson-correlation coefficient (short: `correlation coefficient` or more simple, just `correlation`) in the next step, as it can give an answer on the strength of a linear dependence.

Bravais-Pearson-correlation coefficient

Definition 4.1.20. *Let $(x_1, y_1), \dots, (x_n, y_n)$ be observations (a data set) of a bivariate metric variable (X, Y) and s_{xy} be the empirical covariance of X and Y. Further, let $s_x > 0$ and $s_y > 0$ be the empirical standard deviation of $x_1, \dots, x_n$ and $y_1, \dots, y_n$, respectively. The Bravais-Pearson-correlation coefficient is then defined by*

$$r_{xy} = \frac{s_{xy}}{s_x s_y} = \frac{\sum_{i=1}^n (x_i - \overline{x}_n)(y_i - \overline{y}_n)}{\sqrt{\sum_{i=1}^n (x_i - \overline{x}_n)^2} \sqrt{\sum_{i=1}^n (y_i - \overline{y}_n)^2}},$$

where $\overline{x}_n$ resp. $\overline{y}_n$ are the empirical means of $x_1, \dots, x_n$ resp. $y_1, \dots, y_n$.

Before continuing with an example, let us first consider the following properties of the correlation coefficient.

Lemma 4.1.21. *Let $(x_1, y_1), \dots, (x_n, y_n)$ be observations of a bivariate metric variable (X, Y) with the corresponding correlation coefficient r_{xy}.*

(i) Consider the bivariate variable (X, X) and the corresponding data set $(x_1, x_1), \dots, (x_n, x_n)$. Then it holds that

$$r_{xx} = 1.$$

(ii) For the linearly transformed data set $(x_1^*, y_1^*), \dots, (x_n^*, y_n^*)$ with

$$x_i^* = a + bx_i, \quad a \in \mathbb{R}, b \neq 0, \qquad \text{and} \qquad y_i^* = c + dy_i, \quad c \in \mathbb{R}, d \neq 0,$$

for $i \in \{1, \dots, n\}$, we have

$$r_{x^*y^*} = \frac{bd}{|bd|} \cdot r_{xy} = \begin{cases} r_{xy}, & \text{if } bd > 0 \\ -r_{xy}, & \text{if } bd < 0 \end{cases}.$$

(iii) For the correlation coefficient r_{xy}, it holds that

$$-1 \le r_{xy} \le 1.$$

The upper and lower limit of the correlation range are attained in the following cases:

- $r_{xy} = 1 \Leftrightarrow$ *There exist $b > 0$ and $a \in \mathbb{R}$ with $y_i = a + bx_i, i \in \{1, \dots, n\}$.*
- $r_{xy} = -1 \Leftrightarrow$ *There exist $b < 0$ and $a \in \mathbb{R}$ with $y_i = a + bx_i, i \in \{1, \dots, n\}$.*

Proof (i) For the variable (X, X), we obtain

$$r_{xx} = \frac{\sum_{i=1}^{n}(x_i - \overline{x}_n)(x_i - \overline{x}_n)}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x}_n)^2}\sqrt{\sum_{i=1}^{n}(x_i - \overline{x}_n)^2}} = \frac{\sum_{i=1}^{n}(x_i - \overline{x}_n)^2}{\sum_{i=1}^{n}(x_i - \overline{x}_n)^2} = 1\,.$$

(ii) For the linearly transformed bivariate variable (X^*, Y^*) we calculate the following correlation coefficient

$$\begin{aligned} r_{x^*,y^*} &= \frac{\sum_{i=1}^{n}(x_i^* - \overline{x}_n^*)(y_i^* - \overline{y}_n^*)}{\sqrt{\sum_{i=1}^{n}(x_i^* - \overline{x}_n^*)^2}\sqrt{\sum_{i=1}^{n}(y_i^* - \overline{y}_n^*)^2}} \\ &= \frac{\sum_{i=1}^{n}\left(a + bx_i - \frac{1}{n}\sum_{j=1}^{n}(a + bx_j)\right)\left(c + dy_i - \frac{1}{n}\sum_{j=1}^{n}(c + dy_j)\right)}{\sqrt{\sum_{i=1}^{n}\left(a + bx_i - \frac{1}{n}\sum_{j=1}^{n}(a + bx_j)\right)^2}\sqrt{\sum_{i=1}^{n}\left(c + dy_i - \frac{1}{n}\sum_{j=1}^{n}(c + dy_j)\right)^2}} \\ &= \frac{bd\sum_{i=1}^{n}(x_i - \overline{x}_n)(y_i - \overline{y}_n)}{\sqrt{b^2\sum_{i=1}^{n}(x_i - \overline{x}_n)^2}\sqrt{d^2\sum_{i=1}^{n}(y_i - \overline{y}_n)^2}} \\ &= \frac{bd}{|bd|}r_{xy} \end{aligned}$$

(iii) We only prove the direction "$\Leftarrow$". Let $b \neq 0, a \in \mathbb{R}$ and $y_i = a + bx_i, i \in \{1, \ldots, n\}$. Since $x_i = c + dx_i$ with $c = 0$ and $d = 1$, we get from (i) and (ii) that

$$r_{xy} = \frac{b}{|b|}r_{xx} = \frac{b}{|b|} = \begin{cases} 1, & \text{if } b > 0 \\ -1, & \text{if } b < 0 \end{cases}.$$

□

Thus, the correlation coefficient is a measure for the strength of linear dependence. A completely negative/positive linear dependence between two variables holds at the boundaries $-1/1$ of the value range. They are not linearly independent for a correlation coefficient of zero since a non-linear dependence can still apply. Based on the correlation coefficient, one expresses the types of dependence in the following way:

Definition 4.1.22. *Two metric variables X and Y are called*

- *positively correlated, if* $r_{xy} > 0$,
- *uncorrelated, if* $r_{xy} = 0$,
- *negatively correlated, if* $r_{xy} < 0$.

Furthermore, we denote X and Y to have a

- *weak correlation, if* $0 < |r_{xy}| < 0.5$,
- *strong correlation, if* $0.8 \leq |r_{xy}| \leq 1$.

The above distinction in weak and strong correlation is, however, not "strict" and generally accepted. One can find different thresholds for weak correlation depending on research areas.

To illustrate the properties introduced in Lemma 4.1.21, consider the following example.

Example 4.1.23. We generate two empirically independent data sets using the funtion `rnorm()`. We will draw more attention to these kind of functions Chapter 6.

```
> x <- rnorm(100)  # generates a data set of length 100
> y1 <- rnorm(100)
> y2 <- 0.7 * x + rnorm(mean = 0, sd = 0.1, 100)
> y3 <- -0.7 * x + rnorm(mean = 0, sd = 0.1, 100)
> y4 <- x^2
```

```
> par(mfrow = c(2, 2))
> plot(x, y1)
> plot(x, y2)
> plot(x, y3)
> plot(x, y4)
```

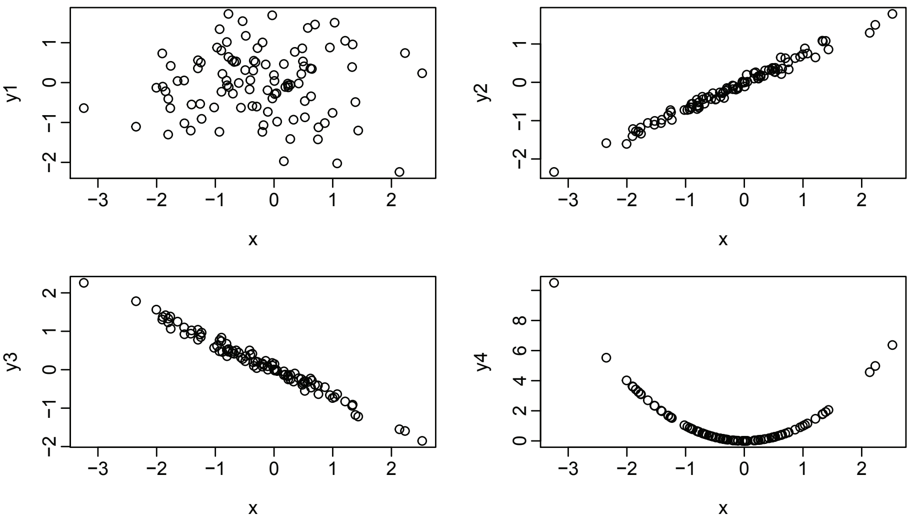


Figure 4.5: Scatter plot of two uncorrelated (extittop left), strongly positively correlated (extittop right), strongly negatively correlated (extitbottom left) and non-linearly dependent variables (extitbottom right).

Let us now compute correlations for the four above examples. We expect no correlation in the first, positive correlation in the second, negative correlation in the third and rather small correlation in the last example.

```
> cor(x, y1)

## [1] 0.01499297

> cor(x, y2)

## [1] 0.9922923
```

```
> cor(x, y3)

## [1] -0.9906092

> cor(x, y4)

## [1] -0.3012178
```

Although different types of dependence are present in the above four data sets, the correlations of data sets one and four are similarly small. There is no dependence in the first data set, whereas there is a very strong but non-linear dependence in the fourth data set. Non-linear dependence is not measured by the correlation coefficient. In the second and third example the correlation coefficient is close to one (in absolute value), just as we expected.

Remark 4.1.24. Take note of the following when interpreting the correlation coefficient. Even if the correlation coefficient indicates a strong correlation between two variables, we cannot conclude, which variable influences the respective other variable. We can already see this since the correlation coefficient is by definition symmetric in two underlying variables. Applied context can only give clarity about the direction of the dependence.

Furthermore, *spurious correlation* can be present. We talk about a spurious correlation, if a third variable Z, which correlates with both X and Y, induces the dependence of X and Y. As an example take the body weight X, the shoe size Y and the height Z. The shoe size will presumably grow with the increasing body weight. Both variables are positively correlated. Nevertheless, both variables are rather influenced by someone's height since the body weight and shoe size increase (normally) when the height grows.

4.1.3 Ordinal variables

In this section we want to draw our attention to a bivariate variable (X, Y) with X and Y being at least ordinal. We will introduce a dependence measure, which is defined by the ranks of the variable values (cf. Definition 3.1.7). Before we define this measure, let us first recall the computation of the rank $Rg(x_j)$ of the value x_j in a data set $x_1, \ldots, x_n$.

```
> x <- c(30, 20, 7, 42, 50, 20)
> sort(x)

## [1]  7 20 20 30 42 50

> rank(x)

## [1] 4.0 2.5 1.0 5.0 6.0 2.5
```

The observation $x_2 = 20 = x_6$ appears twice - at indices 2 and 3. By Definition 3.1.7, its rank is computed by

$$r + \frac{s-1}{2} = 2 + \frac{1}{2} = 2.5\,,$$

which coincides with our computation in R .

Definition 4.1.25. *Let $(x_1, y_1), \dots, (x_n, y_n)$ be observations (a data set) of a (at least) bivariate ordinal variable (X, Y) and $Rg(x_1), \dots, Rg(x_n)$ and $Rg(y_1), \dots, Rg(y_n)$ the ranks of the observations $x_1, \dots, x_n$ and $y_1, \dots, y_n$, respectively. Spearman's rank correlation coefficient r_{xy}^{Sp} is then defined by*

$$r_{xy}^{Sp} = \frac{\sum_{i=1}^{n}(Rg(x_i) - \overline{Rg}_x)(Rg(y_i) - \overline{Rg}_y)}{\sqrt{\sum_{i=1}^{n}(Rg(x_i) - \overline{Rg}_x)^2}\sqrt{\sum_{i=1}^{n}(Rg(y_i) - \overline{Rg}_y)^2}},$$

where $\overline{Rg}_x = \frac{1}{n}\sum_{i=1}^{n} Rg(x_i)$ and $\overline{Rg}_y = \frac{1}{n}\sum_{i=1}^{n} Rg(y_i)$.

From the definition it immediately follows that the rank correlation coefficient coincides with the Bravais-Pearson correlation coefficient applied to the ranks of the observed values. Therefore, we have

$$r_{xy}^{Sp} = r_{Rg(x)Rg(y)}\,.$$

The designation "rank correlation coefficient" is a meaningful notion for this measure since the correlation between the ranks of the observations is actually measured. Further, the values of the variable are not involved in the computation of the rank correlation coefficient and therefore, the rank correlation coefficient can not measure any specific functional relationships between the considered variables. On the contrary, it measures only a monotone relationship between the considered variables. This can be seen in the properties of the rank correlation coefficient, which are summarized in the following lemma.

Lemma 4.1.26. *Let $x_1, \dots, x_n$ and $y_1, \dots, y_n$ be observations of two (at least) ordinal variables with the rank correlation coefficient r_{xy}^{Sp}.*

(i) The value range of the rank correlation coefficient is $[-1, 1]$, i.e.

$$-1 \le r_{xy}^{Sp} \le 1\,.$$

(ii)

$$r_{xy}^{Sp} = 1 \Leftrightarrow Rg(x_i) = Rg(y_i) \text{ for all } i \in \{1, \dots, n\}\,,$$

i.e. $r_{xy}^{Sp} = 1$ if and only if $y_i < y_j$ follows from $x_i < x_j$ and $y_i = y_j$ follows from $x_i = x_j$.

(iii)

$$r_{xy}^{Sp} = -1 \Leftrightarrow Rg(x_i) = n + 1 - Rg(y_i) \text{ for all } i \in \{1, \dots, n\}\,,$$

i.,e. $r_{xy}^{Sp} = -1$ if and only if $y_i > y_j$ follows from $x_i < x_j$ and $y_i = y_j$ follows from $x_i = x_j$.

(iv) If f and g are monotone functions, then for the rank correlation coefficient $r_{f(x)g(y)}^{Sp}$ of the transformed data $f(x_1), \dots, f(x_n)$ and $g(y_1), \dots, g(y_n)$ it holds

$$r_{f(x)g(y)}^{Sp} = \begin{cases} r_{xy}^{Sp} & \text{if } f \text{ and } g \text{ are both either increasing or decreasing} \\ -r_{xy}^{Sp} & \text{if } f \text{ increases and } g \text{ decreasing (or the other way round)} \end{cases}.$$

To summarize, it can be said that the rank correlation coefficient is a measure of the monotone behavioral change of two variables. This of course also includes monotone nonlinear relationships. However, no conclusions can be drawn about the nature of the nonlinear relationship (i.e. quadratic, exponential, etc.).

Example 4.1.27. In Section 4.1.1, we have already looked at the data set `grades`. The variables `prev` and `grade` of this data set are ordinal as grades. Let us now calculate the rank correlation coefficients for these two variables. To compute it in R , the function `cor()` can be used, which is already known from Example 4.1.23 of the previous subsection. This time, however, the option `method` must be set to ''`spearman`''. Further, a direct application to the objects `prev` and `grade` is not possible since they are encoded by letters as it can be seen below.

```
> grades$prev

##   [1] B+  A-  B+  F   F   A   A   C   A   C   C   B
##  [13] A-  B   C+  A   C   A   C   C   F   F   C   A-
##  [25] A   A   A   C   C+  B+  B+  B   A   C   A   C
##  [37] C   A   B   F   B   D   B+  C   C   B+  A   A-
##  [49] A-  B-  C   A   F   A   B   F   C   F   F   C+
##  [61] D   D   F   B   D   C   B-  B   B   A   A   C
##  [73] C   F   C   A   A   C   B   F   A   D   C   A
##  [85] D   B+  A   F   B   C   B+  A   A   C   F   B-
##  [97] A   F   C   F   C   A   C   F   D   D   D   B
## [109] C   B-  F   B   B+  A   F   F   B   F   F   A
## [121] A   B
## Levels: A  A-  B+  B  B-  C+  C  D  F
```

The R function `cor()` requires a numerical input. We can easily transform the underlying grades into the numbers 1 to 9. The function `as.numeric()` does this conversion.

```
> as.numeric(grades$prev)[1:5]

## [1] 3 2 3 9 9

> cor(as.numeric(grades$prev), as.numeric(grades$grade), method = "spearman")

## [1] 0.6536665
```

We recognize a medium to strong correlation of the ranks, which gives us a medium monotonic dependence between the two considered variables.

We can also calculate the rank correlation coefficients for metric variables, since they have a higher measurement level than ordinal variables. We will do this in the next example.

Example 4.1.28. We consider the height and weight of children between 0 and 12 years from the data set `kid.weigths` (`UsingR`). Both variables are obviously metric and it can be suspected that both variables are positively related, i.e. the higher the child the heavier it will be. However, it is not clear whether this relationship is linear. For this reason, let us consider a scatter plot of the data. We distinguish between girls (F) and boys (M).

```
> ggplot(data = kid.weights, aes(x = height, y = weight, colour = gender)) +
+     geom_point(aes(shape = gender))
```

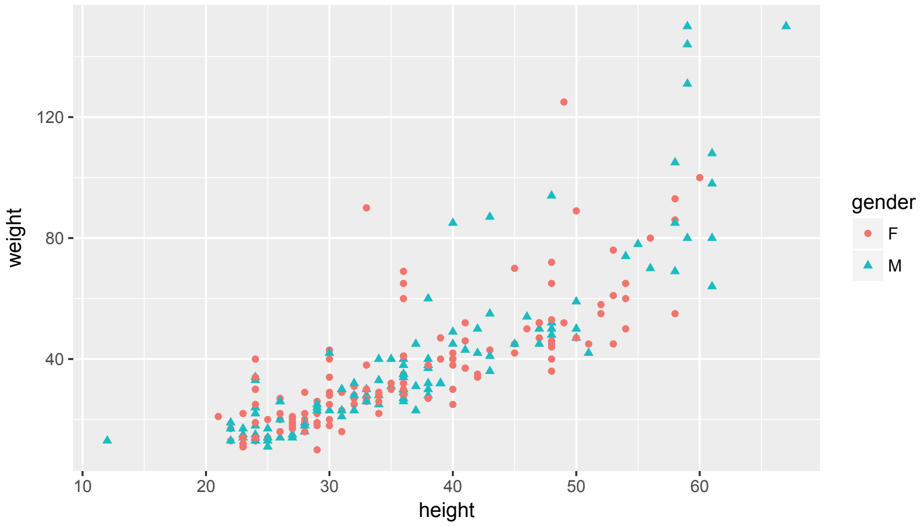


Figure 4.6: Height and weight of children ages 0 to 12 years old subdivided into girls (F) and boys (M).

It can be observed a quadratic relationship, as used in the BMI (kilogram/meter^2), rather than a linear relationship between the two variables. Nevertheless, we calculate the correlation and rank correlation coefficients.

```
> cor(kid.weights$height, kid.weights$weight)

## [1] 0.8237564

> cor(kid.weights$height, kid.weights$weight, method = "spearman")

## [1] 0.8822136
```

The linear approximation for the relationship between two variables is quite good, as one can observe from the correlation coefficient of 0.824. The measure for the monotonic relationship (expressed by the Spearman correlation) between size and weight is even bigger. Finally, we calculate the correlation between the squared height and weight.

```
> cor(kid.weights$height^2, kid.weights$weight)

## [1] 0.8446415
```

This confirms our conjecture of a quadratic relationship.

4.2 Descriptive linear regression

In Section 4.1, dependence measures, especially the Bravais-Pearson correlation coefficient, were introduced to quantify the relationship between two variables X and Y. In the descriptive linear regression analysis, the characterization of the linear relationship between two metric variables X and Y will be discussed. The variable X plays the role of the independent variable, which has an influence on the dependent variable Y. We denote X as a covariable (regressor or covariate) and Y as a response. The assumed relationship is thus of the form

$$Y = a + bX\,,$$

where the parameters a and b are unknown. As in the previous sections, we assume that there is a bivariate data set $(x_1, y_1), \dots, (x_n, y_n)$. The values

$$\tilde{y}_i = a + bx_i, \qquad i \in \{1, \dots, n\}\,,$$

are referred to as regression values (on a regression line $y = a + bx$).

Which of two variables is the covariate and which one is response is often determined from theoretical considerations. The decision whether a linear approach is useful can be checked by a scatter plot (for graphic illustration of the relationship) and by computing the correlation coefficient. However, a conclusion from the scatter plot may be that X has a quadratic impact on Y and not linear, i.e. X^2 influences Y and not X. The resulting relationship between X^2 and Y is still a linear form, since Y is still a linear function of X^2, that is,

$$Y = a + bX^2\,.$$

This can be generalized by the form $Y = a + bf(X)$. An idea for a suitable choice of the function f can always be obtained through a scatter plot.

In reality, the relation $Y = a + bX$ will be often not exact, which means that the regression values $\tilde{y}_i$ will not precisely match the observed values y_i. Reasons for this may be measurement errors and/or measurement inaccuracies as well as natural fluctuations of characteristics of the statistical units. The linear relationship is, in some cases, only an approximation of the real relationship. To take into account all above points, we consider a linear regression model, which has the following form

$$Y = a + bX + \varepsilon.$$

The additive error term ε represents the disturbance term or noise. Equivalently, it describes the deviation between the measured value y and the regression value $\tilde{y}$. The following relationship is assumed for the observations $(x_1, y_1), \dots, (x_n, y_n)$:

$$y_i = a + bx_i + \varepsilon_i, \qquad i \in \{1, \dots, n\}\,, \tag{4.2.1}$$

where the parameter $a \in \mathbb{R}$, which we call the intercept, and the parameter $b \in \mathbb{R}$, which we call the slope, are unknown constants. Similarly, the error ε_i can not be observed.

The aim of the regression analysis is now to fit Model (4.2.1) to the observed data. For this, the parameters a and b must be estimated using the underlying data set $(x_1, y_1), \dots, (x_n, y_n)$. A natural approach is here to "select" a linear function $f_{a,b}(x) = a + bx$ in such a way that the sum over the squared distances between the observed values y_i and the function values $f_{a,b}(x_i)$, $i \in \{1, \dots, n\}$ is minimal. This leads us to the method of the least squares.

4.2.1 Method of least squares and the least squares estimator

In this section we want to choose a specific function from the set of linear functions,

$$\mathcal{H} = \{f_{a,b}(x) = a + bx, x \in \mathbb{R} \mid a, b \in \mathbb{R}\}$$

that minimizes

$$Q(a,b) := \sum_{i=1}^{n}(y_i - f_{a,b}(x_i))^2 = \sum_{i=1}^{n}(y_i - (a + bx_i))^2.$$

Since $Q(a,b)$ is differentiable in a and b, an extreme point $(\widehat{a}, \widehat{b})$ satisfies the conditions

$$\begin{aligned} \frac{\partial}{\partial a} Q(a,b)|_{(a,b)=(\widehat{a},\widehat{b})} &= 0, \\ \frac{\partial}{\partial b} Q(a,b)|_{(a,b)=(\widehat{a},\widehat{b})} &= 0, \end{aligned}$$

where $\frac{\partial}{\partial x}$ denotes the (partial) derivative with respect to x, $x \in \{a, b\}$. This extreme point is a minimum if the Hessian

$$\begin{pmatrix} \frac{\partial^2}{\partial^2 a} Q(a,b) & \frac{\partial^2}{\partial b \partial a} Q(a,b) \\ \frac{\partial^2}{\partial a \partial b} Q(a,b) & \frac{\partial^2}{\partial^2 b} Q(a,b) \end{pmatrix}$$

is positive definite at $(a,b) = (\widehat{a}, \widehat{b})$.

It holds that

$$\begin{aligned} \frac{\partial}{\partial a} Q(a,b) &= \sum_{i=1}^{n} 2(y_i - (a + bx_i))(-1) = \sum_{i=1}^{n} 2(a + bx_i - y_i) \\ \frac{\partial}{\partial b} Q(a,b) &= \sum_{i=1}^{n} 2(y_i - (a + bx_i))(-x_i) = \sum_{i=1}^{n} 2x_i(a + bx_i - y_i) \end{aligned}$$

Equating the partial derivatives with 0 and solving the corresponding equations gives

$$\begin{aligned} \frac{\partial}{\partial a} Q(a,b)|_{(a,b)=(\widehat{a},\widehat{b})} = 0 = \sum_{i=1}^{n} 2(\widehat{a} + \widehat{b}x_i - y_i) &\Leftrightarrow n\widehat{a} = \sum_{i=1}^{n} y_i - \widehat{b} \sum_{i=1}^{n} x_i \\ &\Leftrightarrow \widehat{a} = \overline{y}_n - \widehat{b}\overline{x}_n \end{aligned}$$

and

$$\begin{aligned} \frac{\partial}{\partial b} Q(a,b)|_{(a,b)=(\widehat{a},\widehat{b})} = 0 = \sum_{i=1}^{n} 2x_i(\widehat{a} + \widehat{b}x_i - y_i) &\Leftrightarrow \widehat{b} \sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} x_i y_i - \widehat{a} \sum_{i=1}^{n} x_i \\ \Leftrightarrow \widehat{b} \sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} x_i y_i - \left(\overline{y}_n - \widehat{b}\overline{x}_n\right) \sum_{i=1}^{n} x_i &\Leftrightarrow \widehat{b} \left(\sum_{i=1}^{n} x_i^2 - n\overline{x}_n^2\right) = \sum_{i=1}^{n} x_i y_i - n\overline{x}_n \overline{y}_n \\ &\Leftrightarrow \widehat{b} = \frac{\sum_{i=1}^{n} x_i y_i - n\overline{x}_n \overline{y}_n}{\sum_{i=1}^{n} x_i^2 - n\overline{x}_n^2}. \end{aligned}$$

Furthermore, one can show (we discard this computation here) that the Hessian is positive definite at this point. Thus, the solution of the underlying minimization problem is given by

$$\begin{aligned}\widehat{b} &= \frac{\sum_{i=1}^{n}(x_i - \overline{x}_n)(y_i - \overline{y}_n)}{\sum_{i=1}^{n}(x_i - \overline{x}_n)^2} = \frac{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x}_n)(y_i - \overline{y}_n)}{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x}_n)^2} = \frac{s_{xy}}{s_x^2}\\ \widehat{a} &= \overline{y}_n - \widehat{b}\overline{x}_n\,.\end{aligned}$$

The estimators $\widehat{a}$ and $\widehat{b}$ are called the least-squares estimators of a and b. With these estimators, fitted regression values can now be obtained via

$$\widehat{y}_i = \widehat{a} + \widehat{b}x_i, \qquad i \in \{1, \dots, n\}\,.$$

The difference between the observed value and the fitted value

$$\widehat{\varepsilon}_i = y_i - \widehat{y}_i, \qquad i \in \{1, \dots, n\}\,,$$

is called the residual. For the sum of the residuals it holds that

$$\sum_{i=1}^{n}\widehat{\varepsilon}_i = \sum_{i=1}^{n}(y_i - \widehat{y}_i) = 0\,.$$

Before we think about an assessment of the model fit, let's consider the following example.

Example 4.2.1. In this example, we will again analyze the dataset `homedata` (`UsingR`), which contains the estimated prices of houses in 1970 and in 2000. As before, we only look at the first 150 observations. In R, the function `lm()` calculates the least-squares estimators $\widehat{a}$ and $\widehat{b}$.

```
> y1970_150 <- homedata$y1970[1:150]
> y2000_150 <- homedata$y2000[1:150]
> (lm_homedata <- lm(y2000_150 ~ y1970_150))

##
## Call:
## lm(formula = y2000_150 ~ y1970_150)
##
## Coefficients:
## (Intercept)    y1970_150
##  -93300.497        5.075
```

We obtain the estimators $\widehat{a} = -9.33005 \times 10^4$ und $\widehat{b} = 5.07$. The resulting regression line

$$y = \widehat{a} + \widehat{b}x$$

can be added to the scatterplot of the observation values easily using `abline()`.

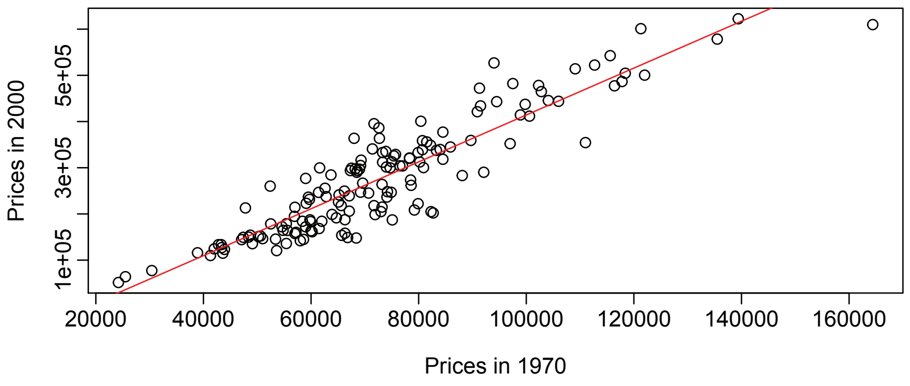


Figure 4.7: Scatter plot of the estimated house prices together with the adjusted regression line.

```
> plot(y1970_150, y2000_150, xlab = "Prices in 1970",
+      ylab = "Prices in 2000")
> abline(lm_homedata, col = "red")
```

Alternatively, you could use ggplot2 to draw the regression line directly into the scatter plot without first calculating the regression model.

```
> ggplot(data = homedata[1:150, ], aes(x = y1970, y = y2000)) + geom_point() +
+     geom_smooth(method = "lm", se = FALSE) + xlab("Prices in 1970") +
+     ylab("Prices in 2000")
```

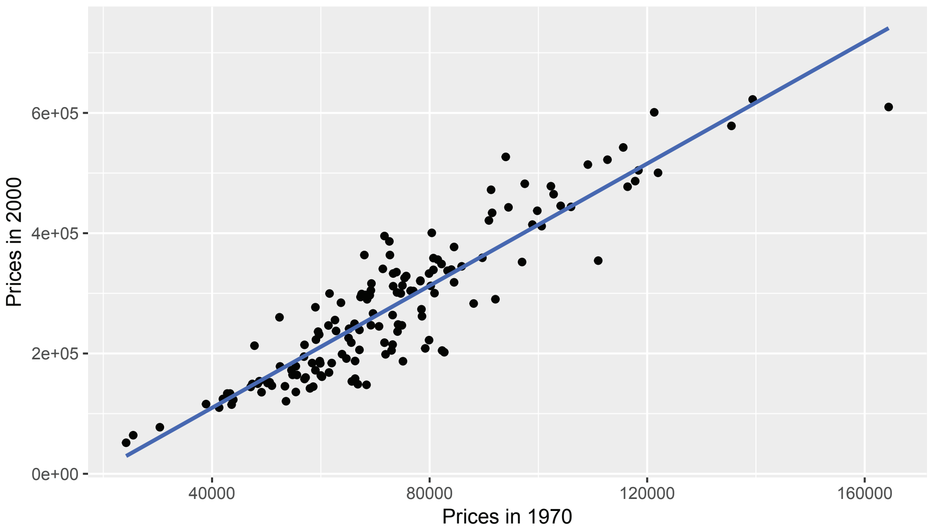


Figure 4.8: Scatter plot of the estimated house prices together with the fitted regression line.

One of the main applications of a regression model is the prediction of a new response value y given a new covariable value $x \in I = [x_{(1)}, x_{(n)}]$.Since there are no observations outside of the interval I, nothing can be said about the relationship between the variables outside of the interval I. Therefore, the predicted value, which is calculated for a covariate value outside of the interval I, should be interpreted with caution. Near the interval, however, a good approximation would usually be expected.

Here we look at the predicted value of a house in the year 2000, which was worth $50000 \in [24200, 164400]$ US dollars in 1970. The value results from the fitted regression line

$$\widehat{y} = \widehat{a} + \widehat{b} \cdot 50000\,.$$

With the estimated coefficients, we obtain

```
> -93300.497 + 5.075 * 50000

## [1] 160449.5
```

The coefficients do not have to be entered by hand. They are stored in the variable `coef` of the regression model `lm_homedata`. They can be read using the `coef()` function or the $ notation.

```
> coef_mod_hd <- coef(lm_homedata)   # or lm_homedata$coef\t
> sum(coef_mod_hd * c(1, 50000))

## [1] 160434.5
```

The easiest way to calculate a prediction is to use the function `predict()`. However, the covariable must have a structure of `data.frame` .

```
> predict(lm_homedata, newdata = data.frame(y1970_150 = 50000))

##        1
## 160434.5
```

Let us now calculate the residual for the data point $(72700, 363700)$.

```
> 363700 - predict(lm_homedata, newdata = data.frame(y1970_150 = 72700))

##        1
## 88069.81
```

However, you can get the residue more easily with the command `residuals()`.

```
> residuals(lm_homedata)[which(y1970_150 == 72700)]

##       63
## 88069.81
```

Remark 4.2.2. In Example 4.2.1, we used prices in 1970 as an independent variable to describe the prices in 2000. This is more reasonable from the application perspective (price dynamics of objects) than to use prices in 2000 as independent variables. Nevertheless, we could fit the corresponding model, since, there is also a linear relationship in this constellation too. The above discussion should make clear that it should be decided in advance, which variable can be considered as a useful influence factor and which can not.

4.2.2 Evaluating the model fit

Using scatter plots, we have gained an intuition that a linear regression model could be useful for describing the relationship between X and Y. Then the least-squares estimators $\widehat{a}$ and $\widehat{b}$ were determined. However, one question is now open: How well does the estimated linear regression model fit the underlying data set. In this section, we will present methods for evaluating model fit.

To determine the least squares estimator, the function

$$Q(a,b) = \sum_{i=1}^{n}(y_i - a - bx_i)^2 = \sum_{i=1}^{n}\varepsilon_i^2$$

was minimized. Since this function is based on the deviations ε_i, $i \in \{1,\ldots,n\}$, the evaluation of the model fit should include the estimated errors (or residuals)

$$\widehat{\varepsilon}_i = y_i - \widehat{y}_i = y_i - \widehat{a} - \widehat{b}x_i, \qquad i \in \{1,\ldots,n\}.$$

Since the range of residuals depends on the range of response variable (i.e. range of Y), an assessment of the residual values is difficult. Therefore, the normalized residuals are often considered.

Definition 4.2.3. *Let $\widehat{\varepsilon}_1,\ldots,\widehat{\varepsilon}_n$ be the residuals (i.e. estimated errors) of a linear regression model. For $\sum_{i=1}^{n}\widehat{\varepsilon}_i^2 > 0$, the normalized residuals are defined as quotients*

$$\widehat{d}_i = \frac{\widehat{\varepsilon}_i}{\sqrt{\sum_{i=1}^{n}\widehat{\varepsilon}_i^2}} = \frac{y_i - \widehat{y}_i}{\sqrt{\sum_{i=1}^{n}(y_i - \widehat{y}_i)^2}}, \qquad i \in \{1,\ldots,n\},$$

In the following lemma, a few properties of the (normalized) residuals are summarized.

Lemma 4.2.4. *Let $\widehat{\varepsilon}_1.\ldots,\widehat{\varepsilon}_n$ and $\widehat{d}_1,\ldots,\widehat{d}_n$ be the residuals and normalized residuals of a linear regression model, respectively.*

(i) For the normalized residuals it holds that

$$-1 \leq \widehat{d}_i \leq 1, \qquad i \in \{1,\ldots,n\},$$

$\sum_{i=1}^{n}\widehat{d}_i = 0$ *and* $\sum_{i=1}^{n}\widehat{d}_i^2 = 1$.

(ii) The sum of the squared residuals is zero if and only if all observed values are on the fitted regression line, that means

$$\sum_{i=1}^{n}\widehat{\varepsilon}_i^2 = 0 \Leftrightarrow y_i = \widehat{y}_i \text{ for all } i \in \{1,\ldots,n\}.$$

(iii) The variability decomposition

$$\sum_{i=1}^{n}(y_i - \overline{y}_n)^2 = \sum_{i=1}^{n}(\widehat{y}_i - \overline{y}_n)^2 + \sum_{i=1}^{n}(y_i - \widehat{y}_i)^2$$

holds - or equivalently

$$s_y^2 = s_{\widehat{y}}^2 + s_{\widehat{\varepsilon}}^2,$$

where s_y^2 is the empirical variance of the observation values $y_1,\ldots,y_n$, $s_{\widehat{y}}^2$ the empirical variance of the regression values $\widehat{y}_1,\ldots,\widehat{y}_n$ and $s_{\widehat{\varepsilon}}^2$ denotes the empirical variance of the residuals.

Proof We only prove the variability decomposition. We have

$$\begin{aligned}\sum_{i=1}^{n}(y_i-\widehat{y}_i)^2 &= \sum_{i=1}^{n}(y_i-\overline{y}_n+\overline{y}_n-\widehat{y}_i)^2 = \sum_{i=1}^{n}\left[(y_i-\overline{y}_n)^2+2(y_i-\overline{y}_n)(\overline{y}_n-\widehat{y}_i)+(\overline{y}_n-\widehat{y}_i)^2\right]\\ &= \sum_{i=1}^{n}(y_i-\overline{y}_n)^2+\sum_{i=1}^{n}(\overline{y}_n-\widehat{y}_i)^2-2\sum_{i=1}^{n}(y_i-\overline{y}_n)(\widehat{y}_i-\overline{y}_n)\end{aligned}$$

The least-squares estimators are given by

$$\widehat{a}=\overline{y}_n-\widehat{b}\overline{x}_n \text{ und } \widehat{b}=\frac{s_{xy}}{s_x^2}\,,$$

such that $\overline{y}_n=\widehat{a}+\widehat{b}\overline{x}_n$ and $s_{xy}=\widehat{b}s_x^2$ holds. Thus, we obtain

$$\widehat{y}_i-\overline{y}_n=\widehat{a}+\widehat{b}x_i-(\widehat{a}+\widehat{b}\overline{x}_n)=\widehat{b}(x_i-\overline{x}_n)\,. \tag{4.2.2}$$

Hence, it follows

$$\begin{aligned}\sum_{i=1}^{n}(y_i-\overline{y}_n)(\widehat{y}_i-\overline{y}_n) &\overset{(4.2.2)}{=} \widehat{b}\sum_{i=1}^{n}(y_i-\overline{y}_n)(x_i-\overline{x}_n)=\widehat{b}(n-1)s_{xy}=(n-1)\widehat{b}^2s_x^2\\ &= \sum_{i=1}^{n}\widehat{b}^2(x_i-\overline{x}_n)^2 \overset{(4.2.2)}{=} \sum_{i=1}^{n}(\widehat{y}_i-\overline{y}_n)^2\,,\end{aligned}$$

and from this it follows

$$\sum_{i=1}^{n}(y_i-\widehat{y}_i)^2=\sum_{i=1}^{n}(y_i-\overline{y}_n)^2-\sum_{i=1}^{n}(\widehat{y}_i-\overline{y}_n)^2\,.$$

From $\overline{\widehat{y}}_n=\frac{1}{n}\sum_{i=1}\widehat{y}_i=\overline{y}_n$ and $\overline{\widehat{\varepsilon}}_n=\frac{1}{n}\sum_{i=1}\widehat{\varepsilon}_i=0$ the desired statement for the empirical variances follows:

$$s_y^2=s_{\widehat{y}}^2+s_{\widehat{\varepsilon}}^2\,.$$

□

The variance $s_{\widehat{y}}^2$ specified in Lemma 4.2.4 quantifies the variability of the fitted regression values. The corresponding summand $\sum_{i=1}^{n}(\widehat{y}_i-\overline{y}_n)^2$ of the variability decomposition formula is therefore also called *variance explained by the regression* . The remaining part of the variance of the observed responses is the variance of the residuals. This proportion of the total variability can not be explained by the linear regression model. The corresponding summand $\sum_{i=1}^{n}(y_i-\widehat{y}_i)^2$ of the variability decomposition formula is called *residual variance*.

Coefficient of determination (or R^2)

Based on the residuals, we now want to introduce a measure, which can assess the fit of the estimated linear regression line.

Definition 4.2.5. *Let $(x_1,y_1),\ldots,(x_n,y_n)$ be observations of a bivariate metric variable (X,Y) with $s_x^2>0$ and $s_y^2>0$. The coefficient of determination R_{xy}^2 of the linear regression is then defined by*

$$R_{xy}^2=1-\frac{\sum_{i=1}^{n}(y_i-\widehat{y}_i)^2}{\sum_{i=1}^{n}(y_i-\overline{y}_n)^2}=1-\frac{s_{\widehat{\varepsilon}}^2}{s_y^2}\,.$$

Due to the variability decomposition, we also get the following representation

$$R_{xy}^2 = \frac{\sum_{i=1}^n (y_i - \overline{y}_n)^2 - \sum_{i=1}^n (y_i - \widehat{y}_i)^2}{\sum_{i=1}^n (y_i - \overline{y}_n)^2} = \frac{\sum_{i=1}^n (\widehat{y}_i - \overline{y}_n)^2}{\sum_{i=1}^n (y_i - \overline{y}_n)^2} = \frac{s_{\widehat{y}}^2}{s_y^2}\,.$$

Thus, the coefficient of determination is the quotient of the regression sum of squares $\sum_{i=1}^n (\widehat{y}_i - \overline{y}_n)^2$, which depends the estimated regression model, and the total sum of squares $\sum_{i=1}^n (y_i - \overline{y}_n)^2$ of the observations $y_1, \ldots, y_n$. Further properties of the coefficient of determination are summarized in the following lemma.

Lemma 4.2.6. *Consider the data set $(x_1, y_1), \ldots, (x_n, y_n)$ of a bivariate metric variable (X, Y) with $s_x^2 > 0$ and $s_y^2 > 0$.*

(i) Let r_{xy} be the Bravais-Pearson-correlation coefficient of the data $(x_1, y_1), \ldots, (x_n, y_n)$. Then it holds that

$$R_{xy}^2 = r_{xy}^2\,.$$

(ii) For the coefficient of determination we have

$$0 \leq R_{xy}^2 \leq 1\,.$$

(iii) The coefficient of determination is equal to one if and only if all observed values coincide with the fitted regression values, i.e.

$$R_{xy}^2 = 1 \Leftrightarrow \widehat{y}_i = y_i \text{ for all } i \in \{1, \ldots, n\}\,.$$

(iv) The coefficient of determination is equal to zero if and only if the fitted regression line is constant, i.e.

$$R_{xy}^2 = 0 \Leftrightarrow \widehat{y}_i = \overline{y}_n \text{ for all } i \in \{1, \ldots, n\}\,.$$

Proof We only prove property (i). From the proof of Lemma 4.2.4, we know

$$\sum_{i=1}^n (\widehat{y}_i - \overline{y}_n)^2 = \sum_{i=1}^n \widehat{b}^2 (x_i - \overline{x}_n)^2\,.$$

It follows now that

$$\begin{aligned} R_{xy}^2 &= \frac{\sum_{i=1}^n (\widehat{y}_i - \overline{y}_n)^2}{\sum_{i=1}^n (y_i - \overline{y}_n)^2} = \frac{\sum_{i=1}^n \widehat{b}^2 (x_i - \overline{x}_n)^2}{\sum_{i=1}^n (y_i - \overline{y}_n)^2} = \frac{\widehat{b}^2 \cdot \frac{1}{n-1} \sum_{i=1}^n (x_i - \overline{x}_n)^2}{\frac{1}{n-1} \sum_{i=1}^n (y_i - \overline{y}_n)^2} = \frac{\widehat{b}^2 s_x^2}{s_y^2} \\ &= \frac{s_{xy}^2 \cdot s_x^2}{(s_x^2)^2 \cdot s_y^2} = \frac{s_{xy}^2}{s_x^2 \cdot s_y^2} = \left(\frac{s_{xy}}{s_x \cdot s_y}\right)^2 = r_{xy}^2\,. \end{aligned}$$

□

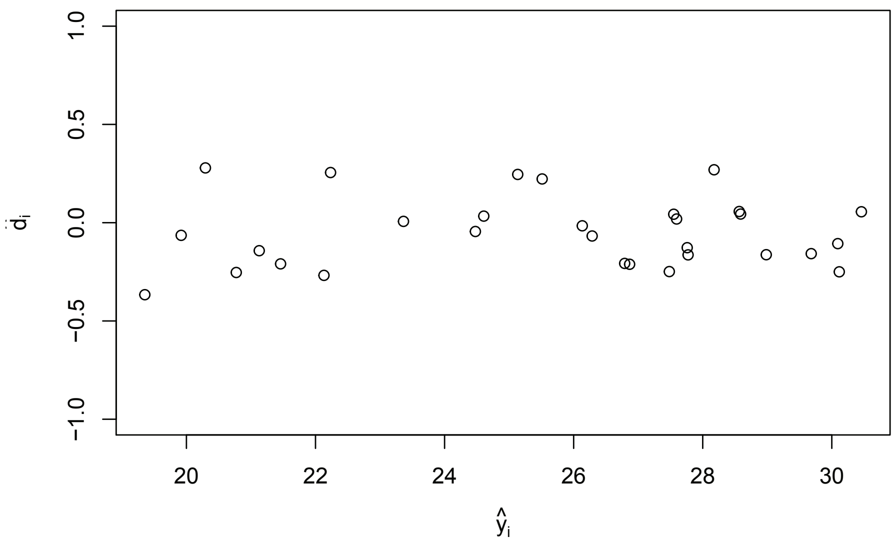
The coefficient of determination equals one if and only if the entire variation in the data can be explained by the regression model. For values close to one (let us say $R_{xy}^2 > 0.8$, although this threshold is not strict and varies depending on research field), a high portion of the variation (scattering) is explained by the model, and therefore we speak of a good model fit to the data.

On the other hand, the model can not explain the variability in the data at all, if the coefficient of determination is zero. For values close to zero (let say $R_{xy}^2 < 0.3$, although this threshold is not strict and varies depending on research field), the linear regression model describes the relationship between the variables insufficiently, and therefore one speaks of a bad model fit.

Residual analysis (Residual plot)

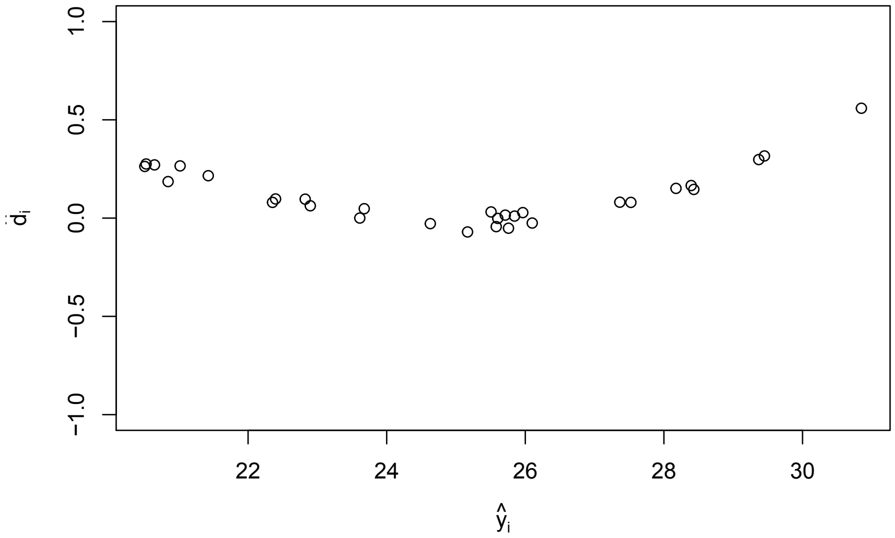
The aim of the residual analysis is a verification of the model assumptions used, i.e. of the assumed linear relationship. The residual plot is a scatter plot in which the fitted responses $\widehat{y}_1, \ldots, \widehat{y}_n$ are plotted against the (normalized) residuals. A use of the normalized residuals $\widehat{d}_1, \ldots, \widehat{d}_n$ has the advantage since the value range is always limited to the interval $[-1, 1]$.

The residual plot is interpreted as follows. If the assumed linear relationship between the variables exists then the deviations between the observations $y_1, \ldots, y_n$ and the fitted responses $\widehat{y}_1, \ldots, \widehat{y}_n$ should be of purely random/chaotic character (measurement errors). Therefore, no regular (systematic) structures should therefore be recognized in the residual plot.



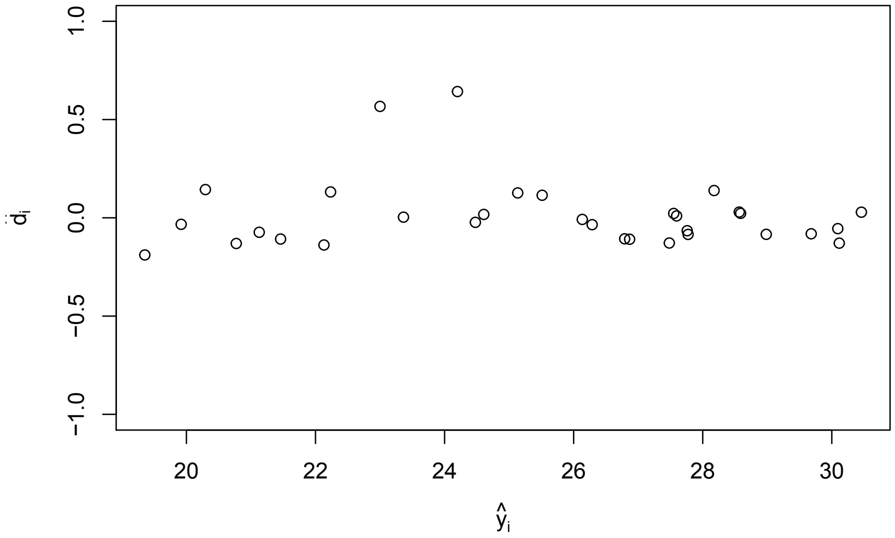
The points in the above residual plot lie in a random manner in approximately equal parts both above and below the x-axis. The deviations of points are distributed irregularly over the range of values of the fitted responses.

However, if the residual plot has e.g. the form like in the figure below then there might be a systematic error present.



In this case, a transformation of the independent variable (consider $f(X)$ instead of X) could help or one must question a linear relationship between variables and completely depart from the linear regression model.

If one discovers some (extremely) large deviations in the residual plot like in the following graph then the corresponding observations have to be examined, whether they can be classified as outliers and should perhaps be removed from the data set.



Example 4.2.7. We consider again the data from Example 4.2.1. The aim of this example is to assess the model fit using the coefficient of determination and the residual plot. To do this, we fit the model in R and save the result in the variable `lm_homedata`.

```
> lm_homedata <- lm(y2000_150 ~ y1970_150)
```

The summary of the variable `lm_homedata` contains a variety of information about the fitted regression model. The least-squares estimators, the resulting residuals and the coefficient of determination (or R^2). The later can be found under the label `Multiple R-squared`.

```
> summary(lm_homedata)

##
## Call:
## lm(formula = y2000_150 ~ y1970_150)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -131180  -31945    1161   35091  143179
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -9.330e+04  1.469e+04  -6.353 2.46e-09 ***
## y1970_150    5.075e+00  1.929e-01  26.308  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 51890 on 148 degrees of freedom
## Multiple R-squared:  0.8238,Adjusted R-squared:  0.8226
## F-statistic: 692.1 on 1 and 148 DF,  p-value: < 2.2e-16
```

The value of the coefficient of determination can also be accessed directly in the summary using the $-notation.

```
> summary(lm_homedata)$r.squared

## [1] 0.8238341
```

The fitted regression model explains about 82% of the variability contained in the data. The model fit is therefore good with respect to the coefficient of determination. Let us now take a look at the residuals to check, whether there is a systematic error, which could be discovered using the residual plot. A detected systematic error would contradict to the assumption of a linear relationship.

```
> residuals <- residuals(lm_homedata)
> norm_residuals <- residuals / sqrt(sum(residuals^2))
> adj_regression_values <- fitted(lm_homedata)
> plot( adj_regression_values, norm_residuals,
+       xlab = "fitted responses", ylab = "normalized residuals")
```

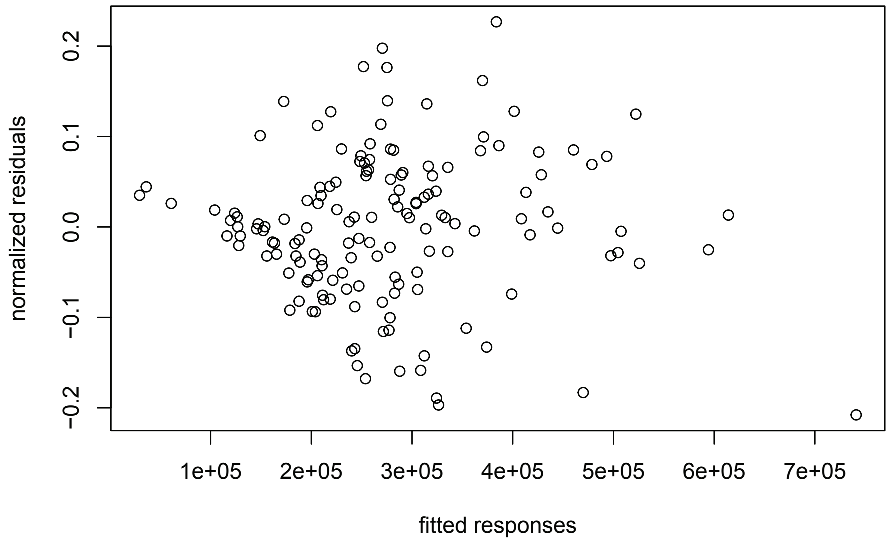


Figure 4.9: Residual plot of the homedata data

In Figure 4.9 one can observe that the normalized residuals have a much larger range for fitted responses between 200000 and 400000 $ than in the remaining areas. i.e. for fitted responses less than 200000 and larger than 400000. This may be an indication that the assumed linear relationship is not the same over the whole value range of the response variable. On the other hand, there are significantly fewer observations outside of this price interval from 200000 to 400000 $, so the above range comparison is not quite obvious. In summary, it can be said that there is no clear indication of a violation of the assumption of a linear relationship.

In the next example, consider the kid.weights data from Example 4.1.28.

Example 4.2.8. In Example 4.1.28, we have recognized that a quadratic influence of the body size is more appropriate than a linear one. Therefore, we want to fit a model of the form

$$y_i = a + bx_i^2 + \varepsilon_i.$$

To specify the squared term in the model notation of R

```
Response ~ Covariable
```

the transformation of the covariable must be stated within the function I() (treats the input variable as is).

```
> lm_kid_weights <- lm(weight ~ I(height^2), data = kid.weights)
> plot(weight ~ I(height^2), data = kid.weights)
> abline(lm_kid_weights)
```

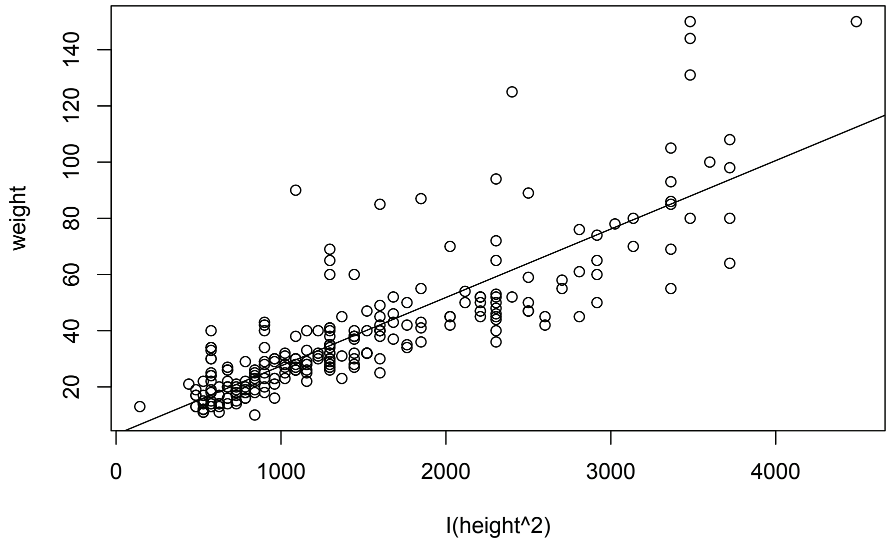


Figure 4.10: Scatter plot of the squared height and weight of 250 children including the fitted regression line.

In order compare the above model with the standard approach, we fit the linear regression model with linear influence of `height`.

```
> lm_kid_weights_linear <- lm(weight ~ height, data = kid.weights)
> summary(lm_kid_weights)$r.squared

## [1] 0.7134193

> summary(lm_kid_weights_linear)$r.squared

## [1] 0.6785746
```

Both regressions do not provide a good model fit according to our empirical rule of thumb ($R^2 < 0.8$). However, we obtain a slightly larger coefficient of determination for the model with the squared variable `height`, which indicates a slight improvement of the model fit. Finally, we consider the residuals for both models.

```
> par(mgp=c(2.2,0.45,0), tcl=-0.4, mar=c(3.3,3.6,1.1,1.1))
> par(mfrow = c(1, 2))
> plot(fitted(lm_kid_weights_linear), residuals(lm_kid_weights_linear),
+      xlab = "fitted responses",
+      ylab = "residuals", ylim = c(-30, 70), main="Linear influence")
> plot(fitted(lm_kid_weights), residuals(lm_kid_weights),
+      xlab = "fitted responses", ylab = "residuals", ylim = c(-30, 70),  main="Quadratic i
```

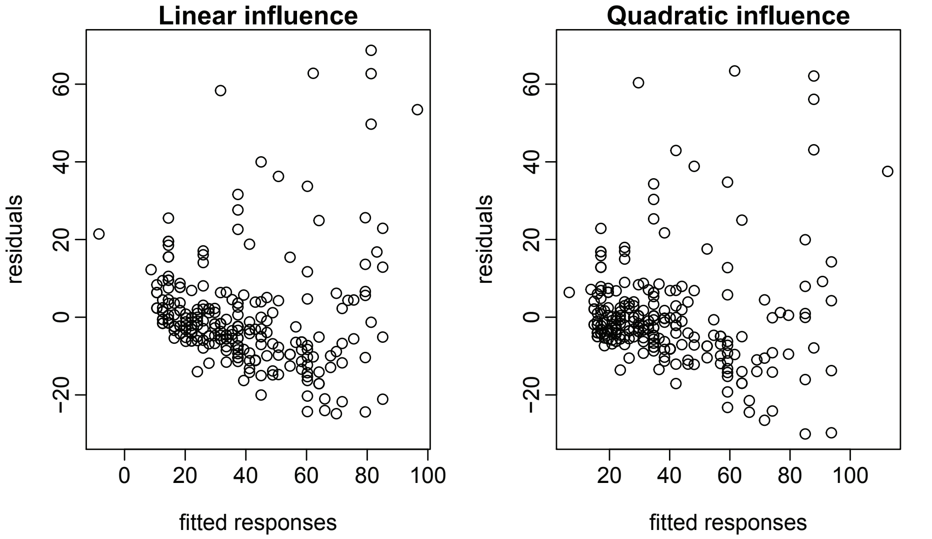


Figure 4.11: Residual plot for the linear model with linear influence (left) and quadratic influence (right) of exttheight.

We can observe in Figure 4.11 that residuals of the linear regression with linear influence of `height` have more variability than ones from the linear regression with quadratic influence of `height`.

Example 4.2.9. Finally, let us consider the results from the US presidential election in 2000, which are contained in the data set `florida` (`UsingR`). We compare the result of the conservative candidates Pat Buchanan and George Bush, because we assume a linear dependence between votes for Buchanan and votes for Bush.

```
> plot(BUCHANAN ~ BUSH, data = florida)
> lm_florida <- lm(BUCHANAN ~ BUSH, data = florida)
> abline(lm_florida)
```

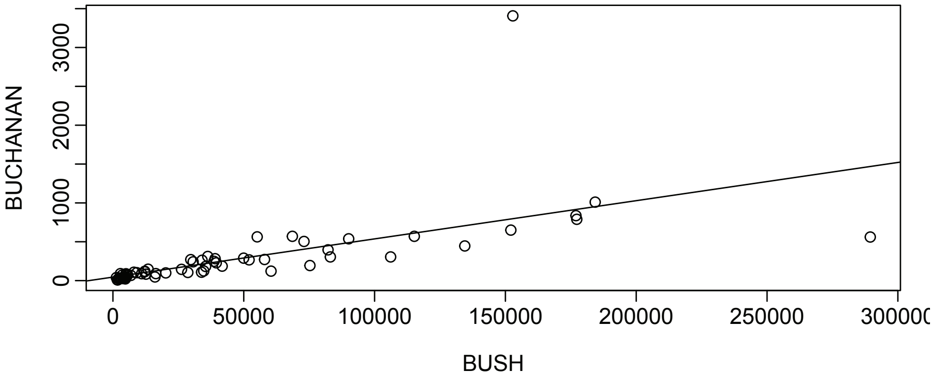


Figure 4.12: Scatter plot of votes for Pat Buchanan and George Bush with the fitted regression line.

There are two obvious outliers. Using the function `identify()`, we can identify those points per mouse click.

```
> with(florida, identify(BUSH, BUCHANAN, n = 2, labels = County))
```

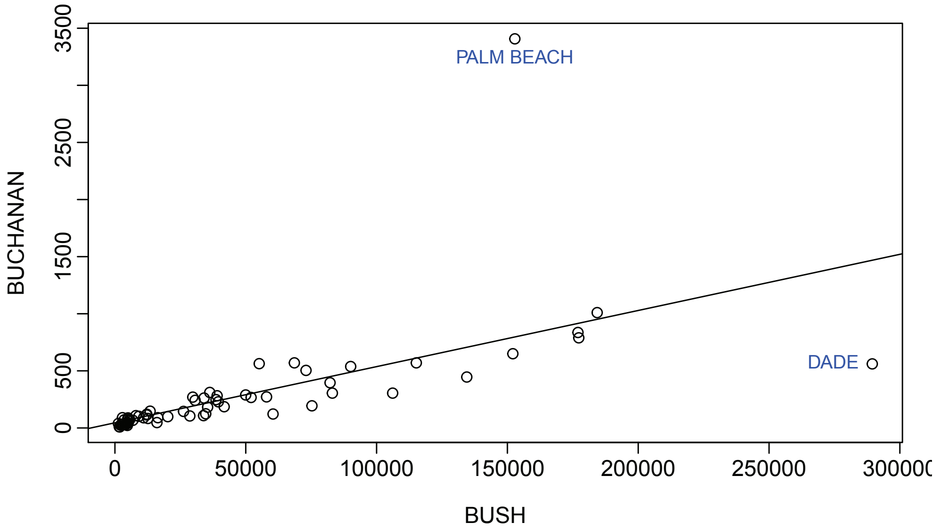
The outliers are observations 13 and 50 from district

```
> florida$County[c(13, 50)]

##        DADE PALM BEACH
##        DADE PALM BEACH
## 67 Levels: ALACHUA BAKER BAY BRADFORD BREVARD ... WASHINGTON
```

We label both observations with their respective district name in the plot.

```
> plot(BUCHANAN ~ BUSH, data = florida)
> abline(lm_florida)
> text(florida$BUSH[c(13, 50)], florida$BUCHANAN[c(13, 50)],
+      labels = florida$County[c(13, 50)], cex = 0.8, col = "blue",
+      pos = c(2, 1))
```



It is suspected that the layout of the ballot paper (Wahlzettel in German) led to many wrong votes for Buchanan, which probably were meant for Al Gore. For Palm Beach, we obtain the residual

```
> residuals(lm_florida)[50]

##       50
## 2610.193
```

Note that the election in 2000 was decided by a difference of 537 votes.

R summary

From Chapter 3 the functions `table()` and `prop.table()` were already known. They can also be used to analyze bivariate nominal data. Marginal distributions are obtained with the function `margin.table()`. We can compare more than five quantiles of two data sets with the function `qqplot()`. A rough impression on the relationship of two metric data sets is obtained with a scatter plot, which is generated with the function `plot()`. The empirical variance and the correlation coefficients for two metric data sets are calculated with the function `cov()` and `cor()`. The function `cor()` can also be used to calculate the rank correlation coefficient of two ordinally scaled variables using the option `method='spearman'`. The least-squares estimator of the parameters of a linear regression model (and much more) is calculated with the function `lm()`. For an overview of the results, use the function `summary()`. The residuals and fitted responses are extracted with the function `residuals()` and `fitted()`. The fitted regression line is drawn with the function `abline()`. A prediction can be calculated with the function `predict()`.