Introduction to statistical data analysis with R

2nd Edition

Matthias Kohl

Contents

List of Figures

List of Tables

Preface

In the meantime, more than five years have passed since the first version of this book was written. Since the statistical software R develops very dynamically some of the features of the first version of the book no longer work and and many new possibilities have emerged. Therefore it was high time to update the book and the R code it contains. In particular I use now for some semesters no longer simple R scripts for my lectures, but corresponding R Markdown documents (see section 1.5).

Other important updates compared to the first version of the book include:

- links to my GitHub repository ISDR
- many more exercises
- brief videos on YouTube
- AL and RMX estimator
- estimation of cut-off values
- boostrap confidence intervals
- more examples about sample size calculation
- statistical test for repeated measurements
- new chapter about multiple testing

The R code used in the individual chapters or sections, can be found on GitHub https://github.com/stamats/ISDR and is contained in text files (R Markdown files) with the file extension `.Rmd`. The R code of each chapter can be run separately from the other chapters. In addition, you can find brief videos on YouTube at https://youtu.be/6E3QJOc1bnM with short explanations.
The second edition of the book was again created with the help of the software package LaTeX and pdfLaTeX. In addition, the extension package `"knitr"` (Xie (2015a)) for the statistical software R was used again, which offers flexible possibilities to combine explanations with inputs and outputs of R.

Villingen-Schwenningen, October 2022

Matthias Kohl

Preface first edition

Statistics is everywhere today and we are steadily, knowingly or unknowingly, confronted with results of statistical procedures. Examples are internet search engines, targeted ads on websites, assessments of our creditworthiness, reference ranges of blood tests, weather forecast, election forecast, and many more. Often, statistical procedures are not appropriately applied or their results are not properly reported. Therefore, basic statistical knowledge is not only important in professional but also in everyday life and helps to distinguish between correct and incorrect information.

The basis of this book are my lecture notes of several statistics courses I gave in recent years at Furtwangen University, Campus Villingen-Schwenningen, in the framework of various bachelor and master programs as well as at Freiburg University in the framework of the international master program in biomedical sciences (IMBS).

As the title of the book already indicates, the introduction to statistical analysis happens by using the statistical software R (R Core Team (2022a)), a free software that is available for most operating systems. The R code used in the book is contained in the file `www.stamats.de/RCodeEN.zip` in form of text files with file extension `.R`. The R code of each chapter runs independent of the other chapters.

> Note:
> For the book several messages generated by R were wittingly suppressed to save space and to keep focus on the essentials. The suppressed messages are of no importance for the presented analyses. Conversely, you should be aware that there might be additional messages when you run the code contained in this book. This also includes innocuous warning messages.

The book was written using the software package LaTeX in combination with pdfLaTeX. In addition, the contributed package `"knitr"` (Xie (2015b)) of the statistical software R was applied, which offers flexible options for combining explanations with input and output of R.

Villingen-Schwenningen, August 2015

Matthias Kohl

1 Statistical Software R

The chapter includes a short introduction to the statistical software R where the following issues are covered:

- development history based on the statistical programming language S
- modular structure in form of packages
- installation on various operating systems
- installation of the integrated development environment (IDE) RStudio

Working with R in practice is introduced in the subsequent chapters in combination with the introduction to statistical data analysis.

1.1 R and its development history

The statistical software R (R Core Team (2022a)) is a free, non-commercial implementation of the statistical programming language R developed at the AT&T Bell Laboratories by Rick Becker, John Chambers and co-workers. It is a development environment and a programming language for statistics and graphics developed under GNU GPL-2/3 and therefore can be installed on arbitrary many computers without any restriction.

R is a function based language. That is, all actions are initiated by calling functions. In doing so additional parameters (arguments) are frequently passed to the functions controlling the concrete execution of the function. The function is identified by its name, the parameters by their name or also by their position. A call has the following structure (not always directly visible):

```
FunctionName(parameter1 = value1, parameter2 = value2, ..., parameterN = valueN)
```

We will see many examples in the course of the book.

We briefly summarize the development history of S and R:

05.05.1976: start of the development of version 1 of S (Chambers (2008, p. 476))

1980: release of version 2 of S (Chambers (2000))

1988: release of version 3 of S (S3) (Chambers (2000))

1992: start of the R project by Ross Ihaka and Robert Gentleman (Hornik (2008))

August 1993: first files of R published on Statlib (Ihaka (1998)).

Juni 1995: publication of the first GPL (GNU General Public License) version of R (Ihaka (1998))

05.12.1997: the R project officially becomes a GNU project (Ihaka (1997).

1998: release of version 4 of S (S4) (Chambers (2000))

29.02.2000: R 1.0.0 released, an implementation of S3 (Hornik (2008))

04.10.2004: R 2.0.0 released, an advanced version of S4 (Chambers (2008), Hornik (2008))

22.04.2010: R 2.11.0 released, support of Windows 64bit-systems (Dalgaard (2010))

03.04.2013: R 3.0.0 released, unlimited memory allocation in case of 64bit-systems (Dalgaard (2013))

18.06.2015: R 3.2.1 released, version used for writing the first edition of the book (Dalgaard (2015))

24.04.2020: R 4.0.0 released, reduced memory requirements, new color palletes and much more (Dalgaard (2020))

In general, there is a new release (version R x.y.0) in spring (March/April) of each year with patches released (R x.y.1, R x.y.2, etc.) over the year as necessary (R Core Team (2022c)).

The base system of R is developed by the so-called R Core Development Team currently consisting of 21 members (The R Foundation (2022a)). In addition, in 2002 the R Foundation (The R Foundation (2022b)) has been founded where the R Core Development Team members participate as ordinary members. The goals of the foundation include continuation of the development of R, the investigation of new methods, teaching and training in the area of computational statistics, and organisation of assemblies and conferences focused on computational statistics.

Furthermore, an R Consortium has been founded in June 2015 under the umbrella of the Linux Foundation for a stronger support of R from industry. Members are companies such as Microsoft, Google, Oracle, and HP (R Consortium (2022)).

Muenchen (2022) tries to estimate the popularity and the market share of data analysis software. The statistical software R performs well in all statistics and today plays a central and in some fields even leading role.

1.2 Structure of R

The statistical software R consists of packages that are organized in one or more libraries. There are three categories of packages. First of all, there are the **base packages** providing the basic functionality of R, which are maintained by the R Core Development Team. Currently, these are the following 14 packages: `"base"`, `"compiler"`, `"datasets"`, `"grDevices"`, `"graphics"`, `"grid"`, `"methods"`, `"parallel"`, `"splines"`, `"stats"`, `"stats4"`, `"tcltk"`, `"tools"`, `"utils"`; for more information see Section 5 in

the FAQs of R (Hornik (2022)).

The second group of packages, which are also part of the default installation of R, are the **recommended packages**. These packages mainly include additional, more complex statistical procedures. Currently, there are the following 15 packages: `"boot"`, `"class"`, `"cluster"`, `"codetools"`, `"foreign"`, `"KernSmooth"`, `"lattice"`, `"MASS"`, `"Matrix"`, `"mgcv"`, `"nlme"`, `"nnet"`, `"rpart"`, `"spatial"`, `"survival"` (Hornik (2022, Section 5)).

Finally, there are the **contributed packages**. Due to the open nature of R, anyone can contribute new packages anytime, which for sure is an important aspect for the success and the wide distribution of R. There is a continuously increasing developer community steadily contributing new packages to R, where the number of contributed packages grows exponentially for more than ten years now. Currently, there are clearly more than 25 000 packages (DataCamp Inc. (2022), Howson Ian (2022)). Those packages are spread over several so-called repositories. The largest number of packages are on CRAN (Comprehensive R Archive Network, http://cran.r-project.org/). It currently contains more than 18 000 packages. Contributed packages for the analysis of genomic data are mainly part of Bioconductor (Gentleman et al. (2004), http://www.bioconductor.org/), which currently provides more than 2 000 packages for download. A strongly growing number of R packages is hosted on GitHub (https://github.com/) and other git repositories.

1.3 Installation of R

The necessary files for installing R under Windows, Mac OS X, or Linux can be downloaded from CRAN (http://cran.r-project.org/) or one of its mirrors. In general, the installation of R does not differ from the installation of other software on these operating systems.

Windows The Windows installer for 32- and 64-bit can be found under http://cran.r-project.org/bin/windows/base/. Further information about the installation, updates or also uninstalling are included in the FAQs for Windows (Ripley and Murdoch (2022)). A short video about the installation is at https://youtu.be/6E3QJOc1bnM.

Mac OS X The necessary files for Mac OS X as well as a brief manual are given at http://cran.r-project.org/bin/macosx/. Similar to Windows there is also a FAQ page for Mac OS X (Iacus et al. (2022)) including additional information.

Linux There are files for

- Debian (http://cran.r-project.org/bin/linux/debian/, Ranke (2022))
- OpenSUSE (http://cran.r-project.org/bin/linux/suse/, Steuer (2022))
- Fedora, Red Hat Enterprise Linux (RHEL), CentOS, Scientific Linux, Oracle Linux (http://cran.r-project.org/bin/linux/fedora/, Úcar (2022))
- Ubuntu (http://cran.r-project.org/bin/linux/ubuntu/, Rutter (2022))

These websites include also brief manuals describing the installation.

The official and comprehensive documentation for the installation of R is the manual "R Installation and Administration" (R Core Team (2022d)). It also includes descriptions on how to install R from the source files.

1.4 Working with R

Starting R under Windows opens a simple graphical user interface (GUI) shown in Figure 1.1. One can

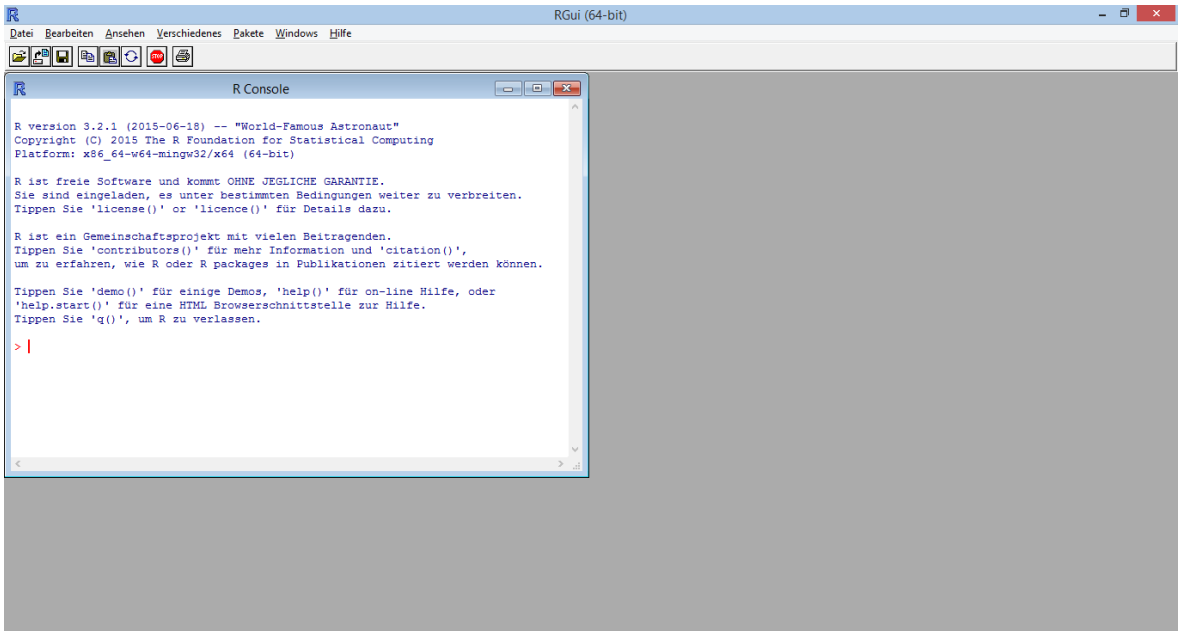


Figure 1.1: R GUI (64-bit) on Windows (German system).

now start to enter R commands in the *R Console* window. This works for simple computations but not for a real data analysis, which should be well documented and which we might want to repeat in the same or a slightly modified form for a different dataset. In this case it is recommended to generate a text file including the R commands. We can use any text editor for this purpose where it is common to use `.r` or `.R` as file extension. However, in programming it is common practice to go one step further and use a text editor with additional functionality or an integrated development environment (IDE).

Depending on the operating system there are several options. It seems that the largest functionality is currently provided by the free and open source IDE RStudio (`http://www.rstudio.org/`). It can be installed under Linux, Windows, and Mac OS X. I use it for data analysis as well as in my lectures. A short video about the installation under Windows is at `https://youtu.be/mHWiuLqsYtM`.

Figure 1.2 shows the RStudio IDE after installation on my Ubuntu Linux system. It looks very similar on Windows and Mac OS X. You can see three of the four panes. On the left hand side there is the *R Console*, in which the statistical software R is running. On the top of the right hand side the windows

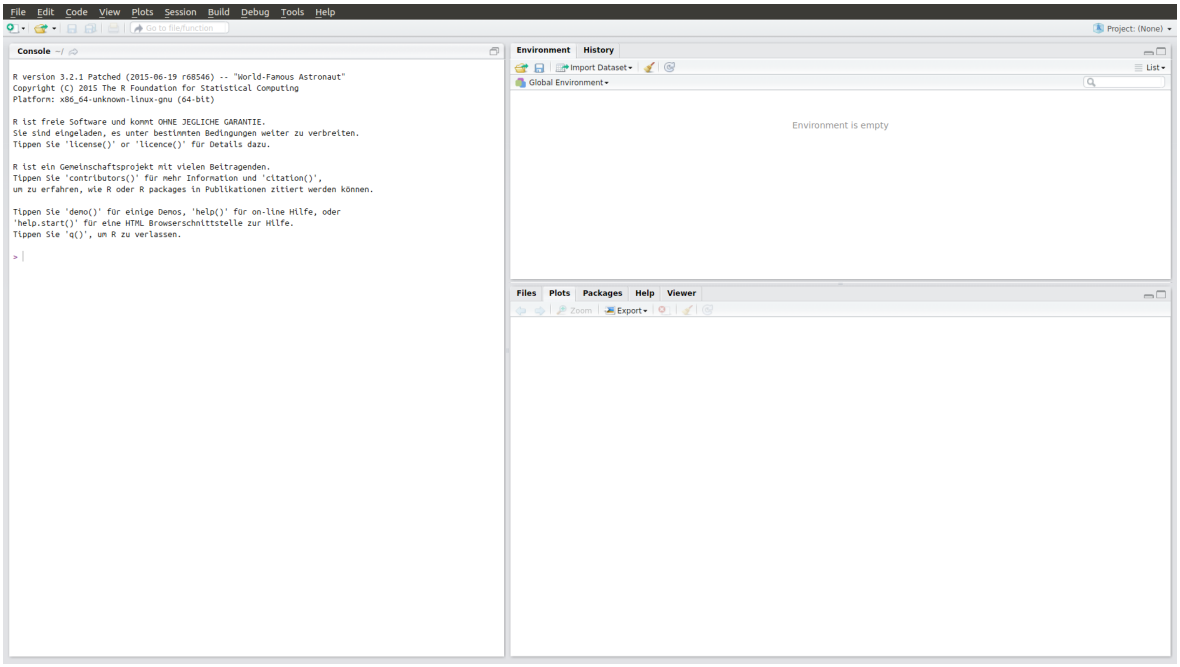


Figure 1.2: RStudio IDE after installation on Ubuntu Linux (German system).

Environment and *History* are shown. *Environment* shows all R objects that are currently loaded or were generated during the current session. As RStudio has just been started, the *Enviroment* is empty. The *History* contains an history of the R commands that have been executed. On the bottom of the right hand side there are the windows *Files*, *Plots*, *Packages*, *Help*, and *Viewer*. *Files* shows a file browser, which after the start shows the current working directory. Window *Plots* includes the plots generated in the current session and hence is empty immediately after starting RStudio. In window *Packages* all packages installed on the system are shown and can also be loaded via this window. Window *Help* provides several ways of help (local and online) for R and RStudio. Finally, in window *Viewer* local websites or web applications can be displayed.

After opening a new R script by using the menu item *File* → *New File* → *R Script*, a fourth window becomes visible (see Figure 1.3). It contains an empty and yet unnamed text file – a so-called R script. Later on, we will see that text input is supported by several interactive functions, which make it easier for beginners to write error free R code. Single R commands or also marked command blocks can be sent to the *R Console* for execution via the menu item *Run*. By means of the menu item *Source* the whole R script can be executed. The arrangement of the panes can be changed via the menu item `Tools` → `Global Options...` → `Pane Layout`. More details about RStudio will be presented in the course of this book.

1.5 Markdown in combination with R

In the recent years, Markdown was added to R in addition to the classic R scripts. Markdown is a simplified markup language. Its aim is to generate a document that is readable and understandable without further transformation (Wikipedia contributors (2022e)). The implementation of markdown in R is called

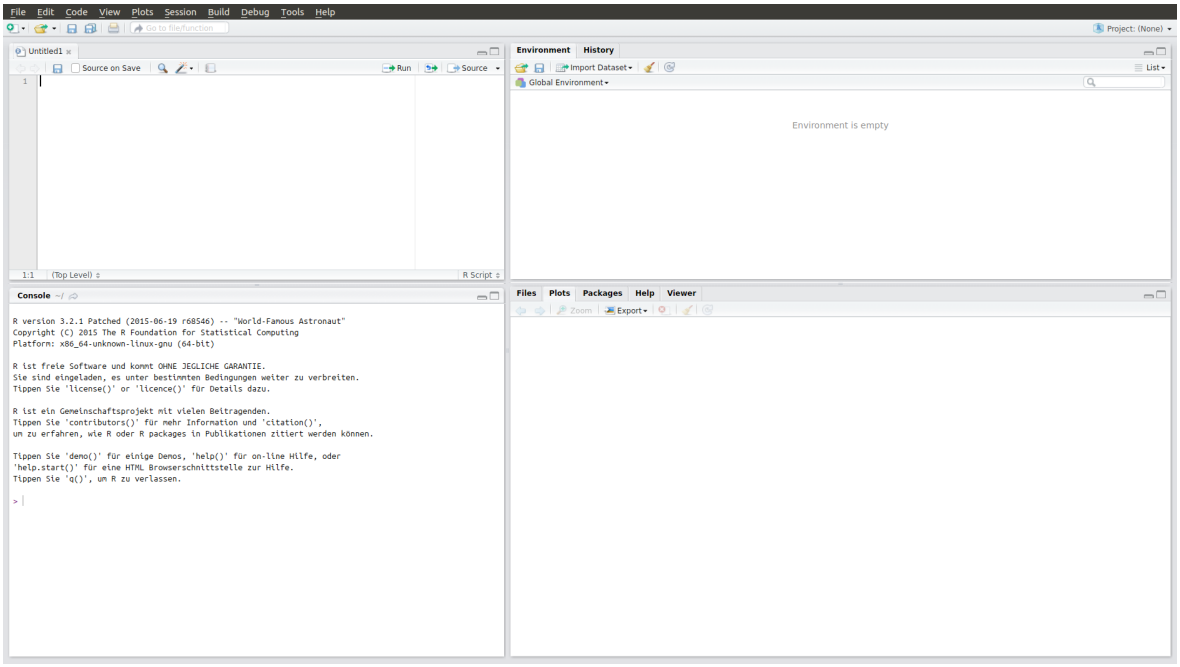


Figure 1.3: RStudio IDE after opening a new R script on Ubuntu Linux (German system).

R Markdown and is supported by RStudio (https://rmarkdown.rstudio.com/). The book Xie et al. (2018), which online version can be found under https://bookdown.org/yihui/rmarkdown/, contains a detailed Rmarkdown documentation

Markdown respectively, R Markdown is a very simple and easy to learn language. Markdown files are simply formatted text files, which can be translated into various formats (html, pdf, doc, etc.). The translation and the formatting of the file is controlled by the header of the file, which starts and ends with a line consisting of `---`. The translation from simple text file to different formats is carried out with the help of various R packages, such as "`knitr`" (Xie (2015a)) and "`rmarkdown`" (Xie et al. (2018)).

For each of the following chapters of the book there is an R Markdown file (file extension: Rmd) containing the R code of the respective chapter in the form of so-called R code chunks. The R Code Chunks begin with ```` ```r ````, and ends with ```` ``` ````. After the letter `r`, additional options can be added. Details about this can be found in the online version of Xie (2015a) at https://yihui.org/knitr/. Between the code chunks any additional text can be added and allows the user to create their own documentation of the individual chapter.

1.6 Exercises

1. Install R and RStudio on your personal computer, notebook, etc.
2. Start RStudio, open a new R script and take a close look at all opened windows and all menu items.
3. Start RStudio, create a new R Markdown file. This requires the installation of additional package,

which should be carried out automatically. Select HTML as output format. You will notice that the generated new Rmd file is not empty, but already contains some examples. Compile the file by using “knit”.

4. Acquaint yourself with the help options available in window *Help*.

5. Check, if the base and recommended packages are installed on your system (window *Packages*). Which R packages are checked after starting RStudio and hence are active, i.e. are loaded and can immediately be applied?

6. Acquaint yourself with R Markdown.

2 Descriptive Statistics

The chapter is about descriptive statistics where the following topics are covered:

- Interplay of probability theory, descriptive and inferential statistics
- Types of attributes and scales of measurement
- Basic function for data import and export with R
- Data import of text files with RStudio
- Frequency tables, bar and pie charts
- Mode, quantile, quartile, median, range, interquartile range (IQR), MAD, box-and-whisker plot
- Cross table, ϕ-coefficient, Pearson's contingency coefficient, Cramér's V
- Spearman's ρ, Kendall's τ, scatter plot
- Arithmetic mean, geometric mean, standard deviation, coefficient of variation, quartile coefficient of dispersion
- histogram, density estimation
- Pearson (product-moment) correlation coefficient

The R code of this chapter is included in the R Markdown file `DescriptiveStatistics.Rmd`, which you can download from my GitHub account (link: https://github.com/stamats/ISDR/blob/main/DescriptiveStatistics.Rmd). Right click on `Raw`. Then you can `Save target as ...`. The least difficulties arise, if you save my R Markdown files in the same folder as the data.

2.1 Basics

Figure 2.1 provides an overview of the interplay between probability theory, descriptive and inferential statistics. The starting point is a **population** or **universe** that has to be clearly characterized. The goal is to obtain some (new, important) insights about this population, e.g. which party will get how many votes in the next election or which disease occurs with which frequency. A complete survey in most cases is impossible, as for instance it would be to expensive due to the size of the population, or as the population is continuously changing over time.

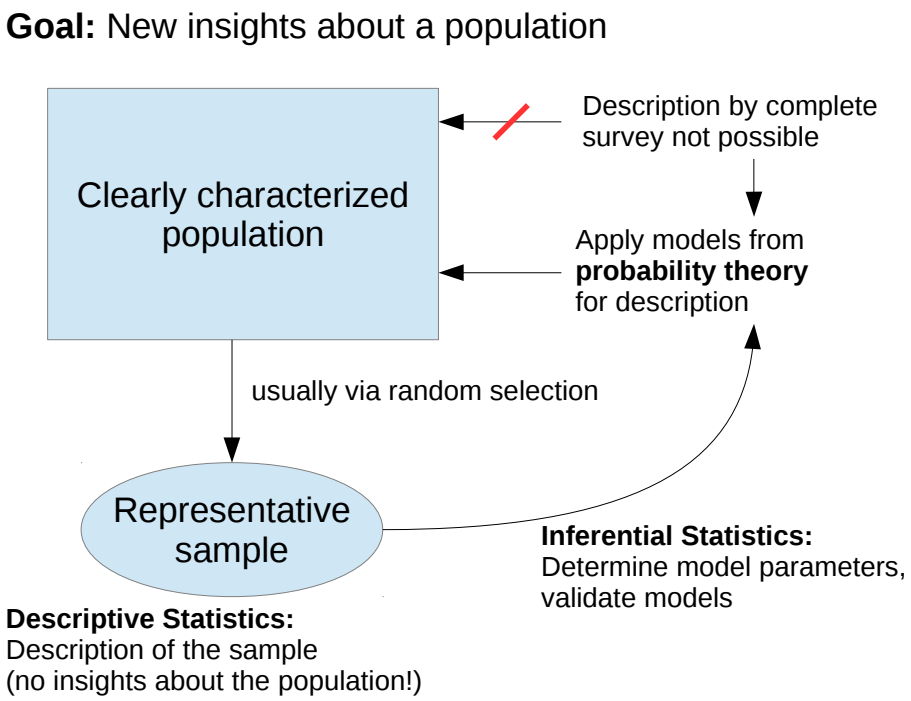


Figure 2.1: Interplay between probability theory, descriptive and inferential statistics.

The statistical way out consists of postulating **models from probability theory** where the model parameters are unknown and have to be determined. For this purpose a **representative sample** is drawn from the population, usually via random selection. The task of **descriptive statistics** is to characterize this random sample as accurately as possible. That is, descriptive statistics gains no insights about the population, but describes "only" the (randomly) selected part from it. Descriptive statistics helps to become acquainted with the data and to identify uncommon or erroneous values in the data. As a consequence, it also makes an important contribution to inferential statistics, as valid inference is only possible by knowing the data and the data quality ("garbage in, garbage out").

The goal of **inferential statistics** is to draw inferences from a representative sample about the corresponding population. An important part is to determine (estimate) the unknown parameters of assumed probability models from the available data. In addition, the validity of existing models can be examined.

Note:
We are dealing with models, i.e. we should not assume that these models exactly reflect the reality. Instead, the models under certain assumptions and at a certain time point offer a quite good description of reality. In this sense, one should interpret the following quote of the famous statistician George E.P. Box (Box and Draper (1987, p. 424)):

"'Essentially, all models are wrong, but some are useful."'

The following example demonstrates that model selection is crucial for the result and that identical data under different assumptions may lead to contradictory results.

Example 2.1. In the Second World War, the goal was to better protect American bombers against fire of the German air defense. For this purpose, the location and number of bullet hols of returning airplanes were analyzed. Based on the collected information the Army concluded that the locations with extraordinary many hits should get an additional armor. A plausible result under the assumption that the German air defense especially aims at these parts of the air planes.
In contrast, the statistician Abraham Wald assumed in his analysis that the hits should be uniformly distributed over the air planes (Wald (1980)). Since this was not the case for the returning air planes, he concluded that the not returning air planes were hit at very vulnerable locations and hence crashed. Consequentially, he recommended to add amor at places where the returning air planes had no or only a few hits.

The elements of a population – which might be persons, items, etc. – are described by a number of **attributes** (variables). These attributes can be divided into several **types of attributes** as shown in Figure 2.2. The main distinction is between qualitative (categorical) and quantitative (metric) attributes. These two categories can be divided by the so-called **scales of measurement** into nominal, ordinal, interval and ratio scaled, where nominal is the lowest and ratio scaled the highest level. In dependence of the scale of measurement, certain arithmetic operations are allowed, where the number of allowed operations increase from the left hand side (nominal) to the right hand side (ratio scaled). Therefore, it is important to know the scales of measurement of the investigated variables. Otherwise, the measured values of the variables – the so-called levels of the attributes – could for instance be wrongly described by descriptive statistical methods.

Note:
The bounds between the scales of measurements are partly fluent; e.g., in practice, a medical score with many levels is often treated like a metric variable.

The information content of variables increases with the scale of measurement. Thus, during the design of a study, one should ideally select a variable with the highest possible scale of measurement to describe an attribute. Unfortunately, this is not always possible in practice, as the measurement of more informative variables usually requires more efforts and is more expensive. As a consequence, one can not always avoid to select a less informative variable for a study.

We consider an example.

Example 2.2. Our goal is to characterize the age distribution of a sample or of the respective population. In this case, the date of birth would be more informative than age in years or age groups, where the effort to collect the data is more or less the same for all three options. Hence, the date of birth should be selected. Furthermore, this selection offers the opportunity to restrict the statistical analysis to age in

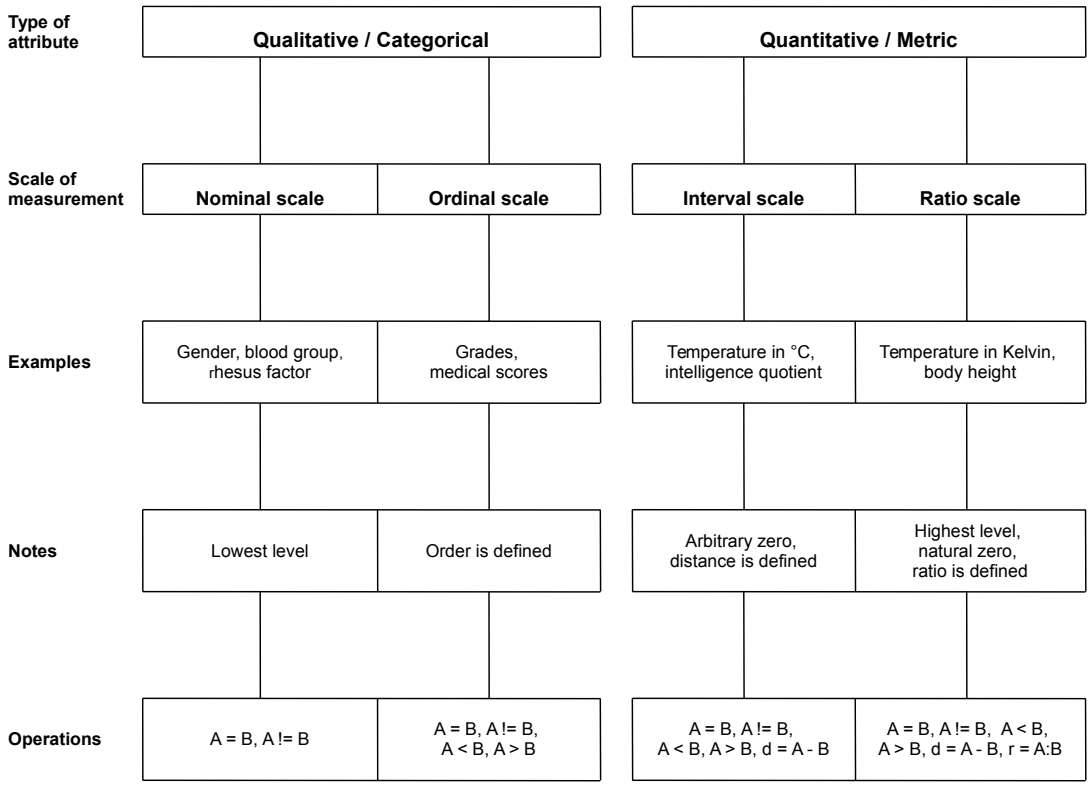


Figure 2.2: Types of attributes and scales of measurement.

years or age groups if it turns out later, that the additional information provided by date of birth is not needed or irrelevant.

2.2 Excursus: Data Import and Export with R

Before we can start with a descriptive analysis, we must first plan and conduct a study and collect data. In doing so, a variety of things have to be considered. We do not elaborate on those things here, as it would go beyond the scope of the book.

In larger studies, the collected data is often saved in specifically designed databases, in smaller studies one or several files of a spreadsheet software are usually used. For the statistical analysis the organization of the data in a table is most effective, where some redundancy may be very helpful. For more details we refer to Broman and Woo (2018).

Nowadays, most software offers an option to export data, where in allmost all cases on cen export the data in one or or several text files. Therefore, we will only consider data import from text files in this section. Beyond this, R offers a variety of options to import data such as the import of files from other statistical software packages or from spread sheet software or by interfaces to databases. An overview of

the various options for data import and export is included in manual “R Data Import/ Export” (R Core Team (2022b)).

The starting point for reading data from text files is function `scan`. With this function, data can be imported from the console or a text file. However, in most cases one needs not to directly apply function `scan`, but one can use function `read.table`, which is much simpler to handle. Furthermore, there are functions `read.csv`, `read.csv2`, `read.delim` oder `read.delim2` that are even more specialized; see Table 2.1.

Function name	**Description**
`scan`	Read data from console or a text file.
`read.table`	Read data from a text file in spreadsheet format.
`read.csv`	Special case of `read.table` with decimal point “.” and column separator “,” (“English csv-file”).
`read.csv2`	Special case of `read.table` with decimal point “,” and column separator “;” (“German csv-file”).
`read.delim`	Special case of `read.table` with decimal point “.” and column separator “\t” (tab).
`read.delim2`	Special case of `read.table` with decimal point “,” and column separator “\t” (tab).

Table 2.1: Overview of some basic functions for data import with R.

We can also use RStudio to import text files, which is especially helpful for beginners. In window *Environment* there is menu item *Import Dataset*. After selecting *From Text File...* a window opens for choosing a text file. After choosing a text file, the window shown in Figure 2.3 opens. The provided options correspond to the most important arguments of the `read.*` functions. The data is imported via one of the `read.*` functions, where the call for reading in the data is subsequently shown in figure *History*. To ensure the exact reproducibility of the import, the R code shown in figure *History* should be transferred to the current R script via the menu item *To Source*.

> Note:
> Even if the import fails, which for instance may happen if special characters are included in the file path, the R code for reading in the data is generated in window *History*. By transfering this R code to the current R script, making necessary corrections (e.g. correcting the file path) and re-running the R code one can after all import the file.

For using the result of the import for subsequent analyses, it must be assigned to some variable. The name of the variable can be specified in field *Name* (see Fig. 2.3). After the import, a data object with the chosen name is visible in window *Environment*; see Figure 2.4. The data object can be viewed in the editor window by clicking on its name.
The data object is a so-called `data.frame`, the basic data structure in R for saving datasets. It is similar

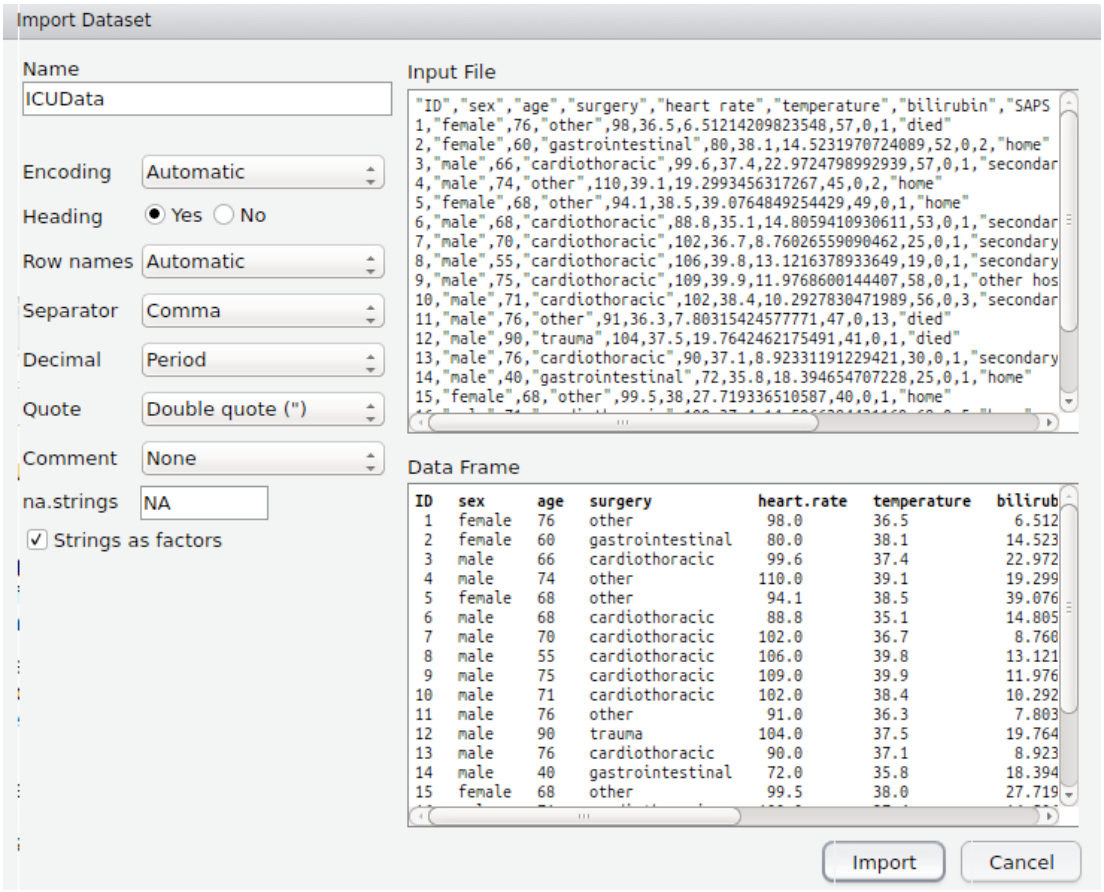


Figure 2.3: RStudio window for import of text files.

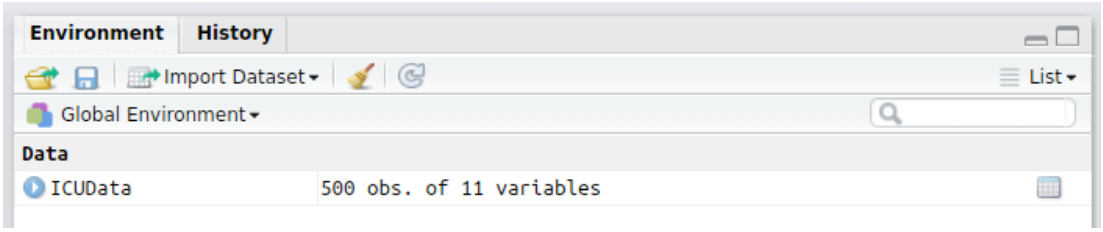


Figure 2.4: RStudio window *Environment* with a data object.

to a table in a spreadsheet program. The columns correspond to the variables (attributes), the rows represent the observed levels of the studied subjects.

The counterpart to the introduced `read.*` functions for exporting data are the functions `write.table`, `write.csv`, and `write.csv2`. If you work with English system settings, you should use `write.csv` for exporting data. The generated file can then be opened without problems in a current spreadsheet software.

Another form of data import are functions `load` and `readRDS`, whichcan be applied to load so-called `.RData` respectively `.rds` files. These files have been generated by R function `save` or `save.image` respectively `saveRDS`. With these functions one can save single objects (`save` or `saveRDS`) or the entire

content of an R session (`save.image`) in an `.Rdata` respectively `.rds` file. In addition, one can specify if the file should be compressed (default) or not.

2.3 Import of ICU-Dataset

In this section, we read in the `ICUData.csv` dataset, which we will analyze in the book in various ways. It consists of data from 500 patients of an intensive care unit (ICU). The data is not from real patients, but I have generated it based on my long-term experience with data of intensive care patients. The data is similar to real data with respect to many aspects.

Please, use the following steps to import the dataset:

1. Download the dataset from my GitHub-Account and save it on your computer. Avoid using special characters in the file path.
   Link: https://github.com/stamats/ISDR/blob/main/ICUData.csv
   Right click on `Raw` and then you can `Save target as...`.

2. Start RStudio.

3. Change the working directory. Click on ... in window *Files* (at right edge) and select the folder, in which you have saved `ICUData.csv`. Next, click on *More* → *Set As Working Directory*.

4. Check the working directory by entering the following R code in window *Console*

   ```
   1 getwd()
   ```

   followed by the *Enter/Return*-key. The output should correspond to the folder, in which you have saved file `ICUData.csv`. If not, please repeat the above steps again.

5. Save the R Markdown file `DescriptiveStatistics.Rmd` in the same folder as the dataset and open it with RStudio.

6. Import the ICU dataset by running the following R code.

   ```
   1 ICUData ← read.csv(file = "ICUData.csv")
   ```

   In your R markdown file, place the cursor in the line with the above R code and click on *Run*. By doing this, the R code is copied to window *Console* and executed. There should be no output. In case there is an error message – probably

   ```
   Error in file(file, "rt"): cannot open the connection
   ```

   either saving the file or changing the working directory has not worked properly. Please, check steps 3 and 4 and run the R code again as described above.
   Alternatively you can import the data directly from my website.

```
Link ← "https://raw.githubusercontent.com/stamats/ISDR/main/ICUData.csv"
ICUData ← read.csv(file = Link)
```

In this case you should check your working directory, too and set it to the folder that include the respective R markdown file.
As a further alternative, you can use the import function of RStudio as described in Section 2.2. Please make sure that your settings match the settings visible in Figure 2.3.

7. Take a look at window *Environment* and check if there is object `ICUData` in the field *Data* (see Fig. 2.4). It must be an object of type `data.frame` with 500 observations (obs.) of 11 variables. If this is true, the import was successful.

Note:
In step 7 we have used the assignment operator `<-` to assign the result of the import via `read.csv` the name `ICUData`. That is, the data are saved in a `data.frame` with name `ICUData` and we can use this object for further analysis.

Although the import looks successful at the first glance, it is still possible that the dataset was not imported as required. Thus, I strongly recommend to check the import more precisely. First, one can use function `View` to take a closer look at the imported dataset – if it is not too large.

```
View(ICUData)
```

You can also achieve this by clicking on the name of the dataset in window *Environment* of RStudio. By doing this, one can for instance see, if the column names and row names (if any) were correctly transferred, if the entries in the columns are correct, and if there are empty lines or columns. As different data types look identical or very similar in this view, one should also take a closer look at the structure of the dataset. For this purpose function `str` is provided.

```
str(ICUData)
```

```
'data.frame':   500 obs. of  11 variables:
$ ID : int 1 2 3 4 5 6 7 8 9 10 ...
$ sex : chr "female" "female" "male" "male" ...
$ age : int 76 60 66 74 68 68 70 55 75 71 ...
$ surgery : chr "other" "gastrointestinal" "cardiothoracic" "other" ...
$ heart.rate : num 98 80 99.6 110 94.1 88.8 102 106 109 102 ...
$ temperature : num 36.5 38.1 37.4 39.1 38.5 35.1 36.7 39.8 39.9 38.4 ...
$ bilirubin : num 6.51 14.52 22.97 19.3 39.08 ...
$ SAPS.II : int 57 52 57 45 49 53 25 19 58 56 ...
$ liver.failure: int 0 0 0 0 0 0 0 0 0 0 ...
$ LOS : int 1 2 1 2 1 1 1 1 1 3 ...
$ outcome : chr "died" "home" "secondary care/rehab" "home" ...
```

A similar result one can obtain in window *Environment* of RStudio by clicking on the blue arrow symbol in front of `ICUData` in the field *Data*. The result is shown in Figure 2.5.

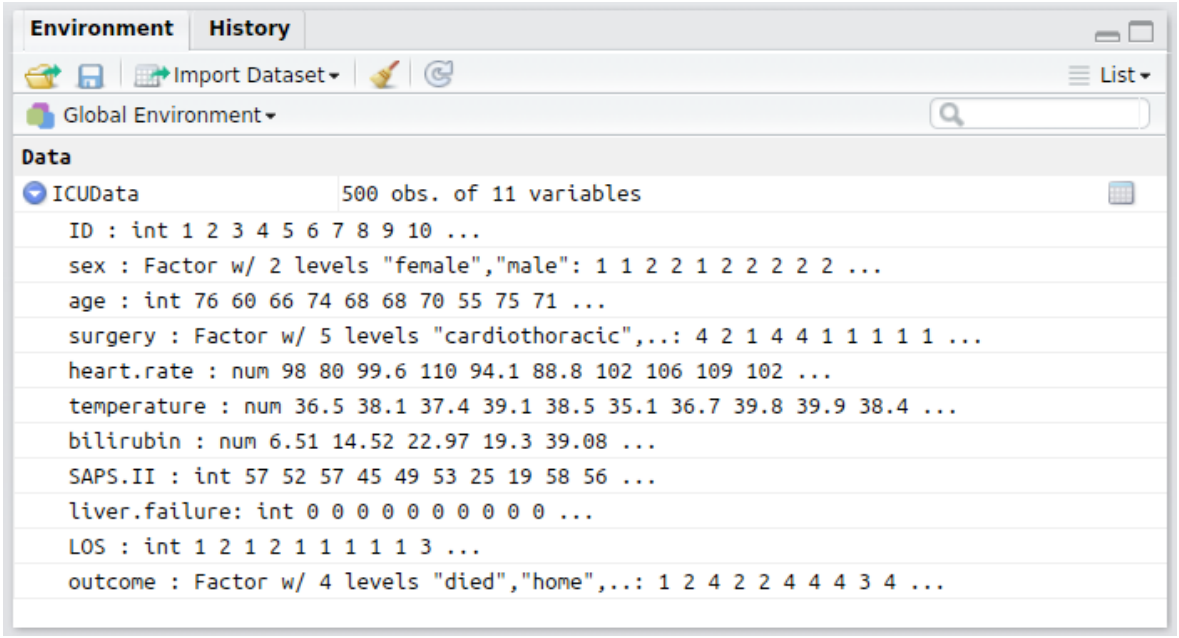


Figure 2.5: View of the exact structure of a dataset in RStudio.

The two displays are not identical in this case. This is because the import shown in Figure 2.5 was created with an older version of R. In R version 4.0.0 an important change was made here and the argument `stringsAsFactors` was changed from `TRUE` to `FALSE`. Since it makes sense for the purposes of our analyses to read the "string" variables as factors (categorical variables), we import the data again. This time with the additional setting `TRUE` for the argument `stringsAsFactors`.

```
ICUData ← read.csv(file = "ICUData.csv", stringsAsFactors = TRUE)
## resp.
Link ← "https://raw.githubusercontent.com/stamats/ISDR/main/ICUData.csv"
ICUData ← read.csv(file = Link, stringsAsFactors = TRUE)
```

We take another look at the structor of the imported data.

```
str(ICUData)
```

```
'data.frame':   500 obs. of  11 variables:
$ ID : int 1 2 3 4 5 6 7 8 9 10 ...
$ sex : Factor w/ 2 levels "female","male": 1 1 2 2 1 2 2 2 2 2 ...
$ age : int 76 60 66 74 68 68 70 55 75 71 ...
$ surgery : Factor w/ 5 levels "cardiothoracic",..: 4 2 1 4 4 1 1 1 1 1
  ...
$ heart.rate : num 98 80 99.6 110 94.1 88.8 102 106 109 102 ...
$ temperature : num 36.5 38.1 37.4 39.1 38.5 35.1 36.7 39.8 39.9 38.4 ...
$ bilirubin : num 6.51 14.52 22.97 19.3 39.08 ...
$ SAPS.II : int 57 52 57 45 49 53 25 19 58 56 ...
$ liver.failure: int 0 0 0 0 0 0 0 0 0 0 ...
$ LOS : int 1 2 1 2 1 1 1 1 1 3 ...
$ outcome : Factor w/ 4 levels "died","home",..: 1 2 4 2 2 4 4 4 3 4 ...
```

The dataset consists of the following variables:

ID: consecutive numbers (`integer`) from 1 to 500 for identification of the patients

sex: a nominal variable (`Factor`) with levels: female and male

age: age in years (`integer`)

surgery: kind of surgery, nominal variable (`Factor`) with levels: cardiothoracic, gastrointestinal, neuro, other, and trauma

heart.rate: maximum heart rate in beats per minute (`numeric` = real number) during the entire stay on the ICU

temperature: maximum body temperature in °C (`numeric`) during the entire stay on the ICU

bilirubin: maximum level of bilirubin in μmol/l (`numeric`) during the entire stay on the ICU. The red dye of human blood is digraded and as an intermediate stage bilirubin emerges, a yellowish substance. Standard values are below 21 μmol/l where higher values for instance may indicate liver problems (Wikipedia contributors (2022b)).

SAPS.II: SAPS-II Score (`integer`) at admission to the ICU. The score reflects the physiological condition of a patient and is used to estimate the severity of disease. The higher the score the more severe is the disease. The range of values is from 0 to 163, where the values are associated with a probability of dying (Wikipedia contributors (2021)).

liver.failure: presence of liver failure (`integer`) where 0 and 1 indicate no and yes, respectively; that is, strictly speaking this is a nominal variable coded by numbers.

LOS: length of stay on the ICU in days (`integer`)

outcome: kind of discharge from the ICU (`Factor`). The possible levels are: died, home, other hospital, and secondary care/rehab.

Note:
The names of the variables `heart.rate`, `SAPS.II`, and `liver.failure` were changed during import. The respective column names include a blank and hence are no syntactically correct variable names in R. Such changes are done automatically during import. One can avoid it by setting the parameter `check.names`. The respective R code would be

```
ICUData ← read.csv(file = "ICUData.csv", check.names = FALSE)
```

However, `check.names = FALSE` should only be used after some experience in working with R, as it may lead to certain unwanted side effects and problems.

2.4 Excursus: Installation of Contributed Packages

First, make sure that you have an active internet connections. For installing packages one needs to apply function `install.packages`. We install the contributed packages `"DescTools"` (Andri et mult. al. (2022)), `"scales"` (Wickham and Seidel (2022)), `"ggplot2"` (Wickham (2009)) and `"MKdescr"` (Kohl (2022a)), which we will use in this chapter.

```
1 install.packages(c("DescTools", "scales", "ggplot2", "MKdescr"))
```

With function `c` (short for concatenate) the four package names are concatenated to a single vector and can be installed together.

> Note:
> When you install for the first time a contributed package, the installation may not start immediately, but a new window may open up, in which the path to the library, in which the package shall be installed, is requested. Here, I would recommend to use the default setting of your operating system; i.e., if necessary, select this option and confirm it.

RStudio offers an alternative way to install a package via the window *Packages*. One can open a window for installation via the menu item *Install*; see Figure 2.6. The default settings shown in the window

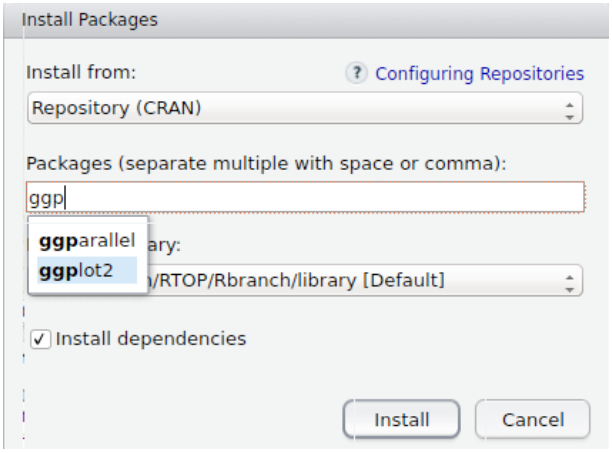


Figure 2.6: Installation of R packages in RStudio.

should only be changed, if you are very experienced in using R. In particular, it is important that *Install dependencies* is checked. Since most of the R packages require other R packages to work properly, this option ensures that these additional R package are automatically installed together with the specified package.

> Note:
> Please run the installation only once. Afterwards, the packages are installed and need not be reinstalled every time when you run an analysis. On the contrary, a repeated installation may damage the current installation and the respective package can not be used (loaded) any more. In such a case, you should close RStudio without saving the workspace and afterwards start it again. After the start the workspace

should be empty. If this is not the case, you have a file named `.RData` in your current working directory that is automatically loaded at start-up. In such a case, you should close RStudio again and then remove this file. Afterwards, start RStudio again and run function `remove.packages` with the name of the package that does not load any more. Finally, install the respective package again; for instance

```
remove.packages("ggplot2")
install.packages("ggplot2")
```

As explained in Section 1.2, there are thousands of R packages. Hence, it is reasonable, that installed package are not loaded automatically. Otherwise your system would become slower and slower the more package you have installed. Instead, all packages except the base packages (see Section 1.2) must be explicitly loaded via function `library`. We load the four packages we have installed previously.

```
library(DescTools)
library(scales)
library(ggplot2)
library(MKdescr)
```

In most of the cases, a successful loading of a package can only be recongnized by the lack of an error message. However, nowadays more and more packages print short messages during the load process confirming the successful loading. Sometimes there are also warnings. One should carefully read such warnings, but in most of the cases they can be ignored. Repeatedly running `library` is unproblematic. The function recognizes that a given package is already loaded and does not reload the package.

Note:
Sometimes the installation of a package may also fail as the installation of dependent packages failed. In such a case, you should read the error message(s) of the installation or loading of the package very carefully and you should try to remove the errors step by step. Because of the wide distribution of R, searching through the internet may help to find a solution in case of unknown or unclear error messages.

2.5 Categorical Variables

2.5.1 Univariate Analysis

First, we consider all variables separately (univariate) and start with nominal variables. That is, we analyze a single variable, whose levels are a set of possible names without any ordering. Examples are sex, blood group, rhesus factor, or also surgery, liver failure and outcome as in case of our ICU dataset (cf. Section 2.3).

Please first import the ICU dataset as described in Section 2.3, if you have not done it yet. In addition, install the contributed packages `"DescTools"` (Andri et mult. al. (2022)) and `"ggplot2"` (Wickham

(2009)) as explained in Section 2.4.

In case of nominal variables, descriptive statistics consists of calculating and visualizing absolute and relative frequencies. With the following R Code we compute the **absolute frequencies** of the kind of surgery the ICU patients obtained.

```
table(ICUData$surgery)
```

```
cardiothoracic gastrointestinal             neuro             other
           223               79                46               121
       trauma
            31
```

The computation is done by function `table`. With symbol `$` we can access the variables of a dataset (`data.frame`). In this case, we access variable `surgery`, which includes the kind of surgery. We obtain the **relative frequencies** by dividing these numbers by the number of patients. This is also called the **empirical frequency distribution**. It is not recommended to use 500 here, even if it would be correct. It is better and more general to divide by the number of rows of the dataset, which can be obtained by function `nrow`.

```
table(ICUData$surgery)/nrow(ICUData)
```

```
cardiothoracic gastrointestinal             neuro             other
         0.446             0.158             0.092             0.242
       trauma
         0.062
```

That is, almost half of the patients underwent a cardiothoracic surgery. This most frequent level is also called **mode**. At second position, we have the other surgeries, followed by gastrointestinal surgeries. The smallest number of surgeries were caused by trauma, slightly more by neurological causes. A quicker way to calculate absolute and relative frequencies is offered by function `Freq` from package `"DescTools"` (Andri et mult. al. (2022)).

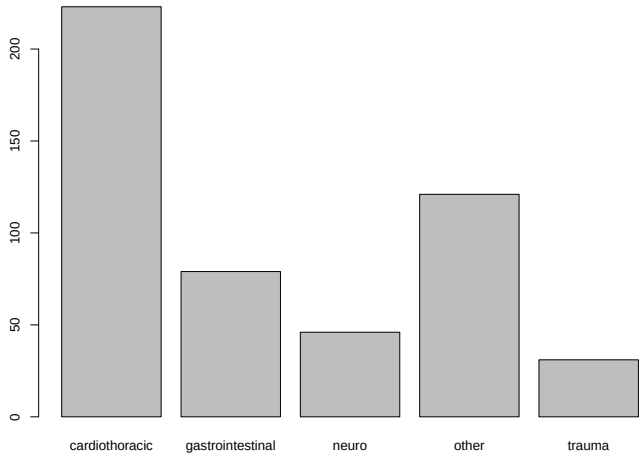
```
Freq(ICUData$surgery)
```

```
              level  freq   perc  cumfreq  cumperc
1    cardiothoracic   223  44.6%      223    44.6%
2  gastrointestinal    79  15.8%      302    60.4%
3             neuro    46   9.2%      348    69.6%
4             other   121  24.2%      469    93.8%
5           trauma    31   6.2%      500   100.0%
```

The function `Freq` additionally computes the **cumulative** absolute and relative frequencies.

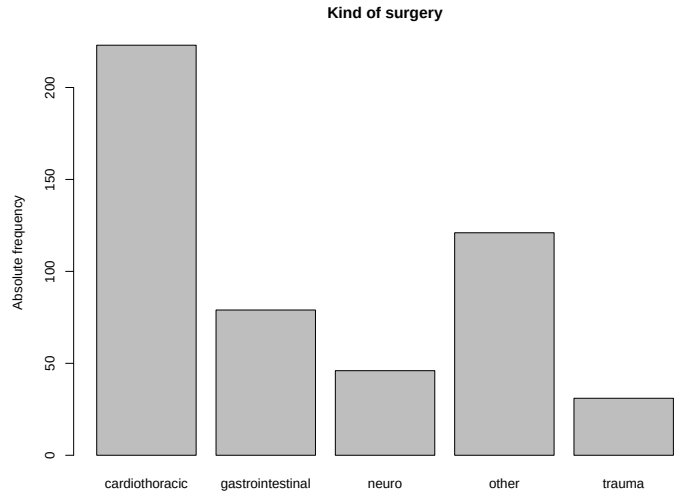
The graphical representation of relative and absolute frequencies is best done by **bar plots**. We first depict the absolute frequencies applying function `barplot`.

```
1 barplot(table(ICUData$surgery))
```



We add a title (argument `main`) and label the y axis (argument `ylab`) of the bar plot.

```
1 barplot(table(ICUData$surgery), main = "Kind of surgery",
2         ylab = "Absolute frequency")
```



There are many more arguments that can be used to further adapt the plot. We will get to know some more of them in the course of the book. Various examples of how to configure bar plots are also provided by the help page of `barplot`, which will be shown in Window *Help* of RStudio after running `?barplot`. Alternatively, one can search for help using the search field included in window *Help* of RStudio.

Current versions of RStudio also offers an interactive way of help. If you start writing code in an R script, the names of matching objects and, with some delay, matching help is shown; see Figure 2.7. By

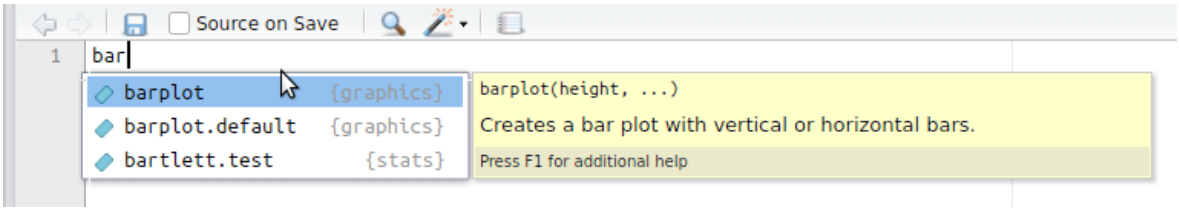


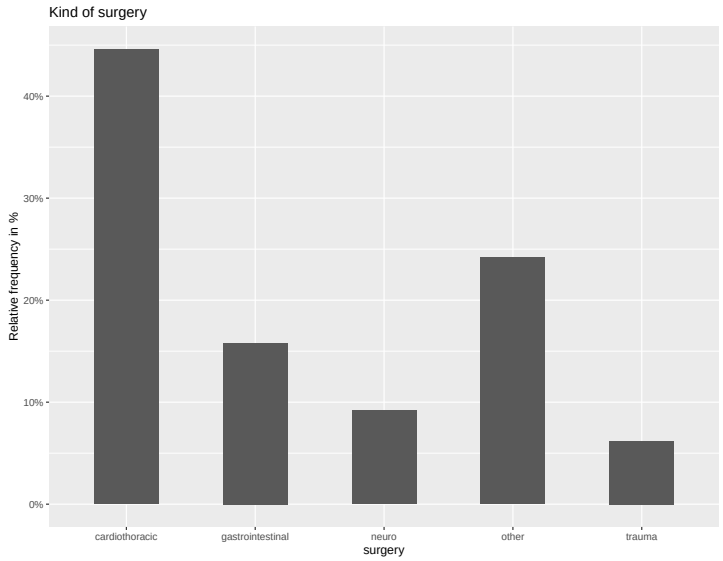
Figure 2.7: Interactive context based help in RStudio.

pressing the F1 key, the related help page opens in window *Help*.

A bar plot of the relative frequencies can be generated with a very similar R code as in case of the absolute frequencies. One just has to replace the absolute by relative frequencies. In addition to the standard graphics, there are other graphic systems implemented in R. Currently, the most frequently used system beside the standard system is probably the implementation of grammar of graphics in package `"ggplot2"` (Wickham (2009)). It is one of the packages that made an important contribution to the success of R. Thus, we use this system to display the relative frequencies.

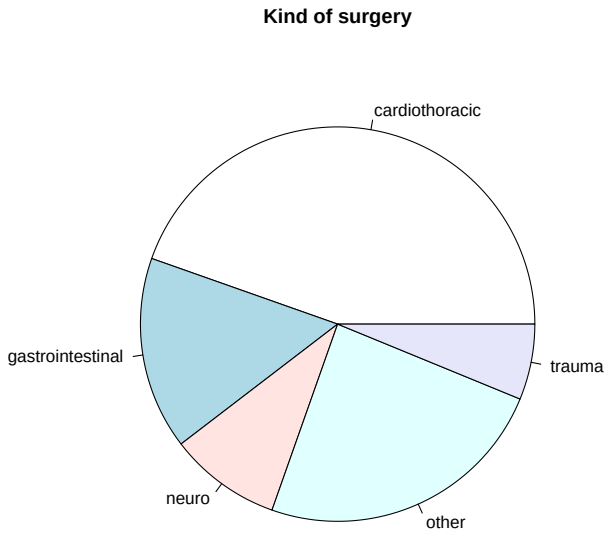
We generate a bar plot of the relative frequencies using functions `ggplot` and `geom_bar`, where the width of the bars is reduced by argument `width`. With the help of function `aes` we can set the representation of the data. In the case at hand, we use the relative frequencies as percentages, which are calculated by the fixed expression `(..count..)/sum(..count..)`. For the scaling of the y-axis there are many options, which can be generated by `scale_y_*`. In this case we use a continuous scaling of the y-axis, where we define the label format using `percent_format`. This function is included in the package `"scales"` (Wickham and Seidel (2022)). Finally, the functions `ggtitle` and `ylab` are applied to add a title and label the y axis of the plot.

```
## Assign data
ggplot(ICUData, aes(x=surgery)) +
  ## Add bars with relative frequencies
  geom_bar(aes(y = (..count..)/sum(..count..)), width = 0.5) +
  ## The percent of relative frequency
  scale_y_continuous(labels = percent_format(accuracy = 1)) +
  ## Title and label of y axis
  ggtitle("Kind of surgery") + ylab("Relative frequency in %")
```



In practice, pie charts are frequently used instead of bar plots. Of course, this is also possible with R. The respective function is `pie`.

```
pie(table(ICUData$surgery), main = "Kind of surgery")
```



This kind of diagram has some drawbacks (see also Chapter 3). On the help page of `pie` you can read:

> "Pie charts are a very bad way of displaying information. The eye is good at judging linear measures and bad at judging relative areas. A bar chart or dot chart is a preferable way of displaying this type of data."

Thus, it is better to use a bar plot or dot chart to make the representation easier to read for the human eye.

Note:
The use of appropriate colors and diagrams is in more detail described in Chapter 3.

In the sequel, we additionally assume that the categories are ordered; that is, we consider ordinal variables. The ordering offers several additional ways for statistical analysis. In particular, quantiles are applicable for various purposes.

Definition 2.3 (Quantile)**.** *Let $x_1, x_2, \dots, x_n \in \mathbb{R}$ ($n \in \mathbb{N}$) be some observations and let $x_{(1)}, x_{(2)}, \dots, x_{(n)}$ be the increasingly sorted observations. Then, the **α-quantile** for $\alpha \in (0,1)$ is defined by*

$$q_\alpha = \begin{cases} x_{\left(ceiling(n\alpha)\right)} & \text{if } n\alpha \notin \mathbb{Z} \\ [x_{(n\alpha)}, x_{(n\alpha+1)}] & \text{if } n\alpha \in \mathbb{Z} \end{cases} \tag{2.1}$$

The following remark includes some additional explanations about α-quantiles.

Remark 2.4. ***(a)*** *If $n\alpha$ is no integer, the α-quantile corresponds to the ceiling$(n\alpha)$-th observation. Here "ceiling" means rounding to the next larger integer. In R there is function* `ceiling`*; e.g.*

```
ceiling(c(2.01, 3.88))
```

```
[1] 3 4
```

If $n\alpha$ is integer, the α-quantile is not unique and all values in the bounded interval $[x_{(n\alpha)}, x_{(n\alpha+1)})$ are valid α-quantiles. In practice, this is not satisfactory. Therefore, there is a number of proposals regarding the value of the interval that should by chosen as representative of the α-quantile. The most obvious approach probably is to use the midpoint of the interval. In R function `quantile` *nine different approaches are implemented; see also Example 2.5.*

(b) *Important special cases of quantiles are **percentiles** for $\alpha \in \{0.01, 0.02, \dots, 0.99, 1.00\}$, **quartiles** for $\alpha \in \{0.25, 0.50, 0.75\}$, and the **median** for $\alpha = 0.5$.*

Example 2.5. We consider the numbers $2, 4, 6, \dots, 20$ and want to compute the 20-th percentile, i.e. $\alpha = 0.2$. Hence, we get $n\alpha = 10 \cdot 0.2 = 2$. Therefore, the 20-th percentile is each number in the bounded interval $[x_{(2)}, x_{(3)}] = [4, 6]$. For performing this computation in R, we first have to enter the data. In the case at hand, the functions `c` (short for concatenate) or `seq` (short for sequence) can be used.

```
## Concatenating numbers to a vector
x ← c(2, 4, 6, 8, 10, 12, 14, 16, 18, 20)
## Sequence: begin = 2, end = 20, distance = 2
x ← seq(from = 2, to = 20, by = 2)
```

In both cases the result is the vector `x` including the required numbers. We apply function `quantile` to the vector.

```
x
```

```
 [1]  2  4  6  8 10 12 14 16 18 20
```

```
## R default
quantile(x, probs = 0.2)
```

```
20%
5.6
```

```
## Type used by SAS software
quantile(x, type = 3, probs = 0.2)
```

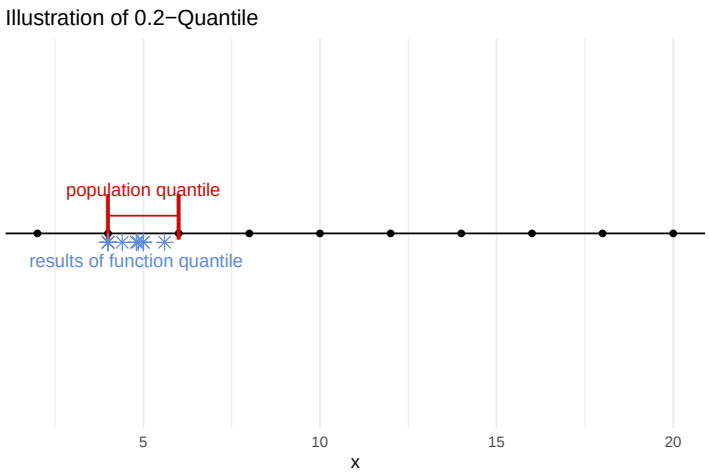
```
20%
  4
```

```
## Type used by SPSS and Minitab software
quantile(x, type = 6, probs = 0.2)
```

```
20%
4.4
```

We visualize the results using function `illustrate.quantile` of package `"MKdescr"` (Kohl (2022a)).

```
illustrate.quantile(x, alpha = 0.2)
```



Note:
As Example 2.5 demonstrates, we must be aware that different software programs may give different results in case of quantiles.

We return to our ICU dataset. The medical score SAPS II is a typical example of an ordinal attribute. We first determine the median of the values via function `median`.

```
median(ICUData$SAPS.II)
```

```
[1] 42
```

```
## also possible
quantile(ICUData$SAPS.II, probs = 0.5)
```

```
50%
 42
```

That is, 50% of the patients have a SAPS II score ≤ 42 and 50% of the patients have a score ≥ 42. The median is a so-called **location parameter** and does not give us any information about the variability of the values. For this purpose we can use quantiles, too. A very frequently used **scale** or **dispersion parameter** is the so-called **interquartile range** (IQR), the distance between third and first quartile (i.e. $q_{0.75} - q_{0.25}$). In R we can use function `IQR` to compute the IQR.

```
## length of the interval
IQR(ICUData$SAPS.II)
```

```
[1] 26
```

```
## IQR-interval
quantile(ICUData$SAPS.II, probs = c(0.25, 0.75))
```

```
25% 75%
 31  57
```

Consequently, the middle 50% of our patients possess a range of 26 SAPS II points. Another option to evaluate the disperson of the values is the **median absolute deviation** (MAD)

$$\text{MAD}\,(x_1, x_2, \ldots, x_n) = \text{median}\left\{|x_1 - M|, |x_2 - M|, \ldots, |x_n - M|\right\} \tag{2.2}$$

where $M = \text{median}\,\{x_1, x_2, \ldots, x_n\}$. We obtain with help of function `mad`

```
mad(ICUData$SAPS.II, constant = 1.0)
```

```
[1] 13
```

We have to set the additional parameter `constant`, otherwise a standardized version of the MAD is calculated.

```
mad(ICUData$SAPS.II)
```

```
[1] 19.2738
```

The standardized version of MAD is

$$\text{MAD}\,(x_1, x_2, \ldots, x_n) = 1.4826 \cdot \text{median}\left\{|x_1 - M|, |x_2 - M|, \ldots, |x_n - M|\right\} \tag{2.3}$$

By applying this standarization, the MAD under certain assumptions (normally distributed data) becomes comparable to the standard deviation, which we will introduce in Section 2.6. Also in the case of the interquartile range it is possible to standardize accordingly, which in the case of normally distributed data leads to a value that can be compared with the standard deviation.

$$\text{IQR}\,(x_1, x_2, \ldots, x_n) = 0.7413 \cdot (q_{0.75} - q_{0.25}) \tag{2.4}$$

The standardized interquartile range can be calculated with function `sIQR` from package `"MKdescr"` (Kohl (2022a)).

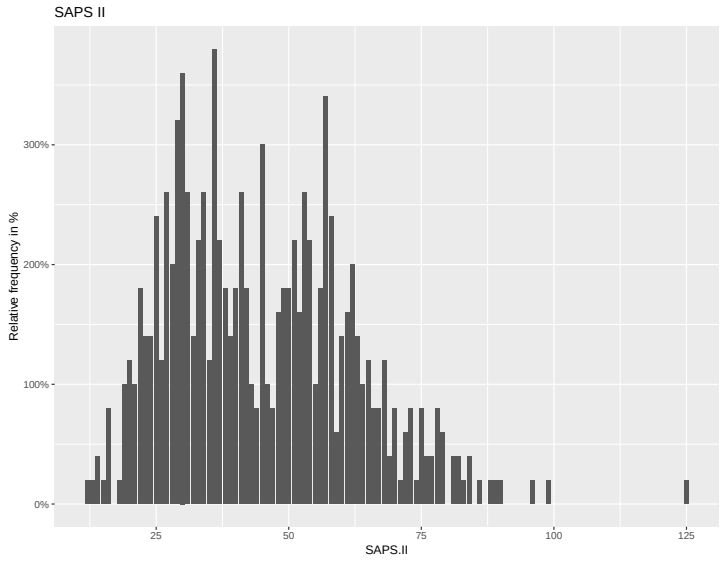
```
sIQR(ICUData$SAPS.II)
```

```
[1] 19.27383
```

The result is identical to the standardized MAD. This is because in the present case the unstandardized MAD is just half the size of the interquartile range, where the standardization constant of MAD is just twice the size of the standardization constant of the interquartile range.

For depicting ordinal data we can again use bar plots.

```
## Assign data
ggplot(ICUData, aes(x=SAPS.II)) +
  ## Add bars
  geom_bar(aes(y = 100*(..count..)/sum(..count..))) +
  ## Percent of the relative frequency
  scale_y_continuous(labels = percent_format(accuracy = 1)) +
  ## Title and label of y axis
  ggtitle("SAPS II") + ylab("Relative frequency in %")
```

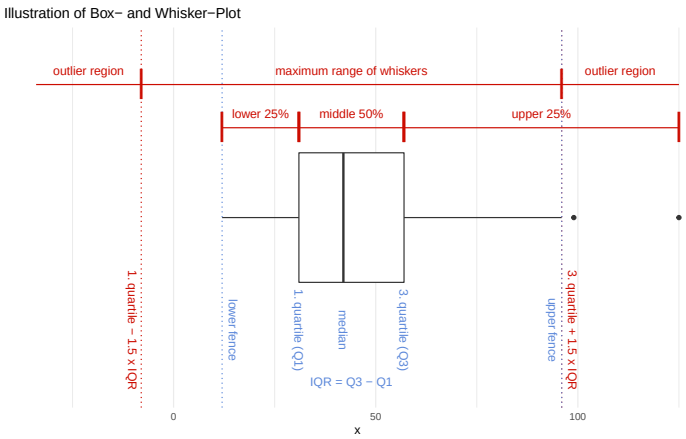


Due to the large number of possible characteristic values, the bar chart is already quite reminiscent of a histogram, which we will get to know in Section 2.6.1.

Quantiles are also the basis for one of the most important graphical display in descriptive statistics, the so-called **box-and-whisker plot**. The box-and-whisker plot very well summarizes the information of median, IQR and range of the observations. In addition, it can be applied to identify suspicious observations (outliers).

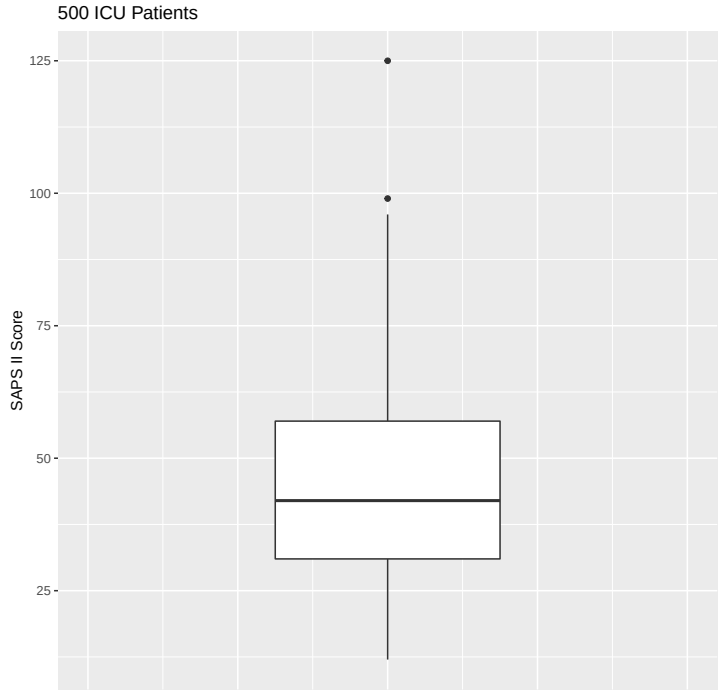
We generate a box-and-whisker plot of the SAPS II values by applying function `illustrate.boxplot` of package `"MKdescr"` (Kohl (2022a)) to illustrate the definition of the box-and-whisker plot by real data.

```
illustrate.boxplot(ICUData$SAPS.II)
```



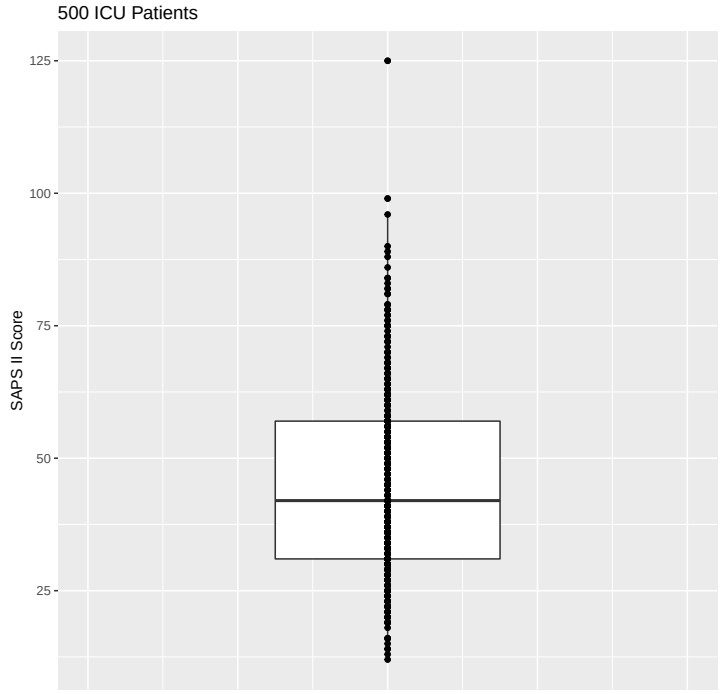
As we already know, the median is 42. The box of the box-and-whisker plot represents the middle 50% of the observations, which lie in the bounded interval [31, 57], whose length corresponds to the IQR, which is 26 points. Moreover, 25% of the values are smaller than 31 and accordingly, 25% of the values are larger than 57. Obviously, two patients were very severely sick with scores of 99 and 125 shown as outliers. Consequentially, the probability of surviving for these two patients was very small and hence, it is no surprise that both patients died. Nine of the ten patients with the highest SAPS II scores (≥ 83) died. We repeat the plot applying function `geom_boxplot` of package `"ggplot2"` (Wickham (2009)).

```
ggplot(ICUData, aes(x = 1, y = SAPS.II)) +
  geom_boxplot() + xlim(0, 2) + ylab("SAPS II Score") +
  ggtitle("500 ICU Patients") +
  ## remove labels from x-axis
  xlab("") + theme(axis.ticks.x = element_blank(),
                   axis.text.x = element_blank())
```



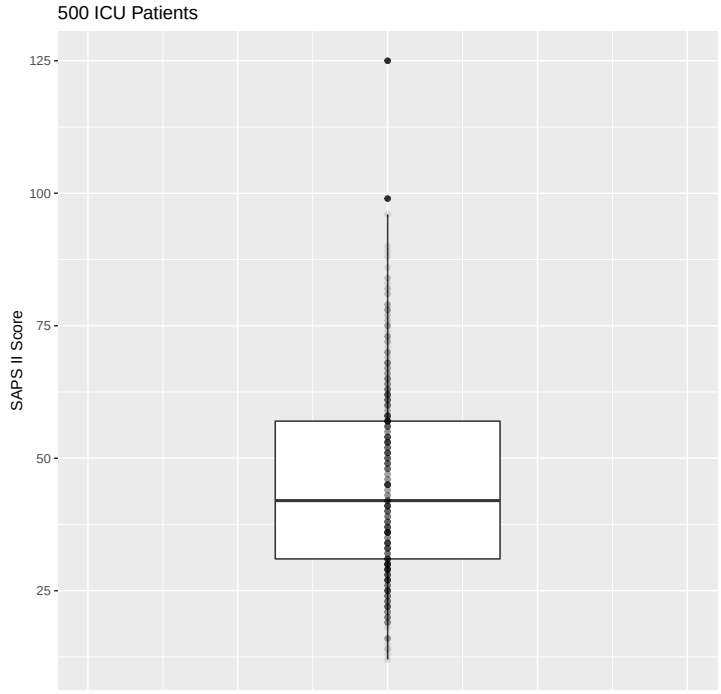
We use function `xlim` to increase the limits of the x-axis, i.e. the box appears narrower. The two values 0 and 2 represent the limits (i.e. start and end) of the x-axis. Since the x-axis does not include any relevant information and any labels may be rather irritating, we remove all labels from this axis applying functions `xlab` and `theme`.
When working with only a few data points, it is also recommended to display the observed values in the boxplot. Here we additionally use function `geom_point`.

```
ggplot(ICUData, aes(x = 1, y = SAPS.II)) +
  geom_boxplot() + xlim(0, 2) + ylab("SAPS II Score") +
  geom_point() + ggtitle("500 ICU Patients") +
  ## remove labels from x-axis
  xlab("") + theme(axis.ticks.x = element_blank(),
                   axis.text.x = element_blank())
```



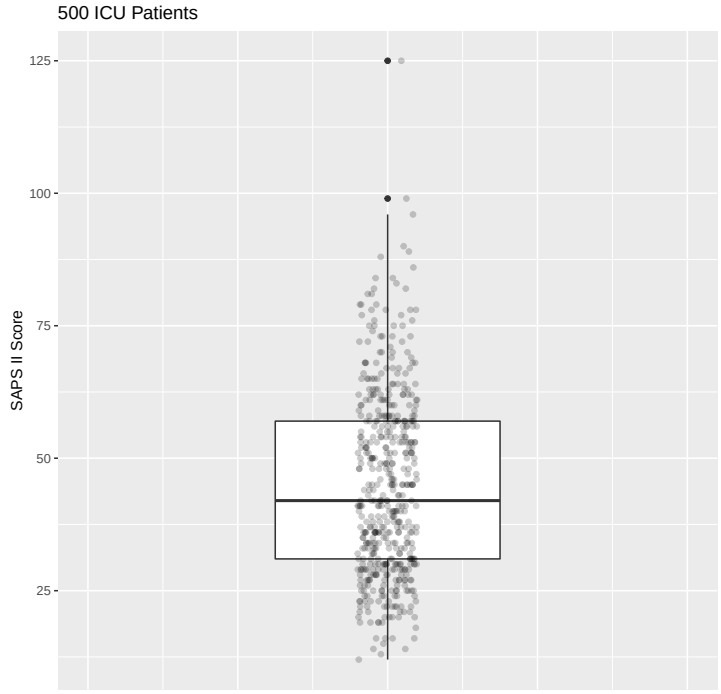
The disadvantage is that values overlap. A possible solution is **alpha blending**. In this way, the points are still drawn on the top of each other. However, the number of points is represented by the color intensity; i.e., the darker a point is, the more points are overlapping.

```
ggplot(ICUData, aes(x = 1, y = SAPS.II)) +
  geom_boxplot() + xlim(0, 2) + ylab("SAPS II Score") +
  geom_point(alpha = 0.1) + ggtitle("500 ICU Patients") +
  ## remove labels from x-axis
  xlab("") + theme(axis.ticks.x = element_blank(),
                   axis.text.x = element_blank())
```



Another possibility is the use of **jittering** In this case all the points are randomly jittered in x- and/or y-direction. This is implemented in the function `geom_gitter`.

```
ggplot(ICUData, aes(x = 1, y = SAPS.II)) +
  geom_boxplot() + xlim(0, 2) + ylab("SAPS II Score") +
  geom_jitter(height = 0, width = 0.1, alpha = 0.2) +
  ggtitle("500 ICU Patients") +
  ## remove labels from x-axis
  xlab("") + theme(axis.ticks.x = element_blank(),
                   axis.text.x = element_blank())
```



In summary, the box-and-whisker plot is one of the most important graphics in descriptive statistics and should be used in any descriptive analysis of ordinal or quantitative data.

Another interesting property of the α-quantile is its robustness against outliers (gross errors) or more generally incorrect values, respectively. More precisely, up to $\alpha\%$ of the data for $\alpha \in (0, 0.5]$ and $1 - \alpha\%$ of the data for $\alpha \in [0.5, 1)$ may be incorrect values. This fact makes the median especially attractive as it possesses the maximum robustness in this sense.

Example 2.6. Wie again consider the sequence $2, 4, 6, \ldots, 20$ and compute median and third quartile as well as 90% and 95% quantile.

```
x ← c(2, 4, 6, 8, 10, 12, 14, 16, 18, 20)
quantile(x, probs = c(0.5, 0.75, 0.9, 0.95))
```

```
 50%  75%  90%  95%
11.0 15.5 18.2 19.1
```

Now, we increase the largest number from 20 to 200, which corresponds to 10% outliers in the case at hand. We obtain

```
x ← c(2, 4, 6, 8, 10, 12, 14, 16, 18, 200)
quantile(x, probs = c(0.5, 0.75, 0.9, 0.95))
```

```
  50%   75%   90%   95%
 11.0  15.5  36.2 118.1
```

The 95% and also the 90% quantile are affected and are clearly increased. In contrast, median and third quartile show no change.

Another option to visualize the distribution of the data, is the so-called empirical cumulative distribution function.

Definition 2.7 (Empirical cumulative distribution function)**.** *Let $x_1, x_2, \ldots, x_n \in \mathbb{R}$ ($n \in \mathbb{N}$) be some observations and let $x_{(1)}, x_{(2)}, \ldots, x_{(n)}$ be the increasingly sorted observations. Furthermore, let $h_{(1)}$, $h_{(2)}$, …, $h_{(n)}$ be the associated relative frequencies. Then, the* ***empirical cumulative distribution function*** *is*

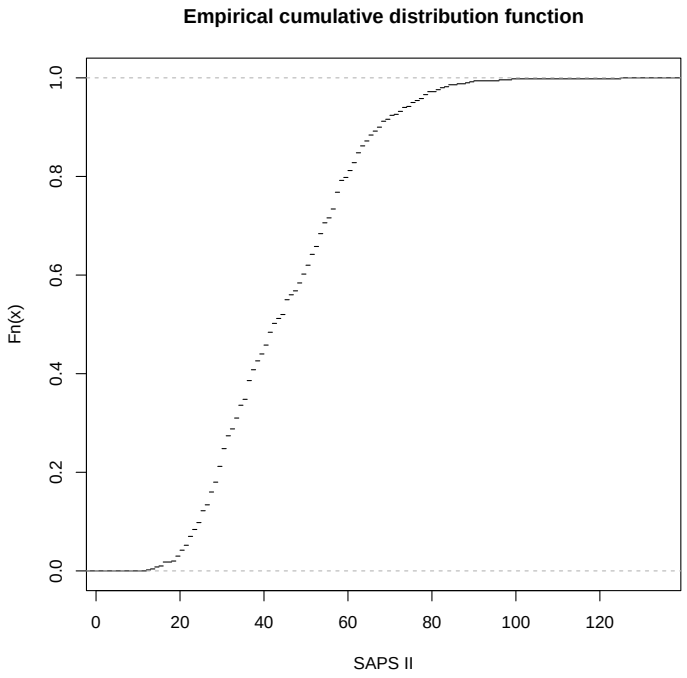
$$\hat{F}_n(x) = \begin{cases} 0 & \text{if } x < x_{(1)} \\ \sum_{i=1}^{k} h_{(i)} & \text{if } x_{(k)} \leq x < x_{(k+1)} \\ 1 & \text{if } x > x_{(n)} \end{cases} \tag{2.5}$$

The definition implies certain properties.

Remark 2.8. *Looking at the definition, the empirical cumulative distribution function is a monotone increasing step function, which is continuous from above.*

We use functions `ecdf` and `plot` to compute and plot the empirical cumulative distribution function of the SAPS II values.

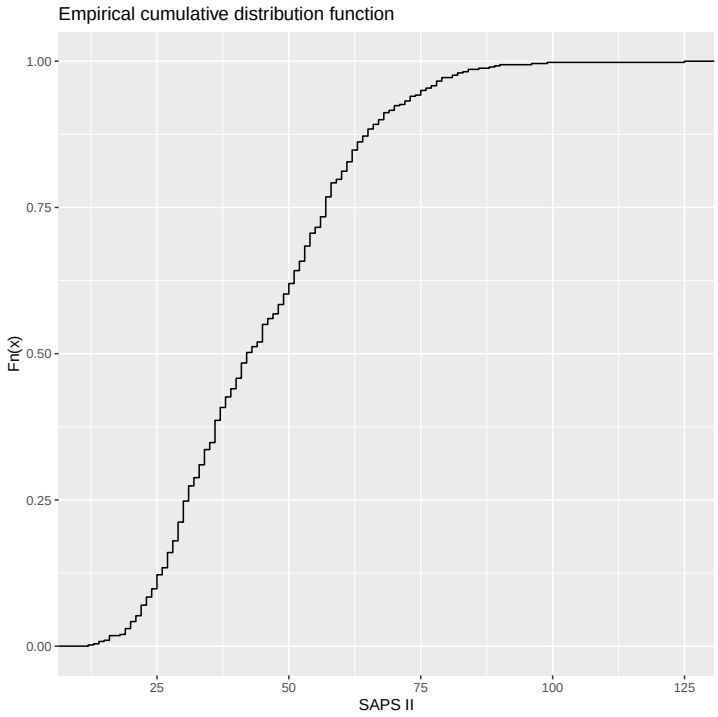
```
plot(ecdf(ICUData$SAPS.II), xlab = "SAPS II", do.points = FALSE,
     main = "Empirical cumulative distribution function")
```



Because of the quite large number of observations leading to a fine partition of the x-axis and many small jumps, we do not plot points (`do.points = FALSE`). The points can be used to illustrate that the

function is continuous from above. We can generate a similar plot with function `stat_ecdf` of package `"ggplot2"` (Wickham (2009)).

```
ggplot(ICUData, aes(x = SAPS.II)) + stat_ecdf() + xlab("SAPS II") +
  ylab("Fn(x)") + ggtitle("Empirical cumulative distribution function")
```



However, in practice, this plot is used rather rarely, because the interpretation requires some experience and is rather difficult for users.

2.5.2 Bivariate Analysis

So far we have analyzed the variables separately, but now we want to investigate the relationship between pairs of variables. We start with nominal variables. In this case, the analysis consists of calculating and plotting absolute or relative frequencies of all possible combinations of levels. This leads to a so-called **contingency table** or **cross table**. We analyse variables sex and surgery of the ICU dataset. We can compute the absolute frequencies of all level combinations with function `table`.

```
table(ICUData$sex, ICUData$surgery)
```

```
        cardiothoracic gastrointestinal neuro other trauma
female              61               31    19    57      7
male               162               48    27    64     24
```

The absolute numbers suggest that men undergo clearly more cardiothoracic surgeries than women. Since the dataset includes clearly more males than females, we should secure this hypothesis by additionally considering relative frequencies. We apply function `proportions` to the cross table to compute relative

frequencies. The argument `margin` controls if the relative frequencies are computed row- (`margin = 1`) or column-wise (`margin = 2`). In our example, we need the row-wise calculation.

```
proportions(table(ICUData$sex, ICUData$surgery), margin = 1)
```

```
        cardiothoracic gastrointestinal      neuro      other     trauma
female     0.34857143       0.17714286 0.10857143 0.32571429 0.04000000
male       0.49846154       0.14769231 0.08307692 0.19692308 0.07384615
```

For improving the representation, we compute percentages and round the results via function `round` to one decimal place.

```
round(100*proportions(table(ICUData$sex, ICUData$surgery), margin = 1), 1)
```

```
        cardiothoracic gastrointestinal neuro other trauma
female           34.9             17.7  10.9  32.6    4.0
male             49.8             14.8   8.3  19.7    7.4
```

For representing absolute and relative frequencies in a cross table we can also use function `PercTable` of package `"DescTools"` (Andri et mult. al. (2022)).

```
PercTable(table(ICUData$sex, ICUData$surgery), rfrq = "010")
```

```
                cardiothoracic   gastrointestinal   neuro   other

female   freq               61                31      19      57
         p.row          34.9%             17.7%   10.9%   32.6%

male     freq              162                48      27      64
         p.row          49.8%             14.8%    8.3%   19.7%


                trauma

female   freq        7
         p.row    4.0%

male     freq       24
         p.row    7.4%
```

The computation of the relative frequencies is controlled by argument `rfrq`. The value ‘‘`010`’’ represents row-wise relative frequencies. The results confirm our first impression that cardiothoracic surgeries were more frequent in case of men. Conversely, females had remarkably more “other” surgeries.

The strength of the relationship of two (or more) nominal (or also ordinal) variables can be determined by so-called contingency coefficients.

Definition 2.9 (Contingency coefficients)**.** *Let us assume $n \in \mathbb{N}$ observations of two variables with $l \in \mathbb{N}$ and $m \in \mathbb{N}$ levels, respectively. That is, the observed pairs of values can be represented by a matrix with l rows and m columns, where the total number of entries is $k = l \cdot m$. Furthermore, let n_i $(i = 1, \ldots, k)$ be the number of observations in cell i, p_i $(i = 1, \ldots, k)$ the theoretical probability of cell i, and hence $e_i = N \cdot p_i$ $(i = 1, \ldots, k)$ the expected number of observations in cell i. Then, the* ***χ^2-statistics*** *is*

$$\chi^2 = \sum_{i=1}^{k} \frac{(n_i - e_i)^2}{e_i} \tag{2.6}$$

Based on χ^2 we get the following ***contingency coefficients***

(i) ***ϕ-coefficient***

$$\phi = \sqrt{\frac{\chi^2}{n}} \tag{2.7}$$

(ii) ***Pearson's contingency coefficient***

$$C = \sqrt{\frac{\chi^2}{n + \chi^2}} \tag{2.8}$$

(iii) ***Cramér's V***

$$V = \sqrt{\frac{\chi^2}{n \cdot (M - 1)}} \qquad M = \min\{l, m\} \tag{2.9}$$

We give some further explanations.

Remark 2.10. ***(a)*** *In practice, it is important to be aware of the maximum possible value of the computed contingency coefficient. Furthermore, a clear disadvantage of contingency coefficients is that they only measures the strength of a relationship, but are not able to identify the direction of a relationship, which for instance is of interest in case of ordinal attributes.*

(b) *The ϕ-coefficient attains values in the interval $[0, 1]$, where 1 is only possible under certain circumstances. If the result is 0, the two attributes are independent.*

(c) *The range of Pearson's contingency coefficient is $\left[0, \sqrt{\frac{M}{M-1}}\right]$ $(M = \min\{l, m\})$, where 0 indicates independence of the investigated attributes.*

(d) *Cramér's V attains values in the interval $[0, 1]$, where again 0 stands for independence. One speaks of week dependence if $V \leq 0.3$, moderate dependence if $0.3 < V \leq 0.7$, and strong dependence if $V > 0.7$.*

We apply functions `Phi`, `ContCoef`, and `CramerV` of package `"DescTools"` (Andri et mult. al. (2022)) to determine the strength of the relationship between sex and surgery.

```
## phi coefficient
Phi(table(ICUData$sex, ICUData$surgery))
```

```
[1] 0.1846974
```

```
## Pearson's contingency coefficient
ContCoef(table(ICUData$sex, ICUData$surgery))
```

```
[1] 0.1816254
```

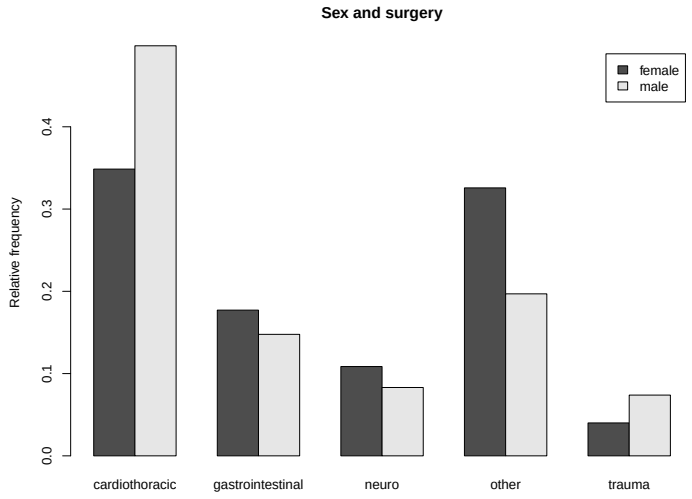
```
## Cramer's V
CramerV(table(ICUData$sex, ICUData$surgery))
```

```
[1] 0.1846974
```

We obtain only a weak dependence between sex and surgery. As $M = 2$, the ϕ-coefficient and Cramér's V are identical.

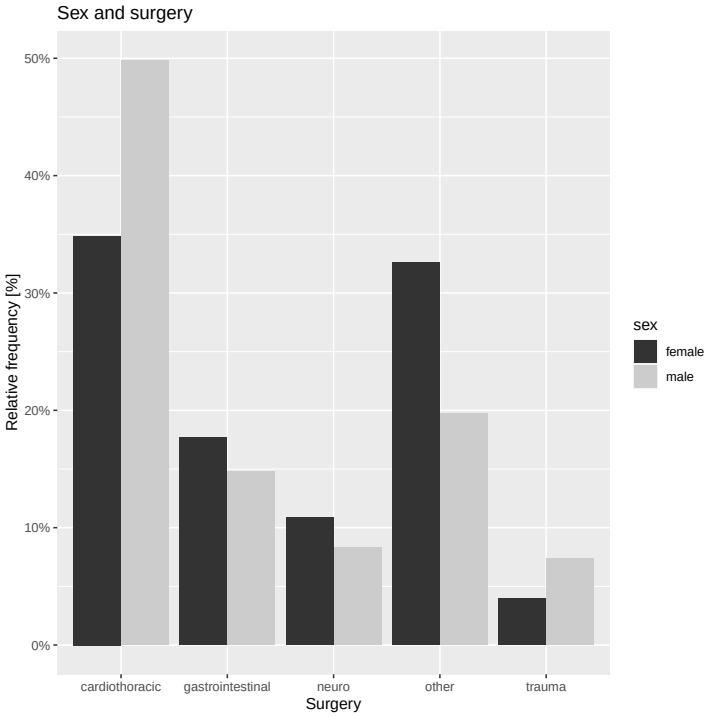
Bar charts are the usual way to graphically represent contingency tables. We plot the variables sex and surgery, where we apply function `barplot` in combination with `table` and `prop.table`.

```
barplot(prop.table(table(ICUData$sex, ICUData$surgery), margin = 1),
        beside = TRUE, legend.text = TRUE, ylab = "Relative frequency",
        main = "Sex and surgery")
```



The argument `beside = TRUE` guarantees that the bars of females and males are beside and not above each other. By `legend.text = TRUE` we obtain a legend explaining the relation between colors and sex. We will now regenerate the graph by using the function of the package `"ggplot2"`(Wickham (2009)). The relative frequencies are calculated within the function `geom_bar` using the function `tapply`. With ''`dodge`'' as position, the bars will be displayed side by side (and not on the top of each other). We also use the additional function `scale_fill_grey`, which fills the bars with shades of gray.

```
ggplot(ICUData, aes(x = surgery, fill = sex)) +
  geom_bar(aes(y = (..count..)/tapply(..count.., ..fill.., sum)[..fill..]),
           position = "dodge") + scale_fill_grey() +
  scale_y_continuous(labels = percent_format(accuracy = 1)) +
  ylab("Relative frequency [%]") + xlab("Surgery") +
  ggtitle("Sex and surgery")
```



With the help of the function `tapply` we can split the values of one variable with the values of another variable and apply a function to this.
In case of ordinal attributes, we can use rank correlations instead of contingency coefficients, which show not only the strength, but also the direction of a relationship. The **rank** of an observation corresponds to its position inside the sample after decreasingly sorting the observations; i.e., the largest observation has rank 1, the second largest rank 2, etc.

Definition 2.11 (Spearman's ρ)**.** *Let* $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ *(*$n \in \mathbb{N}$*) be pairs of observations with ranks* $(rx_1, ry_1), (rx_2, ry_2), \ldots, (rx_n, ry_n)$*. Then,* ***Spearman's*** $\boldsymbol{\rho}$ *is*

$$\rho = \frac{\sum_{i=1}^{n} \left(rx_i - mr_x\right)\left(ry_i - mr_y\right)}{\sqrt{\sum_{i=1}^{n} \left(rx_i - mr_x\right)^2 \sum_{i=1}^{n} \left(ry_i - mr_y\right)^2}} \tag{2.10}$$

where mr_x *and* mr_y *are the respective average ranks; i.e.*

$$mr_x = \frac{1}{n}\sum_{i=1}^{n} rx_i \qquad \textit{und} \qquad mr_y = \frac{1}{n}\sum_{i=1}^{n} ry_i \tag{2.11}$$

Spearman's ρ *attains values in* $[-1, 1]$*, where* 1 *represents a perfect monotone increasing relation and* -1 *a perfect monotone decreasing relation.*

We give some additional explanations.

Remark 2.12. ***(a)*** *If a value was observed several times (at least twice) this is called a* ***binding****. If there*

are no bindings, the computation of Spearman's ρ simplifies and it holds

$$\rho = 1 - \frac{6 \sum_{i=1}^{n} (rx_i - ry_i)^2}{n(n^2 - 1)} \tag{2.12}$$

(b) *Beside Spearman's ρ,* ***Kendall's τ*** *is a frequently applied rank correlation coefficient. It compares the number of concordant and discordant pairs of observations. The result is in $[-1, 1]$. A value of 1 implies that both variables have exactly the same order, and -1 that they are in perfect inverse order. Kendall's τ is more appropriate than Spearman's ρ in case of small samples or scores with uneven scales. For more details we refer to Section 3.2.5 of Hedderich and Sachs (2018).*

(c) *Rank correlations are also very useful in case of metric variables and can help to identify monotone relations.*

We compute the correlation between SAPS II and length of stay (LOS), where we apply function `cor`.

```
## Spearman's rho
cor(ICUData$SAPS.II, ICUData$LOS, method = "spearman")
```

```
[1] 0.3379928
```

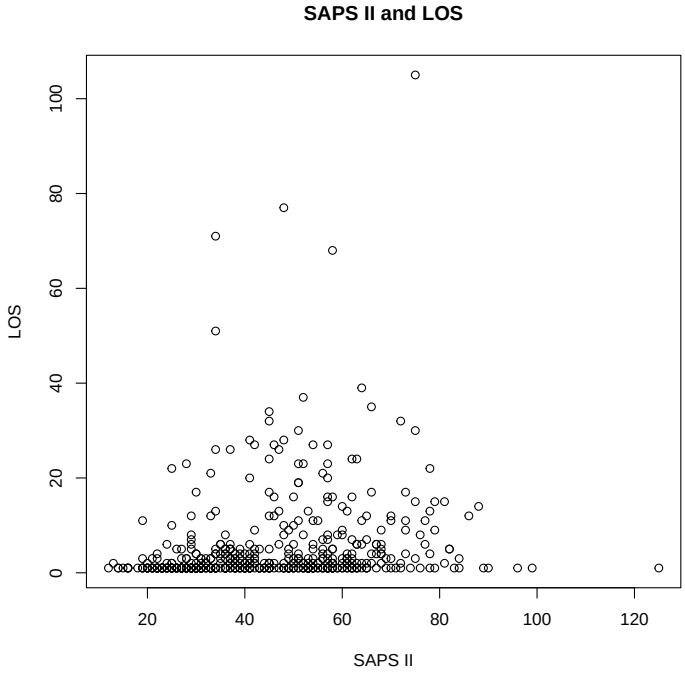
```
## Kendall's tau
cor(ICUData$SAPS.II, ICUData$LOS, method = "kendall")
```

```
[1] 0.2518917
```

Note:
The numbers we have obtained above describe the strength of a monotone relationship, but only if the relationship is truly monotone. Monotony is an assumption that should be verified before the computation is performed. In general, one should be aware that most statistical analyses are based on certain assumptions. If these assumptions are violated, the meaning of the results is unclear. That is, if there is no simple monotone relationship between SAPS II and length of stay (LOS), the above results for Spearman's ρ or Kendall's τ may lead to misinterpretations. The verification of assumptions usually requires a sound knowledge about the investigated relationships.

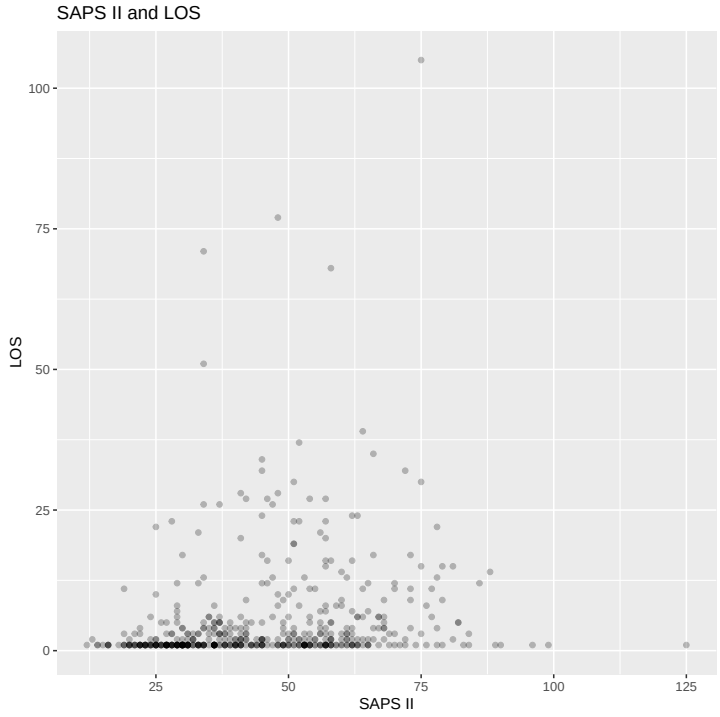
As expected, there is a positive relationship. Patients with a high SAPS II score are more severely ill and thus have to stay on the ICU for a longer time period. What works against this, is the fact that patients with a very high SAPS II value also have a high probability of dying, hence might die just after admission to ICU. We display the observed values in a **scatter plot** to check, if this is actually true. We apply function `plot`.

```
plot(ICUData$SAPS.II, ICUData$LOS, xlab = "SAPS II", ylab = "LOS",
     main = "SAPS II and LOS")
```



Indeed, the patients with the highest SAPS II values have a small LOS and died quite rapidly. The ordinal or discrete structure of the attributes leads to an overlap of observations. We can use a so-called **alpha blending** to better visualize the structure of the point cloud; that is, the final color emerges from a combination of the original colors. We demonstrate this by means of package `"ggplot2"` (Wickham (2009)). The possible shaps of points that are available in R by default are given on the help page of function `points`.

```
ggplot(ICUData, aes(x=SAPS.II, y=LOS)) +
  ## shape = 19: slightly larger point
  ## alpha = 0.25: strength of blending
  geom_point(shape=19, alpha=0.25) +
  ## title and labels
  ggtitle("SAPS II and LOS") + xlab("SAPS II") + ylab("LOS")
```



The darker the color the more observations overlap. In summary, we can assume a monotone increasing connection for a certain range of SAPS II scores but surely not for the full range. Therefore, the computed rank correlations should be interpreted with care. It would even be better to investigate the relationship in a different way, by some kind of regression analysis. However, this is beyond the scope of this book.

2.6 Metric Variables

2.6.1 Univariate Analysis

As distances and even ratios are defined, further analyses are possible in case of metric variables. If not explicitly mentioned, the introduced analyses are possible for interval and ratio scaled variables. Probably the most frequently used statistics to describe data is the arithmetic mean.

Definition 2.13 (Arithmetic mean)**.** *Let $x_1, x_2, \ldots, x_n \in \mathbb{R}$ ($n \in \mathbb{N}$) be some observations. Then, the **arithmetic mean** is*

$$AM(x_1, \ldots, x_n) = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{2.13}$$

In this section, we again use our ICU dataset; see Section 2.3. We compute the arithmetic mean of the maximum body temperature during the stay on the ICU applying function `mean`.

```
1 mean(ICUData$temperature)
```

```
[1] 37.6632
```

That is, the arithmetic mean is only slightly above the normal range, where the result suggests a precision that is actually not true. The temperatures are only given with one decimal place. Consequentially, the arithmetic mean should be rounded to one decimal place. For this, we use function `round`.

```
round(mean(ICUData$temperature), 1)
```

```
[1] 37.7
```

It is advisable, to always compare the arithmetic mean with the median, as the median gives another description of the middle of the data and is very robust against outliers (see Example 2.6).

```
median(ICUData$temperature)
```

```
[1] 37.7
```

As median and arithmetic mean can be regarded as identical, it is likely, that the distribution of the maximum body temperature is quite symmetric around the arithmetic mean (resp. median). In addition, there are either no outliers or positive and negative outliers neutralize each other. We repeat the analysis using variable LOS (length of stay) given in days.

```
round(mean(ICUData$LOS),1)
```

```
[1] 5.3
```

```
median(ICUData$LOS)
```

```
[1] 1
```

In this case, we see a clear difference between arithmetic mean and median. Either the distribution of LOS is skewed (more precisely right-skewed, see also Remark 2.23) or there are outliers pulling the arithmetic mean to the right. We will be able to distinguish these two cases below, where we consider diagrams of the data.

Another location parameter is the geometric mean, which is applied in case of relative changes. This measure of location is only meaningfully defined for strictly positive data.

Definition 2.14 (Geometric mean)**.** *Let $x_1, x_2, \ldots, x_n \in (0, \infty)$ ($n \in \mathbb{N}$) be some observations. Then, the* ***geometric mean*** *is*

$$GM(x_1, \ldots, x_n) = \sqrt[n]{x_1 \cdot x_2 \cdot \ldots \cdot x_n} \tag{2.14}$$

In the following remark, we describe an important connection between geometric and arithmetic mean.

Remark 2.15. *By applying the rules of logarithm we obtain*

$$\begin{aligned} AM(\log(x_1), \ldots, \log(x_n)) &= \frac{1}{n}\sum_{i=1}^{n}\log(x_i) = \frac{1}{n}\log\left(x_1 \cdot x_2 \cdot \ldots \cdot x_n\right) \\ &= \log\left(\sqrt[n]{x_1 \cdot x_2 \cdot \ldots \cdot x_n}\right) \\ &= \log\left(GM(x_1, \ldots, x_n)\right) \end{aligned} \tag{2.15}$$

That is, the arithmetic mean of the logarithmized observations is equal to the logarithm of the geometric mean where the base of logarithm is irrelevant. If we select the natural logarithm (ln), we can rewrite it by applying the e-function to

$$GM(x_1, \ldots, x_n) = e^{AM(\ln(x_1), \ldots, \ln(x_n))} \tag{2.16}$$

If one observes processes following an exponential growth or decay, it is often easier to take the logarithm of the data and analyze the logarithmized observations. This is for instance true for the bilirubin measurements included in our ICU dataset. The base and recommended packages do not include the geometric mean, but we can apply function `Gmean` of package `"DescTools"` (Andri et mult. al. (2022)). We compute the natural logarithm of the geometric mean.

```
1 log(Gmean(ICUData$bilirubin))
```

```
[1] 2.847326
```

As our derivation in Remark 2.15 shows, the following R code must yield the same result, which is actually true.

```
1 mean(log(ICUData$bilirubin))
```

```
[1] 2.847326
```

Consequentially, we may compute the geometric mean not only via function `Gmean`, but also by

```
1 exp(mean(log(ICUData$bilirubin)))
```

```
[1] 17.24162
```

where `exp` calculates the e-function. In addition, this form of computation has numerical advantages as summation is numerically more stable than calculating products. Therefore, the geometric mean is usually implemented in this way.

In practice, not only location but also dispersion of the observations is of interest. The probably most frequently applied measure of dispersion is the standard deviation, which is the square root of the variance.

Definition 2.16 (Variance, standard deviation)**.** *Let $x_1, x_2, \dots, x_n \in \mathbb{R}$ ($n \in \mathbb{N}$) be some observations. Then, the sample **variance** is*

$$Var(x_1, \dots, x_n) = \frac{1}{n} \sum_{i=1}^{n} \left(x_i - AM(x_1, \dots, x_n)\right)^2 \tag{2.17}$$

*and sample **standard deviation** reads*

$$SD(x_1, \dots, x_n) = \sqrt{Var(x_1, \dots, x_n)} \tag{2.18}$$

We give some additional explanations.

Remark 2.17. ***(a)*** *Instead of $\frac{1}{n}$ one often uses $\frac{1}{n-1}$ for computing variance and standard deviation. This minor diffence also makes the difference between descriptive and inferential statistics. With standarization $\frac{1}{n}$ we describe the sample, whereas with standardization $\frac{1}{n-1}$ we obtain an unbiased parameter estimate for the underlying population; for more details see Example 5.3. If the sample size n is not too small, we can neglect the difference in practice.*

(b) *Let us assume the observations were measured in unit U. Then, variance has unit U^2 and standard deviation unit U. This is one reason why standard deviation is more frequently applied in practice than variance.*

We compute variance and standard deviation for the maximum body temperature. The respective functions in R are `var` and `sd` both using standardization $\frac{1}{n-1}$.

```
1 var(ICUData$temperature)
```

```
[1] 3.011869
```

```
1 sd(ICUData$temperature)
```

```
[1] 1.735474
```

By multiplying the result with $\frac{n-1}{n}$, we obtain the "true" sample values.

```
1 n ← nrow(ICUData)
2 (n-1)/n*var(ICUData$temperature)
```

```
[1] 3.005846
```

```
1 sqrt((n-1)/n)*sd(ICUData$temperature)
```

```
[1] 1.733738
```

Rounding to one decimal place, which should be done based on the given precision, would lead to identical results. Similarly to the comparison of arithmetic mean and median, we now compare standard deviation and the standardized MAD (cf. equations (2.3) and (2.4)).

```
sd(ICUData$temperature)
```

```
[1] 1.735474
```

```
mad(ICUData$temperature)
```

```
[1] 1.18608
```

```
sIQR(ICUData$temperature)
```

```
[1] 1.111952
```

There are clear differences between the results. Either the temperature distribution can not be described by a distribution that is symmetric around the arithmetic mean or there are outliers distorting the standard deviation. We will identify the cause below.

In case of positive measurements, one in practice often uses the following standardized dispersion measure.

Definition 2.18 (Coefficient of variation)**.** *Let $x_1, x_2, \dots, x_n \in [0, \infty)$ ($n \in \mathbb{N}$) be some positive observations. Then, the **coefficient of variation** is*

$$CV(x_1, \dots, x_n) = \frac{SD(x_1, \dots, x_n)}{AM(x_1, \dots, x_n)} \tag{2.19}$$

We give some additional explanations.

Remark 2.19. ***(a)*** *The coefficient of variation is a dimensionless quantity, which is frequently given in percent; that is, percental dispersion with reference to the arithmetic mean. Consequentially, it should only be applied to ratio scaled variables.*

(b) *There are variants of the coefficient of variation based on quantiles. One option is based on median and MAD*

$$medCV(x_1, \dots, x_n) = \frac{MAD(x_1, \dots, x_n)}{median(x_1, \dots, x_n)} \tag{2.20}$$

*Alternatively, one can use quartiles leading to the so-called **quartile coefficient of dispersion***

$$QCD(x_1, \dots, x_n) = \frac{IQR(x_1, \dots, x_n)}{median(x_1, \dots, x_n)} \tag{2.21}$$

We apply these standardized dispersion measures to the maximum body temperature and use the functions `CV`, `medCV` and `iqrCV` from package `"MKdescr"` (Kohl (2022a)).

```
CV(ICUData$temperature)
```

```
[1] 0.04607877
```

```
medCV(ICUData$temperature)
```

```
[1] 0.03146106
```

```
iqrCV(ICUData$temperature)
```

```
[1] 0.02949474
```

We get only minor variations around the arithmetic mean respectively, median in the range of about 3-5%, where the values of `medCV` and `iqrCV` are quite similar.

In the following definition we give the standard deviation for the geometric mean.

Definition 2.20 (Geometric standard deviation)**.** *Let $x_1, x_2, \dots, x_n \in (0, \infty)$ ($n \in \mathbb{N}$) be some positive observations. Then, the **geometric standard deviation** is*

$$SD_{GM}(x_1, \dots, x_n) = e^{\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\ln(x_i) - \ln\left(GM(x_1,\dots,x_n)\right)\right)^2}} \tag{2.22}$$

We briefly motivate this definition.

Remark 2.21. *It holds*

$$SD\left(\ln(x_1), \dots, \ln(x_n)\right) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\ln(x_i) - AM\left(\ln(x_1), \dots, \ln(x_n)\right)\right)^2} \tag{2.23}$$

By using the connection (2.15) and by analogously introducing the geometric standard deviation, we get

$$\begin{aligned} SD\left(\ln(x_1), \dots, \ln(x_n)\right) &= \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\ln(x_i) - \ln\left(GM(x_1, \dots, x_n)\right)\right)^2} \\ &= \ln\left(SD_{GM}(x_1, \dots, x_n)\right) \end{aligned} \tag{2.24}$$

By applying the e-function, Definition 2.20 follows. Futhermore, the expression below the sigma sign may be rewritten as

$$\ln(x_i) - \ln\left(GM(x_1, \dots, x_n)\right) = \ln\left(\frac{x_i}{GM(x_1, \dots, x_n)}\right) \tag{2.25}$$

We check equation (2.24) using the bilirubin values of our ICU dataset, where we use function `Gsd` of package `"DescTools"` (Andri et mult. al. (2022)) for computing the geometric standard deviation.

```
log(Gsd(ICUData$bilirubin))
```

```
[1] 0.7238379
```

```
sd(log(ICUData$bilirubin))
```

```
[1] 0.7238379
```

In addition to location and scale measures, **shape measures** are used in case of metric variables. A shape measure of symmetry is skewness.

Definition 2.22 (Skewness)**.** *Let $x_1, x_2, \ldots, x_n \in \mathbb{R}$ ($n \in \mathbb{N}$) be some observations. Then, the **skewness** is*

$$Skew(x_1, \ldots, x_n) = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{x_i - AM(x_1, \ldots, x_n)}{SD(x_1, \ldots, x_n)} \right)^3 \tag{2.26}$$

*If $Skew(x_1, \ldots, x_n) < 0$, the data distribution is **left-skewed** or **negatively skewed**, if $Skew(x_1, \ldots, x_n) > 0$, it is **right-skewed** or **positively skewed**.*

We give some additional explanations.

Remark 2.23. ***(a)*** *By centering the date with respect to the arithmetic mean and standardizing it by the standard deviation, which is also called **z-transformation**, one gets the so-called **z-score**, a dimensionless score. As skewness is defined based on the z-score, it is also a dimensionless measure.*

(b) *The skewness of a distribution can also be identified by using arithmetic mean and median. If $AM(x_1, \ldots, x_n) < median(x_1, \ldots, x_n)$, the distribution is left-skewed. Conversely, if $AM(x_1, \ldots, x_n) > median(x_1, \ldots, x_n)$ the distribution is right-skewed; see also Figure 2.8.*

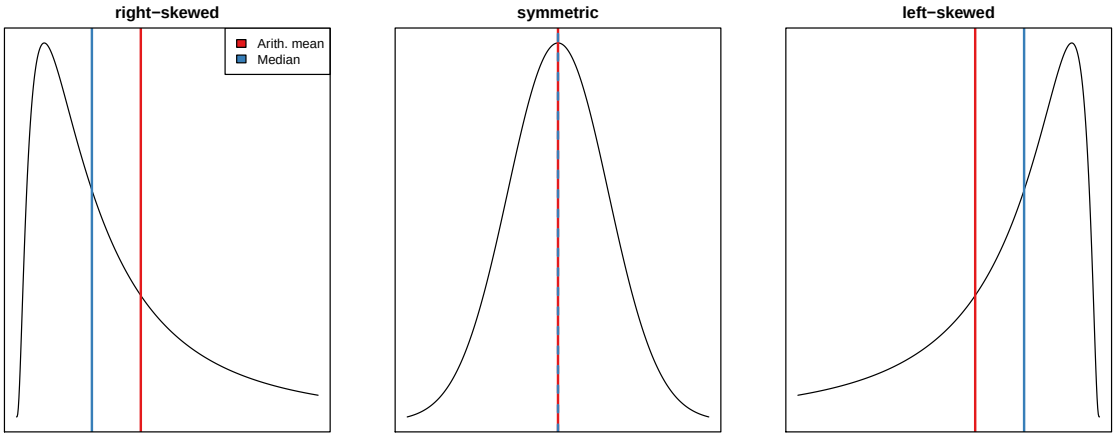


Figure 2.8: Examples of skewness.

We compute the skewness of the maximum body temperature by applying function `Skew` of package `"DescTools"` (Andri et mult. al. (2022)).

```
1 Skew(ICUData$temperature)
```

```
[1] -8.77457
```

The result, which indicates a strongly left-skewed distribution, contradicts our observation above, where median and arithmetic mean were (more or less) identical giving evidence for a symmetric distribution. A closer look at the measured temperatures shows that patient 398 had an abnormally low maximum (!) body temperature of $9.1°C$ (measurement or transcription error?). We repeat the computation without

patient 398. For accessing the maximum body temperature of patient 398, we can use square brackets [and his index.

```
## Patient 398
ICUData$temperature[398]
```

```
[1] 9.1
```

A negative index means that this index is omitted. We obtain

```
Skew(ICUData$temperature[-398])
```

```
[1] 0.3142909
```

Now, the skewness is close to 0 and confirms our first impression. The distribution of the values, without patient 398, is quite symmetric around the arithmetic mean. Furthermore, omitting patient 398 also clearly reduces the standard deviation

```
sd(ICUData$temperature[-398])
```

```
[1] 1.173187
```

which is now very close to the standardized MAD and the standardized IQR.

> Note:
> Single outliers may have a strong influence on certain statistical procedures and may clearly distort the results. Examples are arithmetic mean, variance/standard deviation, and skewness. Therefore, it is important to always investigate the data with respect to suspicious values.

We compute the skewness for length of stay (LOS). Based on arithmetic mean and median, we concluded above that the distribution must be right-skewed. Thus, we would expect a positive value of skewness.

```
Skew(ICUData$LOS)
```

```
[1] 4.880826
```

Indeed, the result confirms our first analysis.

Another measure of shape is the kurtosis.

Definition 2.24 (Kurtosis)**.** *Let $x_1, x_2, \ldots, x_n \in \mathbb{R}$ ($n \in \mathbb{N}$) be some observations. Then, the **kurtosis** is*

$$Kurt(x_1, \ldots, x_n) = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{x_i - AM(x_1, \ldots, x_n)}{SD(x_1, \ldots, x_n)} \right)^4 - 3 \tag{2.27}$$

*If $Kurt(x_1, \ldots, x_n) < 0$, the data distribution is **platykurtic**, if $Kurt(x_1, \ldots, x_n) > 0$, it is **leptokurtic**.*

We give some additional explanations.

Remark 2.25. *The reference for defining the kurtosis is the normal distribution (see Section 4.2). By subtracting* 3 *in the above definition, the normal distribution has kurtosis* 0*. For clarification one speaks in this case also from* ***excess*** *or* ***excess kurtosis****. If we observe a negative (excess) kurtosis, the distribution is flatter and less curved than the normal distribution. If the (excess) kurtosis is positive, the distribution is steeper and more curved than the normal distribution; see also Figure 2.9.*

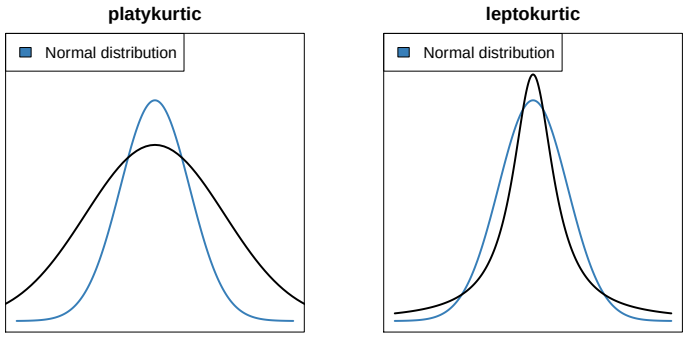


Figure 2.9: Examples of kurtosis.

We compute the kurtosis of the maximum body temperature of the ICU patients using function `Kurt` of package `"DescTools"` (Andri et mult. al. (2022)). Due to the strong impact of patient 398 on skewness, we compare the kurtosis with and without this patient.

```
Kurt(ICUData$temperature)
```

```
[1] 144.4649
```

```
Kurt(ICUData$temperature[-398])
```

```
[1] 0.3431707
```

This shows once again how big the influence of one single observation can be. We conclude that the distribution is not extremely leptokurtic, but except for one observation can be quite well described by a normal distribution. We determine the kurtosis of length of stay (LOS).

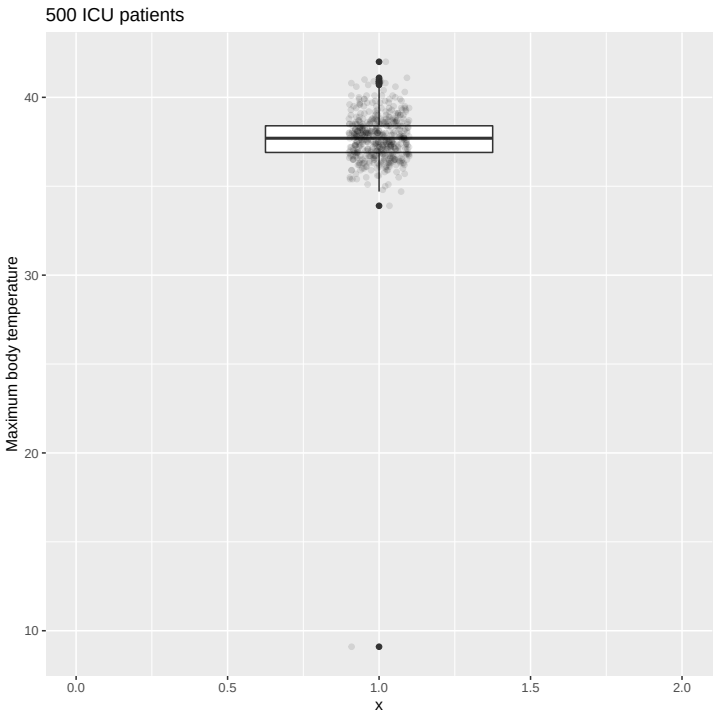
```
Kurt(ICUData$LOS)
```

```
[1] 33.59482
```

That is, the distribution of LOS is leptokurtic.

We proceed with various options for plotting metric variables. We start with a box-and-whisker plot of the maximum body temperature. We apply the functions of package `"ggplot2"` (Wickham (2009)).

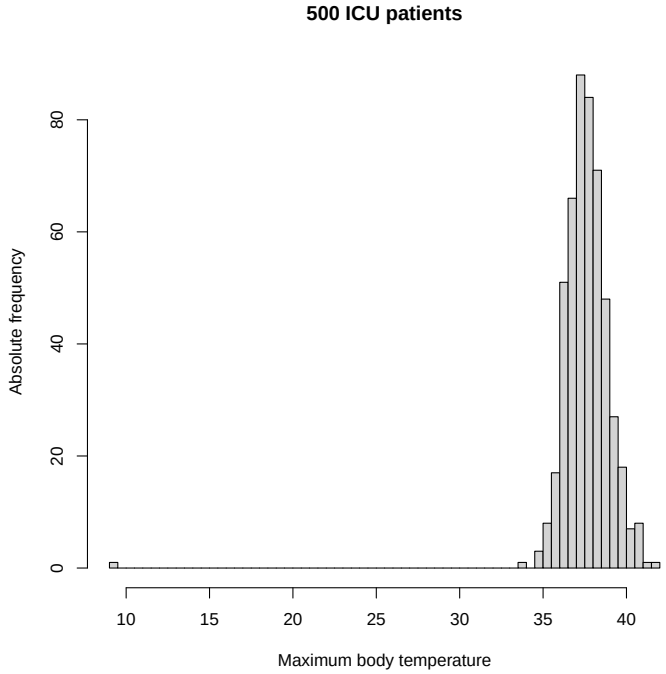
```
## Box-and-whisker plot at position x = 1
ggplot(ICUData, aes(x = 1, y = temperature)) +
  geom_boxplot() + xlim(0, 2) + ylab("Maximum body temperature") +
  geom_jitter(height = 0, width = 0.1, alpha = 0.1) +
  ggtitle("500 ICU patients")
```



We see some minor outliers and the value of patient 398, which extremly differs from all other observations.

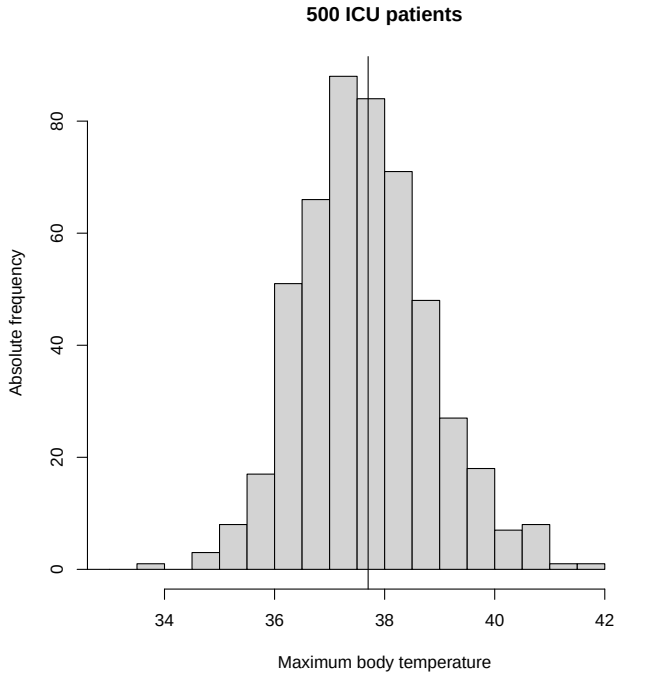
We futher analyse the distribution using histograms. A **histogram** is a special kind of bar chart, that is obtained by splitting the range of a metric variable in consecutive intervals. For each interval the absolute or relative frequency of the included observations is visualized by a bar. For choosing the intervals, there are some rules of thumb, which are used by software programs to automatically select a number of equal length intervals. However, in most cases it is better to select the intervals by hand and choose a division that fits to the context. We generate a histogram of the maximum body temperatures, where we use intervals of length $0.5°C$. We can specify the intervals by argument `breaks`.

```
hist(ICUData$temperature, breaks = seq(from = 9.0, to = 42, by = 0.5),
     main = "500 ICU patients", xlab = "Maximum body temperature",
     ylab = "Absolute frequency")
```



Again, we clearly see the extreme value of patient 398. To get a better view of the distribution, we can either remove the value of patient 398 or restrict the range of the x-axis by argument `xlim`. We select the second option. Furthermore, we add a vertical line for the median by applying function `abline`.

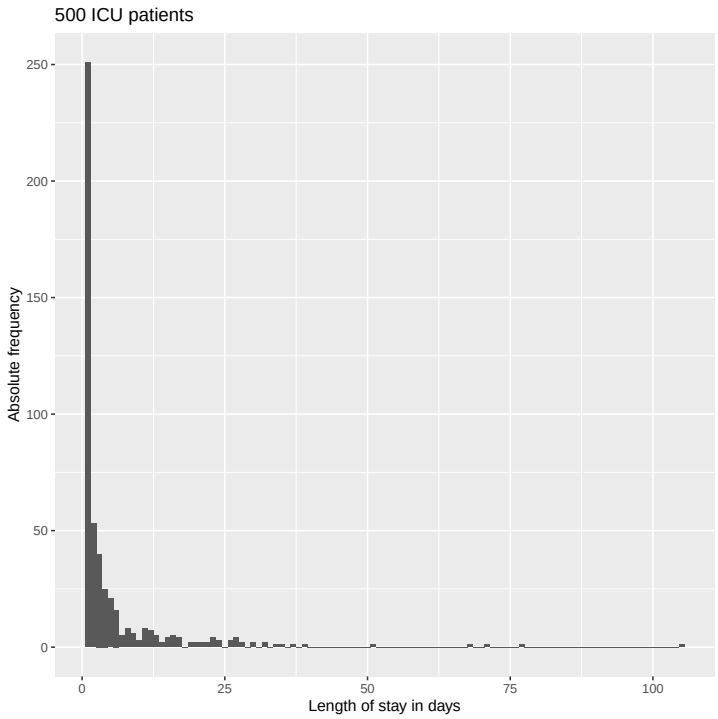
```
hist(ICUData$temperature, breaks = seq(from = 9.0, to = 42, by = 0.5),
     main = "500 ICU patients", xlab = "Maximum body temperature",
     ylab = "Absolute frequency", xlim = c(33,43))
abline(v = median(ICUData$temperature))
```



The plot confirms our previous computations; i.e., the distribution is quite symmetric around the arithmetic mean respectively median and the distribution of the maximum body temperature in the ICU population (except for patients with strong undercooling/hypothermia) is probably well described by a normal distribution.

Next, we take a look on length of stay, where we use function `geom_histogram` of package `"ggplot2"` (Wickham (2009)) to generate a histogram. As length of the intervals we use one day, which we can specify by argument `binwidth`.

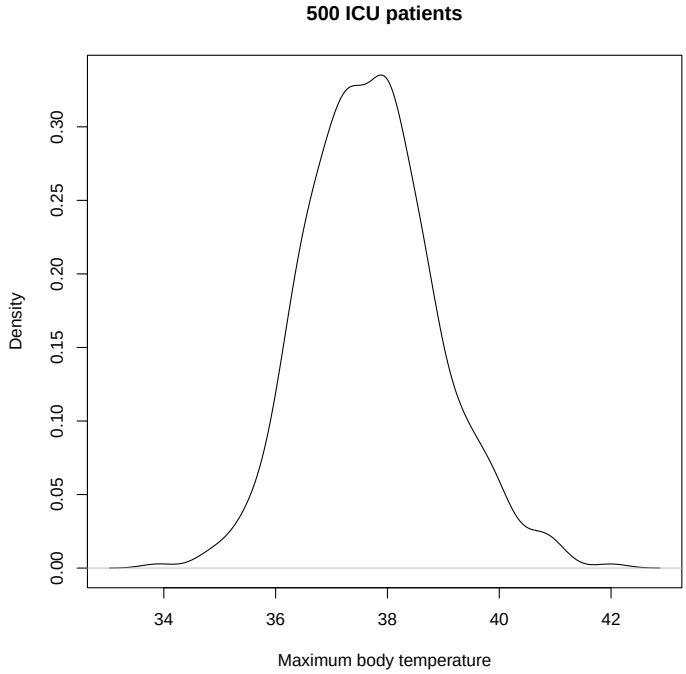
```
ggplot(ICUData, aes(x = LOS)) + geom_histogram(binwidth = 1) +
  xlab("Length of stay in days") + ylab("Absolute frequency") +
  ggtitle("500 ICU patients")
```



The figure confirms our previous computations. We get a clearly right-skewed and quite spiky distribution. The majority of patients had a LOS of only a few days. The maximum LOS was 105 days.

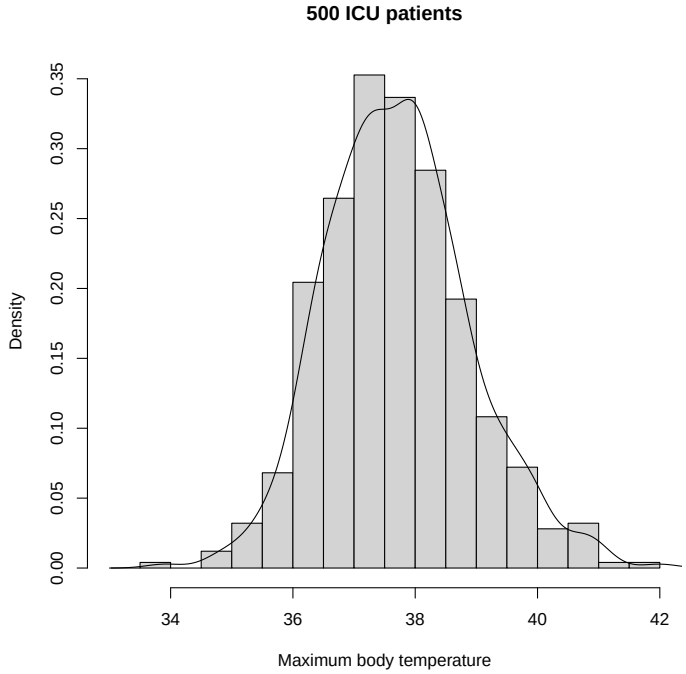
Alternatively, we can visualize the distribution of the observed values by means of their estimated density. The empirical **density** may be regarded as a smoothed version of a histogram. In R we can apply function `density` to compute the density (more precisely: the kernel density estimation). The result can be visualized via function `plot`. We consider the maximum body temperature and omit patient 398.

```
plot(density(ICUData$temperature[-398]), xlab = "Maximum body temperature",
     ylab = "Density", main = "500 ICU patients")
```



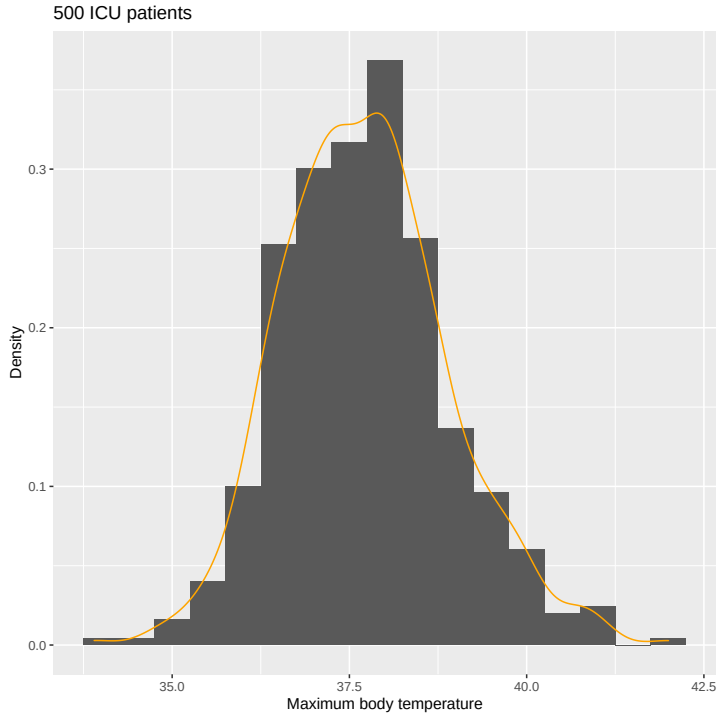
We get a density that is quite symmetric around the arithmetic mean. If we want to display histogram and density together, we must use argument `freq = FALSE` in the call of function `hist`. With this setting the density scale is used for plotting the histogram. The function `lines` adds a line to an already existing plot and can be used to add the estimated density to the histogram.

```
hist(ICUData$temperature[-398], breaks = seq(from = 33, to = 42, by = 0.5),
     xlab = "Maximum body temperature", ylab = "Density", freq = FALSE,
     main = "500 ICU patients")
lines(density(ICUData$temperature[-398]))
```



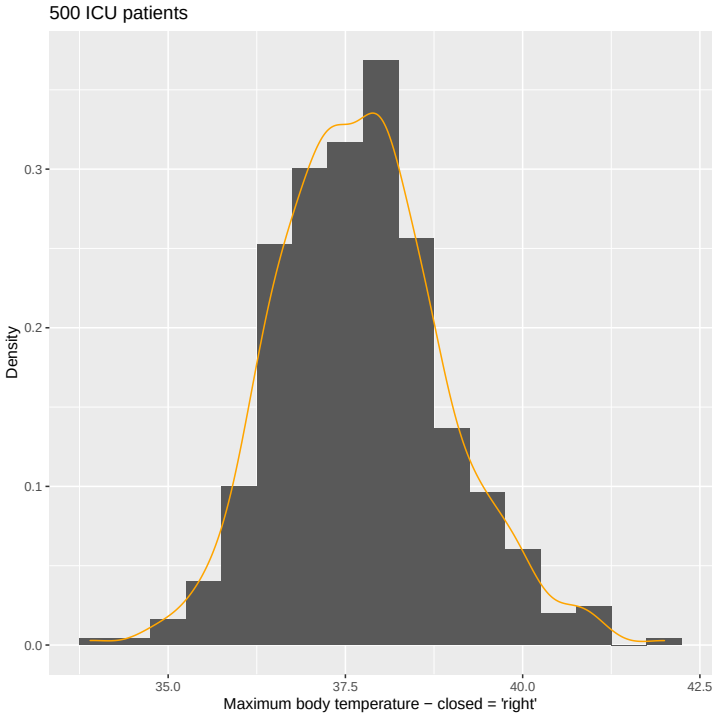
The density curve adapts well to the histogram. We use function `ggplot` in combination with functions `geom_histogram` and `geom_density` to generate a similar plot with package `"ggplot2"` (Wickham (2009)).

```
ggplot(ICUData[-398,], aes(x=temperature)) +
  geom_histogram(aes(y=..density..), binwidth = 0.5) +
  geom_density(color = "orange") + ylab("Density") +
  xlab("Maximum body temperature") +
  ggtitle("500 ICU patients")
```

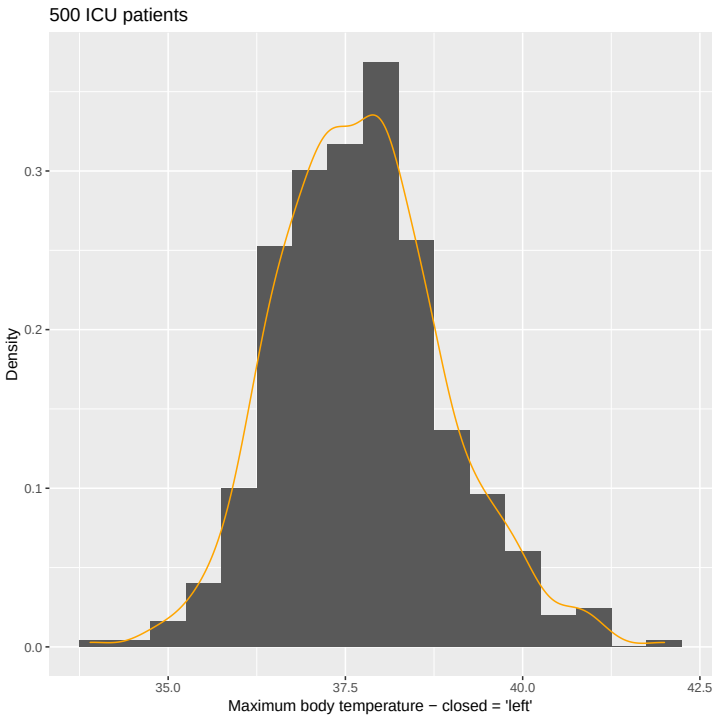


The estimated densities are very similar or even identical in both figures, however the histograms differ. That happens, because in case of `geom_histogram` one considers intervals that are open to the right-hand side and closed to the left-hand side, whereas in case of `hist` it is the other way round, i.e. open to the left and closed to the right. In priciple, this could be achieved, by additionally setting `closed = "right"` in function `geom_histogram`. Unfortunately, this does not currently work and `closed = "left"` and `closed = "right"` give identical results.

```
ggplot(ICUData[-398,], aes(x=temperature)) +
  geom_histogram(aes(y=..density..), binwidth = 0.5, closed = "right") +
  geom_density(color = "orange") + ylab("Density") +
  xlab("Maximum body temperature - closed = 'right'") +
  ggtitle("500 ICU patients")
```




```
ggplot(ICUData[-398,], aes(x=temperature)) +
  geom_histogram(aes(y=..density..), binwidth = 0.5, closed = "left") +
  geom_density(color = "orange") + ylab("Density") +
  xlab("Maximum body temperature − closed = 'left'") +
  ggtitle("500 ICU patients")
```

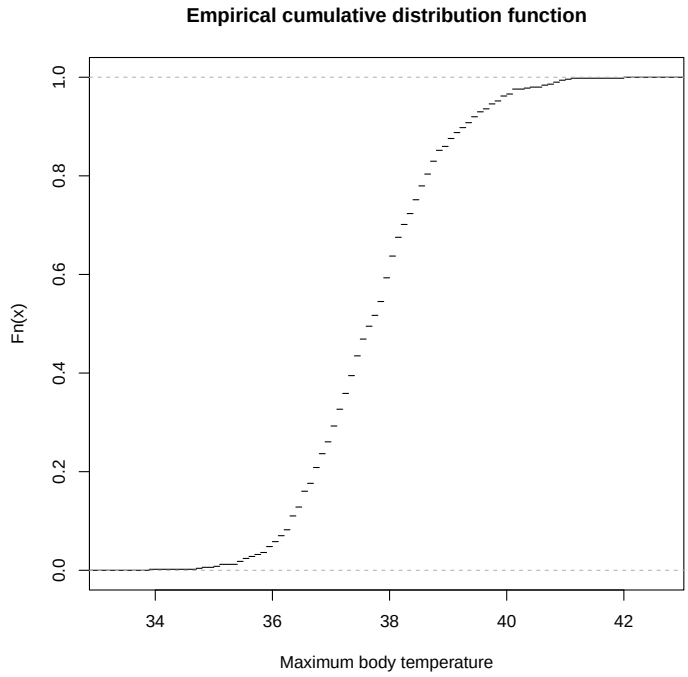


Both figures are identical. For practical applications this small bug in the function `geom_histogram` is not relevant.

Note:
In case of `ggplot`, we not only omit the temperature of patient 398 but by `[-398,]` remove all data of patient 398. More precisely, we remove row 398 from the dataset.

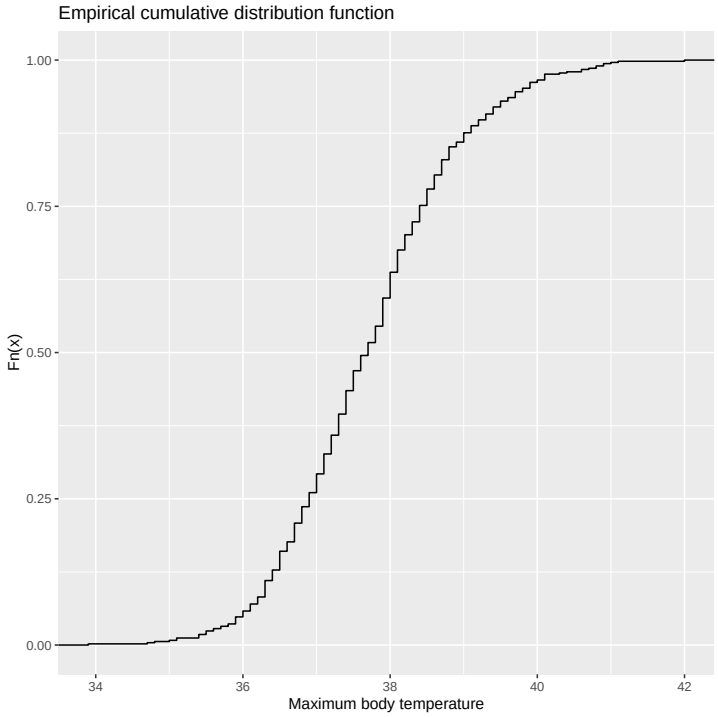
We may also visualize the distribution of the maximum body temperature by means of the empirical cumulative distribution function (cf. Definition 2.7). We first apply functions `ecdf` and `plot` where we again omit patient 398.

```
plot(ecdf(ICUData$temperature[-398]), xlab = "Maximum body temperature",
     main = "Empirical cumulative distribution function", do.points = FALSE)
```



Because of the large number of small jumps, we do not plot points (i.e., `do.points = FALSE`). We can also generate an analogous figure by means of function `stat_ecdf` of package `"ggplot2"` (Wickham (2009)).

```
ggplot(ICUData[-398,], aes(x = temperature)) + stat_ecdf() +
  xlab("Maximum body temperature") + ylab("Fn(x)") +
  ggtitle("Empirical cumulative distribution function")
```



Another important application of these way to display the empirical distribution of the data, is to compare it with the distribution of an assumed probability model. In doing so, a graphical validation of an assumed model is possible. We will investigate this in more detail in Chapter 5.

2.6.2 Bivariate Analysis

The strength and direction of the relationship between metric variables can be described by means of correlation, similar to the case of ordinal data (see Section 2.5.2). Beside rank correlations one can use the Pearson correlation.

Definition 2.26 (Pearson correlation)**.** *Let* $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n) \in \mathbb{R}^2$ *be some pairs of observations. Then, the* ***Pearson (product-moment) correlation (coefficient)*** *is*

$$r = \frac{\sum_{i=1}^{n} \left(x_i - AM(x_1, \ldots, x_n)\right) \left(y_i - AM(y_1, \ldots, y_n)\right)}{\sqrt{\sum_{i=1}^{n} \left(x_i - AM(x_1, \ldots, x_n)\right)^2 \sum_{i=1}^{n} \left(y_i - AM(y_1, \ldots, y_n)\right)^2}} \tag{2.28}$$

The Pearson correlation may attain values in $[-1, 1]$ *where* 1 *represents a perfect positive linear relation and* -1 *a perfect negative linear relation.*

We give some additional explanations.

Remark 2.27. ***(a)*** *The assumption that there is a linear relation and hence, the Pearson correlation is appropriate to describe the strength of the relationship is a rather strong assumption. Rank correlations are more flexible, as they can be used to describe monotone relationships.*

(b) *A closer look at the equations shows that Spearman's ρ (cf. Definition 2.11) is nothing else but the Pearson correlation of the ranks.*

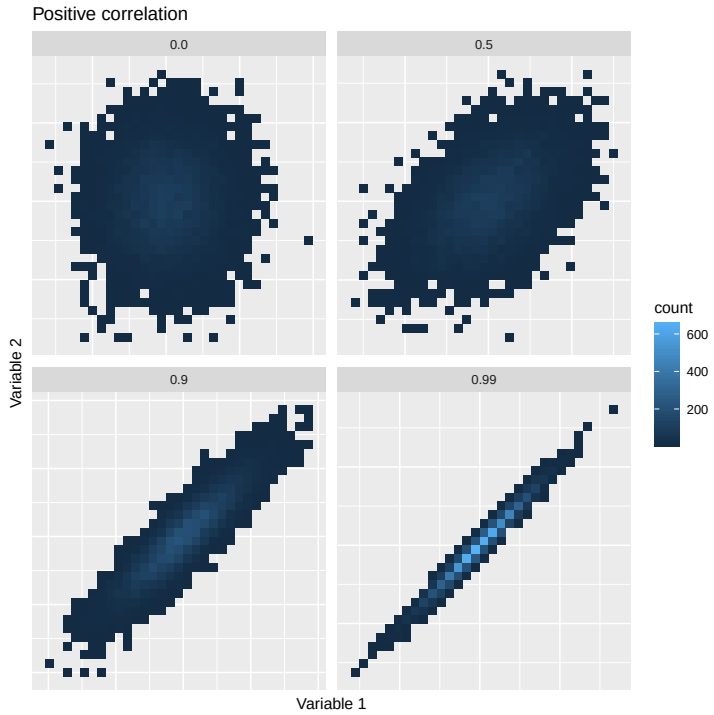
(c) *By adding $\frac{1}{n}$ to the numerator of the Pearson correlation, the numerator is identical to the* ***sample covariance*** *of the two variables. By expanding the denominator analogously, it becomes the product of the two standard deviations. That is, the Pearson correlation can be regarded as a normalized covariance.*

We investigate the connection between maximum body temperature and maximum heart rate.

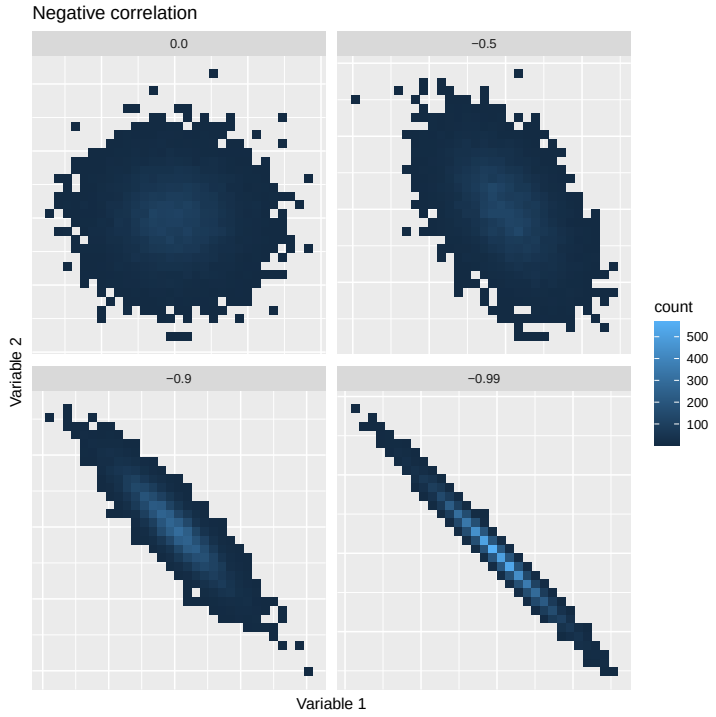
```
cor(ICUData$temperature, ICUData$heart.rate)
```

```
[1] 0.1763067
```

There is a weak positive relation; i.e., with increasing body temperature also the heart rate tends to increase. The relevance of this result is again difficult to judge, as it is unclear whether there really is a linear relation between these two variables. Before we plot the data, let's first see how point clusters generated by two linearly related variables look like. For this we use the function `simCorVars` from the package `"MKdescr"` (Kohl (2022a)).

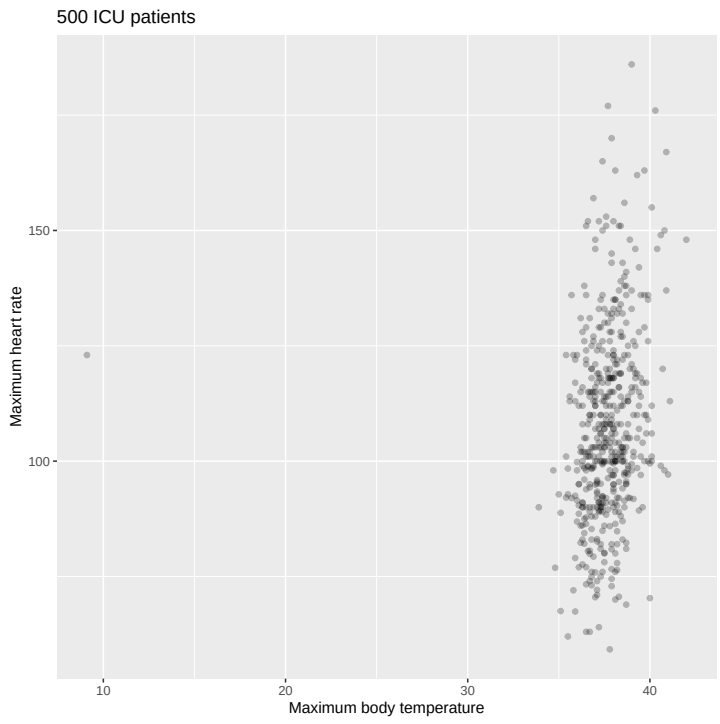


We thus obtain ellipsoidal point clusters and, in the case of a correlation of 0, a circle. In the case of negative correlations we get correspondingly mirrored point clusters.



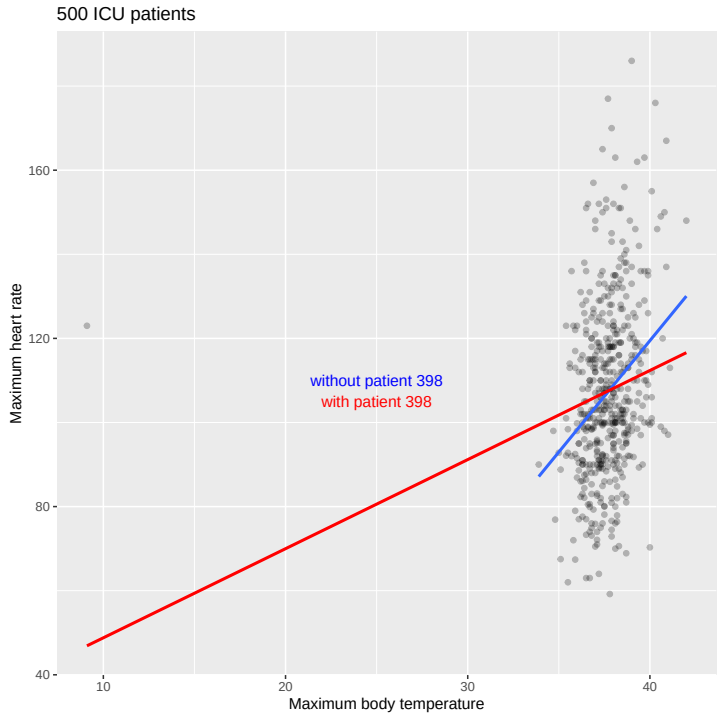
We plot the data by means of a scatter diagram.

```
ggplot(ICUData, aes(x=temperature, y=heart.rate)) +
  ## shape = 19: somewhat larger point
  ## alpha = 0.25: strength of alpha blending
  geom_point(shape=19, alpha=0.25) +
  ## title and axes labels
  ggtitle("500 ICU patients") + xlab("Maximum body temperature") +
  ylab("Maximum heart rate")
```



We again see that patient 398 is an outlier. Furthermore, we see rather narrow ellipse-shaped point cluster, which indicates a high positive correlation and seems to contradict the calculation. Here the value of patient 398 represents a so-called **leverage point**; that is, it leverages our analysis. This can be immediately visualized by using `geom_smooth` to plot the corresponding two linear regression lines (with and without patient 398). We also label the plot by applying the function `annotate`.

```
ggplot(ICUData, aes(x=temperature, y=heart.rate)) +
  ## shape = 19: somewhat larger point
  ## alpha = 0.25: strength of alpha blending
  geom_point(shape=19, alpha=0.25) +
  ## Linear regression line
  geom_smooth(data = ICUData[-398,], method = "lm", se = FALSE) +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  annotate("text", x = c(25, 25), y = c(110, 105),
           label = c("without patient 398", "with patient 398"),
           color = c("blue", "red")) +
  ## title and axes labels
  ggtitle("500 ICU patients") + xlab("Maximum body temperature") +
  ylab("Maximum heart rate")
```



The blue regression line, computed without patient 398, clearly tilts when we add patient 398 and we get a clearly different result. We repeat the analysis without this patient.

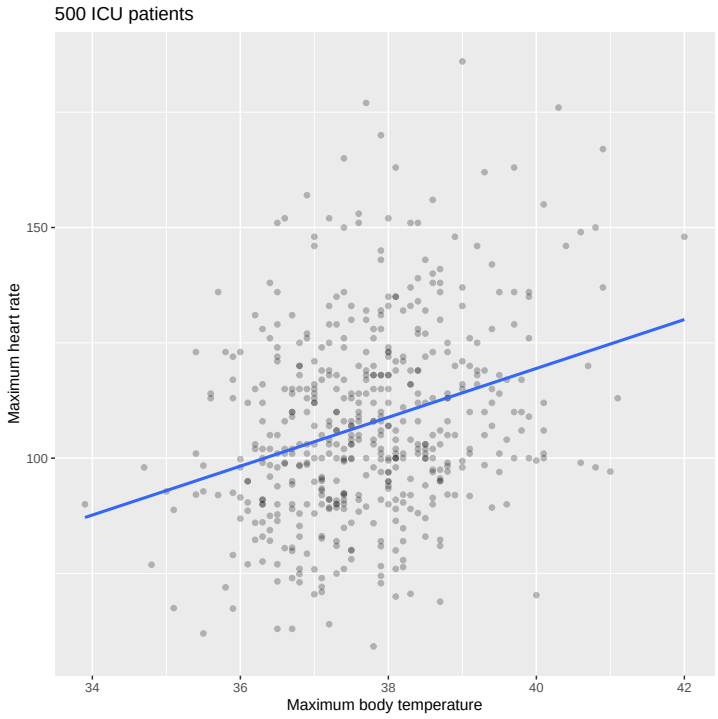
```
cor(ICUData$temperature[-398], ICUData$heart.rate[-398])
```

```
[1] 0.2978033
```

By omitting a single value, the Pearson correlation is almost twice as large, but still does not correspond to the visual impression. The very narrow ellipsoidal point cluster that can be seen suggests an even much

higher positive correlation. The reason of this is that the outlier not only interferes in the calculation, but also clearly changes the axis scaling. We repeat the plot without the patient 398.

```
ggplot(ICUData[-398,], aes(x=temperature, y=heart.rate)) +
  ## shape = 19: somewhat larger point
  ## alpha = 0.25: strength of alpha blending
  geom_point(shape=19, alpha=0.25) +
  ## Linear regression line
  geom_smooth(method = "lm", se = FALSE) +
  ## title and axes labels
  ggtitle("500 ICU patients") + xlab("Maximum body temperature") +
  ylab("Maximum heart rate")
```



Now the graphical representation coincides quite well with the calculated value, and a weak elliptical point cluster can be seen.

> Note:
> Single outlier can have a very strong influence on many statistical procedures. These include, for example, Pearson correlation and linear regression analysis. The results can be undermined by them and even the presence of a lot of data ("big data") cannot protect against this effect. In extreme cases, a single observation can cause a completely different statistical result. In statistics this is also called a "breakdown" of the applied statistical procedure.

We investigate the influence of outliers on Spearman's ρ and Kendall's τ.

```
## Spearman's rho
cor(ICUData$temperature, ICUData$heart.rate, method = "spearman")
```

```
[1] 0.2659957
```

```
cor(ICUData$temperature[-398], ICUData$heart.rate[-398], method = "spearman")
```

```
[1] 0.2707241
```

```
## Kendall's tau
cor(ICUData$temperature, ICUData$heart.rate, method = "kendall")
```

```
[1] 0.1826903
```

```
cor(ICUData$temperature[-398], ICUData$heart.rate[-398], method = "kendall")
```

```
[1] 0.1858804
```

Both rank correlations change only slightly. Thus, the transition to ranks generates a certain robustness against outliers comparable to quantiles (cf. Example 2.6). We compute the Pearson correlation and additionally apply function `rank`.

```
cor(rank(ICUData$temperature), rank(ICUData$heart.rate))
```

```
[1] 0.2659957
```

Indeed, it turns out that the Spearman correlation is just the Pearson correlation of ranks; see Remark 2.27 (b).

Note:
The popular saying: "A picture is worth a thousand words" applies also to statistics. Hence, always try to look at your data. It serves as a check of the data, e.g. for identifying wrong or erroneous observations or outliers as well as for confirming computed results.

2.7 Exercises

Use the ICU dataset and always briefly describe your results.

1. Compute absolute and relative frequencies for variable `outcome`.
2. Use a bar chart to visualize the relative frequencies for variable `outcome`. Apply the standard function `barplot` as well as the functions of package `"ggplot2"` (Wickham (2009)).
3. Determine the 95% quantile, median, inter quartile range, MAD, arithmetic mean, standard deviation, coefficient of variation, skewness, and kurtosis of variable `heart.rate`. What do the results tell you about the distribution of the values?

4. Determine the 95% quantile, median, inter quartile range, MAD, arithmetic mean, standard deviation, coefficient of variation, skewness, and kurtosis of variable `age`. What do the results tell you about the distribution of the values?

5. Determine the 95% quantile, median, inter quartile range, MAD, arithmetic mean, standard deviation, coefficient of variation, skewness, and kurtosis of variable `LOS`. What do the results tell you about the distribution of the values?

6. Draw a box-and-whisker plot as well as a histogram combined with a density plot of variable `heart.rate`. Apply the standard functions as well as the functions of package `"ggplot2"` (Wickham (2009)). Describe and interpret the plots.

7. Draw a box-and-whisker plot as well as a histogram combined with a density plot of variable `age`. Apply the standard functions as well as the functions of package `"ggplot2"` (Wickham (2009)). Describe and interpret the plots.

8. Draw a box-and-whisker plot as well as a histogram combined with a density plot of variable `LOS`. Apply the standard functions as well as the functions of package `"ggplot2"` (Wickham (2009)). Describe and interpret the plots.

9. Investigate in a suitable manner (!) the relation between variables `liver.failure` and `outcome`. Interpret the results and plot the data in an appropriate way. Is the graph in accordance with your calculations?

10. Investigate in a suitable manner (!) the relation between variables `sex` and `outcome`. Interpret the results and plot the data in an appropriate way. Is the graph in accordance with your calculations?

11. Investigate in a suitable manner (!) the relation between variables `age` and `SAPS.II`. Interpret the results and plot the data in an appropriate way. Is the graph in accordance with your calculations?

12. Investigate in a suitable manner (!) the relation between variables `age` and `LOS`. Interpret the results and plot the data in an appropriate way. Is the graph in accordance with your calculations?

13. Investigate in a suitable manner (!) the relation between variables `age` and `outcome`. Interpret the results and plot the data in an appropriate way. Is the graph in accordance with your calculations?

14. Investigate in a suitable manner (!) the relation between variables `sex` and `LOS`. Interpret the results and plot the data in an appropriate way. Is the graph in accordance with your calculations?

3 Colors and Diagrams

This rather short chapter deals with the correct use of colors and the generation of diagrams, which represent the available data in a most suitable way. It covers the following topics:

- Recommendations for handling colors
- Use of predefined color palettes
- Export of diagrams
- Recommendations for generating diagrams according to E. Tufte

The R code of this chapter is included in the R Markdown file `Colors.Rmd`, which you can download from my GitHub account (link: https://github.com/stamats/ISDR/blob/main/Colors.Rmd). Right click on `Raw`. Then you can `Save target as ...`. The least difficulties arise, if you save my R Markdown files in the same folder as the data.
First, we will install the required packages for this chapter.

```
install.packages(c("RColorBrewer", "ggsci"))
```

Make sure that you have already installed the packages from chapter 2 (Section 2.4).

```
library(ggplot2)
library(RColorBrewer)
library(ggsci)
```

As explained in Section 2.4, repeated execution of `library` should be without problems.

3.1 Colors

As we have seen in the last chapter, diagrams play a crucial role in understanding the available data. By the proper use of colors, the expressiveness and aesthetics of a graphic can be clearly improved. Figure 3.1 shows a negative example. Neither the type of diagram nor the colors are appropriately chosen. The selected type of diagram makes it hard to tell the exact proportions (as absolute or relative frequencies). The colors red and blue are very intense and not adapted to the answers. Furthermore, the graphic does not make clear, if there is another category between “We don’t care about colors” and “We love colors”. The category might exist, but nobody selected it or the category was not provided. In pie charts “empty” categories can not be directly seen.

At the DSC conference 2003, Ross Ihaka (Ihaka (2003)) specified the following options for handling colors:

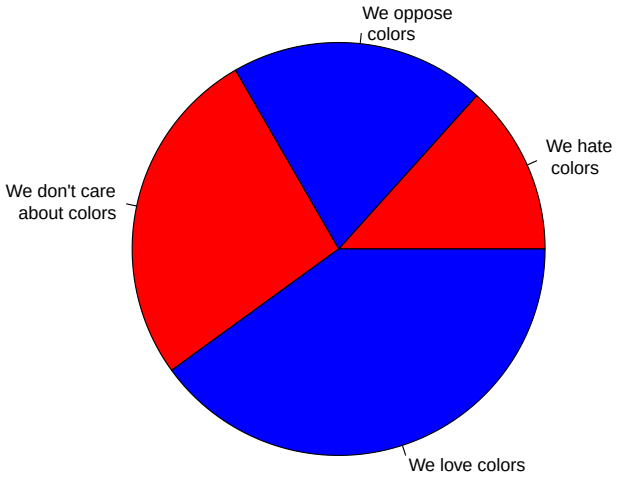


Figure 3.1: A negative example for using colors and diagrams.

1. Avoid colors.
2. Determine colors by experimentation.
3. Use “good taste” or expertise.
4. Use fixed palettes designed by an expert.
5. Look for guiding principles.

We briefly comment on these options:

Ad 1: Of course, one can try to avoid colors, but colors can be very helpful and can clearly improve the expressiveness of a graphic.

Ad 2: The determination of colors by experimentation is usually very time consuming.

Ad 3: This requires a special talent for colors or a respective experience in handling colors.

Ad 4: Good idea!

Ad 5: Where can we find such guiding principles?

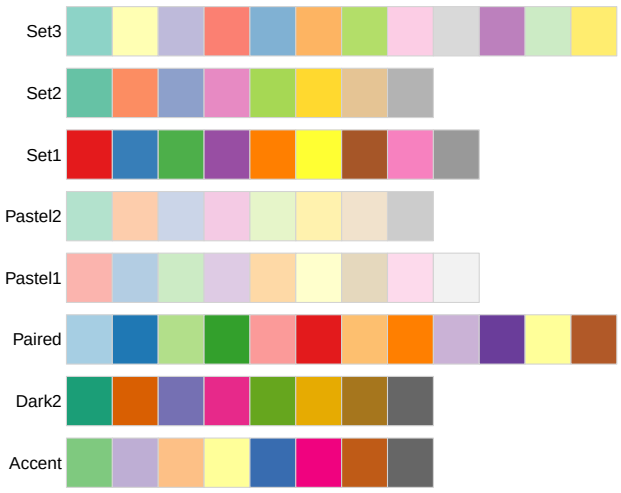
The following basic principles in handling colors are given in Zeileis et al. (2009):

- The colors should not be unappealing.

- The colors in a statistical graphic should cooperate with each other.
- The colors should work everywhere.

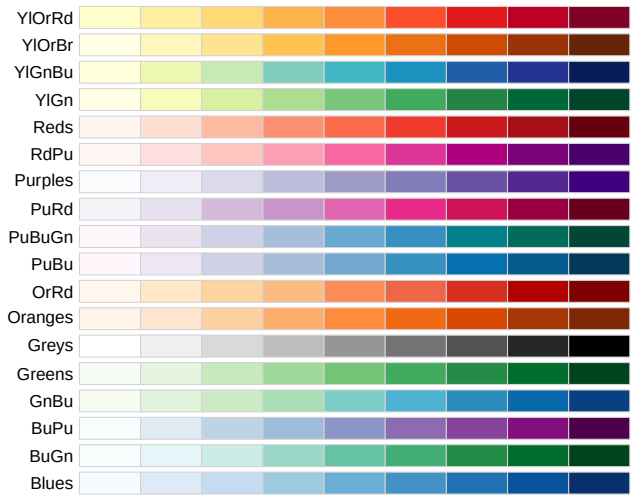
A project that applies these principles is ColorBrewer (Harrower and Brewer (2003)). It provides color palettes for various purposes on the website `www.colorbrewer2.org`. These color palettes can be applied in R by package `"RColorBrewer"` (Neuwirth (2014)). We load the package and take a look at the various color palettes using function `display.brewer.all`. First, we consider the qualitative color palettes, which can be used for displaying categorical variables.

```
1 display.brewer.all(type = "qual")
```



In this case, it is important that there is no color that dominates the others. All colors should appeal equally "important". The second group of color palettes provides colors for attributes, whose levels range from unimportant or uninteresting to important or interesting.

```
1 display.brewer.all(type = "seq")
```



Finally, there is a third group of color palettes for variables with a range from negative to neutral to positive.

```
display.brewer.all(type = "div")
```

On the website www.colorbrewer2.org one can additionally choose colors by the following criteria:

- colorblind safe

- print friendly
- photocopy safe
- LCD friendly

Figure 3.2 compares the introductory negative example with a corresponding pie chart, where the colors are adapted to the categories. For the new diagram ColorBrewer palette `RdYlGn` was applied. A further

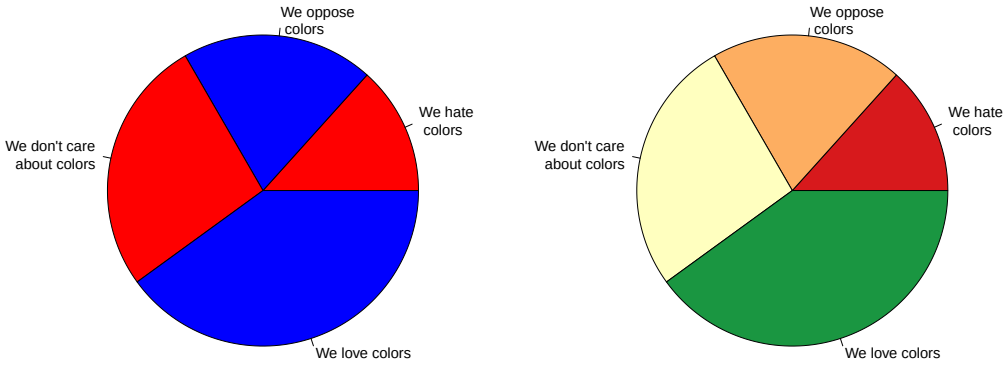


Figure 3.2: A negative example with improved colors.

improvement of the diagram could either consist of labeling the pieces by (absolute or relative) frequencies or by transferring the results to a bar chart. In particular, in case of a bar chart one could make clear that there is a category "We tolerate colors" by adding a bar of height 0; see Figure 3.3.

We plot the absolute frequencies of the types of surgeries as in Section 2.5.1, where we additionally use color palette `Set1` of ColorBrewer. For this, we first generate a vector of colors by applying function `brewer.pal` of package `"RColorBrewer"` (Neuwirth (2014)).

```
## n = 5 colors of palette with name Set1
cols ← brewer.pal(n = 5, name = "Set1")
cols
```

```
[1] "#E41A1C" "#377EB8" "#4DAF4A" "#984EA3" "#FF7F00"
```

That is, the colors are saved in hexadecimal code. R includes several functions for various color spaces, which can be used to determine the hexadecimal code of colors. For instance, if the red-green-blue (RGB) code is known, one can use function `rgb`.

```
rgb(red = 228, green = 26, blue = 28, maxColorValue = 255)
```

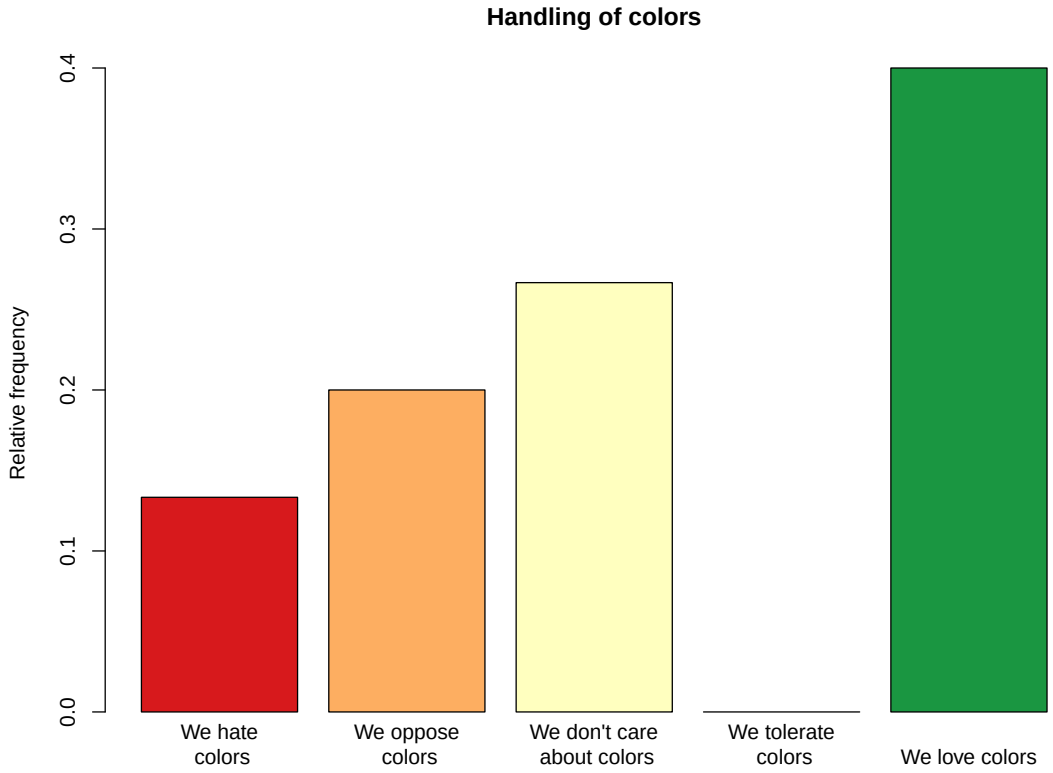


Figure 3.3: From a negative to a positive example.

```
[1] "#E41A1C"
```

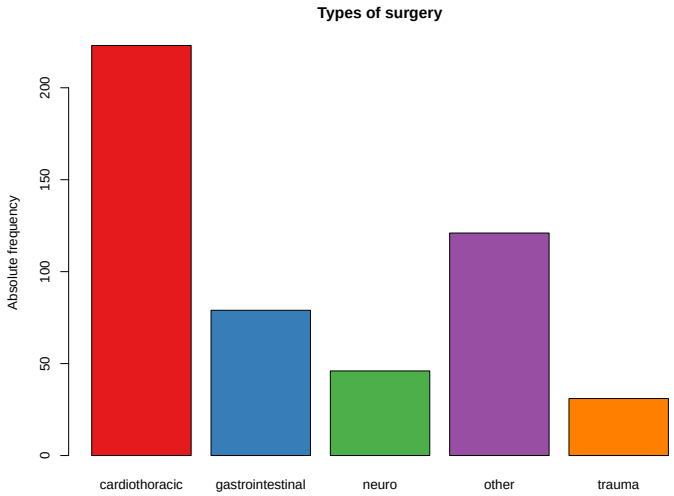
A large number of colors can also be specified by their names. More precisely, there are 657 colors in R that are saved by their names. One can use function `colors` to display these colors and function `col2rgb` to determine their red-green-blue code; e.g.

```
1 col2rgb("royalblue")
```

```
      [,1]
red     65
green  105
blue   225
```

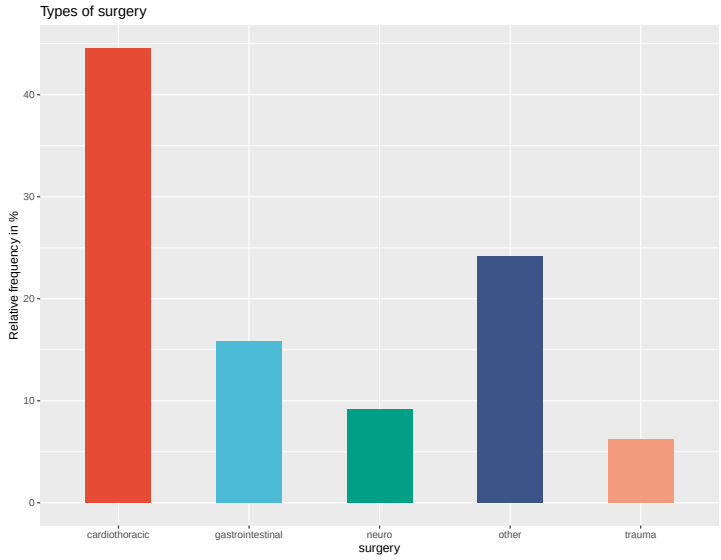
The standard functions for plotting data all have argument `col`, which can be used to specify colors. We now generate the bar chart.

```
1 ICUData ← read.csv(file = "ICUData.csv")
2 barplot(table(ICUData$surgery), main = "Types of surgery",
3         ylab = "Absolute frequency", col = cols)
```



Package "ggplot2" (Wickham (2009)) also provides various ways of using colors. We generate the bar chart of the relative frequencies included in Section 2.5.1, where we this time color the bars via palette NPG from "ggsci" package.

```
## Define data
ggplot(ICUData, aes(x=surgery)) +
  ## Add bars of relative frequencies
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5,
           ## Fill bars with color
           fill = pal_npg()(5)) +
  ## Title and label of y-axis
  ggtitle("Types of surgery") + ylab("Relative frequency in %")
```



The used function pal_npg does not directly return the color palette, but creates a function that can be used to select a given number of colors from the NPG color palette.

```
pal_npg()
```

```
function (n)
{
    n_values ← length(values)
    if (n > n_values) {
        warning("This manual palette can handle a maximum of ",
            n_values, " values. You have supplied ", n, ".",
            call. = FALSE)
    }
    unname(values[seq_len(n)])
}
<bytecode: 0x55f76b379c88>
<environment: 0x55f766bc62e8>
```

Therefore, in the code you can see two consecutive pairs of round brackets. First, with the function call the named function is generated and then immediately evaluated with the value 5.

```
pal_npg()(5)
```

```
[1] "#E64B35FF" "#4DBBD5FF" "#00A087FF" "#3C5488FF" "#F39B7FFF"
```

Besides the presented packages "RColorBrewer" (Neuwirth (2014)) and "ggsci" (Xiao (2018)) there are a lot of other packages, which provide different color palettes.

At the end of this section, we would like to discuss three situations that occur again and again in connection with color palettes. The first situation is specifically related to the "RColorBrewer" package.

```
cols2 ← brewer.pal(n = 2, name = "Set1")
```

```
Warning in brewer.pal(n = 2, name = "Set1"):
        minimal value for n is 3, returning requested palette
        with 3 different levels
```

```
cols2
```

```
[1] "#E41A1C" "#377EB8" "#4DAF4A"
```

The warning message triggered by the above code indicates that in case of this package it is not possible to select less than three colors from a color palette. The package strictly adheres to the default of the web page www.colorbrewer2.org, where it is also not possible to select fewer than three colors. So if we need only one or two colors from a ColorBrewer 2.0 color palette, we have to additionally select them with the help of the square brackets. In the following we select the first and second color.

```
cols2 ← brewer.pal(n = 3, name = "Set1")[1:2]
cols2
```

```
[1] "#E41A1C" "#377EB8"
```

We have now created a color vector that contains only the first two colors of the color palette. The second situation we want to deal with, is the specific selection of certain colors from a color palette. Suppose we want to use the colors orange, red and green from the NPG color palette in exactly this order. Here again the square brackets will help us in combination with the function `c`, with which we can create an arbitrary index vector for the selection.

```
1 cols3 ← pal_npg()(5)[c(5,1,3)]
2 cols3
```

```
[1] "#F39B7FFF" "#E64B35FF" "#00A087FF"
```

Accordingly, we first select color 5 (orange), then color 1 (red) and finally color 3 (green). Finally, we want to deal with the situation where we need more colors than are contained in the selected color palette. We can use the function `colorRampPalette` to interpolate colors. Of course, this does not make sense for every color palette. Sequential or diverging color palettes are particularly suitable for this. For diverging color palettes ColorBrewer 2.0 provides a maximum of eleven colors as we see here at the example of the color palette `RdYlBu` (red - yellow - blue).

```
1 cols11 ← brewer.pal(n = 11, name = "RdYlBu")
2 cols11
```

```
 [1] "#A50026" "#D73027" "#F46D43" "#FDAE61" "#FEE090" "#FFFFBF" "#E0F3F8"
 [8] "#ABD9E9" "#74ADD1" "#4575B4" "#313695"
```

We will now expand these eleven colors to 32 colors.

```
1 cols32 ← colorRampPalette(cols11)(32)
2 cols32
```

```
 [1] "#A50026" "#B50F26" "#C51E26" "#D52E26" "#DF412F" "#E85538" "#F26941"
 [8] "#F67D4A" "#F99254" "#FCA75E" "#FDB96B" "#FDC97A" "#FDD989" "#FEE599"
[15] "#FEF0A8" "#FEFAB7" "#FAFDC8" "#F0F9DA" "#E6F5EC" "#D9EFF6" "#C8E7F1"
[22] "#B6DEEC" "#A5D4E6" "#93C6DE" "#82B8D7" "#70A9CF" "#6197C5" "#5285BC"
[29] "#4472B3" "#3D5EA9" "#374A9F" "#313695"
```

We get the required number of colors. The function `colorRampPalette` creates a function similar to the function `pal_npg`, which can then be used to obtain the desired number of colors. This explains why two pairs of round brackets are necessary here as well.

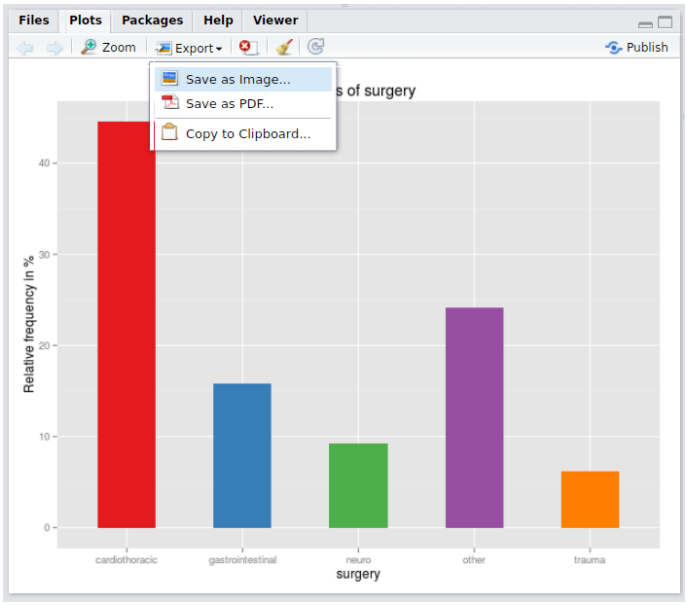


Figure 3.4: RStudio window *Plots* with an example.

3.2 Excursus: Export of Diagrams

In RStudio the generated diagrams are shown in window *Plots* and can be exported by menu item *Export* to various graphic formats or pdf; see Figure 3.4. By clicking on *Save as Image...* a new window opens, in which the size of the image, the file name, the folder and the graphic format can be chosen (see Figure 3.5). It depends on the operating systen and maybe additionally installed graphics software, which graphic formats are available. By choosing *Save as PDF...*, the window shown in Figure 3.6 opens. One can specify the size, the file name and the folder. For a quick import of plots for example in a document or email, one can use menu item *Copy to Clipboard...*. In most cases however, it is preferable to first save the graphic as image or pdf and then import the generated file.

The described options are convenient and quick and in most cases lead to the wanted result. But, especially in case of complex graphics there may be problems under certain circumstances. Moreover, sometimes it is necessary to adapt further parameters during the export such as resolution, compression rate or font size. In such a situation, one directly has to use the export functions of R. As already mentioned, the available devices depend on the operating system and maybe additionally installed software. The main functionality is provided by base package `"grDevices"` (R Core Team (2022a)). In addition, there are some contributed packages offering further options. Table 3.1 contains an overview of common devices supported by R.

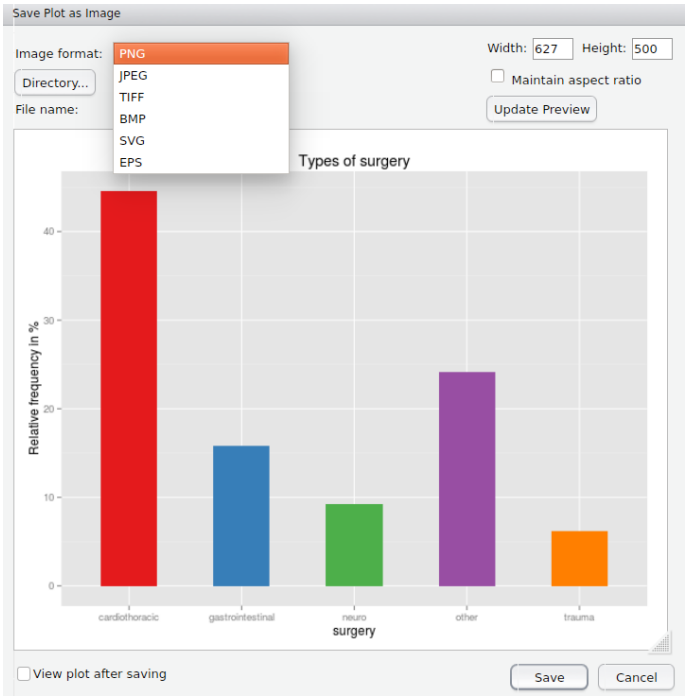


Figure 3.5: RStudio window for saving a plot as image.

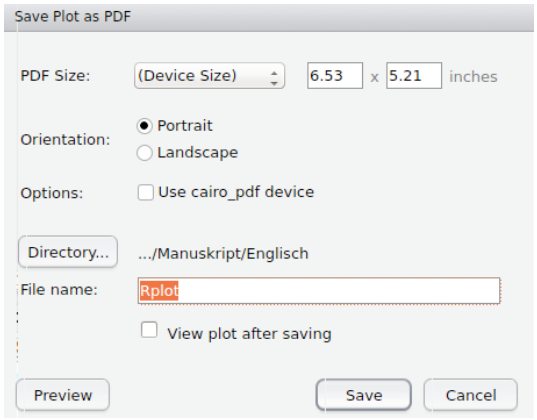


Figure 3.6: RStudio window for saving a plot as pdf file.

Note:
Raster graphics are based on a grid of pixels, where every pixel has a certain color. The best possible way to display such images is the resolution, in which they were generated. In case of rescaling, especially enlarging, the quality of these images declines. In contrast, vector graphics are based on a description of the image and can be rescaled without any problems. In addition, vector graphics often require clearly less memory.

The export of a plot to some file always consists of the following three steps:

1. Open the desired device.
2. Generate the plot.

Function name	Description
bmp	Bitmap (bmp) a standard format of raster graphics in Microsoft Windows.
jpeg	Compressed image files of raster graphics, very common in Internet.
png	Portable network graphics (png) for lossless compressed image files of raster graphics, usually more appropriate for statistical graphics than jpeg.
tiff	Tagged image file format (tiff) especially used for high-resolution printable raster graphics.
pdf	Portable document format (pdf) a very common file format that embeds graphics as vector graphics.
postscript	PostScript (ps) a vector graphics format frequently used for printing, especially the further developed Encapsulated PostScript (eps) is of interest for graphics.
svg	Scalable vector graphics (svg) a vector graphics format for web browsers.

Table 3.1: Overview of devices supported by R.

3. Close the device using function `dev.off`.

As an example, we generate a png image.

```
## 1. Open the device
## height and width in number of pixels
png(file = "Example_Image.png", height = 640, width = 640)
## 2. Generate the plot
barplot(table(ICUData$surgery), main = "Type of surgery",
        ylab = "Absolute frequency", col = cols)
## 3. Close the device
dev.off()
```

After running this code, there is an image file called `Example_Image.png` in the current working directory, which includes the generated plot.
Besides single images, one can even generate movies with R; e.g., package `"animation"` (Xie (2013)) and `"gganimate"` (Pedersen and Robinson (2022)) provides various options and contains some interesting examples.

3.3 Diagrams

The negative example of Section 3.1 (see Figure 3.1) confirms the statement in Section 2.5.1, that pie charts are not the best option for displaying information. Please, try to order the categories shown in Figure 3.7 or even try to determine the plotted frequencies. This gets even worse by introducing a further dimension in form of a three-dimensional pie chart; see Figure 3.8.

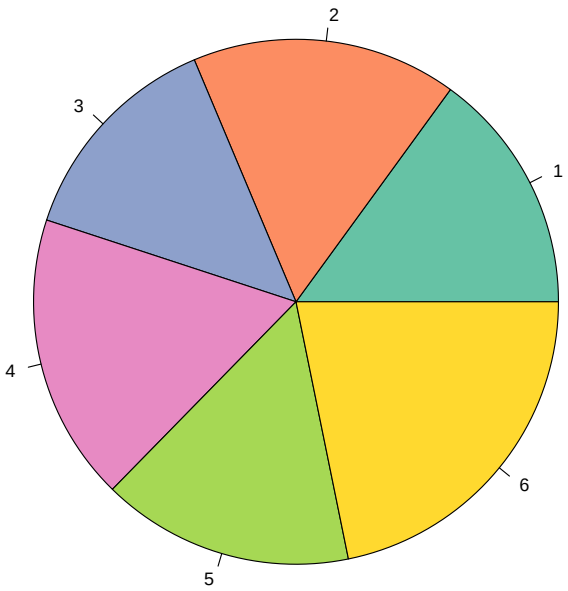


Figure 3.7: Order the categories!

Note:
The third dimension in three-dimensional pie and bar charts, which are frequently used nowadays, leads to a perspective distortion. Moreover, it contradicts one of the recommendations of E. Tufte given below, as the number of information carrying dimensions (= 3) is larger than the dimension of the plotted data (= 2).

In contrast, the order of the categories is immediately visible by using a bar chart as in Figure 3.9.

The following recommendations go back to Eduard Tufte (see Globus (1994)):

- The numbers, that can be measured off the graphic, should be directly proportional to the numerical quantities represented by them.
- Use a clear, detailed and complete labeling to avoid a graphical bias and ambiguity.
- Explanations of the data should be given on the graphic itself.
- Important events in the data should be labeled.
- It is important to show the variation of the data and not of the design.
- The number of information carrying dimensions should not exceed the dimension of the data.
- Never use graphics outside of their context.

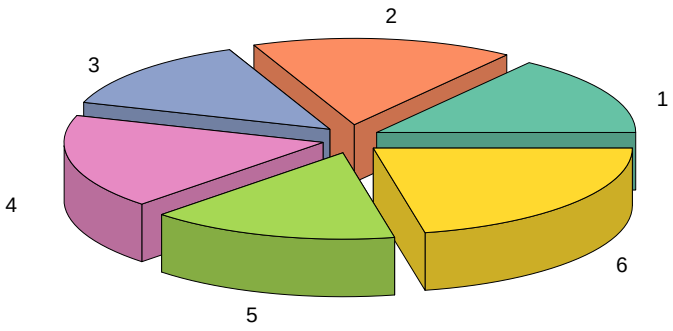


Figure 3.8: Once again: Order the categories!

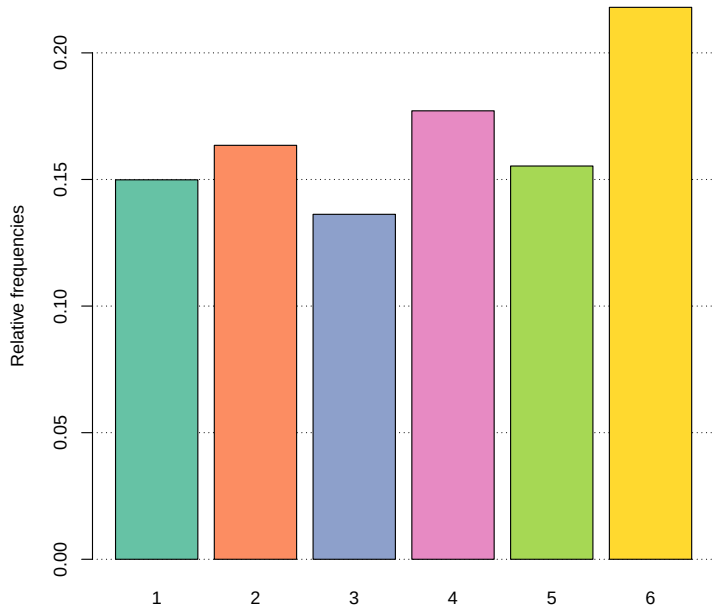
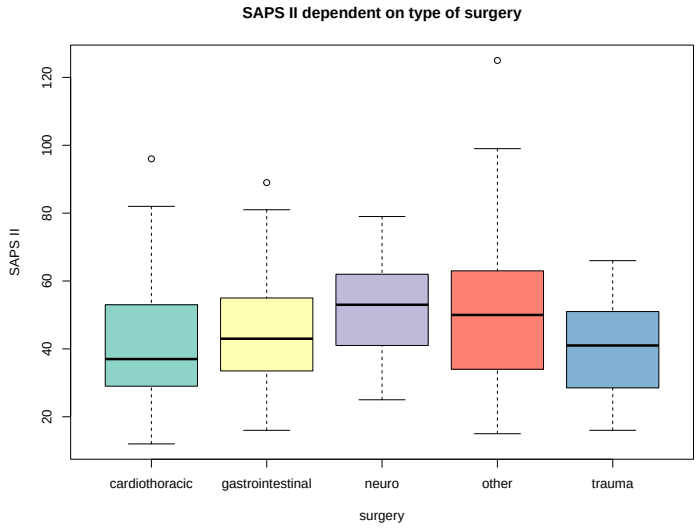


Figure 3.9: And once again: Order the categories!

In the sequel, we present some more examples for using diagrams in combination with colors. We start with a plot of the SAPS II scores for the different types of surgery, where we use box-and-whisker plots. As there is no obvious order between the types of surgery, we choose a qualitative color palette, in this case `Set3` of ColorBrewer (Harrower and Brewer (2003)). First, we apply function `boxplot`.

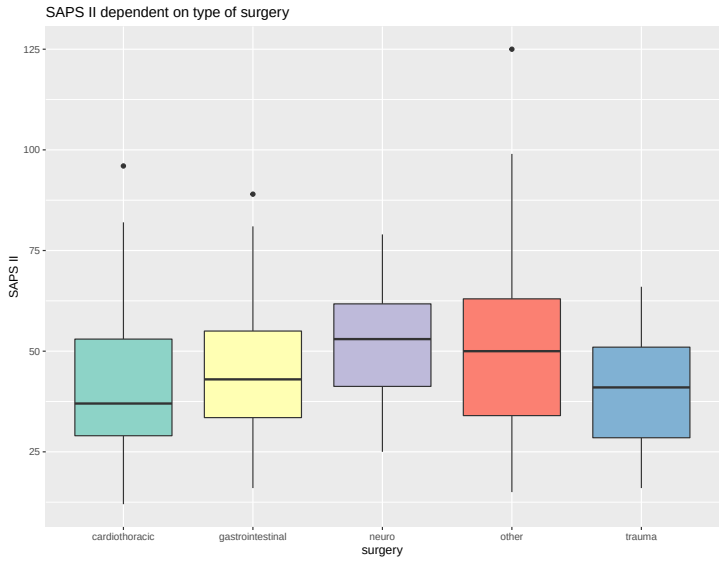
```
cols ← brewer.pal(n = 5, name = "Set3")
boxplot(SAPS.II ~ surgery, data = ICUData, ylab = "SAPS II",
        main = "SAPS II dependent on type of surgery", col = cols)
```



For splitting the scores by types of surgery, we have used a so-called formula. The expression `SAPS.II ~ surgery` means that the left-hand side `SAPS.II` has to be considered in dependence of the right hand side `surgery`. Not surprisingly, we see the largest range in case of other surgeries and the values in case of neurological surgeries tend to be higher.

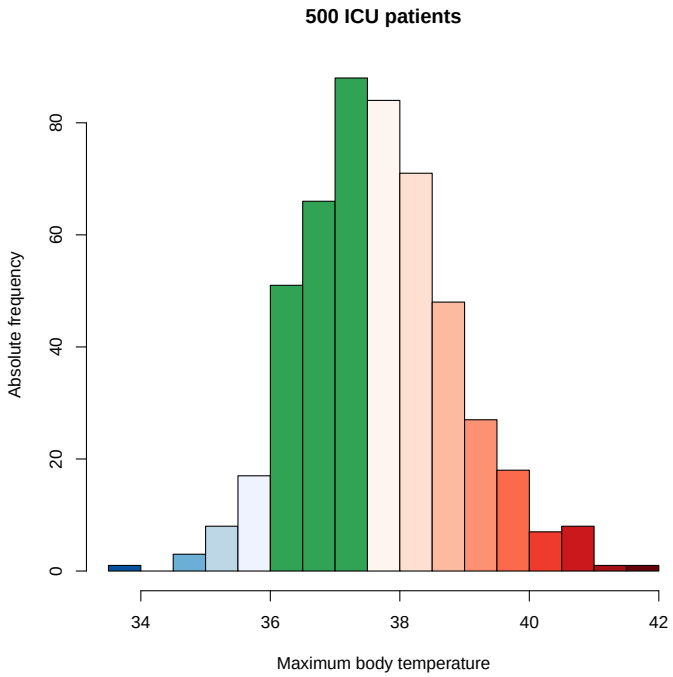
We repeat the plot using package `"ggplot2"` (Wickham (2009)). The colors can be specified by argument `fill` of function `geom_boxplot`.

```
## Define data
ggplot(ICUData, aes(x = surgery, y = SAPS.II)) +
  ## Box-and-whisker plot with colors
  geom_boxplot(fill = cols) +
  ## Labeling
  ylab("SAPS II") + ggtitle("SAPS II dependent on type of surgery")
```



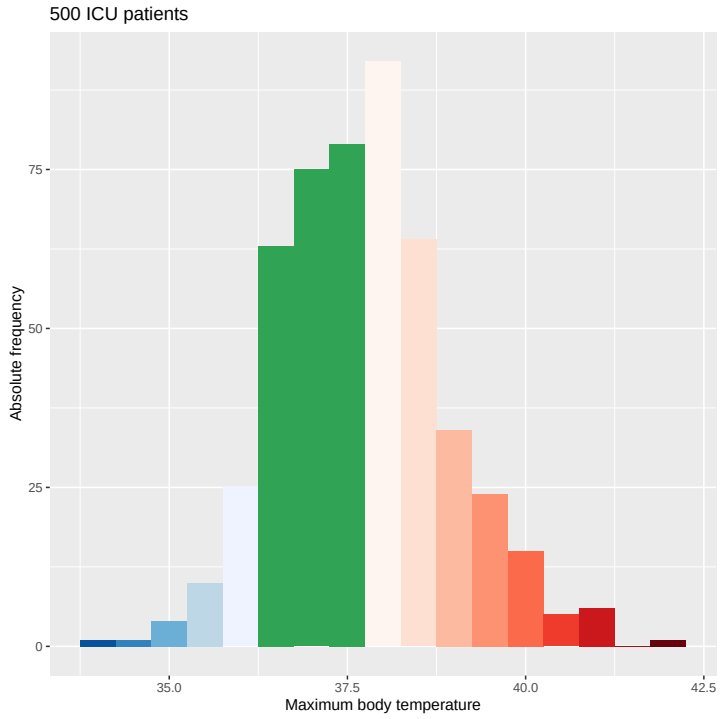
The use of colors is often also useful in case of histograms. We repeat the histogram of the maximum body temperature generated in Section 2.6.1. We split the maximum body temperature in the following three intervals: $< 36°C$ (too low), $36 - 37.5°C$ (normal), $> 37.5°C$ (too high). For the first interval, consisting of five sub-intervals, we use ColorBrewer palette `Blues` and revert the order of the colors with function `rev`. For the normal range, consisting of three sub-intervals, we use color green (more precisely: #31A354) and replicate the color with function `rep`. For the third interval, consisting of nine sub-intervals, we select ColorBrewer palette `Reds`. For getting a better overview, we omit patient 398.

```
cols1 ← rev(brewer.pal(5, "Blues"))
cols2 ← rep("#31A354", 3)
cols3 ← brewer.pal(9, "Reds")
hist(ICUData$temperature[-398], breaks = seq(from = 33.5, to = 42, by = 0.5),
     main = "500 ICU patients", ylab = "Absolute frequency",
     xlab = "Maximum body temperature", col = c(cols1, cols2, cols3))
```



We generate a similar figure by means of package "ggplot2" (Wickham (2009)). Here, there is an additional (empty) sub-interval on the left- and right-hand side. Thus, we have to add one more color in category one (too low) and three (too high).

```
ggplot(ICUData[-398,], aes(x=temperature)) +
  geom_histogram(binwidth = 0.5, fill = c(cols1, cols2, cols3)) +
  ylab("Absolute frequency") + xlab("Maximum body temperature") +
  ggtitle("500 ICU patients")
```

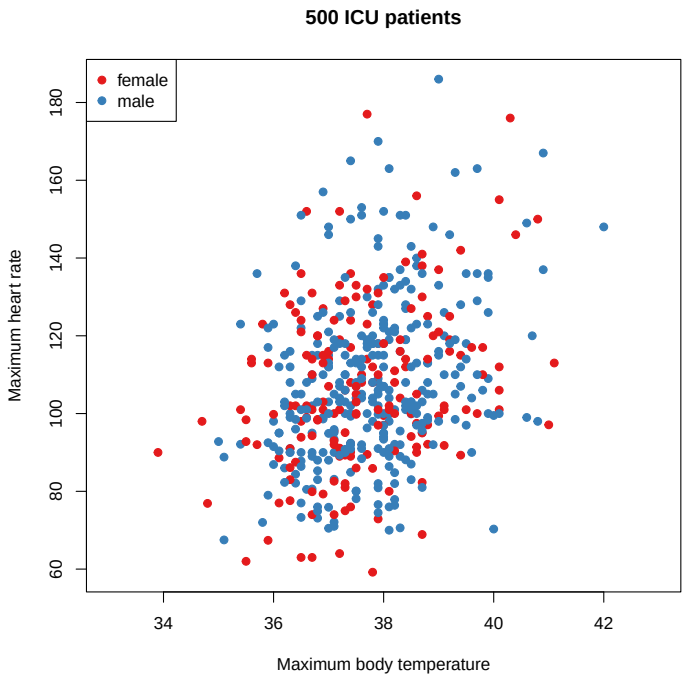


The use of colors is also helpful in case of scatter diagrams and can for example be used to visualize a third variable in addition to the variables on x and y axis. First, we apply function `plot`. We generate a vector of colors that has entry red (more precisely: #E41A1C) for females and entry blue (more precisely: #377EB8) for males. For this, we start with an empty vector generated by function `character`. Accordingly, it is a vector that can include letters or strings. By using the square brackets `[`, the vector is filled with red at positions of female patients and with blue at positions of male patients. The sign `==` is a so-called **logical operator** that can be used to check for equality.

```
## Generate empty vector
colsSex ← character(nrow(ICUData))
## Fill with colors
colsSex[ICUData$sex == "female"] ← "#E41A1C"
colsSex[ICUData$sex == "male"] ← "#377EB8"
```

Using argument `pch = 19` (pch = point character), we select a thicker point as plot symbol. The possible plot symbols are specified in the help page of function `points`. We restrict the x axis to the interval $[33, 43]$ to obtain a better overview. We also add a legend to the plot via function `legend` to explain the meaning of the colors.

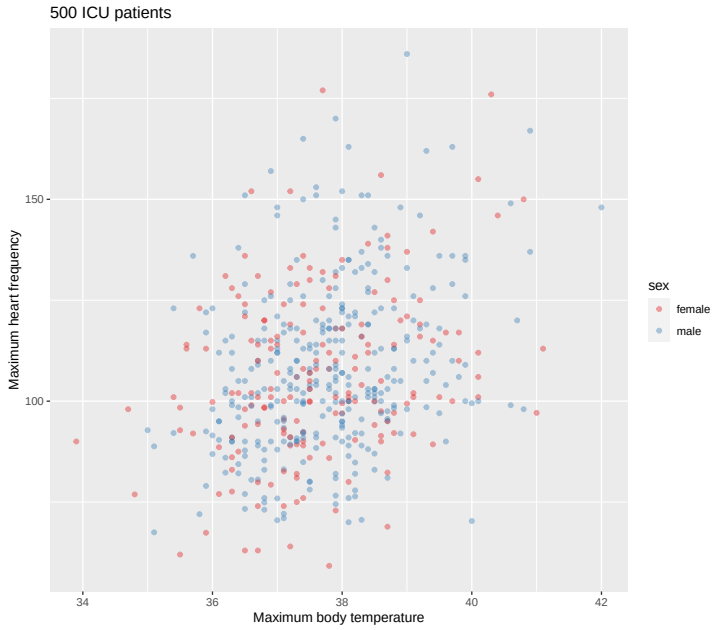
```
plot(x = ICUData$temperature, y = ICUData$heart.rate, pch = 19,
     xlab = "Maximum body temperature", ylab = "Maximum heart rate",
     main = "500 ICU patients", col = colsSex, xlim = c(33,43))
legend(x = "topleft", legend = c("female", "male"), pch = 19,
       col = c("#E41A1C", "#377EB8"))
```



The observations of females and males are quite uniformly distributed over the whole scatter diagram, which indicates that there is no influence of sex on maximum body temperature and maximum heart rate.

In case of package "ggplot2" (Wickham (2009)), it is very easy to additionally use alpha blending. Furthermore, the assignment of colors to the sexes is much easier and can be done by applying function scale_colour_manual. The order of the colors should match the order of sexes. This by default is alphabetical; i.e the first color will be female and the second will be assigned to male.

```
ggplot(ICUData[-398,], aes(x=temperature, y=heart.rate, colour=sex)) +
  ## shape = 19: somewhat larger point
  ## alpha = 0.4: strength of blending
  geom_point(shape=19, alpha=0.4) +
  ## colors
  scale_colour_manual(values = c("#E41A1C", "#377EB8")) +
  ## labeling
  ggtitle("500 ICU patients") + xlab("Maximum body temperature") +
  ylab("Maximum heart frequency")
```



3.4 Exercises

Use the ICU dataset.

1. Generate a bar chart to plot the relative frequencies of variable outcome. Use the standard function barplot as well as the functions of package "ggplot2" (Wickham (2009)) in combination with color palette Set2 of package "RColorBrewer" (Neuwirth (2014)). Save the plots as jpeg files.

2. Generate a bar chart to plot the relative frequencies of variable sex. Use the standard function barplot as well as the functions of package "ggplot2" (Wickham (2009)). Use an appropriate color palette of package "RColorBrewer" (Neuwirth (2014)). Save the plots as jpeg files.

3. Generate a bar chart to plot the relative frequencies of variable liver.failure. Use the standard function barplot as well as the functions of package "ggplot2" (Wickham (2009)). Use an appropriate color palette of package "ggsci" (Xiao (2018)). Save the plots as jpeg files.

4. Draw box-and-whisker plots of variable `age` where you split the values by variable `outcome`. Use the standard function `boxplot` as well as the functions of package `"ggplot2"` (Wickham (2009)) in combination with appropriate colors for the boxes. Save the plots as svg files.

5. Draw box-and-whisker plots of variable `bilirubin` where you distinguish between patients with and without liver failure. Use the standard function `boxplot` as well as the functions of package `"ggplot2"` (Wickham (2009)) in combination with appropriate colors for the boxes. Save the plots as svg files.

6. Draw box-and-whisker plots of variable `LOS` where you distinguish between female and male patients. Use the standard function `boxplot` as well as the functions of package `"ggplot2"` (Wickham (2009)) in combination with appropriate colors for the boxes. Save the plots as svg files.

7. Generate a histogram of variable `heart.rate`. Consider the range from 70 to 100 as normal. Use appropriate colors for the histogram and apply the standard function `hist` as well as the functions of package `"ggplot2"` (Wickham (2009)). Save the plots as png files.

8. Generate a histogram of variable `bilirubin`. Consider concentrations from 1 to 20 as normal. Use appropriate colors for the histogram and apply the standard function `hist` as well as the functions of package `"ggplot2"` (Wickham (2009)). Save the plots as png files.

9. Generate a histogram/bar chart of variable `SAPS.II`. Fill the bars in a way which makes it clear that the severity of disease increases with an increasing SAPS-II score. Use appropriate colors for the histogram/bar chart and apply the standard function `hist` as well as the functions of package `"ggplot2"` (Wickham (2009)). Save the plots as png files.

10. Draw a scatter diagram of the heart rate dependent on age and additionally mark female and male patients by colors. Use the standard function `plot` as well as the functions of package `"ggplot2"` (Wickham (2009)). Save the plots in pdf files.

11. Draw a scatter diagram of the bilirubin concentration dependent on age and additionally mark patients with and without liver failure. Use the standard function `plot` as well as the functions of package `"ggplot2"` (Wickham (2009)). Save the plots in pdf files.

12. Draw a scatter diagram of the heart rate dependent on the body temperature and additionally mark the outcome of the patients by colors. Use the standard function `plot` as well as the functions of package `"ggplot2"` (Wickham (2009)). Save the plots in pdf files.

4 Probability Distributions

We need models of probability theory to be able to infer from a sample to the underlying population (cf. Section 2.1). The basis of such models are probability distributions, where in the simplest case, the probability distributions are already the models that shall be investigated. In this case, the goal of inferential (parametric) statistics consists of estimating the unknown parameters of the assumed probability distributions from the given data.

This chapter introduces the probability, cumulative distribution, and quantile functions of discrete and (absolutely) continuous probability distributions. It covers the following probability distributions:

- Bernoulli distribution Bernoulli (p)
- Binomial distribution Binom (m, p)
- Hypergeometric distribution Hyper (m, n, k)
- Negative binomial distribution Nbinom (r, p)
  Special cases: Pascal distribution, Pólya distribution, geometric distribution
- Poisson distribution Pois (λ)
- Normal distribution Norm (μ, σ^2)
- Log-normal distribution Lnorm (μ, σ)
- Gamma distribution Gamma (σ, α)
  Special cases: Exponential distribution, Erlang distribution, χ^2 distribution
- Weibull distribution Weibull (σ, α)
- Distributions arising in connection with normal distributions: χ^2 distribution Chisq (n), t distribution t (n), F distribution F (m, n)

The R code of this chapter is included in the R Markdown file `ProbabilityDistributions.Rmd`, which you can download from my GitHub account (link: https://github.com/stamats/ISDR/blob/main/ProbabilityDistributions.Rmd). Right click on `Raw`. Then you can `Save target as ...`. The least difficulties arise, if you save my R Markdown files in the same folder as the data.
We will now install the additional packages required for this chapter. We begin with package `"distr"` (Ruckdeschel et al. (2006)).

```
1 install.packages("distr")
```

Another interesting package is package "distr6" (Sonabend and Kiraly (2022)), which unfortunately is not (resp. no longer) on CRAN. We may install the package as usual by adding the repository of "distr6" to the predefined repositories.

```
# Add repository raphaels1
options(repos = c(raphaels1 = "https://raphaels1.r-universe.dev",
                  CRAN = "https://cloud.r-project.org"))
# Install distr6
install.packages("distr6")
```

Alternatively, we can install the package directly from its GitHub repository by using package "remotes" (Csárdi et al. (2021)).

```
install.packages("remotes")
remotes::install_github("alan-turing-institute/distr6")
```

By remotes::install_github the function install_github from package "remotes" is called without explicitly loading package "remotes".
The packages from previous Chapters 2 and 3 should have been installed, too. Now, we will load all packages required for this chapter.

```
library(distr)
library(distr6)
```

As explained in Section 2.4, repeated execution of library is not problematic.

4.1 Discrete Distributions

We consider a function X, which attains its values in the space of natural numbers with certain probabilities. Such a function X is called a **discrete random variable**. The values of a random variable are called **realisations**.

We can uniquely describe the **discrete probability distribution** or **discrete distribution** of a random variable X by specifying the **probability $P(X = k)$** of all possible values $k \in \mathbb{N}$ of X. The function

$$d(k) = P(X = k) \tag{4.1}$$

is called **probability mass function** of X. The function

$$p(k) = P(X \leq k) = \sum_{i=0}^{k} P(X = i) = \sum_{i=0}^{k} d(i) \tag{4.2}$$

is called **cumulative distribution function** of X. Its inverse is the **quantile function**

$$q(p) = \min \{k \in \mathbb{N} \mid p(k) \geq p\} \qquad p \in [0, 1] \tag{4.3}$$

Important parameters of a distribution, which can also be used for its characterization, are **expectation** and **variance**. The expectation of X, $\mathrm{E}(X)$ for short, is the value of X that we can expect in mean. It holds

$$\mathrm{E}(X) = \sum_{k \in \mathbb{N}} k \cdot d(k) \tag{4.4}$$

i.e., the possible levels of X are multiplied by their probabilities and added. The variance of X, $\mathrm{Var}(X)$ for short, is the expected value of the quadratic deviations from the expectation

$$\mathrm{Var}(X) = \sum_{k \in \mathbb{N}} (k - \mathrm{E}(X))^2 \cdot d(k) \tag{4.5}$$

Often, the square root of the variance is considered, which is called **standard deviation** of X, $\sigma_X = \sqrt{\mathrm{Var}(X)}$ for short.

In this section, several important discrete distributions are introduced.

Bernoulli distribution

The simplest discrete distribution is the so-called Bernoulli distribution, for which two applications are sketched in the following example.

Example 4.1. **(a)** We consider the production of bulbs, where 1% of the bulbs are defective. That is, we can describe the production process by a discrete random variable X, which may attain the values $0 =$ defective and $1 =$ not defective. This leads to the following probability mass function

$$P(X = 0) = 0.01 \qquad \text{and} \qquad P(X = 1) = 1 - 0.01 = 0.99 \tag{4.6}$$

(b) In a randomized controlled clinical trial two interventions are compared where 65% of the patients are randomly assigned to intervention I and accordingly, 35% of the patients to intervention II. This procedure can be described by the discrete random variable X, which attains value $0 =$ intervention I with probability 65% and value $1 =$ intervention II with probability 35%, respectively. It yields the following probability mass function

$$P(X = 0) = 0.65 \qquad \text{and} \qquad P(X = 1) = 1 - 0.65 = 0.35 \tag{4.7}$$

The probability distribution that underlies both examples is defined as follows.

Definition 4.2 (Bernoulli distribution)**.** *Let X be some discrete random variable, that may only attain values* 0 *and* 1*. Then, the probability mass function of the distribution of X is*

$$d(k) = P(X = k) = p^k (1-p)^{1-k} = \begin{cases} p & \text{if } k = 1 \\ 1 - p & \text{if } k = 0 \end{cases} \tag{4.8}$$

*where $p \in [0, 1]$. The distribution is called **Bernoulli distribution** with parameter p, abbreviated by $X \sim$ Bernoulli (p).*

Binomial distribution

The Bernoulli distribution can be generalized to the so-called binomial distribution. The following example shows two possible applications.

Example 4.3. **(a)** We again consider the production of bulbs, where 1% of the bulbs is defective and want to check the quality of the last batch. For this purpose, we randomly draw **with** replacement a sample of size $m = 20$ bulbs from the batch (=population). Let X be the random variable describing the number of defective bulbs. By means of the distribution of X, we can for instance specify how likely it is to draw exactly one defective bulb. We get

$$P(X = 1) = 20 \cdot 0.01 \cdot 0.99^{19} = 16.5\% \tag{4.9}$$

Because of the 20 draws, there are 20 possibilities to draw a defective bulb, which happens with a probability of 0.01. In the remaining 19 draws a properly functioning bulb is drawn with probability 0.99 in each draw. Due to the (stochastic) independence of the draws, the probabilities are multiplied.

(b) In 2014, the prevalence of diabetes among adults amounted to about 9% (WHO (2015b)), where **prevalence** is the proportion of a population that has a disease. We conduct a trial and randomly draw **with** replacement a sample of 50 persons. The number of persons having diabetes in our sample is denoted by the random variable X. How likely is it, that our sample contains at least two persons with diabetes? In this case, it is simpler to consider the so-called complementary event: the sample contains no or exactly one person with diabetes. We obtain

$$P(X = 0) = 0.91^{50} = 0.9\% \qquad \text{and} \qquad P(X = 1) = 50 \cdot 0.09 \cdot 0.91^{49} = 4.4\% \tag{4.10}$$

Thus, the wanted probability reads

$$P(X \geq 2) = 1 - P(X \leq 1) = 1 - P(X = 0) - P(X = 1) = 1 - 0.009 - 0.044 = 94.7\% \tag{4.11}$$

In general, we get the following discrete probability distribution.

Definition 4.4. *We consider a box (urn) with black and white balls, where the proportion of white balls is equal to $p \in [0, 1]$. We randomly draw m-times ($m \in \mathbb{N}$) with replacement from this box and describe the number $k \in \{1, 2, \ldots, n\}$ of drawn white balls by the random variable X. Then, the probability mass function of X reads*

$$d(k) = P(X = k) = \binom{m}{k} p^k (1 - p)^{m-k} \tag{4.12}$$

where

$$\binom{m}{k} = \frac{m!}{k!(m - k)!} \tag{4.13}$$

is the binomial coefficient and ! indicates factorials. This distribution is called ***Binomial distribution*** *with parameters m and p, abbreviated by $X \sim Binom\,(m, p)$.*

We give some additional explanations.

Remark 4.5. ***(a)*** *The factorial of $k \in \mathbb{N}$ is defined as*

$$k! = \begin{cases} 1 & \text{if } k = 0 \\ 1 \cdot 2 \cdot \ldots \cdot k & \text{if } k \geq 1 \end{cases} \tag{4.14}$$

(b) *A closer look at the probability mass functions of the Bernoulli and the binomial distribution shows $Bernoulli\,(p) = Binom(1, p)$.*

(c) *Expectation and variance of $Binom\,(m, p)$ are*

$$E(X) = m \cdot p \qquad Var(X) = m \cdot p \cdot (1 - p) \tag{4.15}$$

The statistical software R includes the probability mass functions, cumulative distribution functions and quantile functions of many discrete probability distributions. In general, the names of these basic functions always consist of a prefix and some abbreviation of the name of the probability distribution. The possible prefixes are

`d`: probability mass function

`p`: cumulative distribution function

`q`: quantile function

`r`: function for generating (pseudo) random numbers

Therefore, it is sufficient to know the abbreviation of the distribution to apply the respective functions.

In case of the binomial distribution, the abbreviation is `binom` and the respective functions are: `dbinom`, `pbinom`, `qbinom`, and `rbinom`. The parameters m and p of the binomial distribution are called `size` and `prob` in R.

We compute the probabilities of Example 4.3 using R.

```
## a) Exactly one defective bulb
dbinom(1, size = 20, prob = 0.01)
```

```
[1] 0.1652337
```

```
## b) No person with diabetes
dbinom(0, size = 50, prob = 0.09)
```

```
[1] 0.008955083
```

```
## b) Exactly one person with diabetes
dbinom(1, size = 50, prob = 0.09)
```

```
[1] 0.04428338
```

For determining the probability that at least two persons with diabetes are drawn, it is easier to apply the cumulative distribution function.

```
## b) At least two persons with diabetes: 1-P(X ≤ 1)
1 - pbinom(1, size = 50, prob = 0.09)
```

```
[1] 0.9467615
```

Alternatively and numerically somewhat more precise, we can compute this probability by using argument `lower.tail = FALSE`. Then, the probability $P(X > k)$ instead of $P(X \leq k)$ is computed.

```
## b) At least two persons with diabetes: P(X > 1)
pbinom(1, size = 50, prob = 0.09, lower.tail = FALSE)
```

```
[1] 0.9467615
```

By means of the quantile function, we can for instance determine how many defective bulbs we can **at most** expect with a probability of **at least** 99%.

```
qbinom(0.99, size = 20, prob = 0.01)
```

```
[1] 2
```

Consequentially, if there are more than two defective bulbs during quality control, it may indicate a quality problem; i.e., a larger proportion of defective bulbs. Because it is very unlikely ($< 1\%$) to draw three or more defective bulbs, if there are only 1% defective bulbs in the batch. The probability is about 0.1%.

```
pbinom(2, size = 20, prob = 0.01, lower.tail = FALSE)
```

```
[1] 0.001003576
```

Function `rbinom` can be used to generate random numbers. If we adapt this to our diabetes example, every random number represents a trial, more precisely, the number of persons with diabetes in that trial. We simulate ten trials.

```
rbinom(10, size = 50, prob = 0.09)
```

```
 [1] 2 5 6 3 5 1 6 7 4 6
```

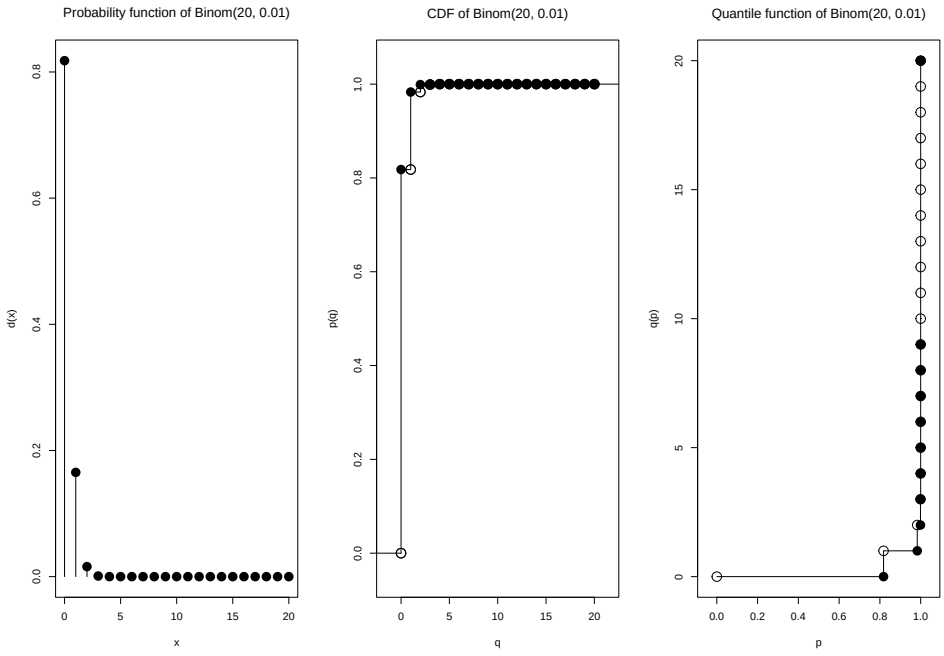
Note:
The increase of the power of computers raised the popularity of simulations considerably. Nowdays, complex probability models can be simulated thousand of times. Simulations are used, for instance, for planning of clinical trials, weather reports or currently to predict the course of the COVID19 pandemic.

We can also plot the probability mass function, the cumulative distribution function, and the quantile function of this binomial distribution. For this, package `"distr"` (Ruckdeschel et al. (2006)) can be used, which includes an object-oriented implementation of probability distributions. We load the package and by function `Binom` generate a random variable X with distribution Binom $(20, 0.01)$ matching our bulb example.

```
library(distr)
X ← Binom(size = 20, prob = 0.01)
```

By means of function `plot`, we can display the probability (mass) function, the cumulative distribution function (CDF), and the quantile function of a given random variable.

```
plot(X)
```

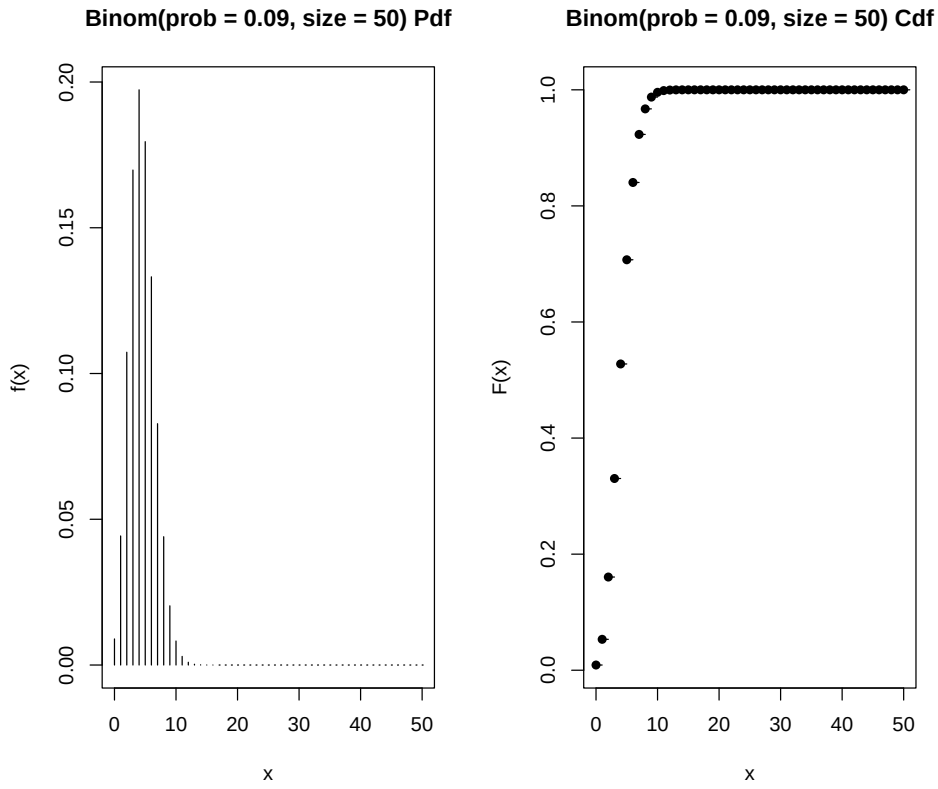


Even though `"distr6"` package is relatively new package, it can be seen as further development of `"distr"`. Instead so called S4 object oriented programming, it uses R6 object oriented programming. We load the package and create a random variable $X6$ with distribution Binom $(50, 0.09)$, which corresponds to our diabetes example.

```
X6 ← Binomial$new(prob = 0.09, size = 50)
```

By means of function `plot`, we can the probability (mass) function and cumulative distribution function (CDF) of the given random variable.

```
1 plot(X6)
```



With function `summary` (more precisely, the respective method for distr6-objects) one can get an overview of the most important properties of the given random variable.

```
1 summary(X6)
```

```
Binomial Probability Distribution.
Parameterised with:

      Id Support Value            Tags
1:  prob    [0,1]  0.09 linked,required
2: qprob    [0,1]       linked,required
3:  size       ℕ+    50        required


Quick Statistics
        Mean:            4.5
        Variance:        4.095
        Skewness:        0.4052163
        Ex. Kurtosis:    0.1242002

Support: {0, 1,...,49, 50}      Scientific Type: ℕ0
```

```
Traits:          discrete; univariate
Properties:      asymmetric; leptokurtic; positive skew
```

Hypergeometric distribution

If we draw without instead of with replacement, it results in the so-called hypergeometric distribution. The following example is very similar to Example 4.3.

Example 4.6. **(a)** We consider a box (= population) with $m+n = 500$ bulbs, where $m = 5$ are defective and randomly draw **without** replacement a sample of $k = 20$ bulbs. Let X be the random variable describing the number of defective bulbs in our sample. By means of the distribution of X, we can for instance determine how likely it is to draw no defective bulb. It holds

$$P(X = 0) = \frac{495}{500} \cdot \frac{494}{499} \cdot \ldots \cdot \frac{476}{481} = 81.5\% \tag{4.16}$$

There are 20 draws, where in each draw a functioning bulb is drawn and put aside. Hence, numerator and denominator are reduced by one after each draw; i.e., the proportion of defective bulbs changes from draw to draw.

(b) In 2000, the population of Andorra was about 66000 inhabitants (Wikipedia contributors (2022a)), where about 6000 inhabitants (WHO (2015a)) had diabetes. We conduct a trial in Andorra and randomly draw **without** replacement 50 inhabitants. Let X be the random variable describing the number of persons having diabetes in our sample. How likely is it that there is at least one person in our sample having diabetes? We consider the complementary event: the sample includes no person with diabetes and obtain

$$P(X = 0) = \frac{60000}{66000} \cdot \frac{59999}{65999} \cdot \ldots \cdot \frac{59951}{65951} = 0.9\% \tag{4.17}$$

Thus, the wanted probability is

$$P(X \geq 1) = 1 - P(X = 0) = 1 - 0.009 = 99.1\% \tag{4.18}$$

We define the hypergeometric distribution.

Definition 4.7. *We consider a box (urn) with $m \in \mathbb{N}$ white and $n \in \mathbb{N}$ black balls and randomly draw $k \in \mathbb{N}$ balls without replacement ($k < m+n$). The random variable X, describing the number j of white balls in the sample ($j \leq m$), has the following probability mass function*

$$d(j) = P(X = j) = \frac{\binom{m}{j}\binom{n}{k-j}}{\binom{m+n}{k}} \tag{4.19}$$

The distribution is called ***hypergeometric distribution*** *with parameters m, n and k, abbreviated by $X \sim Hyper(m, n, k)$.*

We give some additional explanations.

Remark 4.8. ***(a)*** *For large populations and samples the computation of the binomial coefficients included in the definition of the hypergeometric distribution is difficult. This is caused by the fact that factorials of large numbers have to be determined and the factorial grows exponentially.*

(b) *Already for populations of a moderate size and if the sample is not too large compared to the population, the difference between hypergeometric and binomial distribution is very small. It means, it only happens with a small probability that the same ball is drawn more than once.*

(c) *Expectation and variance of Hyper*(m,n,k) *read*

$$E(X) = k \cdot \frac{m}{m+n} \qquad Var(X) = k \cdot \frac{m}{m+n} \cdot \frac{n}{m+n} \cdot \frac{m+n-k}{m+n-1} \tag{4.20}$$

The formulas show a certain analogy to the binomial distribution. The factor $\frac{m+n-k}{m+n-1}$ *representing the essential difference to the binomial distribution is called* ***finite population correction****. We will meet it once again in Example 5.13.*

The hypergeometric distribution is abbreviated by `hyper` in R leading to functions `dhyper`, `phyper`, `qhyper`, and `rhyper`. We compute the probabilities of Example 4.6 using R.

```
## a) no defective bulb
dhyper(0, m = 5, n = 495, k = 20)
```

```
[1] 0.8146893
```

```
## b) no person with diabetes
dhyper(0, m=6000, n=60000, k = 50)
```

```
[1] 0.008502747
```

We can compute the probability of at least one person with diabetes by directly applying function `phyper` with argument `lower.tail = FALSE`

```
## b) at least one person with diabetes
phyper(0, m=6000, n=60000, k=50, lower.tail = FALSE)
```

```
[1] 0.9914973
```

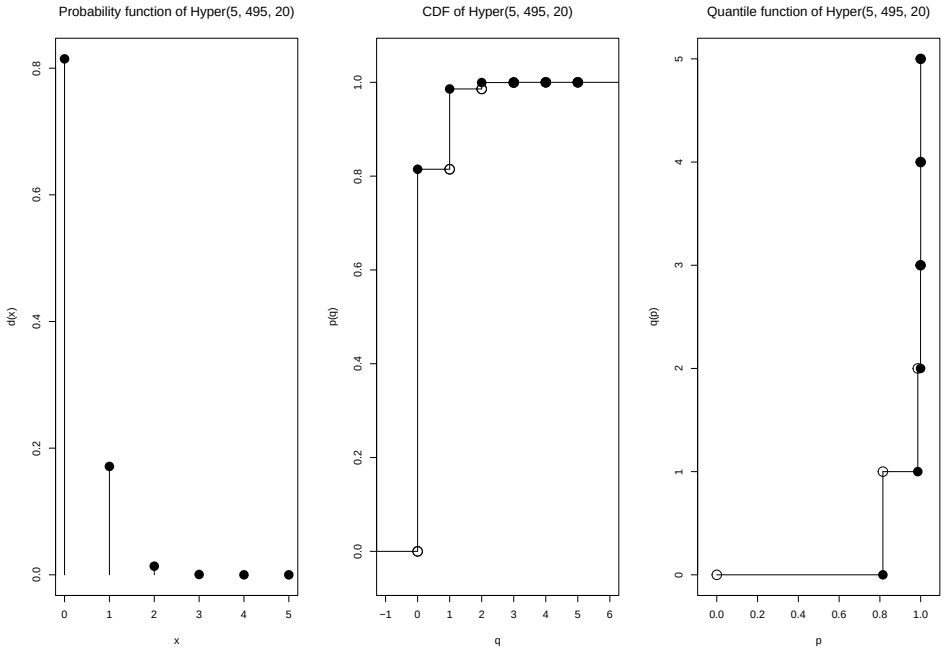
We simulate 10 samples of size 50 for our diabetes example.

```
rhyper(10, m=6000, n=60000, k=50)
```

```
 [1] 5 3 3 5 5 3 9 4 8 6
```

That is, every number represents the number of persons with diabetes in a random sample of size 50. We visualize the distribution Hyper$(5, 495, 20)$ of the bulb example by means of package `"distr"` (Ruckdeschel et al. (2006)).

```
X ← Hyper(m=5, n=495, k=20)
plot(X)
```



We use function `summary` of package `"distr6"` (Sonabend and Kiraly (2022)) to summarize the distribution of the light bulb example.

```
X6 ← Hypergeometric$new(size = 500, successes = 5, draws = 20)
summary(X6)
```

```
Hypergeometric Probability Distribution.
Parameterised with:

          Id              Support Value            Tags
1:     draws {0, 1,...,499, 500}    20        required
2:  failures {0, 1,...,499, 500}       linked,required
3:      size                   ℕ0   500        required
4: successes {0, 1,...,499, 500}     5 linked,required


Quick Statistics
        Mean:           0.2
        Variance:       0.1904609
        Skewness:       2.074205
        Ex. Kurtosis:   3.837489

Support: {0, 1,...,4, 5}        Scientific Type: ℕ0

Traits:         discrete; univariate
Properties:     asymmetric; leptokurtic; positive skew
```

As the following comparison for our bulb example shows, the hypergeometric and the binomial distribution already yield quite similar results, although the population is relatively small.

```
## probability of 0, 1, 2, 3 defective bulbs
## with replacement
dbinom(0:3, size=20, prob=0.01)
```

```
[1] 0.8179069376 0.1652337248 0.0158557615 0.0009609552
```

```
## without replacement
dhyper(0:3, m=5, n=495, k=20)
```

```
[1] 0.8146893166 0.1711532178 0.0136348475 0.0005134461
```

With operator : we can quickly generate integer sequences; e.g.

```
0:3
```

```
[1] 0 1 2 3
```

```
8:11
```

```
[1]  8  9 10 11
```

```
-3:5
```

```
[1] -3 -2 -1  0  1  2  3  4  5
```

Negative binomial distribution

Another important discrete distribution, which is in a certain way related to the binomial distribution, is the negative binomial distribution. We start with an introductory example showing possible applications of this distribution.

Example 4.9. **(a)** We again consider the production of bulbs, where 1% of the bulbs is defective. Let X be the random variable that describes the number of functioning bulbs drawn (with replacement) until the first defective bulb is obtained. How likely is it that exactly the 20th bulb is the first defective bulb? That is, we first get 19 functioning bulbs leading to

$$d(19) = P(X = 19) = 0.99^{19} \cdot 0.01 = 0.8\% \tag{4.21}$$

(b) In 2014, the worldwide prevalence (disease frequency) of diabetes in adults was about 9% (WHO (2015b)). We conduct a trial and draw (with replacement) a sample of 250 persons. We need at least 20 persons with diabetes such that our trial has the required validity (power). How likely is it that we get the necessary number of diabetes patients at the latest with inclusion of the 250th person? That is, that we have to draw at most 230 persons without diabetes. The answer is, as we will see below,

$$p(230) = P(X \leq 230) = \sum_{l=0}^{230} \binom{l + 20 - 1}{l} \cdot 0.91^{l} \cdot 0.09^{20} = 74.1\% \tag{4.22}$$

Thus, we will get 20 diabetes patients with a probability of about 74%.

We define the negative binomial distribution.

Definition 4.10 (Negative binomial distribution)**.** *We consider a box (urn) with black and white balls, where the proportion of white balls is $p \in [0,1]$. We randomly draw with replacement from the box until we have got $r \in \mathbb{N}$ white balls. Let X be the random variable describing the number $k \in \mathbb{N}_0 = \{0,1,2,\ldots\}$ of black balls that we obtain until we have got r white balls for the first time. The probability mass function of X is*

$$d(k) = P(X = k) = \binom{k+r-1}{k}(1-p)^k p^r \tag{4.23}$$

*The distribution is called **negative binomial distribution** with parameters r and p, abbreviated by: $X \sim Nbinom(r,p)$.*

We give some additional explanations.

Remark 4.11. ***(a)*** *The negative binomial distribution is a so-called **waiting time distribution**. We can apply it to specify how many unsuccessful attempts or eventless time intervals we have to wait until the required number of successes or events has occurred.*

(b) *The negative binomial distribution can be generalized such that parameter $r \in (0,\infty) \subset \mathbb{R}$.*

(c) *The negative binomial distribution is sometimes also called **Pascal distribution** or **Pólya distribution**. This mainly happens when the range of r is important. The name Pascal distribution is usually used if $r \in \mathbb{N}$ and the name Pólya distribution if $r \in (0,\infty)$. In case of $r = 1$, it is also called **geometric distribution**.*

(d) *Expectation and variance of $Nbinom(r,p)$ are*

$$E(X) = r\frac{p}{1-p} \qquad Var(X) = r\frac{p}{(1-p)^2} \tag{4.24}$$

In R, the negative binomial distribution is abbreviated by `nbinom` and the parameters are called `size` and `prob` as in case of the binomial distribution. Thus, we get functions `dnbinom`, `pnbinom`, `qnbinom`, and `rnbinom`. We compute the probabilities of Example 4.9 using R.

```
## a) 20th bulb = 1st defective bulb
dnbinom(19, size = 1, prob = 0.01)
```

```
[1] 0.008261686
```

```
## b) At most 250 persons to get 20 patients with diabetes
pnbinom(230, size = 20, prob = 0.09)
```

```
[1] 0.7407983
```

We can also use the quantile function in case of the diabetes example. We can for instance determine the sample size, which is needed, such that we achieve our goal of 20 diabetes patients with a given (high) probability. In such cases 90%, 95%, or even 99% are frequently used. In case of **at least** 95% certainty, we obtain

```
qnbinom(0.95, size = 20, prob = 0.09)
```

```
[1] 286
```

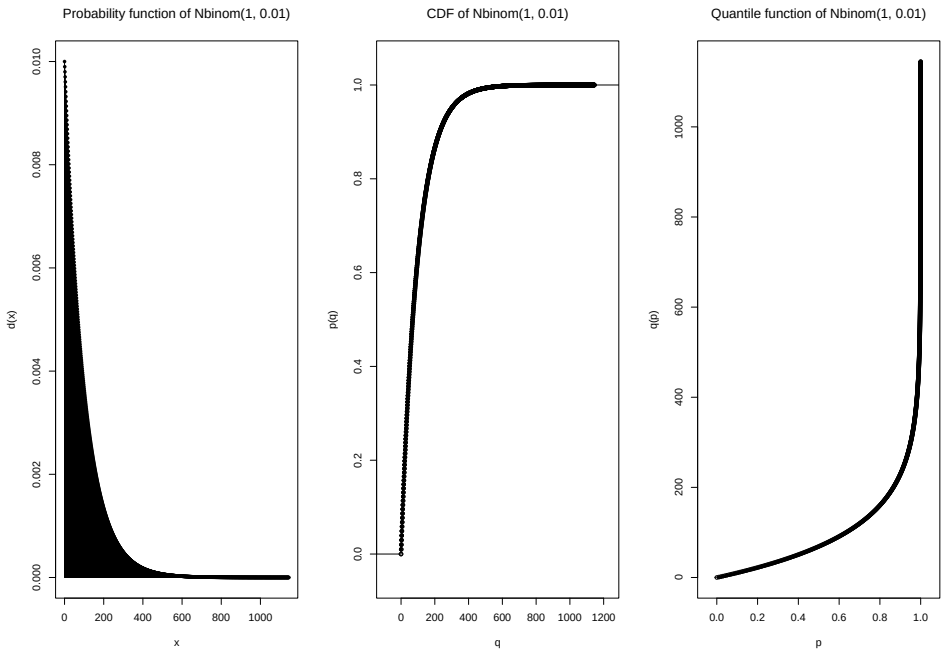
The number represents the number of persons without diabetes, i.e., in total we should randomly draw 306 persons. We simulate 10 diabetes trials.

```
rnbinom(10, size = 20, prob = 0.09)
```

```
 [1] 240 148 151 222 146 227 234 265 303 141
```

Each of the numbers above states how many persons without diabetes had to be drawn to get the required number of 20 persons with diabetes. We visualize the negative binomial distribution of our bulb example by means of package `"distr"` (Ruckdeschel et al. (2006)).

```
X ← Nbinom(size = 1, prob = 0.01)
plot(X, cex.points = 0.75)
```



With the help of argument `cex.points` we reduce the size of the plotted points. We can also use function `summary` of package `"distr6"` (Sonabend and Kiraly (2022)) to summarize the distribution.

```
X6 ← NegativeBinomial$new(size = 1, prob = 0.01)
summary(X6)
```

```
Negative Binomial (fbs) Probability Distribution.
Parameterised with:

      Id              Support Value               Tags
1:  form {fbs, sbf, tbf, tbs}   fbs required,immutable
```

```
2:  mean                    ℝ+           required,linked
3:  prob                 (0,1)  0.01     required,linked
4: qprob                 (0,1)           required,linked
5:  size                    ℕ+      1            required


Quick Statistics
        Mean:           99
        Variance:       9900
        Skewness:       2.000025
        Ex. Kurtosis:   6.000101

Support: ℕ0     Scientific Type: ℕ0

Traits:         discrete; univariate
Properties:     asymmetric; leptokurtic; positive skew
```

Poisson distribution

As last discrete distribution, we introduce the Poisson distribution, which has various applications.

Example 4.12. **(a)** A conventional bulb today has an average (median) lifespan of 1000 hours. Assuming an exponential decrease of the number of functioning bulbs, we obtain

$$50\% = 0.5 = P(\text{Time till failure} > 1000h) = e^{-1000\lambda} \tag{4.25}$$

which leads to a failure rate per hour of about $\lambda = 0.0007$. We assume that we have 20 bulbs in our home that are on for 100 hours per month. Let X be the random variable describing the number of bulbs failing per month. How likely is it, that we have to change at least one bulb per month? We obtain

$$P(X \geq 1) = 1 - P(X = 0) = 1 - e^{-20 \cdot 100 \cdot 0.0007} = 1 - e^{-1.4} = 75.3\% \tag{4.26}$$

(b) The proportion of persons newly falling ill in a certain time period is called **incidence** or **incidence rate**. Finland has worldwide the highest incidence rate of type 1 diabetes for children up to an age of 15 years. On average, every year 55 of 100 000 children in that age newly fall ill with type 1 diabetes (Harjutsalo et al. (2013)), which corresponds to a rate of $\lambda = 0.00055$.
According to Wikipedia contributors (2022c) there live about 900 000 children in that age in Finland; that is, on average we have to expect 495 new cases per year. Let X be the number of new cases per year. How likely is it, that there are more than 450 new cases in one year in Finland? As we will see below, we get

$$P(X > 450) = 1 - P(X \leq 450) = 1 - \sum_{k=0}^{450} \frac{495^k}{k!} e^{-495} = 97.8\% \tag{4.27}$$

We define the Poisson distribution.

Definition 4.13 (Poisson distribution)**.** *A random variable X follows a **Poisson distribution** with parameter $\lambda \in (0, \infty)$, if it has the following probability mass function*

$$P(X = k) = \frac{\lambda^k}{k!} e^{-\lambda} \qquad k \in \mathbb{N}_0$$

abbreviated by: $X \sim Pois(\lambda)$.

We give some additional explanations.

Remark 4.14. ***(a)*** *The parameter λ describes the number of events that we can expect on average in a predefined time period.*

(b) *The Poisson distribution has various applications and can also be used as an approximation of the binomial distribution. The approximation works well if the probability p of the event is small and the sample size n is large. In this case, we may use Pois(np) as approximation for Binom(n, p). Therefore, the Poisson distribution is also called the distribution of rare events.*

(d) *Expectation and variance of Pois(λ) are*

$$E(X) = \lambda \qquad Var(X) = \lambda \tag{4.28}$$

The Poisson distribution is abbreviated by `pois` in R leading to functions `dpois`, `ppois`, `qpois`, and `rpois`. We compute the probabilities of Example 4.12 using R.

```
## a) no defectice bulb
dpois(0, lambda = 1.4)
```

```
[1] 0.246597
```

By means of `ppois` and `lower.tail = FALSE` we obtain

```
## a) at least one defective bulb
ppois(0, lambda = 1.4, lower.tail = FALSE)
```

```
[1] 0.753403
```

```
## b) at least 450 new cases
ppois(450, lambda = 495, lower.tail = FALSE)
```

```
[1] 0.9784977
```

By applying the quantile function, we can determine how many bulbs per month we have to change at most with a high probability (here at least 99%).

```
qpois(0.99, lambda = 1.4)
```

```
[1] 5
```

That is, a stock of five bulbs should suffice for more than one month with a high probability ($\geq$ 99%). The exact probability is

```
ppois(5, lambda = 1.4)
```

```
[1] 0.9967989
```

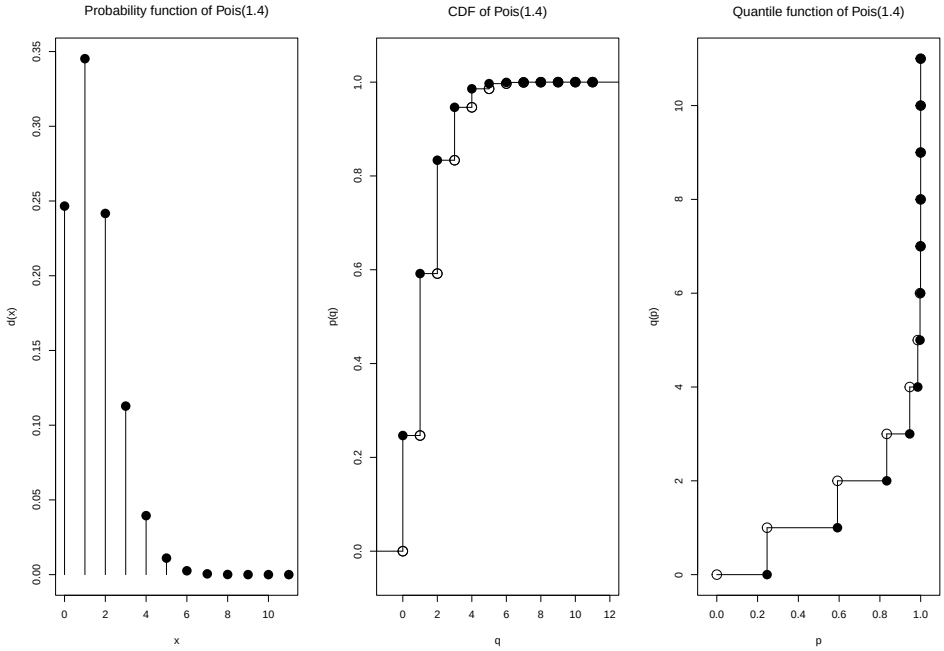
We simulate the number of new cases of type 1 diabetes in Finland for ten years.

```
rpois(10, lambda = 495)
```

```
 [1] 526 488 498 503 519 507 521 547 518 471
```

We visualize the distribution of the bulb example by means of package "distr" (Ruckdeschel et al. (2006)).

```
X ← Pois(lambda = 1.4)
plot(X)
```



Finally, we summarize the distribution by using function summary of package "distr6" (Sonabend and Kiraly (2022)).

```
X6 ← Poisson$new(rate = 1.4)
summary(X6)
```

```
Poisson Probability Distribution.
Parameterised with:

     Id Support Value      Tags
1: rate      ℝ+   1.4 required


Quick Statistics
        Mean:           1.4
        Variance:       1.4
        Skewness:       0.8451543
        Ex. Kurtosis:   0.7142857
```

```
Support: ℕ0      Scientific Type: ℕ0

Traits:          discrete; univariate
Properties:      asymmetric; leptokurtic; positive skew
```

4.2 Continuous Distributions

A random variable X, which may attain all values in an interval $I \subset \mathbb{R}$, is called **continuous random variable**.

Note:
This notion of continuity does not reflect a property of function X, i.e., the random variable X in not necessarily a continuous function. This notion of continuity – more precisely absolute continuity – is derived from the distribution of X. It means that the cumulative distribution function p of X is (almost everywhere) differentiable with derivative $d = p'$ respectively, p is the indefinite integral of d

$$p(x) = \int_{-\infty}^{x} d(t)\, dt \tag{4.29}$$

We may describe the **continuous probability distribution** of random variable X, or **continuous distribution** of X for short, by the so-called **probability density** or **density** d, where

$$d(x) \geq 0 \qquad \text{for (almost) all } x \in \mathbb{R} \text{ and } \qquad \int_{-\infty}^{\infty} d(x)\, dx = 1 \tag{4.30}$$

must hold. The probability $P(X \in [a, b])$ of some interval $[a, b] \in \mathbb{R}$ is given by

$$P(X \in (a, b]) = \int_{a}^{b} d(x)\, dx = p(b) - p(a) \tag{4.31}$$

Thus, the probability is nothing else but the area under the density curve. In particular, it follows

$$P(X = x) = 0 \tag{4.32}$$

i.e., single points possess probability 0. Consequentially, it holds

$$P(X \in (a, b)) = P(X \in (a, b]) = P(X \in [a, b)) = P(X \in [a, b]) \tag{4.33}$$

That is, it does not make any difference, if we consider open, semi-open or closed intervals. Similar to the discrete case, the quantile function in general reads

$$q(p) = \min \{x \in \mathbb{R} \mid p(x) \geq p\} \qquad p \in [0, 1] \tag{4.34}$$

Every cumulative distribution function is monotonically increasing, if it is even strictly monotonically increasing, the quantile function is just the usual inverse function of the cumulative distribution function.

As in case of the computation of probabilities, one has to integrate to determine expectation and variance of continuous random variables. The expectation reads

$$\mathrm{E}(X) = \int_{-\infty}^{\infty} x\, d(x)\, dx \tag{4.35}$$

and the variance is

$$\mathrm{Var}(X) = \int_{-\infty}^{\infty} (x - \mathrm{E}(X))^2\ d(x)\, dx \tag{4.36}$$

Note:
Strictly speaking, there are no continuous random variables in practice, as all measurements that we make can only be done with restricted precision and hence, may at most produce finitely many results. Therefore, continuous random variables can be regarded as an abstract description of reality, in which the restricted precision of our measurements is ignored. Nevertheless, they are very useful and yield sufficiently precise descriptions in many practical applications.

Normal distribution

In the sequel, we will introduce some important continuous distributions. We start with the probably most important continuous distribution in statistics, the normal or Gaussian distribution.

Definition 4.15 (Normal distribution)**.** *A real random variable X follows a **Normal** or **Gaussian distribution** with mean $\mu \in \mathbb{R}$ and standard deviation $\sigma \in (0, \infty)$, if it has the following density*

$$d(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{4.37}$$

It is abbreviated by $X \sim Norm\,(\mu, \sigma^2)$.

We give some additional explanations.

Remark 4.16. ***(a)*** *The central role of the normal distribution follows from the fact that a superposition (sum) of independent factors, under quite weak assumptions can, at least approximately, be described by this distribution. This is a paraphrase of the statement of one of the most important theorems of probability theory, the **central limit theorem**.*

(b) *In presence of a normal distribution, we can make quite precise statements about the probabilities of certain intervals using only its mean and standard deviation. It holds*

$$\begin{aligned} P(X \in [\mu - \sigma, \mu + \sigma]) &= 68.3\% \\ P(X \in [\mu - 2\sigma, \mu + 2\sigma]) &= 95.4\% \\ P(X \in [\mu - 3\sigma, \mu + 3\sigma]) &= 99.7\% \end{aligned} \tag{4.38}$$

This yields the often handy and easy to remember **2σ *rule***: *Within a distance of 2σ around the mean (expectation) about 95% of the values are located. The 2σ rule is relatively robust and approximately holds for quite many distributions.*

(c) *The normal distributions also plays an important role in quality and process control. The name of one of the most famous quality management systems – Six Sigma – comes from the normal distribution. Thus, the goal of this system is an extremely low failure probability.*

(d) *As the names of the parameters already indicate, the expectation and variance of $Norm(\mu, \sigma^2)$ are*

$$E(X) = \mu \qquad Var(X) = \sigma^2 \tag{4.39}$$

(e) *If $X \sim Norm(\mu, \sigma^2)$, it holds*

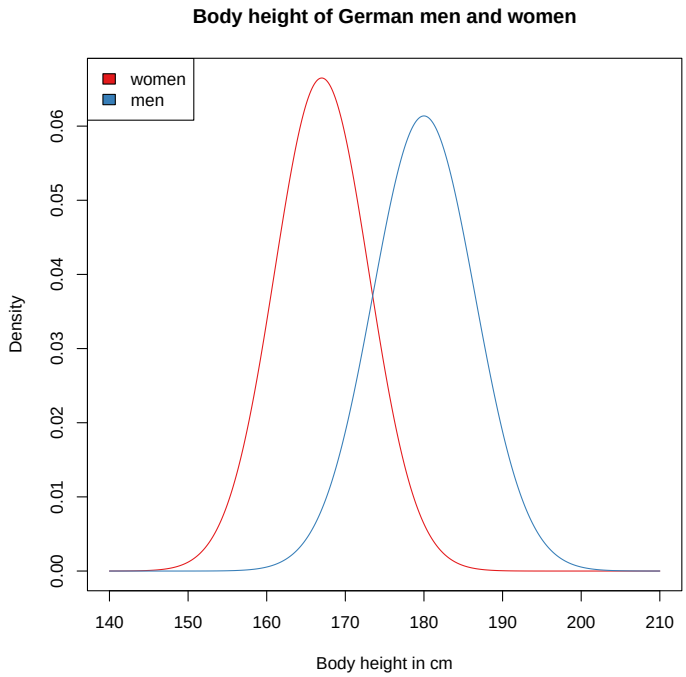
$$Z = \frac{X - \mu}{\sigma} \sim Norm(0, 1) \tag{4.40}$$

and one also calls $Norm(0, 1)$ the ***standard normal distribution***. *We selected the letter Z intendendly, since this distribution is sometimes also called Z-distribution and correspondingly we also refer to it as z-transformation or z-score (cf. Remark 2.23)).*

The normal distribution is abbreviated by `norm` in R leading to the functions `dnorm` (density), `pnorm`, `qnorm`, and `rnorm`. The names of the parameters are `mean` and `sd`. In the following example, we present two applications of the normal distribution.

Example 4.17. **(a)** The body height of adults in a country can be well described by normal distributions. In case of the women in Germany, we get a mean of about 167 cm and a standard deviation of about 6.0 cm. In case of the men in Germany, the mean is about 180 cm and the standard deviation about 6.5 cm (Wikipedia (2015)). We plot the density function of men and women using function `curve`.

```
curve(expr = dnorm(x, mean = 167, sd = 6.0), from = 140, to = 210, n = 501,
      col = "#E41A1C", xlab = "Body height in cm", ylab = "Density",
      main = "Body height of German men and women")
curve(expr = dnorm(x, mean = 180, sd = 6.5), from = 140, to = 210, n = 501,
      add = TRUE, col = "#377EB8")
legend("topleft", legend = c("women", "men"), fill = c("#E41A1C", "#377EB8"))
```



The argument `expr` is the R expression that shall be plotted. With `from` and `to` one can specify the range of the x axis where expression `expr` is evaluated and drawn on a grid of `n` equidistant points. Finally, by using `add = TRUE` we can add further curves to an already existing plot. The proportion of women larger than 175 cm accordingly is

```
pnorm(175, mean = 167, sd = 6.0, lower.tail = FALSE)
```

```
[1] 0.09121122
```

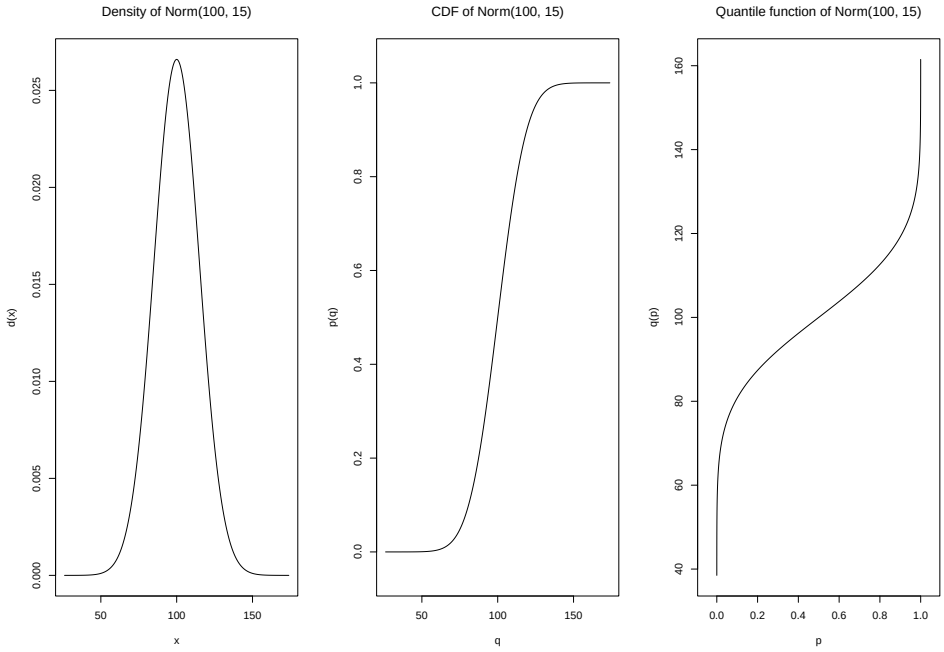
The tallest 5% of men are taller than

```
qnorm(0.95, mean = 180, sd = 6.5)
```

```
[1] 190.6915
```

(b) The intelligence quotient (IQ) can also very well be described by a normal distribution. The IQ scales have a mean of 100 and a standard deviation of 15 (Wikipedia contributors (2022d)). We plot the respective normal distribution by means of package `"distr"` (Ruckdeschel et al. (2006)).

```
X ← Norm(mean = 100, sd = 15)
plot(X)
```



Thus, the 2σ rule states that about 5% of the population have an IQ score smaller than 70 or larger than 130. Reversely, about 2.5% of the population have an IQ smaller than 70 or larger than 130, respectively. Finally, we summarize the distribution using function `summary` of package `"distr6"` (Sonabend and Kiraly (2022)).

```
X6 ← Normal$new(mean = 100, sd = 15)
summary(X6)
```

```
Normal Probability Distribution.
Parameterised with:

     Id Support Value            Tags
1: mean       ℝ   100        required
2: prec      ℝ+        linked,required
3:   sd      ℝ+    15 linked,required
4:  var      ℝ+        linked,required


Quick Statistics
        Mean:           100
        Variance:       225
        Skewness:       0
        Ex. Kurtosis:   0

Support: ℝ      Scientific Type: ℝ

Traits:         continuous; univariate
Properties:     symmetric; mesokurtic; no skew
```

Log-normal distribution

The second important continuous distribution is closely related with the normal distribution and is called log-normal distribution. We first give the definition.

Definition 4.18 (Log-normal distribution)**.** *A real random variable X attaining only positive values follows a **log-normal distribution** with mean $\mu \in \mathbb{R}$ and standard deviation $\sigma \in (0, \infty)$, if it has the following density*

$$d(x) = \begin{cases} \frac{1}{\sqrt{2\pi}\sigma x} e^{-\frac{1}{2}\left(\frac{\log(x)-\mu}{\sigma}\right)^2} & \text{if } x > 0 \\ 0 & \text{else} \end{cases} \tag{4.41}$$

It is abbreviated by $X \sim Lnorm(\mu, \sigma)$.

We give some additional explanations.

Remark 4.19. ***(a)*** *The log-normal distribution occurs in many scientific disciplines. In particular, many biological processes happen on an exponential scale and thus many parameters in biology and medicine can be described by a log-normal distribution. That is, in a similar way as additive superpositions in the sense of the central limit theorem lead to a normal distribution, multiplicative superpositions lead to a log-normal distribution.*

(b) *If X is log-normal distributed, $\log(X)$ is normal distributed. In view of part (a) we can say, that a multiplicative superposition by applying the logarithm becomes an additive superposition.*

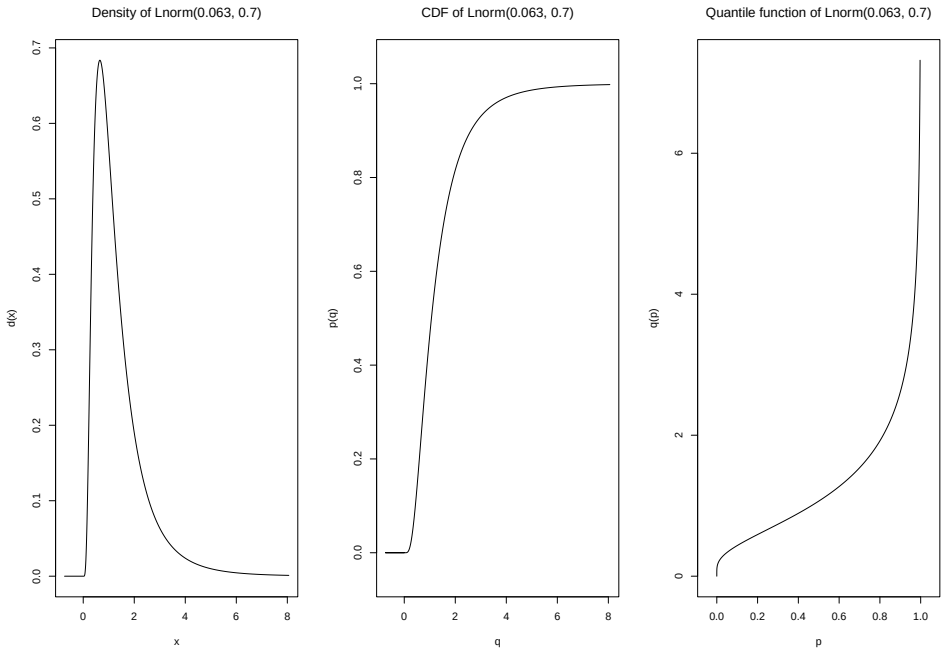
(c) *The parameters of the log-normal distribution are nothing else but the expectation and the variance of $\log(X)$. For the random variable X itself we get*

$$E(X) = e^{\mu + \frac{\sigma^2}{2}} \qquad Var(X) = e^{2\mu+\sigma^2}(e^{\sigma^2} - 1) \tag{4.42}$$

The log-normal distribution is abbreviated by `lnorm` in R. Accordingly, we obtain functions `dlnorm`, `plnorm`, `qlnorm`, and `rlnorm`, where the names of the parameters are `meanlog` and `sdlog`. We give an examples for an application of the log-normal distribution.

Example 4.20. **(a)** For examining the thyroid function the concentration of thyrotropin (TSH) in the blood is analyzed. Its concentration in persons with normal thyroid function can be described by a log-normal distribution (Hamilton et al. (2008)). The declarations of the normal range vary especially with regard to the upper bound. In this example, we use a normal range of 0.27-4.2 $\mu lU/ml$ for adults (Hagemann (2014)). By using the connection between log-normal and normal distribution, we can determine the distribution of TSH in persons with normal thyroid function. In addition, we use the information that the **normal or reference range** of a parameter is always chosen such that 2.5% of the healthy persons may have lower or higher values, respectively (Wikipedia contributors (2022f)). In case of the normal distribution, the normal range approximately corresponds to the 2σ interval.
After log-transforming, the normal range reads [-1.309,1.435]. Since the normal distribution is symmetric, the expectation must be the middle of this interval, i.e., $\mu = 0.063$. The length of the interval roughly is 4σ, more precisely it is 3.92σ. Starting with the interval length of 2.744, the division by 3.92 leads to $\sigma = 0.7$. Therefore, the distribution of log-TSH is $\text{Norm}(0.063, 0.7^2)$, thus TSH is $\text{Lnorm}(0.063, 0.7)$ distributed. We plot the distribution of TSH by means of package `"distr"` (Ruckdeschel et al. (2006)).

```
X ← Lnorm(meanlog = 0.063, sdlog = 0.7)
plot(X)
```



Finally, we summarize the distribution by using function `summary` of package `"distr6"` (Sonabend and Kiraly (2022)).

```
X6 ← Lognormal$new(meanlog = 0.063, sdlog = 0.7)
summary(X6)
```

```
Lognormal Probability Distribution.
Parameterised with:

        Id Support Value           Tags
1:    mean      ℝ+       required,means
2: meanlog       ℝ 0.063 required,means
3:    prec      ℝ+         required,vars
4: preclog      ℝ+         required,vars
5:      sd      ℝ+         required,vars
6:   sdlog      ℝ+   0.7   required,vars
7:     var      ℝ+         required,vars
8:  varlog      ℝ+         required,vars


Quick Statistics
        Mean:           1.360701
        Variance:       1.170738
        Skewness:       2.888357
        Ex. Kurtosis:   17.79117

Support: ℝ+     Scientific Type: ℝ+
```

```
Traits:          continuous; univariate
Properties:      asymmetric; leptokurtic; positive skew
```

(b) Several examples of applications of the log-normal distribution from various scientific disciplines are collected in Limpert et al. (2001) and Limpert and Stahel (2011). In particular, both articles give recommendations for handling log-normal distributed data in practice.

Gamma distribution

A very flexible distribution with many application is the so-called gamma distribution.

Definition 4.21 (Gamma distribution)**.** *A real random variable X attaining only positive values follows a **gamma distribution** with scale parameter $\sigma \in (0, \infty)$ and shape parameter $\alpha \in (0, \infty)$, if it has the following density*

$$d(x) = \begin{cases} \frac{1}{\sigma^\alpha \Gamma(\alpha)} x^{\alpha-1} e^{-\frac{x}{\sigma}} & \text{if } x > 0 \\ 0 & \text{else} \end{cases} \tag{4.43}$$

*where the **gamma function** Γ is*

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t}\, dt \tag{4.44}$$

It is abbreviated by $X \sim Gamma(\sigma, \alpha)$.

We give some additional explanations.

Remark 4.22. ***(a)*** *The shape parameter makes the gamma distribution very flexible, thus it has many applications for instance in insurance mathematics, genetics or also medicine.*

(b) *An important special case of the gamma distribution is the **exponential distribution**, which is obtained for $\alpha = 1$. In addition, one usually uses the rate $\lambda = \frac{1}{\sigma}$ as parameter leading to the following density*

$$d(x) = \begin{cases} \lambda e^{-\lambda x} & \text{if } x > 0 \\ 0 & \text{else} \end{cases} \tag{4.45}$$

It is abbreviated by $X \sim Exp(\lambda)$. One can consider it as the continuous counterpart of the geometric distribution, a special case of the negative binomial distribution. It describes the time between two events of a process, where the events occur continuously and independently from each other at a fixed rate. It is for instance used to estimate survival probabilities.

(c) *If we simultaneously consider $k \in \mathbb{N}$ independent processes, whose events follow $Exp(\lambda)$, their sum follows a so-called **Erlang distribution**. The Erlang distribution itself is a special case of the gamma distribution, where it holds $\alpha = k$ and one usually uses the rate $\lambda = \frac{1}{\sigma}$ as second parameter as in case of the exponential distribution. Hence, the density reads*

$$d(x) = \begin{cases} \frac{\lambda^k}{(k-1)!} x^{k-1} e^{-\lambda x} & \text{if } x > 0 \\ 0 & \text{else} \end{cases} \tag{4.46}$$

The Erlang distribution can for example be used to model the time between calls in a call center, where the number of calls may for instance be described by a Poisson distribution. The Erlang distribution is therefore a waiting time distribution and can be regarded as a continuous variant of the negative binomial distribution.

(d) *Another important special case of the gamma distribution is the* χ^2 ***distribution*** *with* $n \in \mathbb{N}$ *degrees of freedom, Chisq* (n) *for short. It holds* $\sigma = 2$ *and* $\alpha = \frac{n}{2}$*. Thus, the density reads*

$$d(x) = \begin{cases} \frac{1}{2^{\frac{n}{2}}\Gamma(\frac{n}{2})} x^{\frac{n}{2}-1} e^{-\frac{1}{2}x} & \text{if } x > 0 \\ 0 & \text{sonst} \end{cases} \tag{4.47}$$

The χ^2 *distribution also arises in the framework of the normal distribution as we will see later in this section.*

(e) *Expectation and variance of* $X \sim Gamma\,(\sigma, \alpha)$ *are*

$$E(X) = \alpha\sigma \qquad Var(X) = \alpha\sigma^2 \tag{4.48}$$

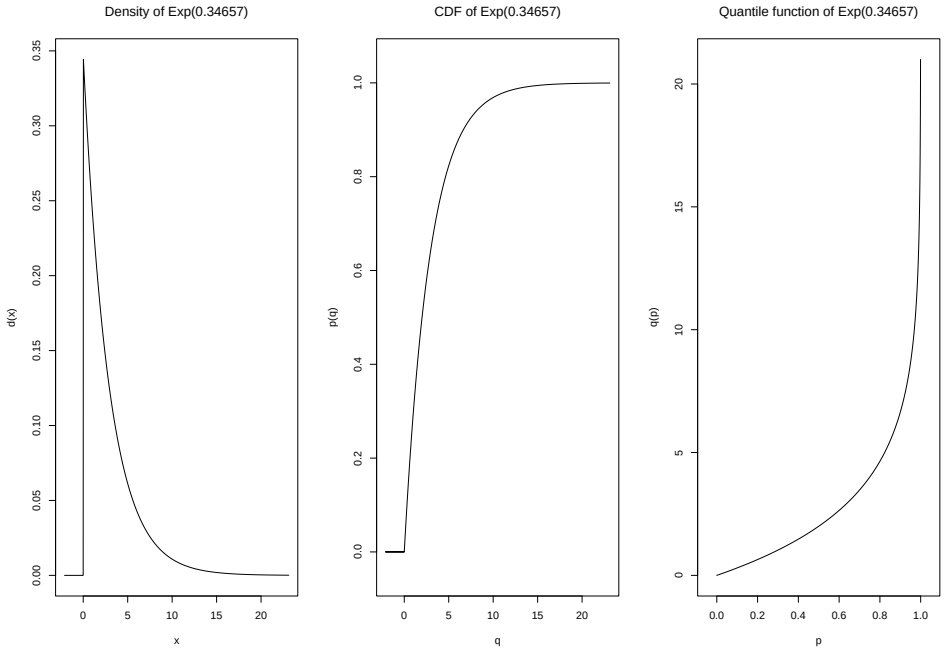
We introduce some applications of the gamma distribution. The gamma distribution is available in R in form of the functions `dgamma`, `pgamma`, `qgamma`, and `rgamma`, where the parameters are called `scale` and `shape`. The exponential distribution is provided by functions `dexp`, `pexp`, `qexp`, and `rexp` with parameter `rate`.

Example 4.23. (a) A modern battery of a smart phone has a median life expectancy of two years. We use the exponential distribution to model the life expectancy, which yields

$$0.5 = P(X \leq 5\text{years}) = 1 - e^{-2\text{years}\cdot\lambda} \tag{4.49}$$

This leads to a failure rate per year of $\lambda = 0.34657$. We plot the distribution by means of package `"distr"` (Ruckdeschel et al. (2006)).

```
1 X ← Exp(rate = 0.34657)
2 plot(X)
```



We summarize the distribution by using the function `summary` of the package `"distr6"` (Sonabend and Kiraly (2022)).

```
X6 ← Exponential$new(rate = 0.34657)
summary(X6)
```

```
Exponential Probability Distribution.
Parameterised with:

      Id Support   Value            Tags
1:   rate      ℝ+ 0.34657 linked,required
2:  scale      ℝ+         linked,required


Quick Statistics
        Mean:           2.88542
        Variance:       8.325648
        Skewness:       2
        Ex. Kurtosis:   6

Support: ℝ0+    Scientific Type: ℝ0+

Traits:         continuous; univariate
Properties:     asymmetric; leptokurtic; positive skew
```

Thus, how likely is it that the battery fails already in the first year?

```
pexp(1, rate = 0.34657)
```

```
[1] 0.2928907
```

That is, more than 30% of the batteries fail already in the first year. After how many years are 99% of the batteries out of order? We obtain

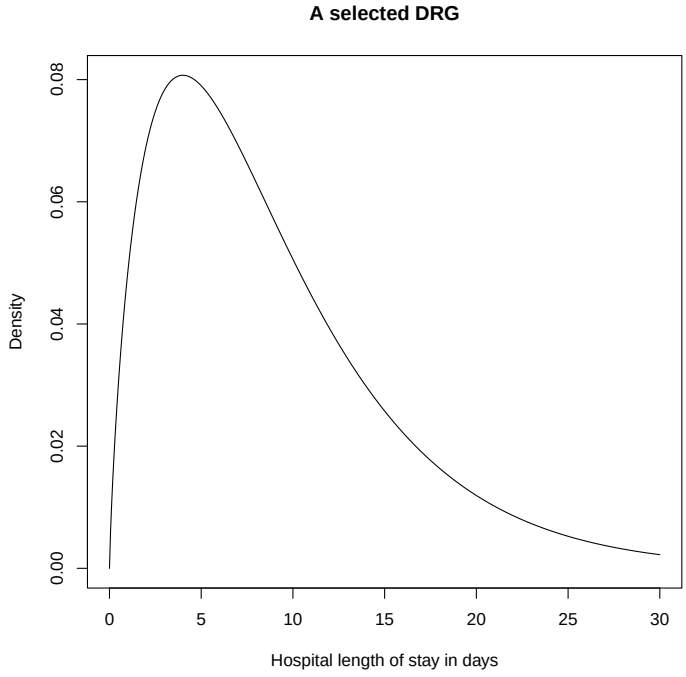
```
qexp(0.99, rate = 0.34657)
```

```
[1] 13.28785
```

That is, in the extreme case a battery may theoretically work for more than 13 years.

(b) The gamma distribution offers a way to model the hospital length of stay of a group of patients; e.g., all patients with a certain diagnosis or more precisely belonging to a certain DRG (diagnosis related group). Assuming a scale parameter of $\sigma = 5$ and a shape parameter of $\alpha = 1.8$ for a selected DRG, we obtain the following density, which we plot using function `curve`

```
curve(dgamma(x, scale = 5, shape = 1.8), from = 0, to = 30, n = 501,
      xlab = "Hospital length of stay in days", ylab = "Density",
      main = "A selected DRG")
```



We summarize the distribution by using function `summary` of package `"distr6"` (Sonabend and Kiraly (2022)).

```
X6 ← Gamma$new(shape = 1.8, scale = 5)
summary(X6)
```

```
Gamma Probability Distribution.
Parameterised with:

      Id Support Value             Tags
```

```
1:  mean      ℝ+       linked,required
2:  rate      ℝ+       linked,required
3: scale      ℝ+     5 linked,required
4: shape      ℝ+   1.8        required


Quick Statistics
        Mean:           9
        Variance:       45
        Skewness:       1.490712
        Ex. Kurtosis:   3.333333

Support: ℝ+     Scientific Type: ℝ+

Traits:         continuous; univariate
Properties:     asymmetric; leptokurtic; positive skew
```

That is, most of the patients of this DRG will be discharged within a few days. But, it may happen that patients have to stay in the hospital for more than two weeks. How likely is it, that a randomly selected patient has to stay in the hospital for more than ten days? We get

```
pgamma(10, scale = 5, shape = 1.8, lower.tail = FALSE)
```

```
[1] 0.3472818
```

Thus, slightly more than one third of the patients have to stay for more than ten days. After how many days 99% of the patients have been discharged? We obtain

```
qgamma(0.99, scale = 5, shape = 1.8)
```

```
[1] 31.3043
```

That is, it happens only very rarely that a patient has to stay for more than one month.

Weibull distribution

If the failure rate changes over time, the so-called Weibull distribution offers a way to model the process. We start with its definition.

Definition 4.24 (Weibull distribution)**.** *A real random variable X attaining only positive values follows a **Weibull distribution** with scale parameter $\sigma \in (0, \infty)$ and shape parameter $\alpha \in (0, \infty)$, if it has the following density*

$$d(x) = \begin{cases} \frac{\alpha}{\sigma}\left(\frac{x}{\sigma}\right)^{\alpha-1} e^{-\left(\frac{x}{\sigma}\right)^{\alpha}} & \text{if } x > 0 \\ 0 & \text{else} \end{cases} \tag{4.50}$$

It is abbreviated by $X \sim Weibull(\sigma, \alpha)$

We give some additional explanations.

Remark 4.25. ***(a)*** *The Weibull distribution plays an important role in the reliability analysis of parts and components for instance in the automobile industry. In contrast to the exponential distribution, the shape parameter offers a possibility to model also aging.*

(b) *The Weibull distribution belongs to the class of* ***extreme value distributions****, more precisely it is an extreme value distribution of Typ III. By the theorem of Fisher–Tippett–Gnedenko, this distribution, under certain assumptions, arises as the maximum of independent random variables.*

(c) *In case $\alpha = 1$, the Weibull distribution is identical to the exponential distribution.*

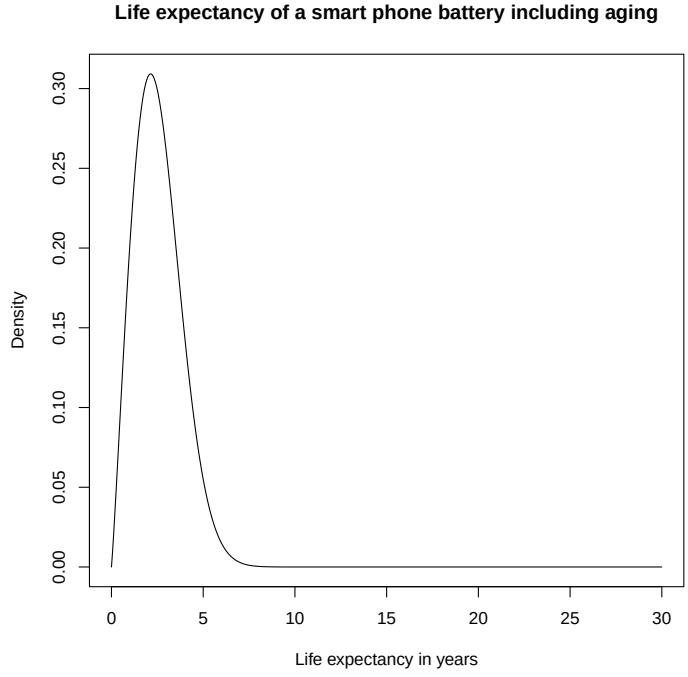
(d) *Expectation and variance are*

$$E(X) = \sigma\Gamma\left(1 + \frac{1}{\alpha}\right) \qquad Var(X) = \sigma^2\left(\Gamma\left(1 + \frac{2}{\alpha}\right) - \Gamma\left(1 + \frac{1}{\alpha}\right)^2\right) \tag{4.51}$$

We introduce some applications.

Example 4.26. **(a)** We again consider the battery of a modern smart phone with a failure rate of 0.34657 as in Example 4.23 (a), i.e. $\sigma = \frac{1}{0.34657}$. In addition, we assume that its aging can be described by a shape parameter of $\alpha = 2.12$. We plot the distribution using function `curve`

```
curve(dweibull(x, scale = 1/0.34657, shape = 2.12), from = 0, to = 30, n = 501,
      xlab = "Life expectancy in years", ylab = "Density",
      main = "Life expectancy of a smart phone battery including aging")
```



We summarize the distribution by using the function `summary` of package `"distr6"` (Sonabend and Kiraly (2022)).

```
X6 ← distr6::Weibull$new(shape = 2.12, scale = 1/0.34657)
summary(X6)
```

```
Weibull Probability Distribution.
Parameterised with:

         Id Support   Value            Tags
1: altscale      ℝ+         linked,required
2:    scale      ℝ+ 2.88542 linked,required
3:    shape      ℝ+    2.12         required


Quick Statistics
        Mean:           2.555462
        Variance:       1.606889
        Skewness:       0.5551175
        Ex. Kurtosis:   0.1124261

Support: ℝ0+    Scientific Type: ℝ0+

Traits:         continuous; univariate
Properties:     asymmetric; leptokurtic; positive skew
```

Since there is also a function in the package "distr" called Weibull, we have to specify distr6:: to distinguish it. More precisely, the operator :: is used to access the objects of the **Namespace** of a package, here "distr6".
There are less defective batteries in the first year than in case of the exponential distribution, namely

```
pweibull(1, scale = 1/0.34657, shape = 2.12)
```

```
[1] 0.1003672
```

only around 10%. However, it takes less time until 99% of the batteries are out of order i.e., only about

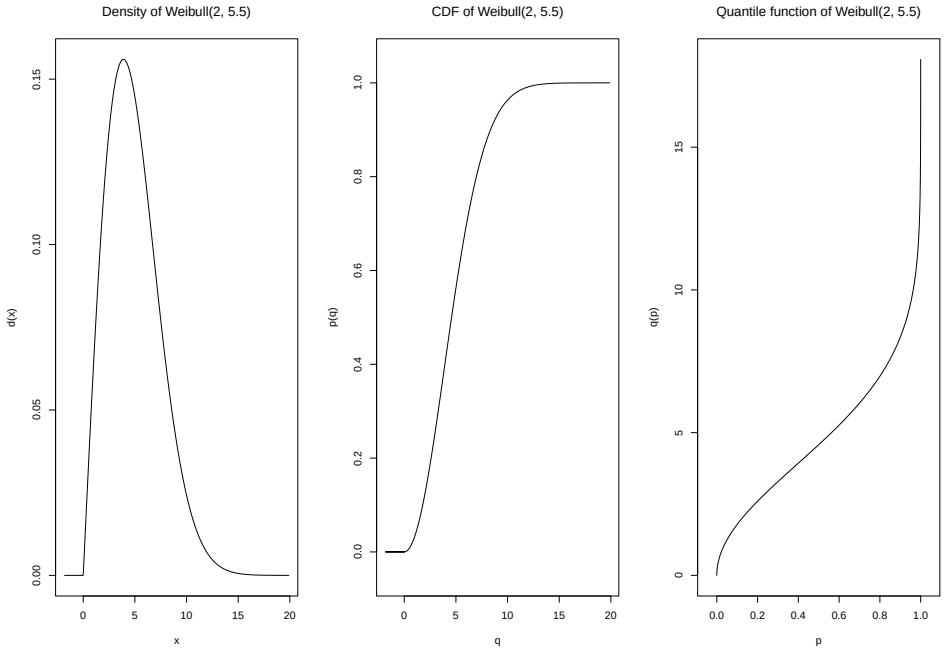
```
qweibull(0.99, scale = 1/0.34657, shape = 2.12)
```

```
[1] 5.930083
```

years.

(b) The Weibull distribution is also used to model wind speed. We assume that the maximum wind speeds (in $\frac{m}{s}$) per day at a selected place may be described by a Weibull distribution with $\sigma = 5.5$ and $\alpha = 2$. We plot the distribution by means of package "distr" (Ruckdeschel et al. (2006)).

```
X ← Weibull(scale = 5.5, shape = 2)
plot(X)
```



We summarize the distribution by using function `summary` of package `"distr6"` (Sonabend and Kiraly (2022)).

```
X6 ← distr6::Weibull$new(shape = 2, scale = 5.5)
summary(X6)
```

```
Weibull Probability Distribution.
Parameterised with:

         Id Support Value            Tags
1: altscale      ℝ+       linked,required
2:     scale      ℝ+   5.5 linked,required
3:     shape      ℝ+     2        required


Quick Statistics
        Mean:           4.874248
        Variance:       6.491706
        Skewness:       0.6311107
        Ex. Kurtosis:   0.2450893

Support: ℝ0+    Scientific Type: ℝ0+

Traits:         continuous; univariate
Properties:     asymmetric; leptokurtic; positive skew
```

We get as median wind speed

```
qweibull(0.5, scale = 5.5, shape = 2)
```

```
[1] 4.57905
```

which is a gentle breeze. How likely is at least a strong breeze, i.e., a wind speed of at least $11\frac{m}{s}$? We obtain

```
1 pweibull(11, scale = 5.5, shape = 2, lower.tail = FALSE)
```

```
[1] 0.01831564
```

That is, it happens only in about 2% of the days.

χ^2, t and F distribution

Finally, we introduce some continuous distributions that arise in the context of the normal distribution and play an important role in inferential statistics. We first give the definitions.

Definition 4.27 (χ^2, t and F distribution)**.** ***(a)*** *A real random variable X attaining only positive values follows a χ^2 **distribution** with $n \in \mathbb{N}$ degrees of freedom, if it has the following density*

$$d(x) = \begin{cases} \frac{1}{2^{\frac{n}{2}}\Gamma(\frac{n}{2})} e^{-\frac{1}{2}x} x^{(\frac{n}{2}-1)} & \text{if } x > 0 \\ 0 & \text{else} \end{cases} \tag{4.52}$$

It is abbreviated by $X \sim Chisq(n)$.

(b) *A real random variable X follows a **t distribution** with $n \in \mathbb{N}$ degrees of freedom, if it has the following density*

$$d(x) = \frac{\Gamma(\frac{n+1}{2})}{\Gamma(\frac{n}{2})\sqrt{\pi n}} \left(1 + \frac{t^2}{n}\right)^{-\frac{n+1}{2}} \tag{4.53}$$

It is abbreviated by $X \sim t(n)$

(c) *A real random variable X attaining only positive values follows an **F distribution** with $m \in \mathbb{N}$ and $n \in \mathbb{N}$ degrees of freedom, if it has the following density*

$$d(x) = \begin{cases} \frac{\Gamma(\frac{n+m}{2})}{\Gamma(\frac{n}{2})\Gamma(\frac{m}{2})} n^{\frac{n}{2}} m^{\frac{m}{2}} \frac{x^{\frac{n}{2}-1}}{(m+nx)^{\frac{n+m}{2}}} & \text{if } x > 0 \\ 0 & \text{else} \end{cases} \tag{4.54}$$

It is abbreviated by $X \sim F(m, n)$.

We give some additional explanations.

Remark 4.28. ***(a)*** *The χ^2 distributions arises as the sum of the square of n independent standard normal random variables. In inferential statistics, the distribution for instance occurs in connection with estimating the variance.*

(b) *Let Z be some standard normal random variable and Y an independent Chisq(n) distributed random variable. Then, it holds*

$$\sqrt{n}\frac{Z}{\sqrt{Y}} \sim t(n) \tag{4.55}$$

The distribution arises in inferential statistics for example by considering standardized arithmetic means.

(c) *Let $X \sim Chisq(m)$ and $Y \sim Chisq(n)$ be some independent random variables. Then, it holds*

$$\frac{n \cdot X}{m \cdot Y} \sim F(m, n) \tag{4.56}$$

The distribution arises in inferential statistics for instance by investigating the ratio of two variances.

Note:
Of course, there are many more probability distributions, which can be used as models for various applications. In particular, these basic distributions can be applied for constructing more complex models such as regression models.

Table 4.1 includes important notions from statistics and their counterparts in probability theory. There

Statistics	**Probability theory**
attribute/variable	random variable
levels	possible values of a random variable
relative frequency	probability
frequency distribution	probability mass function
density estimation	(probability) density
empirical cumulative distribution function	cumulative distribution function
(sample) quantile	quantile
arithmetic mean	expectation
(sample) variance	variance

Table 4.1: Notions from statistics and their counterparts in probability theory.

are also counterparts to (sample) correlation and covariance in probability theory. For their definition one has to consider the common distribution of two random variables, which goes beyond this introductory book.

4.3 Exercises

Please always describe and explain your results in detail.

1. Plot the distribution Binom $(20, p)$ for $p \in \{0.1, 0.2, \ldots, 0.9\}$ by means of package `"distr"` (Ruckdeschel et al. (2006)).

2. Determine expectation and variance of Binom $(2, p)$ without using the explicit formulas for expectation and variance.

3. People with blood group 0-negative are universal donors, where about 7% of the humans have this blood type. Let us assume you conduct a trial, in which 20 persons are randomly selected. How likely is it that there are at least three universal donors in the sample? Use the binomial distribution.

4. The lethality – i.e. the proportion of ill people who die from the disease – of Sars-CoV-2 (COVID-19) in Korea through March 20, 2020, was approximately. 1.09% (Shim et al. (2020)). Assume that this lethality rate also applies to the approx. 83.2 million people in Germany (2019). Assume further that 70% of the inhabitants in Germany must contract the virus before herd immunity develops. How many deaths are then to be expected in Germany (expected value)? How many deaths are then to be expected in Germany with a probability of at least 99% (quantile)? Use the binomial distribution.

5. In a certain hospital the mean birth rate (expected value) is 1.8 births per hour. How many delivery rooms does the hospital need, such that each birth is in a delivery room with 95% probability? Use the Poisson distribution.

6. In the European Union, a rare disease is defined as one with an incidence lower than 1 in 2000 people. The incidence rate per year of sarcoidosis is about 5-60 cases per 100 thousand population. How many new cases can be expected in Germany per year if we assume the worst case (60 cases per 100 thousand) and a population of 83.2 million in Germany (2019)? What is the maximum number of new cases to be expected in Germany per year with a probability of at least 99%. Use the Poisson distribution.

7. An oil company conducts a geological study in a certain region where it is drilled for oil at randomly selected positions. Let us assume that the probability of finding oil in the selected region is 20%. How likely is it that the company has to drill at least five times until the first oil find? How often the company has to drill to find oil twice with 99% certainty? Apply the negative binomial distribution.

8. The probability of getting all the numbers right in the Eurojackpot, is

$$\frac{1}{\binom{50}{5} \times \binom{10}{2}} \approx 1.049 \times 10^{-8}$$

   How many times do you have to play one field every Friday until you have at least a 50% or 99% probability of getting all the numbers right for the first time? What would this cost you if one field currently cost 2.20 euros? Use negative binomial distribution.

9. Plot the distribution Gamma $(1, \alpha)$ for $\alpha \in \{0.1, 0.5, 1.0, 2.0, 5.0, 10.0\}$ by means of package `"distr"` (Ruckdeschel et al. (2006)). The function to generate gamma distributed random variables is called `Gammad`.

10. Determine expectation and variance of Exp (1) without using the explicit formulas for expectation and variance.

11. The expected birth weight of healthy boys is $\mu = 3.35$ kg with a standard deviation of $\sigma = 0.43$ kg. How likely is it that a healthy boy with a birth weight of less than 3 kg is born? What is the normal range of the birth weight of boys? Apply the normal distribution.

12. We assume that the duration in days between the onset of symptoms in the carrier and the infected in the case of Sars-CoV-2 (COVID-19) can be described by log-normal distribution with $\mu = 2.78$

and $\sigma = 2.02$. Accordingly, how long does it take, on median, for the pairs (carrier and infected) to show symptoms?

13. You want to investigate the impact of gamma rays and conduct an animal experiment with mice. The mice are exposed to a radiation of 2.4 Gray. The survival time of the mice in weeks can be described by a gamma distribution with parameters $\sigma = 7$ and $\alpha = 6$. How likely is it that a randomly chosen mouse lives between 25 and 50 weeks? How many weeks does it take until 95% of the mice have died?

14. The median life time of a common bulb today is 1000 hours. We assume that we can describe the life time by a Weibull distribution with $\sigma = 1250$ and $\alpha = 1.8$. How likely is it that a bulb is defective already in the first 100 hours? After how many hours are 99% of the bulbs defective?

15. The maximum water level [in cm] of the river Neckar in Rottweil (Baden-Württemberg) can be described by a Weibull distribution with shape paramether $\alpha = 4.5$ and scale parameter $\sigma = 270$. Calculate the values for 2-year, 10-year, 50-year and 100-year floods. The values correspond to the 50%, 90%, 98%, and 99% quantiles of the distribution.

5 Estimation

The chapter is about estimating parameters of simple parametric models. It covers the following topics:

- Issues of inferential statistics
- Importance of estimation in the framework of inferential statistics
- Parametric probability models
- Point estimator, estimator
- Unbiasedness, efficiency, consistency
- Maximum likelihood estimator (abbreviated: ML estimator)
- Quantile-quantile plot (abbreviated: qq plot)
- Minimum distance estimator (abbreviated: MD estimator)
- Kolmogorov(-Smirnov)-MD estimator (abbreviated: KS-MD estimator)
- Cramér-von-Mises-MD estimator (abbreviated: CvM-MD estimator)
- Asymptotically linear estimator (abbreviated: AL estimator)
- Radius-minimax estimator (abbreviated: RMX estimator)
- Interval estimator, confidence interval
- Confidence interval for arithmetic mean and standard deviation
- Exact confidence intervals, asymptotic confidence intervals
- Bootstrap confidence intervals
- Confidence intervals for ML estimators
- Continuity correction, finite-sample correction
- Confidence intervals for median and MAD
- Confidence intervals for CvM-MD estimators
- Confidence intervals for MD estimators
- Confidence intervals for AL and RMX estimators

The R code of this chapter is included in the R Markdown file `Estimation.Rmd`, which you can download from my GitHub account (link: https://github.com/stamats/ISDR/blob/main/Estimation.Rmd). Right click on `Raw`. Then you can `Save target as ...`. The least difficulties arise, if you save my R Markdown files in the same folder as the data.
We first install the packages required in this chapter. Since the package `"RobLox"` (Kohl (2019)) depends on the Bioconductor package `"Biobase"` (Huber et al. (2015)), we first install `"Biobase"`. We also need the package `"BiocManager"` (Morgan (2019)) for this purpose.

```
install.packages("BiocManager")
BiocManager::install("Biobase", update = FALSE)
```

Now we can install the required packages.

```
install.packages(c("distrMod", "qqplotr", "RobLox", "gridExtra", "MKinfer",
                   "ROptEst", "RobExtremes", "MKpower", "MKclass"))
```

Furthermore, we install package `"rmx"` (Kohl (2022c)) from GitHub which is not yet on CRAN.

```
## Development version from GitHub
# install.packages("remotes")
remotes::install_github("stamats/rmx", build_vignettes = TRUE)
```

Make sure that you have already installed the packages from the previous Chapters 2, 3 and 4.
We load all packages required for the chapter.

```
library(ggplot2)
library(MKdescr)
library(distrMod)
library(qqplotr)
library(RobLox)
library(gridExtra)
library(MKinfer)
library(ROptEst)
library(RobExtremes)
library(MKpower)
library(MKclass)
library(rmx)
library(MASS)
library(boot)
library(parallel)
```

As already explained in Section 2.4, repeated execution of `library` is not problematic. The packages `"MASS"` (Venables and Ripley (2002)), `"boot"` (Canty and Ripley (2021)) and `"parallel"` (R Core Team (2022a)) do not need to be installed, since they belong to the group of recommended and base packages and are therefore already included in the standard installation of R (cf. Section 1.2).

5.1 Introduction

This introduction provides a brief example to make clear, which questions we can address applying inferential statistics.

We consider a coin and for simplicity label the sides with 0 and 1, where we exclude the possibility that after tossing the coin might land on its edge. We are interested in the question:

Is it a fair coin?

Here, fair means that both sides of the coin occur with equal probability. We can describe the coin toss using the Bernoulli distribution Bernoulli (p), where p is the probability that side 1 is tossed. By means of this probability model, we can state the question more precisely and obtain:

Is the probability of side 1 equal to 50%, abbreviated: $p = 0.5$?

How can we address this question? We could test the coin by a very detailed materials analysis. However, this surely would be very costly and only possible with an appropriate technical equipment. Certainly, a random experiment is faster and simpler: we toss the coin several times and record the results. The results of this random experiment are our sample, which is the basis for our decision by means of statistical procedures.

Before we conduct this random experiment, there are some things to clarify:

I: How often should we toss the coin to get a most reliable result?

II: How do we summarize the results such that we may infer the actual probability p of side 1 in a most optimal way?

III: Is the observed count of side 1 in the range of the expected frequency of a fair coin or is it too small or too large?

The answers of inferential statistics to these questions are:

Ad I: We can perform a so-called sample size calculation using a confidence interval (see Section 5.3) or statistical test (see Chapter 6). With these procedures, we can determine the number of replications in such a way that we can decide, if the coin is fair with a given certainty.

Ad II: By means of point estimators (see Section 5.2) we can summarize the observed values. In case of the coin, the observed relative frequency of side 1 can be compared with the theoretical value ($p = 0.5$).

Ad III: We can again use confidence intervals or statistical tests to correctly answer this question with a given high certainty. For instance, if $p = 0.5$ is covered by the computed confidence interval, we consider the coin as a fair coin.

Note:
Statistics is not able to absolutely answer a question. The possibility of a wrong decision can never be excluded. Some scientists even believe that most of the published research results are wrong (Ioannidis (2005)). This criticism may be approached with a proper methodology, sufficiently large trials, a careful application of statistical procedures, and a cautios interpretation of the results. Moreover, everything should be documentend completely and comprehensively to make the research replicable and reproducible (Gentleman and Temple Lang (2007)).

5.2 Point Estimation

In this section, we want to determine the unknown parameters of simple parametric models. This procedure is called estimation, more precisely we are looking for point estimators of the unknown parameters. We first define the notions parametric model and point estimator.

Definition 5.1 (Parametric model, point estimator)**.** ***(a)*** *A* ***parametric model*** *is a set $\mathcal{P} = \{P_\theta \mid \theta \in \Theta\}$ of probability distributions, where the elements of $\mathcal{P}$ are uniquely identifiable by their parameter $\theta \in \Theta \subset \mathbb{R}^k$ ($k \in \mathbb{N}$). This is also called a* ***parametric family****.*

(b) *Let $\mathcal{P} = \{P_\theta \mid \theta \in \Theta\}$ be a parametric family of probability distributions, where $\Theta \subset \mathbb{R}^k$ ($k \in \mathbb{N}$) is the set of all possible parameters. Furthermore, let $x_1, \ldots, x_n$ be a* ***representative*** *sample of size $n \in \mathbb{N}$ from some element $P_\theta \in \mathcal{P}$ (θ unknown). Then, a* ***point estimator*** *or* ***estimator*** *S_n is a random variable*

$$S_n : \mathbb{R}^n \to \Theta,\ (x_1, \ldots, x_n) \mapsto S_n(x_1, \ldots, x_n) =: \hat{\theta} \tag{5.1}$$

where $\hat{\theta}$ is the ***point estimation*** *or* ***estimation*** *of θ.*

We give some additional explanations.

Remark 5.2. ***(a)*** *The notion parametric family implies that the elements of the set may be identified by their parameters. Formally, there is a function that maps a given θ to a certain P_θ and the mapping is unique.*

(b) *The observations of a representative sample correspond to realizations of independent random variables $X_1, \ldots, X_n$, where it holds $X_i \sim P_\theta$ ($i = 1, \ldots, n$). Therefore, the random variables are also called independent and identical distributed (iid).*

(c) *A point estimator S_n is a random variable, i.e., a random function. Consequentially, an estimator has a certain distribution that depends on the unknown distribution P_θ. The quality of an estimator is usually assessed by $E(S_n)$ und $Var(S_n)$. If $E(S_n) = \theta$, the estimator is called* ***bias-free*** *or* ***unbiased****. It means that the estimator in average estimates the true parameter. If $Var(S_n)$ is additionally minimal, the estimator is called* ***efficient****. That is, there is no unbiased estimator that is able to estimate θ more accurately. For more complex models it is usually difficult or even impossible to find efficient estimators. In such a case one often considers the* ***mean squared error (MSE)***

$$MSE(S_n) = E(S_n - \theta)^2 = \big(E(S_n - \theta)\big)^2 + Var(S_n) = Bias(S_n)^2 + Var(S_n)$$

and tries to minimize it. Instead of unbiasedness, one often has to be satisfied with the so-called ***consistency****, which means that the estimator with increasing sample size more and more approaches (in a*

probability theoretic sense) the true (unknown) parameter; i.e., $\lim_{n\to\infty} S_n = \theta$ *(in a probability theoretic sense). Figure 5.1 illustrates the notions unbiased and efficient, where the center of the circle corresponds to the true (unknown) parameter.*

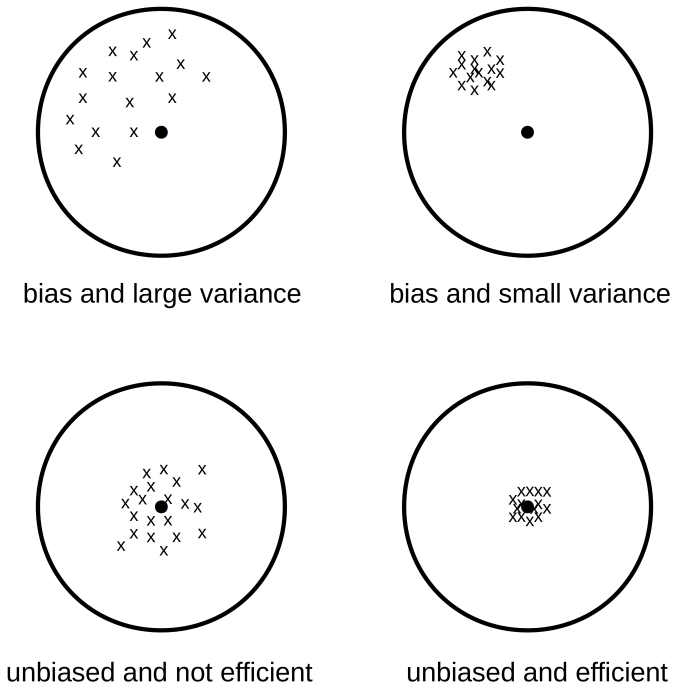


Figure 5.1: Illustration of unbiased and efficient.

In the following example, we introduce some unbiased and efficient estimators.

Example 5.3. **(a)** We consider the probability model $\{\text{Bernoulli}\,(p) \mid p \in (0,1)\}$. The, the relative frequency is an unbiased and efficient estimator of the unknown probability p.

(b) Let the probability model $\{\mathcal{N}(\mu, \sigma^2) \mid \mu \in \mathbb{R}\}$ be given, where $\sigma^2 \in (0,\infty)$ is known. Then, the arithmetic mean is an unbiased and efficient estimator of the unknown expectation μ.

(c) The situation becomes somewhat more complicated in case of the model $\{\mathcal{N}(\mu, \sigma^2) \mid \mu \in \mathbb{R}, \sigma \in (0,\infty)\}$. The sample variance is a possible estimator for the unknown variance σ^2, but it is not unbiased. The bias is $-\frac{1}{n}\sigma^2$. We obtain an unbiased estimator by using the standardization $\frac{1}{n-1}$; i.e.

$$\tilde{S}_n(x_1,\ldots,x_n) = \frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - \text{AM}\,(x_1,\ldots,x_n)\right)^2 \tag{5.2}$$

Therefore, avoiding a bias is the reason why one usually uses $\frac{1}{n-1}$ instead of $\frac{1}{n}$ for computing the empirical variance. Regarding the accuracy of the estimation, the variance of the (true) sample variance is smaller than the variance of $\tilde{S}_n$.

We use our ICU dataset und want to estimate the prevalence (disease frequency) of liver failure on the ICU. Please, import the dataset as described in Section 2.3, if you have not done this already. There you also find more information about the data. In contrast to descriptive statistics, it is now necessary for the

validity of the results that the 500 ICU patients were randomly and representatively selected from the ICU population. We compute the relative frequency as described in Section 2.5.1.

```
ICUData ← read.csv(file = "ICUData.csv", stringsAsFactors = TRUE)
## unbiased and efficient
table(ICUData$liver.failure)/nrow(ICUData)
```

```
   0    1
0.96 0.04
```

That is, 4% of the randomly selected ICU patients had a liver failure. We now regard this as an estimate for all ICU patients and later we will further ensure the result. Therefore, a possible model for the prevalence of liver failure on the ICU is Bernoulli (0.04).

The analysis in Section 2.6.1 suggests that the maximum body temperature of ICU patients – except for strongly undercooled (hypothermic) patients such as patient 398 – is quite well described by a normal distribution. We estimate expectation and variance.

```
## unbiased and efficient
mean(ICUData$temperature[-398])
```

```
[1] 37.72044
```

```
## unbiased
sd(ICUData$temperature[-398])
```

```
[1] 1.173187
```

The results are identical to Section 2.6.1. However, we do not longer use these values for describing the sample, but as parameters of a probability model, which describes the underlying population.

> Note:
> For interpreting the result and inferring to the ICU population, it is of crucial importance, whether we want to include strongly undercooled patients such as patient 398. If this is not the case, we can use $\text{Norm}(37.7, 1.2^2)$ as a model for the maximum body temperature. Otherwise, we have to understand that we can not describe the maximum body temperature by a normal distribution as such an extreme temperature as $9.1°C$ is practically impossible.

We apply the estimated model and compute the probability that the maximum body temperature is less than $10°C$. We get

```
pnorm(10, mean = 37.7, sd = 1.2)
```

```
[1] 3.404114e-118
```

Next, we will address the question how to find optimal or at least good estimators. This is also called **estimator construction**. The probably most frequently applied principle is maximum likelihood, which is defined as follows.

Definition 5.4 (Maximum likelihood estimator)**.** *Let $\mathcal{P} = \{P_\theta \mid \theta \in \Theta\}$, $\Theta \subset \mathbb{R}^k$ ($k \in \mathbb{N}$), be some probability model with probability mass function or density d_θ. Furthermore, let $x_1, \ldots, x_n$ be realizations of independent and P_θ distributed random variables $X_1, \ldots, X_n$. Then, the **likelihood function** is*

$$L(\theta) = \prod_{i=1}^{n} d_\theta(x_i) \tag{5.3}$$

*and the **maximum likelihood estimator** (abbreviated: ML estimator) for θ is the position of the maximum of $L(\theta)$.*

We give some additional explanations.

Remark 5.5. ***(a)*** *In case of the ML estimator, θ is chosen such that the observed data has the maximum possible probability in the assumed probability model.*

(b) *The ML construction principle is generally applicable and usually leads to an (asymptotically) unbiased and efficient estimator. However, there are also probability models, where it is not applicable.*

(c) *In simple cases, the ML estimator can be determined by direct analytical calculations. The numerical computation of the likelihood function is numerically difficult in practice (product of many small numbers) and usually the so-called **log-likelihood function** is used*

$$l(\theta) = \ln\left(L(\theta)\right) = \sum_{i=1}^{n} \ln\left(d_\theta(x_i)\right) \tag{5.4}$$

where the position of the maximum is identical to $L(\theta)$. This simple "trick" clearly simplifies the numerical computations and leads to more stable results.

(d) *There are several R packages that include functions to compute ML estimators. For simple probability models one can for example apply the packages `"stats4"` (R Core Team (2022a)), `"MASS"` (Venables and Ripley (2002)), `"fitdistrplus"` (Delignette-Muller and Dutang (2015)), or `"distrMod"` (Kohl and Ruckdeschel (2010)).*

We present some examples of ML estimators.

Example 5.6. **(a)** In case of the simple Bernoulli model $\mathcal{P} = \{\text{Bernoulli}\,(p) \mid p \in (0, 1)\}$, the ML estimator can explicitly be determined via the first derivative of the log-likelihood function. The likelihood function reads

$$L(p) = \prod_{i=1}^{n} p^{x_i}(1-p)^{1-x_i} \tag{5.5}$$

and thus the log-likelihood function is

$$l(p) = \sum_{i=1}^{n} \ln\left[p^{x_i}(1-p)^{1-x_i}\right] = \sum_{i=1}^{n} \left[x_i \ln(p) + (1-x_i)\ln(1-p)\right] \tag{5.6}$$

We calculate the derivative of the log-likelihood function and obtain

$$\begin{aligned} l'(p) = \frac{d}{dp} l(p) &= \sum_{i=1}^{n} \left[\frac{x_i}{p} - \frac{1 - x_i}{1 - p} \right] = \frac{1}{p} \sum_{i=1}^{n} x_i - \frac{1}{1 - p} \left[n - \sum_{i=1}^{n} x_i \right] \\ &= -\frac{n}{1 - p} + \frac{(1 - p) + p}{p(1 - p)} \sum_{i=1}^{n} x_i \\ &= -\frac{n}{1 - p} + \frac{1}{p(1 - p)} \sum_{i=1}^{n} x_i \end{aligned} \tag{5.7}$$

Setting the first derivative equal to zero ($l'(p) = 0$) yields

$$\frac{n}{1 - p} = \frac{1}{p(1 - p)} \sum_{i=1}^{n} x_i \qquad \Longleftrightarrow \qquad p = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{5.8}$$

As x_i can only take the values 0 and 1, the arithmetic mean of the x_i is nothing else but the relative frequency of 1.

(b) Normal distribution model: The ML estimator for the expectation is the arithmetic mean, for the variance it is the sample variance (i.e. standardization $\frac{1}{n}$).

(c) Poisson model: The ML estimator is the arithmetic mean.

(d) Exponential model: The ML estimator is the inverse of the arithmetic mean.

Instead of the functions `mean` and `sd`, we apply function `fitdistr` of package `"MASS"` (Venables and Ripley (2002)) to determine the ML estimator of the maximum body temperature. We need not to install package `"MASS"`, since it belongs to the group of recommended packages and thus is included in the standard installation of R. We use the normal distribution model and exclude patient 398.

```
fitdistr(ICUData$temperature[-398], densfun = "normal")
```

```
      mean             sd
  37.72044088     1.17201119
( 0.05246643) ( 0.03709937)
```

As the ML estimator for the variance includes the standardization $\frac{1}{n}$, the result slightly differs from the result of function `sd`. In addition to the estimates, we get some additional output showing the **standard errors** of the estimates; see Section 5.3 for more details.

Alternatively, we use package `"distrMod"` (Kohl and Ruckdeschel (2010)), which is derived from package `"distr"` (Ruckdeschel et al. (2006)). Here, the estimation proceeds in two steps. First, we define the probability model and then we estimate the parameters of the generated model by means of function `MLEstimator`.

```
## Change output options
distrModOptions(show.details = "minimal")
## Define probability model
model ← NormLocationScaleFamily()
## Estimate parameters by ML
MLEstimator(ICUData$temperature[-398], model)
```

```
Evaluations of Maximum likelihood estimate:
-------------------------------------------
      mean            sd
  37.72044088    1.17201119
 ( 0.05246643) ( 0.03709937)
```

The normal distribution model is called `NormLocationScaleFamily`, because it is more generally a **location and scale model**, since expectation (location parameter) as well as variance (dispersion or scale parameter) must be estimated. The result for the ML estimate is identical to the result of `fitdistr`.

Note:
The abstract object-oriented implementation in the package `"distrMod"` (Kohl and Ruckdeschel (2010)) allows the calculation of some additional quantities, which can be used, for example, for the calculation of confidence intervals (Section 5.3) or for diagnostics. With the help of `distrModOptions` function and the option `show.details = ‘‘minimal’’`, the corresponding additional outputs were deliberately suppressed. This serves the clarity and the better understanding of the outputs. The additional information would require explanations, which exceed the contents of this book.

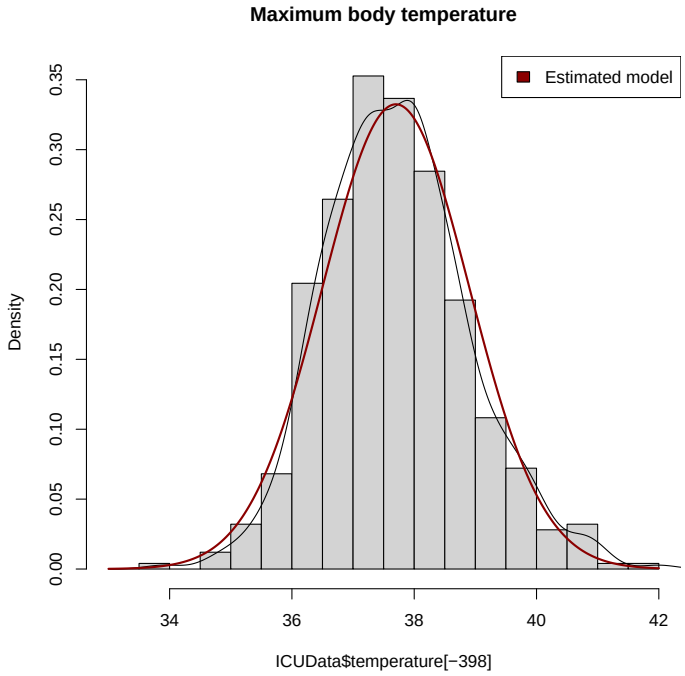
Although the estimation of the unknown parameters worked without any problems, this does not mean that one can blindly trust the estimated values and that the fitted model actually fits the data. Before statistical results and estimated models are further interpreted, a **model validation** should always be performed. This is also called **model diagnostics**. For the validity of each statistical analysis, certain conditions are necessary, which can be more or less strict. The aim of this step is to check the validity of these assumptions and thus of the statistical results, which is difficult in most cases. In case of new data being collected and used for validation, this is also referred to as **external validation**. If the validation is based on the same data that was used for the estimation, this is called an **internal validation**.

Note:
In internal validation, one should proceed very carefully, since every statistical procedure in a certain way also tries to “impose” the model or the assumptions on the data. So there is a risk of underestimating the deviations from the model or the preconditions. This is called **(re-)substitution bias**. The usual approach to reduce or avoid this bias is the use of so-called **resampling methods**. With these methods (e.g. cross-validation or bootstrap), one usually selects a part of the data set at random and performs the statistical method on this part. The remaining part is used for validation. Since this is a random selection, one can repeat this procedure many times and get an impression of the validity of the statistical results and their variance. For simple models resampling methods are usually not used. Therefore I will not pursue this approach here.

Some requirements can be examined with the help of statistical tests. However, I do not recommend this approach, since it is usually unclear, which power (cf. Chapter 6) these tests have for the respective situation. Moreover, these pre tests rely on further assumptions that are necessary for them to keep the type I error (cf. Chapter 6) and to generate valid results. In the following, we will apply diagnostic plots,

which in my opinion represent the best possibility for model validation. In the given example, we are dealing with a simple parametric model, namely the normal distribution. Since the density as well as the (cumulative) distribution function and the quantile function are defining functions for the normal distribution, it makes sense to take a look at these functions for the estimated model and to compare them with the corresponding empirical counterparts obtained from the available data.
We plot the data (without patient 398) and the estimated model by means of a histogram in combination with a density plot.

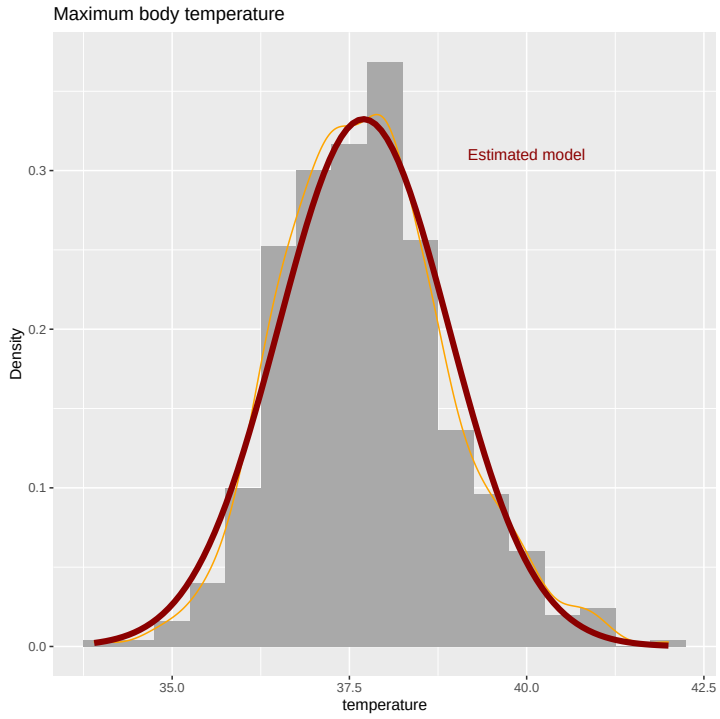
```
hist(ICUData$temperature[-398], breaks = seq(from = 33, to = 42, by = 0.5),
     main = "Maximum body temperature", ylab = "Density", freq = FALSE)
lines(density(ICUData$temperature[-398]))
curve(dnorm(x, mean = 37.7, sd = 1.2), col = "darkred", from = 33, to = 42,
      n = 501, add = TRUE, lwd = 2)
legend("topright", fill = "darkred", legend = "Estimated model")
```



Argument `lwd` controls the thickness of the lines, where the default value is 1 and values larger than 1 lead to thicker lines. We repeat the plot applying the functions of package `"ggplot2"` (Wickham (2009)). Beside the functions `ggplot`, `geom_histogram`, and `geom_density`, we need function `stat_function`, which can be used to add the graph of a function to a plot. With function `annotate` we can add some text to the plot.

```
ggplot(ICUData[-398,], aes(x=temperature)) +
  geom_histogram(aes(y=..density..), binwidth = 0.5, fill = "darkgrey") +
  geom_density(color = "orange") + ylab("Density") +
  stat_function(fun = dnorm, args = list(mean = 37.7, sd = 1.2),
                color = "darkred", lwd = 2) +
  annotate("text", x = 40, y = 0.31, col = "darkred",
```

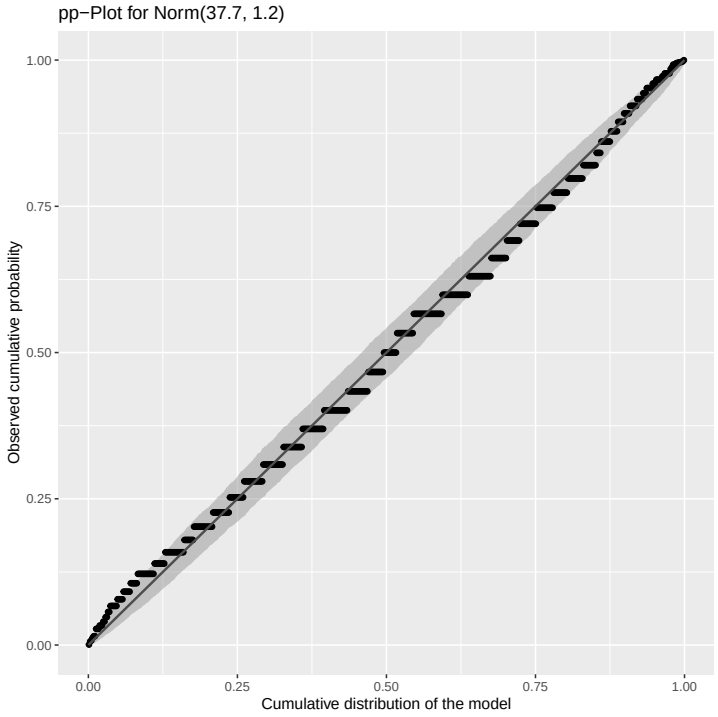
```
7             label = "Estimated model") +
8   ggtitle("Maximum body temperature")
```



We can see that the actual data and the estimated model data are quite similar. Certain random deviations from the model assumptions are always to be expected and further complicate the assessment of validity. The other way around, it is the case that perfect agreement between the model and the data can be an indication for a manipulation of the data. An example for this is the discussion surrounding the work of Fisher (1936), who questioned the results of Mendel (1866), whether Mendel adapted his results on heredity somewhat to his expectations; see also Novitski (2004).

In a similar way, we could compare the empirical cumulative distribution function with the cumulative distribution function of the model. This is called a **pp-plot** (probability-probability plot). A pp-plot can be generated using package `"qqplotr"` (Almeida et al. (2018)) with the functions `stat_pp_point`, `stat_pp_line` and `stat_pp_band` in combination with the package `"ggplot2"` (Wickham (2009)).

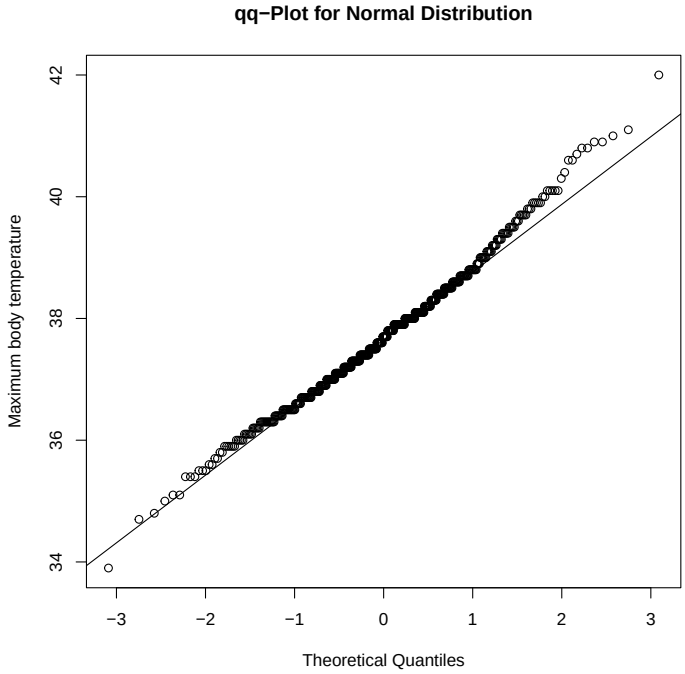
```
ggplot(ICUData[-398,], aes(sample = temperature)) +
  qqplotr::stat_pp_band(dparams = list(mean = 37.7, sd = 1.2)) +
  qqplotr::stat_pp_point(dparams = list(mean = 37.7, sd = 1.2)) +
  qqplotr::stat_pp_line() +
  xlab("Cumulative distribution of the model") +
  ylab("Observed cumulative probability") +
  ggtitle("pp-Plot for Norm(37.7, 1.2)")
```



Again, we can see a great similarity between the actual data and the model. The points line up along the bisector and are to a very large extent within the 95% confidence band, which is formed from point-wise bootstrap confidence intervals. For details about confidence intervals and about bootstrap confidence intervals see Section 5.3.

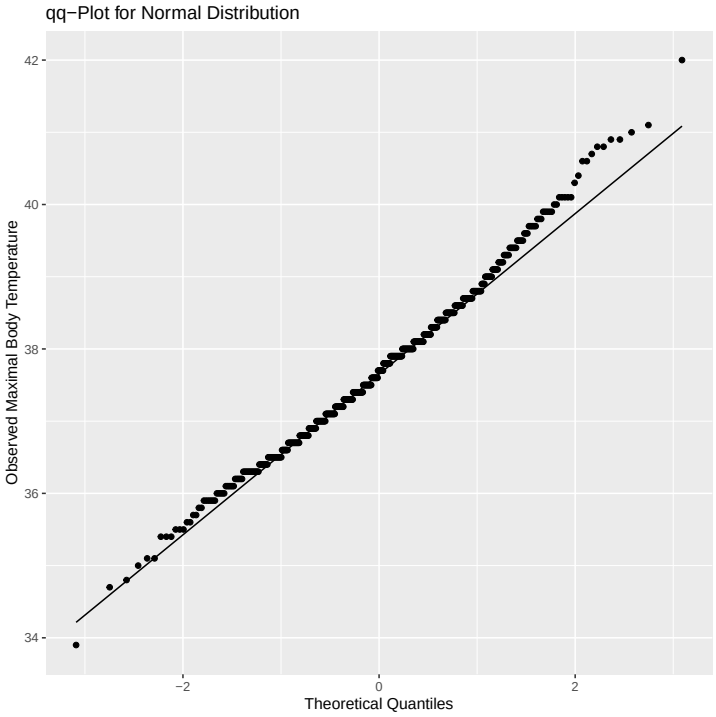
We also want to compare the quantile functions. For such comparisons, the so-called quantile-quantile plot (**qq-plot** for short) is most frequently applied. In this plot, the empirical and theoretical quantiles are compared. The closer the points are to the straight line, the better the theoretical model explains the observations. In case of the normal distribution, we can use R functions `qqnorm` and `qqline`.

```
qqnorm(ICUData$temperature[-398], main = "qq-Plot for Normal Distribution",
       ylab = "Maximum body temperature")
qqline(ICUData$temperature[-398])
```



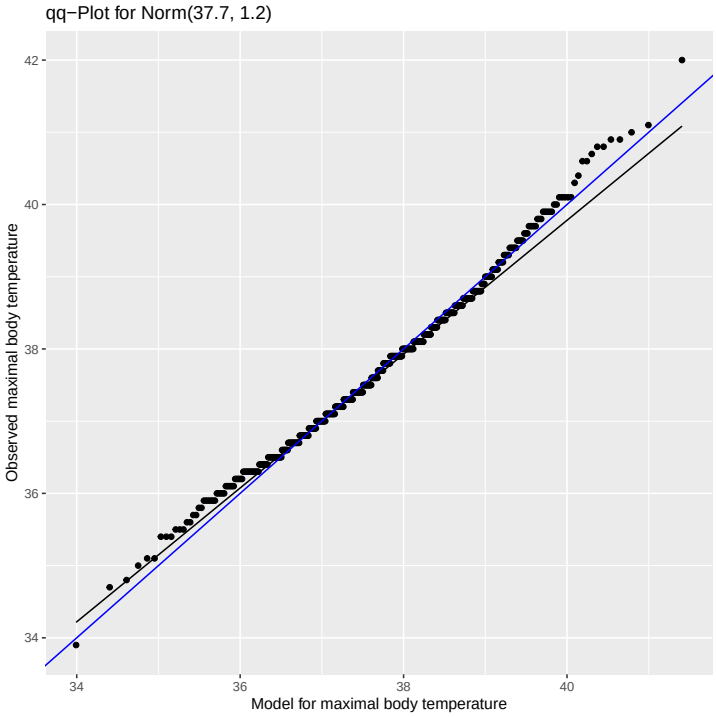
We can also use the functions `stat_qq` and `stat_qq_line` of package `"ggplot2"` (Wickham (2009)) for qq-plots. Since there is also function `stat_qq_line` in package `"qqplotr"` (Almeida et al. (2018)), the additional usage of `ggplot2::` guarantees that the function from package `"ggplot2"` is called.

```
ggplot(ICUData[-398,], aes(sample = temperature)) +
  stat_qq() + ggplot2::stat_qq_line() +
  xlab("Theoretical Quantiles") +
  ylab("Observed Maximal Body Temperature") +
  ggtitle("qq-Plot for Normal Distribution")
```



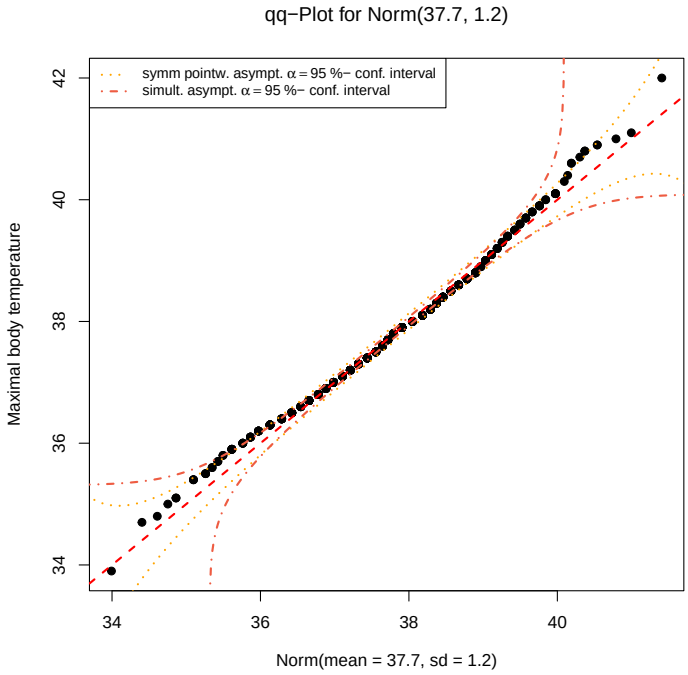
With the help of this plot we can generally check, if the data may stem from a normal distribution. In the current situation, we see that a higher quantity of high temperatures were observed as we would expect in case of the normal distribution. If we want to compare the data with a very concrete distribution, we can use package `"ggplot2"` (Wickham (2009)) and specify the respective parameters.

```
ggplot(ICUData[-398,], aes(sample = temperature)) +
  stat_qq(dparams = list(mean = 37.7, sd = 1.2)) +
  ggplot2::stat_qq_line(dparams = list(mean = 37.7, sd = 1.2)) +
  geom_abline(slope = 1, color = "blue") +
  xlab("Model for maximal body temperature") +
  ylab("Observed maximal body temperature") +
  ggtitle("qq-Plot for Norm(37.7, 1.2)")
```



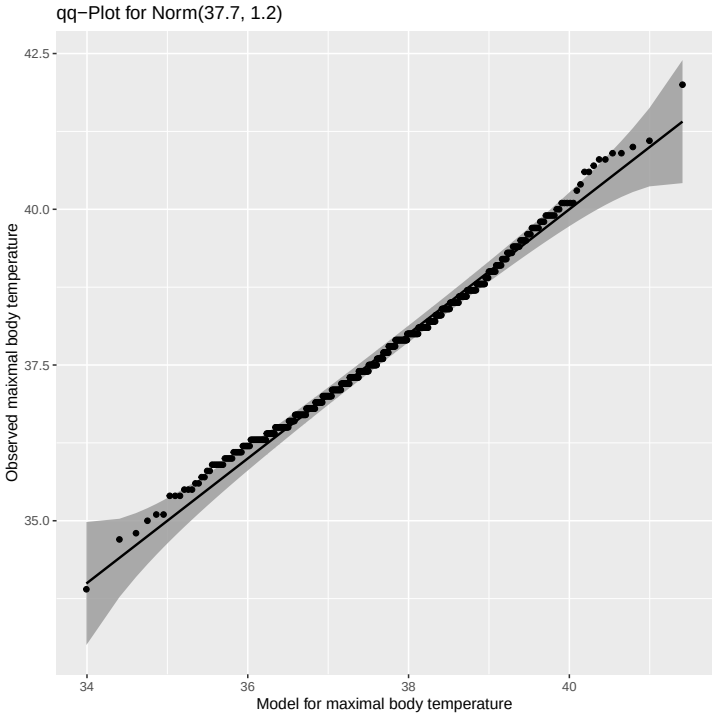
The additional angle bisector drawn in blue shows that the standard qq line (based on the 1st and 3rd quartiles) does not have to be identical to the angle bisector. Deviations from the bisector also indicate deviations between the model and the data. Therefore it is recommended to additionally draw in the angle bisector or to relate the points to the angle bisector, respectively.
An alternative plot is possible with the function `qqplot` from the `"distr"` package (Ruckdeschel et al. (2006)). In this case the plotted straight line corresponds to the bisector.

```
qqplot(ICUData$temperature[-398], Norm(mean = 37.7, sd = 1.2),
       xlab = "Maximal body temperature",
       main = "qq-Plot for Norm(37.7, 1.2)")
```



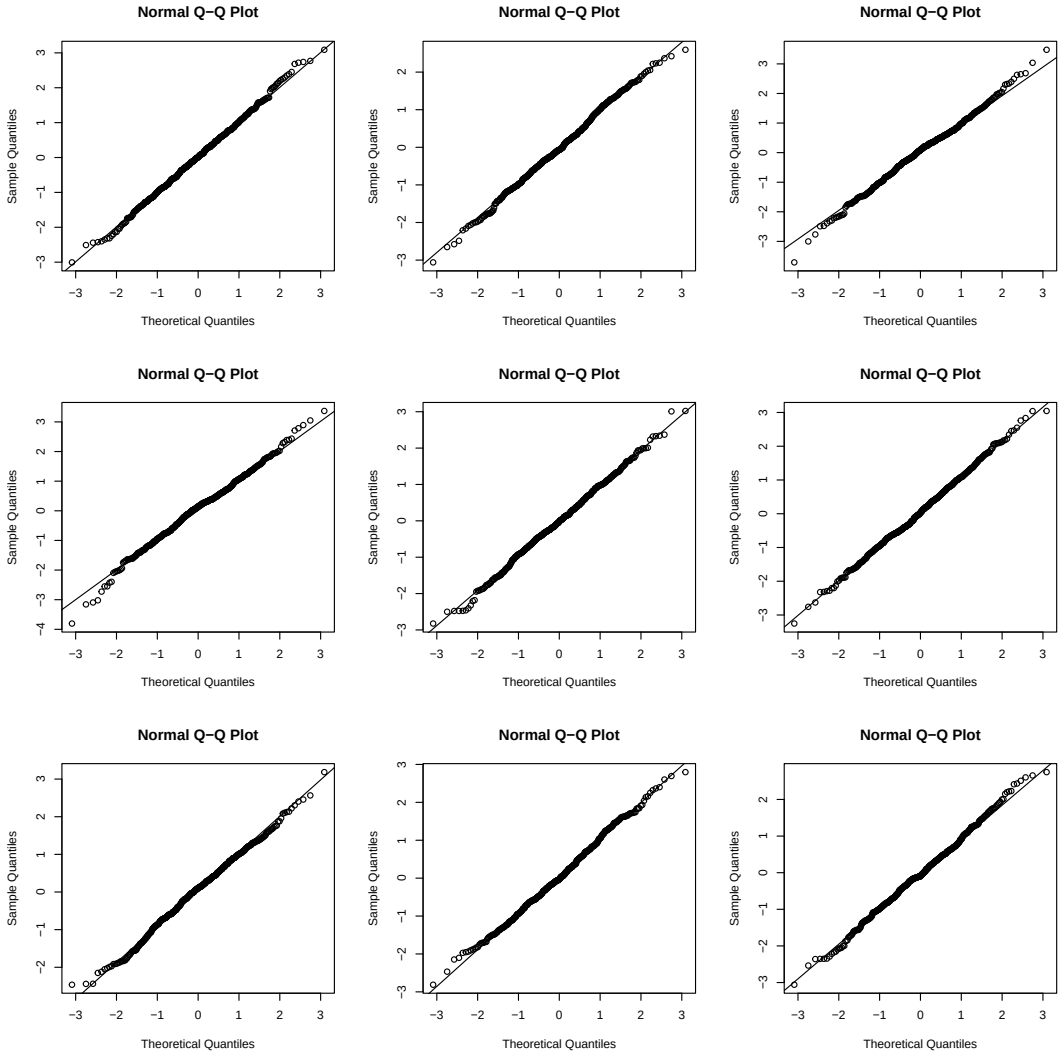
The data also seems to be in good agreement with the estimated model with respect to the qq-plot, although some minor deviations can be seen. The deviations can further be estimated by specifying (pointwise and simultaneous) confidence bands. Confidence intervals will be explained in more detail in Section 5.3. A similar possibility is offered by the package `"qqplotr"` (Almeida et al. (2018)), which provides additional functionality in the form of the functions `stat_qq_point`, `stat_q_line`, and `stat_q_band` for the `"ggplot2"` package (Wickham (2009)). By using `identity = TRUE` we also achieve an alignment to the angle bisector.

```
ggplot(ICUData[-398,], aes(sample = temperature)) +
  qqplotr::stat_qq_band(dparams = list(mean = 37.7, sd = 1.2), identity = TRUE) +
  qqplotr::stat_qq_point(dparams = list(mean = 37.7, sd = 1.2)) +
  qqplotr::stat_qq_line(dparams = list(mean = 37.7, sd = 1.2), identity = TRUE) +
  xlab("Model for maximal body temperature") +
  ylab("Observed maixmal body temperature") +
  ggtitle("qq-Plot for Norm(37.7, 1.2)")
```



For us humans, it is difficult to assess whether the fluctuations we see are still within the expected stochastic deviations. We can also visualize these random deviations from the theoretical model with the help of simulated data. This also helps us to better assess the observed deviations. We compare the qq-plot above with qq-plots of simulated normally distributed data. In the following, we generate standard-normally distributed data with the function `rnorm` (the concrete parameters of the normal distribution are not relevant here) and then display them in the form of qq-plots using `qqnorm` and `qqline`. In order to get an impression of the fluctuations from sample to sample, we repeat the whole process nine times. For this we use a so-called `for` loop. To display all plots in one window we also use the function `par`, which can be used to customize a variety of graphics parameters. With the argument `mfrow` a plot window can be divided into a certain number of rows and columns. In our case, we choose three rows and three columns, which are then filled row by row.

```
par(mfrow=c(3,3))
for(i in 1:9){
  x ← rnorm(499)
  qqnorm(x)
  qqline(x)
}
```



Accordingly, even with normally distributed data, we see certain deviations from the straight line which are quite similar to the deviations in the case of the maximum body temperature. This confirms our first impression that the maximum body temperature of ICU patients (excluding severely hypothermic patients) can be described quite well by a normal distribution.

Note:
As already mentioned above, in the case of real and simulated real data the points will never all lie exactly on the straight line. On the contrary, this would be an indication that the data is not real or that the data has been falsified accordingly. Significant deviations from the straight line indicate that the model does not fit the data (or vice versa). The points should therefore not form a clear "banana shape" (possibly also one-sided or twisted).

In the case of normal distribution, we can alternatively use the median and the MAD and IQR as consistent estimators for the mean and standard deviation. As we have already seen in Section 2.5.1, it is not necessary to remove patient 398.

```
1 median(ICUData$temperature)
```

```
[1] 37.7
```

```
mad(ICUData$temperature)
```

```
[1] 1.18608
```

```
sIQR(ICUData$temperature)
```

```
[1] 1.111952
```

The results are very similar to the ML estimates.

Note:
Median, MAD, and IQR provide consistent estimates for the theoretical median, MAD, and IQR under very general assumptions. In particular, it is not necessary to assume a certain parametric model. Therefore, they are also called non-parametric estimators. Because of this general property and their additional robustness, these estimators are useful for many applications.

Another estimating principle, which works well for simple probability models, is the so-called minimum-distance estimation.

Definition 5.7 (Minimum-distance estimator)**.** *Let $\mathcal{P} = \{P_\theta \mid \theta \in \Theta\}$, $\Theta \subset \mathbb{R}^k$ $(k \in \mathbb{N})$, be some probability model. Furthermore, let $x_1, \ldots, x_n$ be realizations of independent and P_θ distributed random variables $X_1, \ldots, X_n$ and $\hat{F}_n$ their empirical distribution. Then, we consider*

$$D(\theta) = dist\,(P_\theta, \hat{F}_n) \tag{5.9}$$

where dist represents a distance between distributions. The ***minimum-distance estimator*** *(abbreviated: MD estimator) for θ is the position of the minimum of $D(\theta)$.*

We give some additional explanations.

Remark 5.8. ***(a)*** *MD estimators are usually consistent estimators.*

(b) *In the following, we will determine the Cramér-von-Mises-MD estimator (CvM-MD estimator for short) and the Kolmogorov(-Smirnov) MD estimator (KS-MD estimator for short). The definitions of the corresponding distances are based on the cumulative distribution functions. The* ***Cramér-von-Mises distance*** *reads*

$$dist_{CvM}(P_\theta, \hat{F}_n) = \int |P_\theta(x) - \hat{F}_n(x)|^2 \, Q(dx) \tag{5.10}$$

where usually P_θ or $\hat{F}_n$ is chosen for the distribution Q. In case of $\hat{F}_n$, the integral becomes a sum. The ***Kolmogorov(-Smirnov) distance*** *is*

$$dist_{KS}(P_\theta, \hat{F}_n) = \max_{x \in \mathbb{R}} |P_\theta(x) - \hat{F}_n(x)| \tag{5.11}$$

Both MD estimator are very robust against outliers and certain model deviations.

We may apply function `MDEstimator` of package `"distrMod"` (Kohl and Ruckdeschel (2010)) for computing MD estimators. In case of the CvM-MD and the KS-MD estimator the package additionally provides functions `CvMMDEstimator` and `KolmogorovMDEstimator`. We again consider the maximum body temperature of our ICU patients and first compute the CvM-MD estimator. As in case of the ML estimator, we proceed in two steps: First we define the model and then compute the estimator. Without patient 398 we get

```
Model ← NormLocationScaleFamily()
CvMMDEstimator(ICUData$temperature[-398], Model)
```

```
Evaluations of Minimum CvM distance estimate ( mu = model distr. ) :
----------------------------------------------------------------------
      mean           sd
  37.67520679    1.13610915
 ( 0.06932949) ( 0.04461328)
```

The result is very similar to the ML estimator. We repeat the estimation and this time apply the KS-MD estimator.

```
KolmogorovMDEstimator(ICUData$temperature[-398], Model)
```

```
Evaluations of Minimum Kolmogorov distance estimate  :
------------------------------------------------------
estimate:
     mean         sd
37.679362  1.141183
```

Again, we obtain a very similar result. We repeat the estimation and this time do not omit patient 398. The results show the robustness of the MD estimators in contrast to the ML estimator.

```
## ML estimator
MLEstimator(ICUData$temperature, Model)
```

```
Evaluations of Maximum likelihood estimate:
-------------------------------------------
      mean           sd
  37.66320000    1.73373751
 ( 0.07753510) ( 0.05482559)
```

```
## CvM-MD estimator
CvMMDEstimator(ICUData$temperature, Model)
```

```
Evaluations of Minimum CvM distance estimate ( mu = model distr. ) :
----------------------------------------------------------------------
      mean           sd
  37.67176542    1.13941964
 ( 0.06946194) ( 0.04469851)
```

```
## KS-MD estimator
KolmogorovMDEstimator(ICUData$temperature , Model)
```

```
Evaluations of Minimum Kolmogorov distance estimate  :
------------------------------------------------------
estimate:
     mean         sd
37.676420  1.143365
```

Thus, in case of the MD estimators the results remain almost unchanged, whereas in case of the ML estimator especially the estimate of the standard deviation clearly increases.

As a final class of estimators, we consider so-called **optimal-robust asymptotically linear estimators**; in short: **AL-estimators**. In this case in addition to the parametric model $\mathcal{P}$ one considers a so-called shrinking neighborhood around the assumed model distribution. In the vast majority of cases this is the so-called **contamination neighborhood**, which is also known as Tukey's gross-error model

$$Q_n = (1 - r_n)P_\theta + r_n H_n \qquad r_n = \frac{r}{\sqrt{n}} \tag{5.12}$$

Here $r \in [0, \infty)$ is the **neighborhood radius** and H_n is an arbitrary probability distribution. Q_n is therefore the probability distribution, which actually generated the observed data. The majority ($(1-\varepsilon)\%$, $\varepsilon \in [0, 1]$) of the data is from the assumed model distribution, but a small part of the data ($\varepsilon\%$) is erroneous and was caused by an unknown error distribution H_n. For a given sample size n, r is obtained by $r = \sqrt{n}\varepsilon$.

Remark 5.9. ***(a)*** *The shrinking of the neighborhood with the sample size n ensures, that the deviations remain so small that they cannot or only with great difficulty can reliably be detected by statistical methods (e.g. goodness-of-fit or outlier tests).*

(b) *The neighborhood not only ensures that the AL estimators are robust to a certain proportion of outliers, but that they also provide reliable results in the case of other (more subtle) model deviations.*

AL estimators minimize the **maximum asymptotic MSE**, which in this case can be written as

$$\text{asMSE}_\theta(\psi_\theta, r) = \text{E}_\theta\, |\psi_\theta|^2 + r^2 \left(\sup_{P_\theta} |\psi_\theta| \right)^2$$

Here, the AL estimators S_n are described by their so-called **influence function** ψ_θ. The influence function indicates what influence an individual single observation has on the estimation result. It corresponds in a certain sense to the derivative of the estimation function. By determining the influence curve, which minimizes this maximum asymptotic MSE, one also finds the associated corresponding AL estimator. One then calculates the estimator by a so-called 1-step construction

$$S_n^{(1)} = S^{\text{start}} + \frac{1}{n} \sum_{i=1}^{n} \psi_\theta(x_i - S^{\text{start}}) \tag{5.13}$$

for which one needs a suitable initial value/estimator S^{start}. This construction can also be repeated k times ($k \in \mathbb{N}$) or repeated until the estimated value does not change significantly, where in the second

step then $S_n^{(1)}$ takes over the role of the starting estimator, etc. In our experience it is usually sufficient to choose $k = 3$.

> Note:
> The starting estimators should ideally be very robust estimators; i.e., estimators that still provide reliable results even with a high proportion of outliers. Suitable estimators are, median, MAD, IQR or also CvM-MD estimators and KS-MD estimators.

The problem in practice, that the exact proportion of erroneous data is not known, can be solved by using the so-called **radius minimax estimator**, in short: **RMX estimators**. These are AL estimators, which are not optimal for a certain proportion ε of deviations, but for a whole interval $\varepsilon \in [\varepsilon_1, \varepsilon_2]$. For more details on AL and RMX estimators, we refer to Rieder (1994), Kohl (2005), and Kohl et al. (2010). AL and RMX estimators for the normal distribution are implemented in package `"RobLox"` (Kohl (2019)), where median and MAD are used as starting estimators. We again consider our temperature data and assume, based on the qq-plot, that between one (patient 398) and 5% of the observations are erroneous. We calculate the corresponding RMX estimator using the 3-step construction.

```
roblox(ICUData$temperature, eps.lower = 1/500, eps.upper = 0.05, k = 3)
```

```
Evaluations of Optimally robust estimate:
-----------------------------------------
      mean            sd
  37.64509778    1.14027423
 ( 0.05574489) ( 0.04156504)
```

Alternatively, we can apply function `rmx` of package `"rmx"` (Kohl (2022c)), where `k = 3` is already the default setting and hence needs not to be specified. The model is given in a similar way than in case of function `fitdistr` of package `"MASS"` (Venables and Ripley (2002)) by using the short form of the respective probability distribution.

```
res.rmx ← rmx(ICUData$temperature, model = "norm",
              eps.lower = 1/500, eps.upper = 0.05)
res.rmx
```

```
RMX estimator for normal location and scale

      mean            sd
  37.64289993    1.13951576
 ( 0.05609900) ( 0.04181884)

NOTE: asymptotic standard errors are shown

Call:
rmx(x = ICUData$temperature, model = "norm", eps.lower = 1/500,
   eps.upper = 0.05)
```

The result is very similar to the above results of the other robust estimation methods. A more detailed summary of the results can be generated by means of the function `summary`.

```
summary(res.rmx)
```

```
RMX estimator for normal location and scale

     mean           sd
 37.64289993    1.13951576
( 0.05609900) ( 0.04181884)

NOTE: asymptotic standard errors are shown

            Sample size = 500
    Amount gross-errors = 0.2 - 5 %
    FS-corrected radius = 0.488
 Maximum asymptotic MSE = 3.6
Maximum asymptotic bias = 1.07
Asymptotic (co)variance:
     mean    sd
mean 1.57 0.000
sd   0.00 0.874

Call:
rmx(x = ICUData$temperature, model = "norm", eps.lower = 1/500,
   eps.upper = 0.05)
```

The "`rmx`" package (Kohl (2022c)) also includes a set of diagnostic functions. For example, the functions `outlier` and `getOutliers` can be used to output where the outlier areas are located and which data points are to be regarded as outliers. These are by default data points which are smaller than the 0.01% or greater than the 99.9% quantile of the estimated model.

```
outlier(res.rmx)
```

```
      Outlier region for normal location and scale

                   Parameter: mean = 37.6, sd = 1.14
              Outlier region: (-Inf, 33.9) or (41.4, Inf)
      Data in outlier region: 0.2 % + 0.2 % = 0.4 %
     Prob. of outlier region: 0.1 % + 0.1 % = 0.2 %

        Gross-errors for RMX = 0.2 - 5 %
```

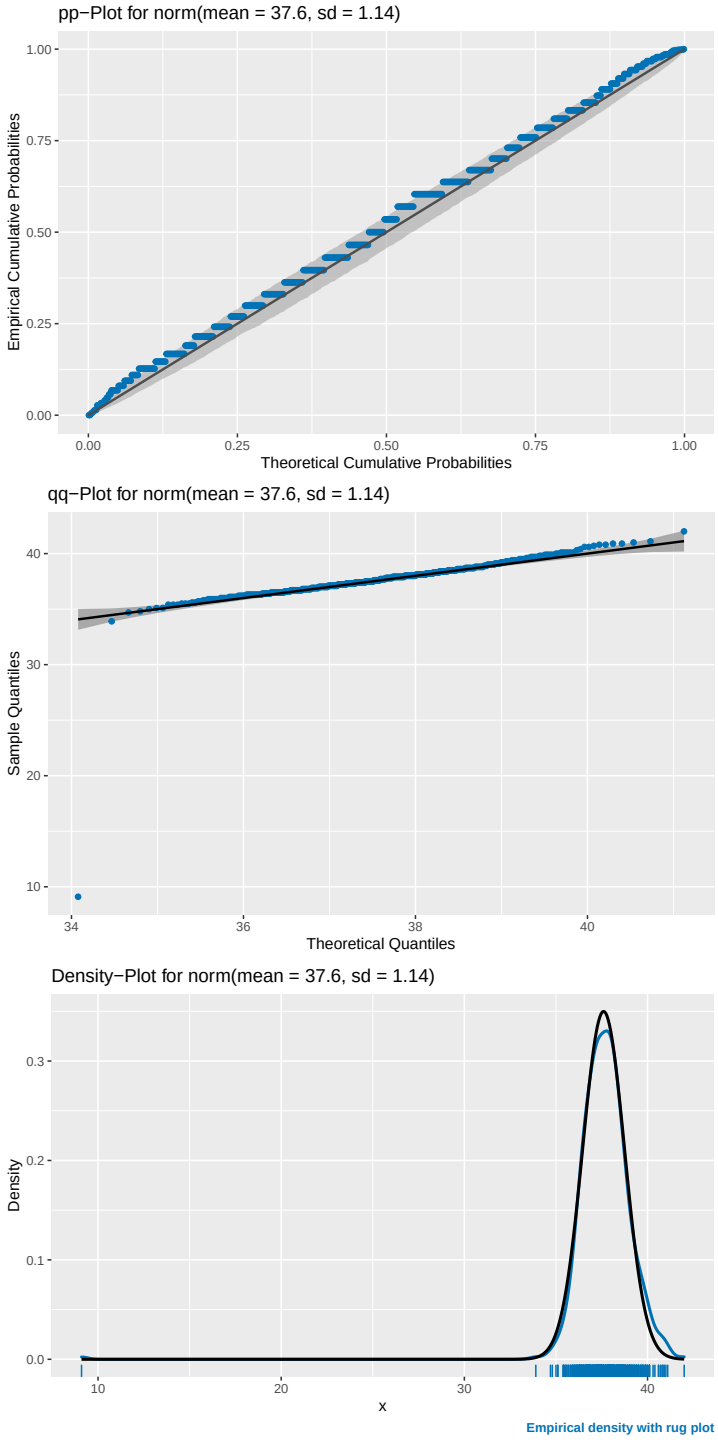
```
getOutliers(res.rmx)
```

```
$values
[1]  9.1 42.0
```

```
$indices
[1] 398 397
```

Accordingly, we get two outliers. Beside the already several times conspicuous patient 398 with a maximum body temperature of 9.1°C also patient 397 with a maximum body temperature of 42.0°C is in the outlier range. For the generation of pp-, qq- and density plot for model validation the functions `ppPlot`, `qqPlot` and `dPlot` are provided. We merge the three plots to form a single figure using the function `grid.arrange` function from the `"gridExtra"` package (Auguie (2017)).

```
gg1 ← ppPlot(res.rmx)
gg2 ← qqPlot(res.rmx)
gg3 ← dPlot(res.rmx)
grid.arrange(gg1, gg2, gg3, ncol = 1)
```



We see a fairly good agreement between data and model, if we disregard patient 398.

Package "RobLox" (Kohl (2019)) contains only an implementation of the RMX estimators for the normal distribution, for other models function roptest from package "ROptEst" (Kohl and Ruckdeschel (2019)) can be used. The "rmx" package (Kohl (2022c)) currently includes an implementation for the normal and binomial distribution. Further models will be added in the future. Package "RobExtremes" (Ruckdeschel et al. (2019)) contains optimized code for the calculation of RMX estimators for extreme

value distributions such as the Weibull distribution. Both packages contain a folder with R scripts demonstrating the use of the main functions; see https://github.com/cran/ROptEst/tree/master/inst/scripts and https://github.com/cran/RobExtremes/tree/master/inst/scripts. These folders are also located on your computer after the installation of the packages. They are subfolders in the directory of the package. The directory, where a package is installed can be determined by the function `path.package` function;

```
path.package("ROptEst")
```

Package `"rmx"` includes a so-called vignette, in which the most important functions of the package are introduced. It can be opened with the help of function `vignette`.

```
vignette("rmx")
```

Note:
There are a number of other important estimator classes such as generalized ML estimators (short: M estimators) or rank estimators (short: R estimators). Another very important construction principle, especially for more complex models, is the least squares estimation (short: KQ-estimation), which goes back to Gauss and Legendre. The model is estimated by minimizing the sum of the squared deviations of the observations from the model.

Another common estimation problem is to determine **optimal thresholds** (cut-offs). We have already seen an example of this, namely the norm range. The **normal range** or **reference range** is an interval, which usually contains the values of 95% of the examined healthy persons. In most cases, it is a two-sided interval. In this case, 2.5% of the healthy individuals have values that are below or above this interval. As seen before, this two-sided interval can be well approximated by the 2σ interval in the case of the normal (and several other) distributions; see Remark 4.16 (b). Usually, this interval is represented by the corresponding quantiles (i.e., 2.5% and 97.5% quantiles). This can be based either on a probability model, as in the example 4.20 (a), or the empirical quantiles of the collected data are used. Often also one-sided intervals are needed. Examples include detection limits **"limit of blank" (LOB)**, **"limit of detection" (LOD)** and **"(lower/upper) limit of quantitation/quantification" (LOQ, LLOQ, ULOQ)** (Armbruster and Pry (2008)). The detection limit LOB corresponds to a one-sided interval, the upper limit of which is the corresponding 95% quantile of the measured blanks. In a normal distribution the result is

$$\mathrm{LOB} = \mathrm{AM}_{\mathrm{Blanks}} + 1.645 \times \mathrm{SD}_{\mathrm{Blanks}} \tag{5.14}$$

in which the constant 1.645 corresponds to the corresponding 95% quantile of the standard normal distribution

```
qnorm(0.95)
```

```
[1] 1.644854
```

If values below this limit are obtained, it must be assumed that it is a blank sample. The detection limit LOD is obtained as

$$\text{LOD} = \text{AM}_{\text{Blanks}} + 3.3 \times \text{SD}_{\text{Blanks}} \tag{5.15}$$

in which

```
pnorm(3.3)
```

```
[1] 0.9995166
```

This is the 99.95% quantile, which is used in this case. Alternatively, instead of blank samples, one can also use samples with a low concentration and calculate LOD as

$$\text{LOD} = \text{LOB} + 1.645 \times \text{SD}_{\text{low conc. samples}} \tag{5.16}$$

If you obtain values above the LOD during a measurement, you can assume a positive signal. The analyzed analyte is therefore contained in the sample. However, a reliable quantification of the amount of the analyte investigated is only carried out for signal values above the (lower) quantification limit LOQ or more precisely LLOQ. This is calculated for example as

$$\text{(L)LOQ} = \text{AM}_{\text{Blanks}} + 10 \times \text{SD}_{\text{Blanks}} \tag{5.17}$$

Instead of the factor 10, 5 or 6 are sometimes used here. In addition, there are also other approaches to determine these detection limits; see for example Bernal (2014).

Another approach to determine cut-offs is based on considering the whole thing as a classification problem for two groups. Thus, one searches for a boundary so that both groups can be correctly classified with the highest possible probability. We must also assume a monotonic relationship between the measurement and the groups; i.e., low values indicate the presence of the first group ("negative") and high values indicate the presence of the second group ("positive"). This leads us to the terms **sensitivity** and **specificity**. Sensitivity is the probability of a member of the positive group being classified as "positive". Specificity describes the probability of a member of the negative group being classified as "negative". If we want to maximize both probabilities, it is obvious to consider the sum of the two criteria. This leads us to **Youdens J statistics** (Youden (1950))

$$\text{Youdens J} = \text{Sensitivity} + \text{Specificity} - 1 \tag{5.18}$$

which is also called **informedness** (Powers (2011)), especially in computer science. Another equivalent criterion is the arithmetic mean of the two criteria, which is also known as the **balanced accuracy (bACC)** (Brodersen et al. (2010))

$$\text{bACC} = 0.5 \times \text{Sensitivity} + 0.5 \times \text{Specificity} \tag{5.19}$$

One could also say that in this case both criteria are weighted equally. If sensitivity and specificity are not equally important, one could express this by a corresponding **weighted accuracy (wACC)** (Brodersen et al. (2010))

$$\text{wACC} = w \times \text{sensitivity} + (1 - w) \times \text{specificity} \tag{5.20}$$

with $w \in [0, 1]$. Corresponding calculations can be done for example with the help of function `optCutOff` from package `"MKclass"` (Kohl (2020a)), in which in addition to the accuracy measures mentioned above, all measures that can be calculated with function `perfMeasures` can be applied.

We consider the ICU data set to determine a cut-off for bilirubin which can be used to distinguish between patients with and without liver failure. For the calculations we round the bilirubin values to one decimal, which realistically corresponds to the accuracy of the corresponding assay.

```
Bili ← round(ICUData$bilirubin , 1)
```

We suspect a one-sided reference range; i.e., patients without liver failure (group "negative") tend to have low bilirubin levels whereas patients with liver failure (group "positive") tend to have high bilirubin levels. We first determine the empirical 95% quantile for the patients without liver failure and compute the sensitivity and specificity of this upper limit for predicting liver failure using the function `perfMeasures` from package `"MKclass"` (Kohl (2020a)).

```
quantile(Bili[ICUData$liver.failure == 0], probs = 0.95)
```

```
   95%
49.615
```

```
perfMeasures(pred = Bili , truth = ICUData$liver.failure ,
             namePos = 1, cutoff = 49.615 , measures = c("SENS", "SPEC"))
```

```
                Performance Measure(s)

             Measure Value
1 sensitivity (SENS)  0.75
2 specificity (SPEC)  0.95
```

Of course, in this case, the specificity is equal to 95%, since we used the empirical 95% quantile of patients without liver failure as the cutoff. So, in a sense, this approach gives us control over the specificity, whereas the sensitivity is unknown and has to be calculated in a second step.
Alternatively, we can also assume that the values of patients without liver failure follow a log-normal distribution and proceed analogously to the example 4.20 (a). We calculate the log values and determine their mean and standard deviation. Then we determine the 95% quantile of the corresponding log-normal distribution and calculate sensitivity and specificity.

```
mean(log(Bili[ICUData$liver.failure == 0]))
```

```
[1] 2.77395
```

```
sd(log(Bili[ICUData$liver.failure == 0]))
```

```
[1] 0.6038337
```

```
qlnorm(0.95, meanlog = 2.774, sdlog = 0.604)
```

```
[1] 43.27139
```

```
perfMeasures(pred = Bili, truth = ICUData$liver.failure,
             namePos = 1, cutoff = 43.27, measures = c("SENS", "SPEC"))
```

```
                Performance Measure(s)

             Measure     Value
1 sensitivity (SENS) 0.7500000
2 specificity (SPEC) 0.9354167
```

The result is similar to the empirical calculation. In the next step we determine the optimal cut-off, which maximizes the Youden J statistic and the balanced accuracy. We use functions `optCutoff` and `perfMeasures` from package `"MKclass"` (Kohl (2020a)).

```
optCutoff(pred = Bili, truth = ICUData$liver.failure, namePos = 1)
```

```
Optimal Cut-off             YJS
     28.4000000       0.7395833
```

```
optCutoff(pred = Bili, truth = ICUData$liver.failure,
          namePos = 1, perfMeasure = "BACC")
```

```
Optimal Cut-off            BACC
     28.4000000       0.8697917
```

```
perfMeasures(pred = Bili, truth = ICUData$liver.failure,
             namePos = 1, cutoff = 28.4, measures = c("SENS", "SPEC"))
```

```
                Performance Measure(s)

             Measure     Value
1 sensitivity (SENS) 0.9000000
2 specificity (SPEC) 0.8395833
```

We obtain a slightly higher sensitivity than in the case of the calculated reference ranges, but at the expense of a somewhat lower specificity.

Note:
The determination of the cut-off and the application of the cut-off should ideally not be performed on the same data set. Otherwise, there is a risk of bias ("resubstitution bias") in the estimation of sensitivity and specificity. If this cannot be avoided, the use of a resampling procedure is recommended, such as bootstrap or cross-validation to minimize this bias.

We calculate the optimal cut-off using 1000 bootstrap replications. Due to the very small proportion of patients with liver failure, we use a stratified version of bootstrap. We again use functions `optCutoff` and `perfMeasures` from package `"MKclass"` (Kohl (2020a)).

```
B ← 1000
n0 ← sum(ICUData$liver.failure == 0)
ind0 ← which(ICUData$liver.failure == 0)
n1 ← sum(ICUData$liver.failure == 1)
ind1 ← which(ICUData$liver.failure == 1)
cutoffs ← numeric(B)
for(i in 1:B){
  selection0 ← sample(ind0, n0, replace = TRUE)
  selection1 ← sample(ind1, n1, replace = TRUE)
  BS.data ← ICUData[c(selection0, selection1),]
  cutoffs[i] ← optCutoff(pred = round(BS.data$bilirubin, 1),
                         truth = BS.data$liver.failure,
                         namePos = 1)[1]
}
summary(cutoffs)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  17.70   28.40   31.60   37.71   53.30  104.70
```

We obtain a mean optimal cut-off, which is somewhat higher than in the case of the calculation on the full data set.

```
perfMeasures(pred = Bili, truth = ICUData$liver.failure,
             namePos = 1, cutoff = mean(cutoffs),
             measures = c("SENS", "SPEC"))
```

```
                Performance Measure(s)

             Measure  Value
1 sensitivity (SENS) 0.7500
2 specificity (SPEC) 0.9125
```

Depending on whether the focus is more on sensitivity or specificity, the corresponding accuracy measure can be adjusted by using weights. We choose a weighting of 3 to 1 in the following and refrain from using the safeguarding by bootstrap.

```
## More emphasis on sensitivity.
optCutoff(pred = Bili, truth = ICUData$liver.failure,
          namePos = 1, perfMeasure = "WACC", wACC = 0.75)
```

```
Optimal Cut-off             WACC
     17.7000000        0.9005208
```

```
perfMeasures(pred = Bili, truth = ICUData$liver.failure,
             namePos = 1, cutoff = 17.7, measures = c("SENS", "SPEC"))
```

```
                Performance Measure(s)

             Measure     Value
1 sensitivity (SENS) 1.0000000
2 specificity (SPEC) 0.6020833
```

```
## More weight on specificity
optCutoff(pred = Bili, truth = ICUData$liver.failure,
          namePos = 1, perfMeasure = "WACC", wACC = 0.25)
```

```
Optimal Cut-off             WACC
     53.3000000        0.9078125
```

```
perfMeasures(pred = Bili, truth = ICUData$liver.failure,
             namePos = 1, cutoff = 53.3, measures = c("SENS", "SPEC"))
```

```
                Performance Measure(s)

             Measure     Value
1 sensitivity (SENS) 0.7500000
2 specificity (SPEC) 0.9604167
```

We obtain modified optimal cut-offs, which lead to a higher estimated sensitivity and higher estimated specificity, respectively. An increase in one criterion necessarily leads to a reduction in the other criterion, since the target criterion is the maximum of the weighted sum of sensitivity and specificity. The corresponding classification function is also called a decision stump ("decision stump") (Iba and Langley (1992)) and can be obtained using, for example, function `decisionStump` of package `"MKclass"` (Kohl (2020a)).

```
stump ← decisionStump(pred = Bili, truth = ICUData$liver.failure,
                      namePos = 1)
stump
```

```
Call:
decisionStump(pred = Bili, truth = ICUData$liver.failure, namePos = 1)
```

```
Cut-off:
Optimal Cut-off
          28.4

Performance:
   YJS
0.7396
```

We can then use it to calculate predictions for new data.

```
predict(stump, newdata = c(4.3, 10.1, 17.4, 28.4, 28.5, 71.0, 93.1))
```

```
[1] 0 0 0 0 1 1 1
Levels: 0 1
```

Note:
There are many other measures of accuracy that could be used for the calculations above. In package "MKclass" (Kohl (2020a)) there are currently 80 such measures implemented. All of these accuracy measures require dichotomization, which can also be seen critically (Harrell (2014, chapter 10)) and one should only use these cut-offs for rough guidance. In most cases, it is advisable to consider continuous classification functions. Such functions can be estimated using accuracy scores such as, for example, the AUC ("Area Under the receiver operating characteristic Curve") (Hanley and McNeil (1982)) or the Brier Score (Brier (1950)).

We consider bilirubin to be a continuous classification function for the prediction of liver failure and calculate the AUC and the Brier score. In the case of the Brier score, the data must be transformed to the interval [0, 1]. We use function perfScores from package "MKclass" (Kohl (2020a)) for this.

```
perfScores(pred = Bili, truth = ICUData$liver.failure, namePos = 1,
           scores = "AUC")
```

```
                Performance Score(s)

                   Score     Value
1 area under curve (AUC) 0.9372917
```

```
perfScores(pred = Bili, truth = ICUData$liver.failure, namePos = 1,
           scores = c("AUC", "BS"), transform = TRUE)
```

```
                Performance Score(s)

                   Score      Value
1 area under curve (AUC) 0.93729167
2       Brier score (BS) 0.02300764
```

An AUC of greater than 0.5 or a Brier score of less than 0.25 indicates that bilirubin contains information to assess liver failure. An AUC of 1.0 or a Brier Score of 0.0 represents an error-free classification. The above calculations confirm that bilirubin can be used to support the diagnosis of liver failure in intensive care patients, where the probability of liver failure increases with increasing bilirubin levels.

5.3 Confidence Intervals

In the previous section, we have learned about several estimating procedures and we now know that we should use unbiased (or at least consistent) and efficient estimators. However, these are only theoretical properties, which in practice can not tell us, how close our point estimator actually is to the considered unknown parameter. A possibility to further safeguard the point estimator, are so-called confidence intervals.

Definition 5.10 (Confidence interval)**.** *Let $\mathcal{P} = \{P_\theta \mid \theta \in \Theta\}$, $\Theta \subset \mathbb{R}^k$ ($k \in \mathbb{N}$), be some probability model. Furthermore, let $x_1, \ldots, x_n$ be realizations of independent and P_θ distributed random variables $X_1, \ldots, X_n$. Then, the* ***interval estimator***

$$\hat{I}(x_1, \ldots, x_n) = [S_u(x_1, \ldots, x_n),\ S_o(x_1, \ldots, x_n)] \tag{5.21}$$

*is called a $(1-\alpha)$-**confidence interval**, if*

$$P(\theta \in \hat{I}) \geq 1 - \alpha$$

for $\alpha \in (0, 1)$. Here, S_u and S_o are estimators for the lower and upper bound of the interval.

We give some additional explanations.

Remark 5.11. ***(a)*** *The definition also allows for one-sided confidence intervals. In this case, one boundary of the interval is free and only S_u or S_o is needed.*

(b) *It is said: A confidence interval covers the true unknown parameter with a probability of $1-\alpha$. This should express, that in 95% of the cases, in which some data is used to compute confidence intervals, these intervals will include the true unknown parameter. The statement, that the true unknown parameter lies in the computed confidence interval with 95% probability strictly speaking is wrong. Because after determining the confidence interval, the true unknown parameter either lies in the interval or not (the parameter is no random variable).*

(b) *We take a more detailed look at the components of a confidence interval. More concretely, they usually are of the following form*

$$\hat{I}(x_1, \ldots, x_n) = [S_n(x_1, \ldots, x_n) - k_1\sigma_{S_n},\ S_n(x_1, \ldots, x_n) + k_2\sigma_{S_n}] \tag{5.22}$$

The components are:

- *A point estimator S_n of the true unknown parameter θ.*
- *The standard deviation of the point estimator σ_{S_n}.*

- *Two constants $k_1, k_2 \in (0, \infty)$ usually depending on α, n and the distribution of S_n (depends on P_θ).*

Moreover, the following notions are used:

Confidence level: *the chosen coverage probability $1 - \alpha$ for the true unknown parameter θ.*

Basis: *point estimator of the true unknown parameter θ, often the center of the interval.*

Confidence bounds: *lower and upper bound of the interval.*

Maximum estimate error: *maximum distance between point estimator and the confidence bounds.*

We give some examples of confidence intervals.

Example 5.12. **(a)** We consider the normal distribution model $\mathcal{P} = \{\mathcal{N}(\mu, \sigma^2) \mid \mu \in \mathbb{R}\}$, where we assume $\sigma^2 \in (0, \infty)$ to be known. As we have learned in Section 5.2, the arithmetic mean is an unbiased and efficient estimator of μ. We assume that the observations $x_1, \ldots, x_n$ are realizations of independent and identical distributed random variables $X_1, \ldots, X_n$ with $X_i \sim \mathcal{N}(\mu, \sigma^2)$ $(i = 1, \ldots, n)$ and obtain

$$\mathrm{AM}(x_1, \ldots, x_n) \sim \mathcal{N}\left(\mu, \frac{1}{n}\sigma^2\right) \tag{5.23}$$

It follows, $\sigma_{S_n} = \frac{1}{\sqrt{n}}\sigma$, which is also called the **standard error** (SE) of the arithmetic mean (SEM). Because of the symmetry of the normal distribution, we get $k_1 = k_2$ and have to choose $k_1 = k_2 = z_{1-\alpha/2}$, the $(1-\alpha/2)$ quantile of the standard normal distribution. Consequentially, the $(1-\alpha)$ confidence interval reads

$$\mathrm{AM}(x_1, \ldots, x_n) \mp z_{1-\alpha/2}\frac{\sigma}{\sqrt{n}} \tag{5.24}$$

In practice, in most cases σ is also unknown and must be estimated, too. As we want to capture the true unknown value of μ, the unbiased sample variance $\tilde{S}$ (i.e. standardization $\frac{1}{n-1}$) is an appropriate candidate for the estimation, where

$$\frac{(n-1)\tilde{S}}{\sigma} \sim \mathrm{Chisq}(n-1) \tag{5.25}$$

That is, the additional estimation of σ leads us away from the normal distribution towards the t distribution with $n-1$ degrees of freedom (see also Remark 4.28 (b)). Thus, we get as confidence interval

$$\mathrm{AM}(x_1, \ldots, x_n) \mp t_{n-1;1-\alpha/2}\frac{\sqrt{\tilde{S}(x_1, \ldots, x_n)}}{\sqrt{n}} \tag{5.26}$$

where $t_{n-1;1-\alpha/2}$ is the $(1-\alpha/2)$ quantile of the t distribution with $n-1$ degrees of freedom.

(b) If we conversely consider the probability model $\mathcal{P} = \{\mathcal{N}(\mu, \sigma^2) \mid \sigma \in (0, \infty\}$, where we more realistically assume μ to be unknown, we obtain the following asymmetric $(1-\alpha)$ confidence interval for σ^2

$$\left[\frac{(n-1)\tilde{S}}{\chi^2_{n-1;1-\alpha/2}}, \frac{(n-1)\tilde{S}}{\chi^2_{n-1;\alpha/2}}\right] \tag{5.27}$$

Here, $\chi^2_{n-1;1-\alpha/2}$ and $\chi^2_{n-1;\alpha/2}$ are the $(1-\alpha/2)$ and the $\alpha/2$ quantile of the χ^2 distribution with $n-1$ degrees of freedom, respectively.

Note:
The above confidence intervals are not only of interest for the normal distribution model, but may also be used as approximations for other probability models. The reason for it is the central limit theorem, which states that the distribution of the arithmetic mean of quite arbitrary independent and identical distributed random variables converges with increasing sample size n towards a normal distribution.

In addition to the point estimates for the maximum body temperature of our ICU patients (cf. Section 5.2), we will now determine 95% confidence intervals (i.e. $\alpha = 0.05$). The confidence interval of the mean μ in case of an unknown standard deviation σ can be determined by function `meanCI` of package `"MKinfer"` (Kohl (2022b)).

```
meanCI(ICUData$temperature[-398])
```

```
        Exact confidence interval(s)

95 percent confidence interval:
        2.5 %   97.5 %
mean 37.61725 37.82363

sample estimates:
     mean         sd
37.720441  1.173187

additional information:
SE of mean
0.05251908
```

The reported interval should be chosen in dependence of the accuracy of the temperature measurement, e.g. $[37.61, 37.83]$ or $[37.60, 37.85]$ or $[37.6, 37.9]$ might be appropriate. Each interval covers the true unknown mean with at least 95% probability. The confidence interval of the arithmetic mean can also be computed by means of function `t.test`. This function can be used for computing t tests, which will be introduced in Chapter 6. However, at this point we only take a look at the confidence interval (`conf.int`) and ignore the remaining results.

```
t.test(ICUData$temperature[-398])$conf.int
```

```
[1] 37.61725 37.82363
attr(,"conf.level")
[1] 0.95
```

As the sample size is quite large in our example, we could, in sense of the central limit theorem, use the quantile of the standard normal distribution instead of the quantile of the t-distribution with $499 - 1$ degrees of freedom. We compare the two quantiles

```
qt(1-0.05/2, df = 499-1)
```

```
[1] 1.964739
```

```
qnorm(1-0.05/2)
```

```
[1] 1.959964
```

and get a difference of less than 0.005. Consequentially, the confidence bounds of the approximative interval are very similar and the differences probably lie below the measurement accuracy.

Based on ML estimators, we can determine a similar approximative confidence interval. We apply function `fitdistr` of package `"MASS"` (Venables and Ripley (2002)) combined with function `confint`.

```
## ML estimator
ML <- fitdistr(ICUData$temperature[-398], densfun = "normal")
## Approximate confidence interval.
confint(ML)
```

```
          2.5 %    97.5 %
mean 37.617609 37.823273
sd    1.099298  1.244725
```

We obtain both an approximate confidence interval for the mean μ as well as for the standard deviation σ. We can also determine these intervals using function `MLEstimator` from package `"distrMod"` (Kohl and Ruckdeschel (2010)) and function `confint`.

```
## Model
Model <- NormLocationScaleFamily()
## ML estimator
ML2 <- MLEstimator(ICUData$temperature[-398], Model)
## Approximate confidence interval
confint(ML2)
```

```
A[n] asymptotic (LAN-based) confidence interval:
          2.5 %    97.5 %
mean 37.617609 37.823273
sd    1.099298  1.244725
```

We compare the approximative confidence interval for the standard deviation, which is symmetric around the ML estimator for the standard deviation, with the asymmetric interval, which we obtain using the χ^2 distribution and functions `sdCI` and `normCI` from package `"MKinfer"` (Kohl (2022b)).

```
## standard deviation only
sdCI(ICUData$temperature[-398])
```

```
        Exact confidence interval(s)

95 percent confidence interval:
      2.5 %   97.5 %
sd 1.104636 1.250879

sample estimates:
     mean        sd
37.720441  1.173187

additional information:
SE of mean
0.05251908
```

```
## mean value and standard deviation
normCI(ICUData$temperature[-398])
```

```
        Exact confidence interval(s)

95 percent confidence intervals:
         2.5 %    97.5 %
mean 37.617255 37.823627
sd    1.104636  1.250879

sample estimates:
     mean        sd
37.720441  1.173187

additional information:
SE of mean
0.05251908
```

The asymmetric confidence interval is slightly different, but the differences are only in the range of permilles.

Note:
If the sample size n is not too small, one can use the approximative confidence intervals emerging from the central limit theorem. Figure 5.2 shows the ratio between the 95% quantile of the t distribution with increasing degrees of freedom and the 95% quantile of the standard normal distribution. From a sample size of about 25 onwards, the difference between the quantiles and thus between the maximum estimate errors is below 5%.

In addition to the approximative intervals based on the normal distribution, there is also a very general data-based approach to calculate confidence intervals, called **bootstrap**. In classical bootstrapping in its simplest form, n (= sample size) observations are drawn from the sample **with replacement**. This random process is repeated B times ($B \in \mathbb{N}$) and for each of these bootstrap samples the corresponding

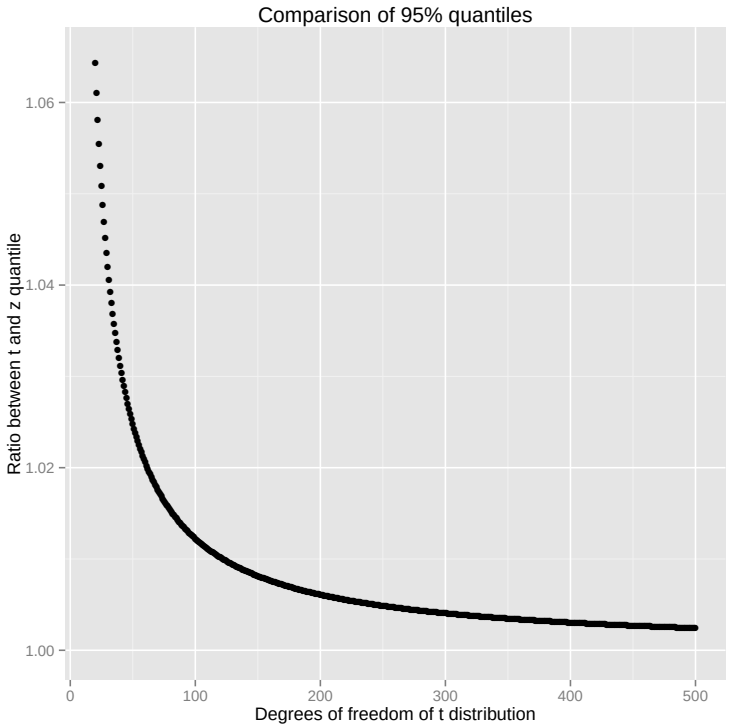


Figure 5.2: Ratio between 95% quantiles of t and standard normal distribution.

estimators are calculated. Descriptively spoken, the sample takes the place of the population and the bootstrap samples take the place of the original sample. If the original sample is representative, then so are the bootstrap samples due to random selection. Each bootstrap sample therefore corresponds to a possible representative sample, that could have been drawn from the population. From the bootstrap samples we consequently obtain many possible estimation results for the population. These estimation results give us a picture of the variability of the estimation results (variance of the estimator) and we can then obtain (without or with few special model assumptions!) a confidence interval for the estimator and for the searched parameter. Bootstrapping is therefore also referred to as a non-parametric method; which means that we do not necessarily need a concrete parametric model for the confidence interval.

Note:
In the case of $n = 20$ and $B = 2000$, the probability of obtaining two or more equal samples is less than 5% (Section 8.1 in Chernick and LaBudde (2011)). The recommended number of bootstrap repetitions start at a few hundred repetitions ($B = 399$ or $B = 599$) and end at values of $B = 10000$ or even $B = 100000$. For R functions that use bootstrap or other simulation techniques, one often finds default values of $B = 999$ or $B = 9999$. In the case of confidence intervals, values of $B = 999$ or greater are recommended. If the calculation time is low or does not matter, $B = 9999$ should give very good and stable results in most of the cases.

The corresponding functionality is implemented in the recommended package `"boot"` (Canty and Ripley (2021)). Function `boot.ci` can compute five different types of bootstrap confidence intervals, namely `"norm"`, `"basic"`, `"stud"`, `"perc"` and `"bca"`. The simplest of the intervals is the percentile interval

("perc"). Here, the confidence interval consists of the empiric quantiles ($\alpha/2$ and $(1-\alpha/2)$ of the bootstrap estimates. The BCa method is a corrected and adjusted variant (bias corrected and accelerated) of the percentile method, which provides the best results in many cases. The method "norm" is based on the formula (5.24) in which mean and standard deviation are calculated from the bootstrap estimates. In the case of the method "basic" the normal approximation (5.24) is used, this time for the bootstrap estimates centered at the mean of the original sample. Finally, the "stud" method uses the bootstrap estimates which are centered at the mean of the original sample and standardized with the standard deviation of the estimator (this is also known as studentizing). We could also call it a bootstrapped z-score; which means that one has to know the standard deviation (or variance) of the estimator and has to calculate it for each bootstrap sample. The method "stud" is therefore also based on the normal approximation. Due to the additional standardization (at least in theory) a reduction of the error is achieved (order of magnitude of the error $O(n^{-1})$ instead of $O(n^{-1/2})$ as in the case of type "basic"). For more details on bootstrap confidence intervals, we refer to chapter 5 of Davison and Hinkley (1997) and chapter 3 of Chernick and LaBudde (2011).

Functions normCI, meanCI and sdCI from package "MKinfer" (Kohl (2022b)) contain simplified interfaces to the functions of package "boot" (Canty and Ripley (2021)).

```
normCI(ICUData$temperature[-398], boot = TRUE)
```

```
        Bootstrap confidence interval(s)

$mean
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 9999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, conf = 1 - alpha, type = bootci.type)

Intervals :
Level      Normal              Basic             Studentized
95%   (37.62, 37.82 )   (37.62, 37.82 )   (37.62, 37.82 )

Level     Percentile            BCa
95%   (37.62, 37.82 )   (37.62, 37.82 )
Calculations and Intervals on Original Scale

$`standard deviation`
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 9999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, conf = 1 - alpha, type = bootci.type)

Intervals :
Level      Normal              Basic             Studentized
95%   ( 1.096,  1.253 )   ( 1.095,  1.254 )   ( 1.100,  1.260 )
```

```
Level     Percentile            BCa
95%   ( 1.092,  1.251 )   ( 1.100,  1.258 )
Calculations and Intervals on Original Scale


sample estimates:
     mean         sd
37.720441  1.173187
```

The results are very similar to the previous results.

> Note:
> For small to moderate sample sizes $n \leq 50$, one must assume, that asymptotic confidence intervals based on the central limit theorem must be regarded as rough approximations. Bootstrapping works very well even for small sample sizes down to $n = 10$ or even somewhat smaller. However, one should also be aware here that Bootstrap confidence intervals for small to moderate numbers of cases ($10 \leq n \leq 50$) and skewed distributions tend to be somewhat too short. In this case Chernick and LaBudde (2011) in section 3.7 recommend to use type `"stud"`.

In the following example we discuss the Bernoulli model.

Example 5.13. We consider the probability model $\mathcal{P} = \{\text{Bernoulli}(p) \mid p \in (0,1)\}$. As we have learned in Section 5.2, the relative frequency $\hat{p}$ is the ML estimator of p and is unbiased and efficient. As the Bernoulli distribution is a discrete distribution, the distribution of $\hat{p}$ is also discrete and quantiles of discrete distribution are not necessarily unique. Consequentially, there is a whole series of proposals for "exact" confidence intervals for the probability p; for example the Clopper-Pearson or the Agresti-Coull interval. I omit the explicit specification of the formulas.
An application of the central limit theorem yields the following approximative confidence interval for p

$$\left(\hat{p} \mp \tfrac{1}{2n}\right) \mp z_{1-\alpha/2}\sqrt{\frac{\hat{p}(1-\hat{p})}{n}} \tag{5.28}$$

The correction term $\frac{1}{2n}$ is called **continuity correction** and improves the approximation. Furthermore, $z_{1-\alpha/2}$ is the $(1-\alpha/2)$ quantile of the standard normal distribution. There exist several rules of thumb to check whether the aymptotic interval is applicable. One of them is

$$n\hat{p} > 5 \qquad \text{and} \qquad n(1-\hat{p}) > 5 \tag{5.29}$$

i.e., the more $\hat{p}$ approaches 0 or 1, the larger the sample size has to be.
If we consider drawing without replacement and the underlying population is small having $N \in \mathbb{N}$ members, it is recommended, to apply the following slightly modified confidence interval

$$\left(\hat{p} \mp \tfrac{1}{2n}\right) \mp z_{1-\alpha/2}\sqrt{\frac{\hat{p}(1-\hat{p})}{n}\frac{N-n}{N-1}} \tag{5.30}$$

The additional factor $\frac{N-n}{N-1}$, which we have already met in Remark 4.8 (c), is called **finite-population correction** and represents the difference between drawing with and without replacement.

We consider the prevalence of liver failure on the ICU and additionally safeguard the estimation by a confidence interval. There are several packages including functions for computing "exact" confidence intervals. We will apply function `binomCI` of package `"MKinfer"` (Kohl (2022b)). We compute the Wilson, the Clopper-Pearson and the Agresti-Coull interval. For this, we need the number of patients with liver failure as well as the total number of patients.

```
## Frequency of liver failure
table(ICUData$liver.failure)
```

```
  0   1
480  20
```

```
## Wilson interval
binomCI(x = 20, n = 500)
```

```
        wilson confidence interval

95 percent confidence interval:
         2.5 %    97.5 %
prob 0.0260408 0.0609736

sample estimate:
prob
0.04

additional information:
standard error of prob
           0.008911592
```

```
## Clopper-Pearson interval
binomCI(x = 20, n = 500, method = "clopper-pearson")
```

```
        clopper-pearson confidence interval

95 percent confidence interval:
          2.5 %     97.5 %
prob 0.02460131 0.06110261

sample estimate:
prob
0.04
```

```
## Agresti-Coull interval
binomCI(x = 20, n = 500, method = "agresti-coull")
```

```
        agresti-coull confidence interval

95 percent confidence interval:
          2.5 %    97.5 %
prob 0.02569479 0.0613196

sample estimate:
     prob
0.0435072

additional information:
standard error of prob
           0.009088128
```

We get minor differences between the intervals, in particular, the Agresti-Coull interval is not based on the relative frequency. The asymptotic interval with and without continuity correction we can also calculate with the help of the function `binomCI`.

```
## without continuity correction
binomCI(x = 20, n = 500, method = "wald")
```

```
        wald confidence interval

95 percent confidence interval:
          2.5 %     97.5 %
prob 0.02282374 0.05717626

sample estimate:
prob
0.04

additional information:
standard error of prob
           0.008763561
```

```
## with continuity correction
binomCI(x = 20, n = 500, method = "wald-cc")
```

```
        wald-cc confidence interval

95 percent confidence interval:
          2.5 %     97.5 %
prob 0.02182374 0.05817626

sample estimate:
prob
```

```
0.04

additional information:
standard error of prob
           0.008763561
```

Finally, the function `binomCI` can also be used to calculate bootstrap confidence intervals.

```
1 binomCI(x = 20, n = 500, method = "boot")
```

```
        boot confidence interval

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 9999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, conf = 1 - alpha, type = bootci.type)

Intervals :
Level      Normal              Basic             Studentized
95%   ( 0.0230,  0.0572 )   ( 0.0220,  0.0560 )   ( 0.0249,  0.0605 )

Level     Percentile            BCa
95%   ( 0.024,  0.058 )   ( 0.024,  0.058 )
Calculations and Intervals on Original Scale

sample estimate:
prob
0.04

additional information:
          standard error of prob bootstrap standard error of prob
                     0.008763561                       0.008711646
```

We can also create an asymptotic confidence interval using the function `MLEstimator` from the package `"distrMod"` (Kohl and Ruckdeschel (2010)) and the function `confint`. We define the Bernoulli model using the function `BinomFamily` and the parameter value `size = 1`.

```
1 ## Bernoulli model
2 Model ← BinomFamily(size = 1)
3 ## ML Estimator
4 MLp ← MLEstimator(ICUData$liver.failure , Model)
5 MLp
```

```
Evaluations of Maximum likelihood estimate:
-------------------------------------------

  0.040000000
 (0.008763561)
```

```
## confidence interval
confint(MLp)
```

```
A[n] asymptotic (LAN-based) confidence interval:
          2.5 %     97.5 %
[1,] 0.02282374 0.05717626
```

The result corresponds to the asymptotic confidence interval above without continuity correction. Roughly summarized, we can assume a prevalence of liver failure on the ICU in the range from 2.2% to 6.1% with relatively high certainty.

Note:
As there is often more than one way to determine a confidence interval of a certain parameter, it is recommended to specify not only the interval but also the type of the interval in practice. Only by doing this, a reader can reproduce the analysis and its results.

A very interesting option to describe the location and scale of data are median and MAD. On the one hand, both estimators are very robust, on the other hand, it is not necessary to assume a specific parametric family.

Example 5.14. Let $x_{(1)}, x_{(2)}, \dots, x_{(n)}$ be the increasingly sorted observations. Then, the $(1-\alpha)$ confidence interval of the median reads

$$\left[x_{(k)}, x_{(n-k+1)}\right] \tag{5.31}$$

where $k \in \mathbb{N}$ has to be determined, such that the following inequality holds

$$1 - 2\sum_{i=1}^{k-1} \binom{n}{i} 0.5^n \geq 1 - \alpha \tag{5.32}$$

This approach can be transferred to the MAD by considering it as the median of $|x_1 - M|, \dots, |x_n - M|$ with $M = \text{median}\,(x_1, \dots, x_n)$. In case of the normal distribution, the MAD is usually standardized by 1.4826 to yield a consistent estimator of the standard deviation; see equation (2.3).

We consider the maximum body temperature of our ICU patients and determine 95% confidence intervals for median and MAD. For this, we apply function `medianCI` of package `"MKinfer"` (Kohl (2022b)). In case of the MAD, we choose the version that is standardized with 1.4826.

```
## Exact confidence interval for the median.
medianCI(ICUData$temperature)
```

```

        exact confidence interval

95 percent confidence interval:
       2.5 % 97.5 %
median  37.5   37.8

sample estimate:
median
  37.7
```

```
## Exact confidence interval for the MAD
madCI(ICUData$temperature)
```

```
        exact confidence interval

95 percent confidence interval:
      2.5 %  97.5 %
MAD 1.03782 1.33434

sample estimate:
    MAD
1.18608
```

Since the sample size in our example is quite large, we may instead turn to the asymptotic confidence interval. We obtain

```
## Asymptotic confidence interval for the median.
medianCI(ICUData$temperature, method = "asymptotic")
```

```
        asymptotic confidence interval

95 percent confidence interval:
       2.5 % 97.5 %
median  37.5   37.8

sample estimate:
median
  37.7
```

```
## Asymptotic confidence interval for the MAD
madCI(ICUData$temperature, method = "asymptotic")
```

```
        asymptotic confidence interval

95 percent confidence interval:
      2.5 %  97.5 %
MAD 1.03782 1.33434

sample estimate:
    MAD
1.18608
```

We obtain identical results as in case of the exact intervals. As an alternative to the asymptotic intervals we can again use bootstrap confidence intervals, which we also calculate with functions `medianCI` and `madCI` from package `"MKinfer"` (Kohl (2022b)).

```
## bootstrap confidence interval for the median.
medianCI(ICUData$temperature, method = "boot")
```

```
        bootstrap confidence interval

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 9999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, conf = 1 - alpha, type = bootci.type)

Intervals :
Level      Normal              Basic
95%   (37.55, 37.91 )   (37.60, 37.90 )

Level     Percentile            BCa
95%   (37.5, 37.8 )   (37.5, 37.8 )
Calculations and Intervals on Original Scale

sample estimate:
median
  37.7
```

```
## bootstrap confidence interval for the MAD.
madCI(ICUData$temperature, method = "boot")
```

```
        bootstrap confidence interval

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 9999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, conf = 1 - alpha, type = bootci.type)

Intervals :
Level      Normal              Basic
95%   ( 1.083,  1.377 )   ( 1.038,  1.334 )

Level     Percentile            BCa
95%   ( 1.038,  1.334 )   ( 1.038,  1.186 )
Calculations and Intervals on Original Scale

sample estimate:
    MAD
1.18608
```

The bootstrap intervals are also very similar to the exact and asymptotic intervals, which is not surprising given the large sample size.

Overall, the intervals are somewhat longer than in the case of the arithmetic mean and the (sample) standard deviation. This is the price we have to pay for the non-parametric nature of these estimation procedures and their robustness.

In the following, we take a look at the MD estimators. Here, the computation of confidence intervals is rather difficult, as the (exact and asymptotic) distribution of these estimators is quite hard to determine. In case of the CvM-MD estimator, we can compute an asymptotic confidence interval by means of function `MDEstimator` of package `"distrMod"` (Kohl and Ruckdeschel (2010)) and function `confint`. First, we again consider the maximum body temperature of our ICU patients.

```
## Model
Model ← NormLocationScaleFamily()
## CvM-MD estimator including variance.
MD ← CvMMDEstimator(ICUData$temperature, Model)
## 95% confidence interval
confint(MD)
```

```
A[n] asymptotic (LAN-based) confidence interval:
          2.5 %    97.5 %
mean 37.535623 37.807908
sd    1.051812  1.227027
```

The confidence intervals are slightly longer than in case of the ML estimator, but we did not have to exclude patient 398 due to the robustness of the CvM-MD estimator.

In a similar fashion, we can also compute the confidence interval for the prevalence of liver failure on the ICU.

```
## Model
Model ← BinomFamily(size = 1)
## CvM-MD estimator incl. variance.
MDp ← CvMMDEstimator(ICUData$liver.failure, Model)
## 95% confidence interval
confint(MDp)
```

```
A[n] asymptotic (LAN-based) confidence interval:
           2.5 %     97.5 %
prob 0.02282334 0.05717567
```

The results are almost identical to the ML estimator. In the case of MD estimators, one can use bootstrap confidence intervals as well. However, we are not aware of any function that does this directly. We use the functions `boot` and `boot.ci` from the package `"boot"` (Canty and Ripley (2021)). We show this for the KS-MD estimator, for which this is of particular interest, since neither an exact nor an asymptotic confidence interval is known. First, we need a function which calculates the estimate and returns it as a scalar (number). The function has two parameters. The first parameter represents the numeric vector

containing the complete data. The second parameter represents the indices of the entries of the numeric vector, which were selected for the respective bootstrap sample.

```
## x: vector of observations
## i: vector of indices of bootstrap sample.
KSMDEst ← function(x, i){
  estimate(KolmogorovMDEstimator(x[i], ParamFamily = NormLocationScaleFamily()))
}
```

Using this function, we first compute the KS-MD estimator for 999 bootstrap samples. The calculations take several minutes. We then use these estimates to determine the bootstrap confidence intervals.

```
## bootstrap estimates
boot.out ← boot(ICUData$temperature, statistic = KSMDEst, R = 999)
## bootstrap confidence intervals
boot.ci(boot.out, index = 1)
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, index = 1)

Intervals :
Level      Normal              Basic
95%   (37.58, 37.79 )   (37.58, 37.79 )

Level     Percentile            BCa
95%   (37.56, 37.78 )   (37.57, 37.79 )
Calculations and Intervals on Original Scale
```

```
boot.ci(boot.out, index = 2)
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, index = 2)

Intervals :
Level      Normal              Basic
95%   ( 1.050,  1.252 )   ( 1.054,  1.265 )

Level     Percentile            BCa
95%   ( 1.021,  1.233 )   ( 1.037,  1.245 )
Calculations and Intervals on Original Scale
```

We obtain very similar results to the other estimators.

Finally, let us apply the RMX estimators to the ICU data. We first compute the RMX estimator for the temperature data using function `roblox` from package `"RobLox"` (Kohl (2019)) and the same settings as in Section 5.2. After that we can use function `confint` to calculate the corresponding asymptotic confidence interval.

```
ALest ← roblox(ICUData$temperature, eps.lower = 1/500,
                eps.upper = 0.05, k = 3)
distrMod::confint(ALest)
```

```
A[n] asymptotic (LAN-based) confidence interval:
         2.5 %   97.5 %
mean 37.535840 37.75436
sd    1.058808  1.22174
```

The confidence interval above ignores a possible bias that is inevitable when considering the described contamination neighborhoods around the parametric model. We can also calculate confidence intervals that take the maximum (asymptotic) bias into account.

```
distrMod::confint(ALest, symmetricBias())
```

```
A[n] asymptotic (LAN-based), uniform (bias-aware)
 confidence interval:
for symmetric Bias
         2.5 %   97.5 %
mean 37.478399 37.81180
sd    1.015979  1.26457
```

We can see that the confidence intervals have become somewhat longer, so more conservative. Alternatively, we can also use the `rmx` function from the package `"rmx"` (Kohl (2022c)). Again, we can use the function `confint` to output the corresponding confidence intervals.

```
RMXest ← rmx(ICUData$temperature, model = "norm",
              eps.lower = 1/500, eps.upper = 0.05)
confint(RMXest)
```

```

 Asymptotic (LAN-based) confidence interval

95 percent confidence intervals:
         2.5 %    97.5 %
mean 37.532948 37.752852
sd    1.057552  1.221479

RMX estimates:
     mean         sd
37.642900  1.139516
```

```
confint(RMXest, method = "as.bias")
```

```
 Asymptotic (LAN-based), uniform (bias-aware) confidence interval

95 percent confidence intervals:
         2.5 %    97.5 %
mean 37.472648 37.813151
sd    1.012602  1.266429

RMX estimates:
     mean         sd
37.642900  1.139516
```

The possibility of bootstrap confidence intervals is integrated in the package "rmx".

```
confint(RMXest, method = "boot")
```

```
 Bootstrap confidence interval

$mean
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 9999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, index = c(1, 3))

Intervals :
Level      Normal              Basic             Studentized
95%   (37.47, 37.67 )   (37.47, 37.67 )   (37.46, 37.67 )

Level     Percentile            BCa
95%   (37.61, 37.82 )   (37.53, 37.67 )
Calculations and Intervals on Original Scale
Warning : BCa Intervals used Extreme Quantiles
Some BCa intervals may be unstable

$sd
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 9999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, index = c(2, 4))

Intervals :
Level      Normal              Basic             Studentized
95%   ( 1.020,  1.187 )   ( 1.021,  1.188 )   ( 1.016,  1.197 )

Level     Percentile            BCa
```

```
95%   ( 1.091,  1.258 )   ( 1.027,  1.187 )
Calculations and Intervals on Original Scale
Some BCa intervals may be unstable


RMX estimates:
     mean         sd
37.642900  1.139516
```

We can also compute bootstrap confidence intervals by using functions `boot` and `boot.ci` from the package `"boot"` (Canty and Ripley (2021)). The following function performs the necessary calculations for the bootstrap samples, where we not only get the estimator, but also the asymptotic variance of the estimator. The return value of the function is therefore a vector of length four, whose first two entries are the estimated values for mean and SD and whose third and fourth entries are the asymptotic variance for these two estimates. The function has the same parameters as the function we defined above for the KS-MD estimator.

```
## x: vector of observations
## i: vector of indices of bootstrap sample.
RMXEst ← function(x, i){
  res ← roblox(x[i], eps.lower = 1/500, eps.upper = 0.05, k = 3)
  c(estimate(res), diag(asvar(res)))
}
```

We determine the bootstrap confidence intervals. In the case of the studentized interval, the asymptotic variance is used for the calculation.

```
## bootstrap estimates
boot.out ← boot(ICUData$temperature, statistic = RMXEst, R = 999)
## bootstrap confidence interval for the mean value.
boot.ci(boot.out, index = c(1,3))
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, index = c(1, 3))

Intervals :
Level      Normal              Basic             Studentized
95%   (37.53, 37.76 )   (37.54, 37.76 )   (37.54, 37.76 )

Level     Percentile            BCa
95%   (37.53, 37.75 )   (37.53, 37.75 )
Calculations and Intervals on Original Scale
```

```
## bootstrap confidence interval for the standard deviation
boot.ci(boot.out, index = c(2,4))
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, index = c(2, 4))

Intervals :
Level      Normal              Basic              Studentized
95%   ( 1.061,  1.226 )   ( 1.059,  1.230 )   ( 1.065,  1.238 )

Level     Percentile            BCa
95%   ( 1.050,  1.221 )   ( 1.061,  1.227 )
Calculations and Intervals on Original Scale
```

The results are very similar to the asymptotic interval without considering the bias.

> Note:
> Beside the introduced options, there are many more possibilities to compute confidence intervals in R. In particular, confidence intervals are usually determined during the computation of statistical tests, which will be introduced in Chapter 6.

To illustrate the statistical procedures presented in this chapter, we will in the following repeat the steps for estimating the parameters, validating the models and calculating the corresponding confidence intervals using an example. For this purpose, we use the SAPS-II score of patients who were discharged home. We assume that we can describe the data by a gamma distribution. We first create a corresponding data subset to simplify the individual steps.

```
1 ICUData.home ← ICUData[ICUData$outcome == "home",]
```

As estimators, we use the ML, the CvM-MD, and the RMX estimator. In the case of the RMX estimator, we use function `roptest` from package `"ROptEst"` (Kohl and Ruckdeschel (2019)) and assume 0% to 5% errors, as is usually the case for routine data.

```
1 Model ← GammaFamily()
2 MLest ← MLEstimator(ICUData.home$SAPS.II, Model)
3 MLest
```

```
Evaluations of Maximum likelihood estimate:
-------------------------------------------
     scale        shape
  5.5804692   6.9961941
 (0.6131542) (0.7414555)
```

```
1 MDest ← CvMMDEstimator(ICUData.home$SAPS.II, Model)
2 MDest
```

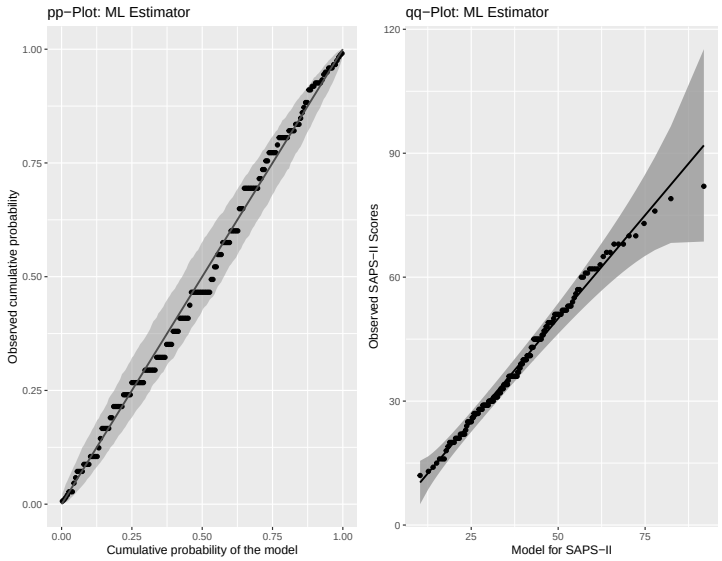
```
Evaluations of Minimum CvM distance estimate ( mu = model distr. ) :
--------------------------------------------------------------------
     scale        shape
  6.0324283   6.4628936
 (0.8239082) (0.8509984)
```

```
RMXest ← roptest(ICUData.home$SAPS.II, Model,
                 eps.lower = 0, eps.upper = 0.05, steps = 3)
RMXest
```

```
Evaluations of 3-step estimate:
-------------------------------
     scale        shape
  5.9019056   6.6572947
 (0.7183910) (0.7803873)
```

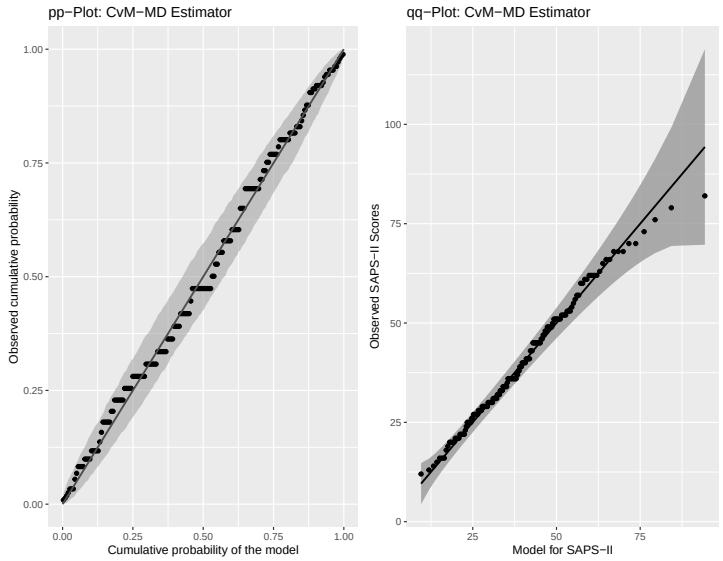
We use pp- and qq-plots to validate the estimated models. We start with the ML estimator.

```
gg1 ← ggplot(ICUData.home, aes(sample = SAPS.II)) +
  qqplotr::stat_pp_band(dparams = list(scale = 5.58, shape = 7.0),
                        distribution = "gamma") +
  qqplotr::stat_pp_point(dparams = list(scale = 5.58, shape = 7.0),
                         distribution = "gamma") +
  qqplotr::stat_pp_line() +
  xlab("Cumulative probability of the model") +
  ylab("Observed cumulative probability") +
  ggtitle("pp-Plot: ML Estimator")
gg2 ← ggplot(ICUData.home, aes(sample = SAPS.II)) +
  qqplotr::stat_qq_band(dparams = list(scale = 5.58, shape = 7.0),
                        distribution = "gamma", identity = TRUE) +
  qqplotr::stat_qq_point(dparams = list(scale = 5.58, shape = 7.0),
                         distribution = "gamma") +
  qqplotr::stat_qq_line(dparams = list(scale = 5.58, shape = 7.0),
                        distribution = "gamma", identity = TRUE) +
  xlab("Model for SAPS-II") +
  ylab("Observed SAPS-II Scores") +
  ggtitle("qq-Plot: ML Estimator")
grid.arrange(gg1, gg2, nrow = 1)
```



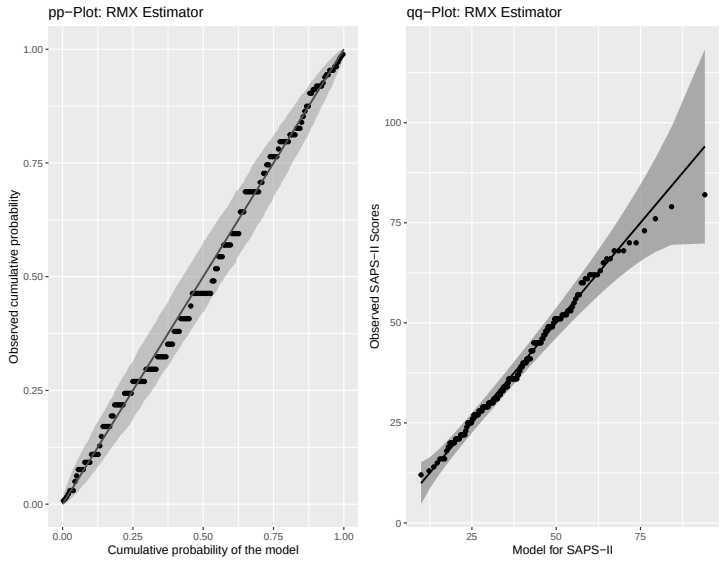
We obtain a good agreement between data and model. We next examine the CvM-MD estimator.

```
gg1 ← ggplot(ICUData.home, aes(sample = SAPS.II)) +
  qqplotr::stat_pp_band(dparams = list(scale = 6.03, shape = 6.46),
                        distribution = "gamma") +
  qqplotr::stat_pp_point(dparams = list(scale = 6.03, shape = 6.46),
                         distribution = "gamma") +
  qqplotr::stat_pp_line() +
  xlab("Cumulative probability of the model") +
  ylab("Observed cumulative probability") +
  ggtitle("pp−Plot: CvM−MD Estimator")
gg2 ← ggplot(ICUData.home, aes(sample = SAPS.II)) +
  qqplotr::stat_qq_band(dparams = list(scale = 6.03, shape = 6.46),
                        distribution = "gamma", identity = TRUE) +
  qqplotr::stat_qq_point(dparams = list(scale = 6.03, shape = 6.46),
                         distribution = "gamma") +
  qqplotr::stat_qq_line(dparams = list(scale = 6.03, shape = 6.46),
                        distribution = "gamma", identity = TRUE) +
  xlab("Model for SAPS−II") +
  ylab("Observed SAPS−II Scores") +
  ggtitle("qq−Plot: CvM−MD Estimator")
grid.arrange(gg1, gg2, nrow = 1)
```



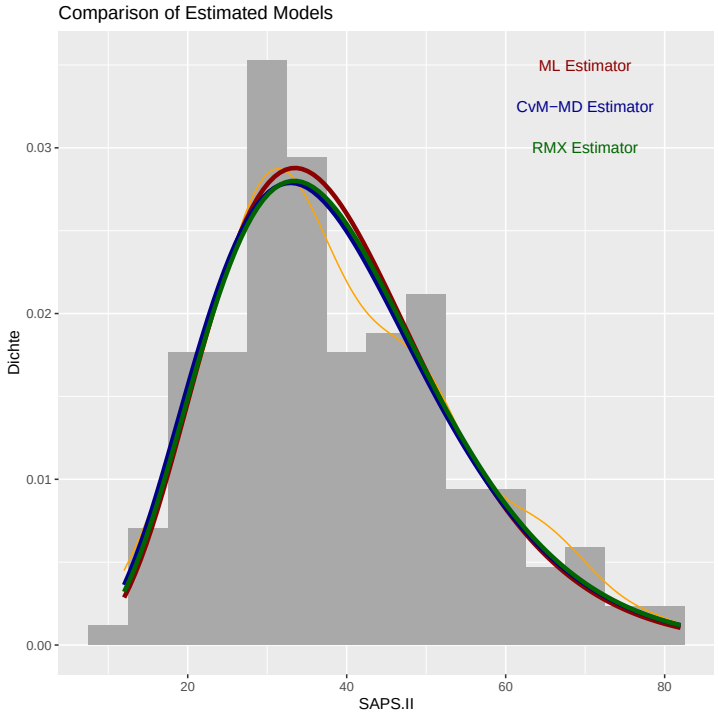
Again, we see a good fit between the data and the model. We consider finally the RMX estimator.

```
gg1 ← ggplot(ICUData.home, aes(sample = SAPS.II)) +
  qqplotr::stat_pp_band(dparams = list(scale = 5.90, shape = 6.66),
                        distribution = "gamma") +
  qqplotr::stat_pp_point(dparams = list(scale = 5.90, shape = 6.66),
                         distribution = "gamma") +
  qqplotr::stat_pp_line() +
  xlab("Cumulative probability of the model") +
  ylab("Observed cumulative probability") +
  ggtitle("pp−Plot: RMX Estimator")
gg2 ← ggplot(ICUData.home, aes(sample = SAPS.II)) +
  qqplotr::stat_qq_band(dparams = list(scale = 5.90, shape = 6.66),
                        distribution = "gamma", identity = TRUE) +
  qqplotr::stat_qq_point(dparams = list(scale = 5.90, shape = 6.66),
                         distribution = "gamma") +
  qqplotr::stat_qq_line(dparams = list(scale = 5.90, shape = 6.66),
                        distribution = "gamma", identity = TRUE) +
  xlab("Model for SAPS−II") +
  ylab("Observed SAPS−II Scores") +
  ggtitle("qq−Plot: RMX Estimator")
grid.arrange(gg1, gg2, nrow = 1)
```



Also in the case of the RMX estimator, we see a good agreement between the estimated model and the data. Overall, we see only minor differences between the three estimators. In particular, since we see no differences between the ML estimator and the two robust estimators, we can interpret this as a sign that the data do not contain any noticeable deviations from the assumed model. We plot the densities of the estimated models.

```
ggplot(ICUData.home, aes(x=SAPS.II)) +
  geom_histogram(aes(y=..density..), binwidth = 5, fill = "darkgrey") +
  geom_density(color = "orange") + ylab("Dichte") +
  stat_function(fun = dgamma, args = list(scale = 5.58, shape = 7.0),
                color = "darkred", lwd = 1.5) +
  stat_function(fun = dgamma, args = list(scale = 6.03, shape = 6.46),
                color = "darkblue", lwd = 1.5) +
  stat_function(fun = dgamma, args = list(scale = 5.90, shape = 6.66),
                color = "darkgreen", lwd = 1.5) +
  annotate("text", x = 70, y = 0.035, col = "darkred",
           label = "ML Estimator") +
  annotate("text", x = 70, y = 0.0325, col = "darkblue",
           label = "CvM-MD Estimator") +
  annotate("text", x = 70, y = 0.030, col = "darkgreen",
           label = "RMX Estimator") +
  ggtitle("Comparison of Estimated Models")
```



Again, there is a high degree of similarity between the estimated models. We calculate the asymptotic confidence intervals for the three estimators.

```
distrMod::confint(MLest)
```

```
A[n] asymptotic (LAN-based) confidence interval:
          2.5 %    97.5 %
scale 4.378709 6.782229
shape 5.542968 8.449420
```

```
distrMod::confint(MDest)
```

```
A[n] asymptotic (LAN-based) confidence interval:
          2.5 %    97.5 %
scale 4.417598 7.647259
shape 4.794967 8.130820
```

```
distrMod::confint(RMXest)
```

```
A[n] asymptotic (LAN-based) confidence interval:
          2.5 %    97.5 %
scale 4.493885 7.309926
shape 5.127764 8.186826
```

The confidence intervals overlap clearly. We compute the corresponding bootstrap confidence intervals for ML and CvM-MD estimator. In the case of the RMX estimator, we omit the calculations due to the high computational time. However, we give the corresponding R code for it. We first define the

functions with which we can determine the estimators and the (asymptotic) variances of the estimators for the bootstrap samples.

```
## x: vector of observations
## i: vector of indices of bootstrap sample.
MLEst ← function(x, i){
  res ← MLEstimator(x[i], ParamFamily = GammaFamily())
  c(estimate(res), diag(asvar(res)))
}
MDEst ← function(x, i){
  res ← CvMMDEstimator(x[i], ParamFamily = GammaFamily())
  c(estimate(res), diag(asvar(res)))
}
RMXEst ← function(x, i){
  res ← roptest(x[i], ParamFamily = GammaFamily(),
                eps.lower = 0, eps.upper = 0.05, steps = 3)
  c(estimate(res), diag(asvar(res)))
}
```

Since the calculations will take some time, we want to accelerate them by parallelization. This is also possible with the function `boot`. First, we determine with the help of the function `detectCores` from package `"parallel"` (R Core Team (2022a)) the number of available CPU cores. We then use all but one core for the calculation.

```
nr.cpus ← detectCores()-1
```

We begin with the calculations for the ML estimator.

```
## bootstrap estimates
boot.out ← boot(ICUData.home$SAPS.II, statistic = MLEst, R = 999,
                parallel = "multicore", ncpus = nr.cpus)
## bootstrap confidence interval for scale
boot.ci(boot.out, index = c(1,3))
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, index = c(1, 3))

Intervals :
Level      Normal              Basic              Studentized
95%   ( 4.522,  6.688 )   ( 4.477,  6.646 )   ( 4.661,  6.900 )

Level     Percentile            BCa
95%   ( 4.515,  6.684 )   ( 4.633,  6.846 )
Calculations and Intervals on Original Scale
```

```
## bootstrap confidence interval for shape
boot.ci(boot.out, index = c(2,4))
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, index = c(2, 4))

Intervals :
Level      Normal              Basic             Studentized
95%   ( 5.563,  8.232 )   ( 5.423,  8.068 )   ( 5.717,  8.268 )

Level     Percentile            BCa
95%   ( 5.924,  8.570 )   ( 5.768,  8.439 )
Calculations and Intervals on Original Scale
```

We get confidence intervals similar to the asymptotic intervals. We perform the corresponding calculations for the CvM-MD estimator.

```
## bootstrap estimates
boot.out ← boot(ICUData.home$SAPS.II, statistic = MDEst, R = 999,
                parallel = "multicore", ncpus = nr.cpus)
## bootstrap confidence interval for scale
boot.ci(boot.out, index = c(1,3))
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, index = c(1, 3))

Intervals :
Level      Normal              Basic             Studentized
95%   ( 4.540,  7.590 )   ( 4.521,  7.554 )   ( 4.829,  8.076 )

Level     Percentile            BCa
95%   ( 4.510,  7.544 )   ( 4.621,  7.723 )
Calculations and Intervals on Original Scale
```

```
## bootstrap confidence interval for shape
boot.ci(boot.out, index = c(2,4))
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, index = c(2, 4))

Intervals :
```

```
Level      Normal              Basic              Studentized
95%   ( 4.728,  7.929 )   ( 4.625,  7.699 )   ( 5.038,  7.999 )

Level     Percentile            BCa
95%   ( 5.227,  8.301 )   ( 5.136,  8.182 )
Calculations and Intervals on Original Scale
```

Once again, we obtain results that deviate only slightly from the asymptotic results. Finally, we give the R code for the calculations of the bootstrap confidence intervals for the RMX estimator. However, as already mentioned, we omit the computations because of the very high computation time. We assume that we would also get results that do not differ strongly from the asymptotic results.

```
## bootstrap estimates
boot.out ← boot(ICUData.home$SAPS.II, statistic = RMXEst, R = 999,
                parallel = "multicore", ncpus = nr.cpus)
## bootstrap confidence interval for scale
boot.ci(boot.out, index = c(1,3))
## bootstrap confidence interval for shape
boot.ci(boot.out, index = c(2,4))
```

Accordingly, we have found three models that can be used to describe the available data well. Which of these three models is ultimately best, would have to be further investigated by using new, independent data.

In addition to validating point estimates as in the above case, confidence intervals can also be used for sample size planning (cf. Section 5.1). We want to demonstrate this in the following example with a simple case.

Example 5.15. We consider the question how many persons polling institutes should ask in opinion polls to get reliable prognoses. Assuming a large population as in case of national elections, we can confidently neglect the finite-sample correction and can apply the asymptotic confidence interval given in Example 5.13. As we are interested in the deviation from the estimated value, i.e. the maximum estimate error, we have to take a closer look at the following expression

$$z_{1-\alpha/2}\sqrt{\frac{\hat{p}(1-\hat{p})}{n}} \tag{5.33}$$

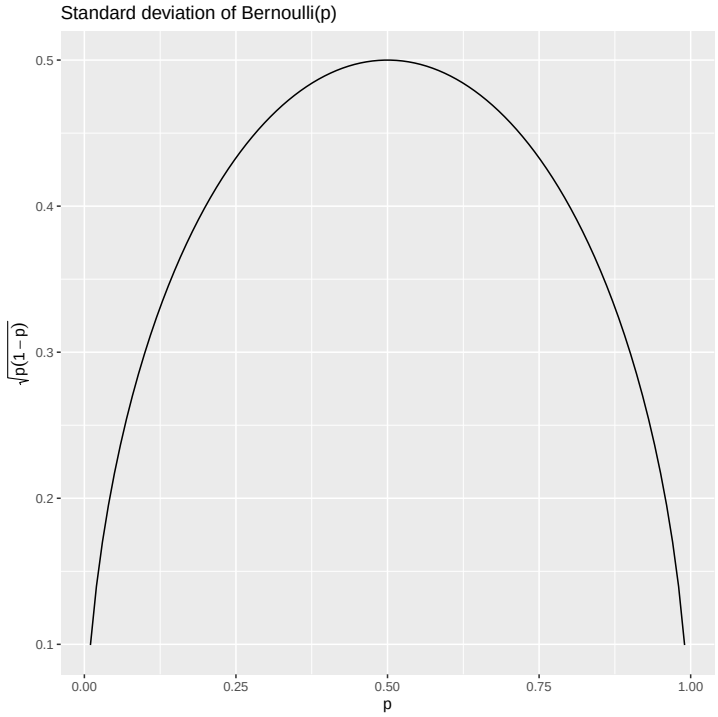
Apparently, the estimate error varies with the confidence level $1-\alpha$, the estimated probability $\hat{p}$ and the sample size n. We assume a 95% confidence interval; that is, we get for $z_{1-\alpha/2} = z_{0.975}$

```
qnorm(0.975)
```

```
[1] 1.959964
```

Next, we take a closer look at the standard deviation $\sqrt{p(1-p)}$ of the Bernoulli distribution.

```
## Values for p
p ← seq(from = 0.01, to = 0.99, length = 100)
## standard deviation
SD ← sqrt(p*(1-p))
## Graphical representation
DF ← data.frame(p, SD)
ggplot(DF, aes(x = p, y = SD)) + geom_line() +
  ylab(expression(sqrt(p*(1-p)))) +
  xlab("p") + ggtitle("Standard deviation of Bernoulli(p)")
```

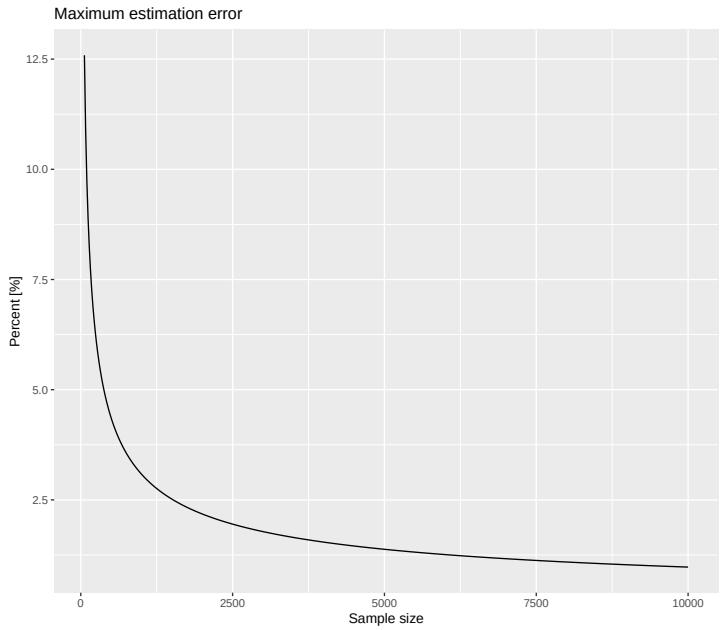


From this plot we can see that the investigated term is largest for $p = 0.5$ and decreases for smaller or larger probabilities. In the case of $p = 0.5$ we get the maximum estimation error. In case of the 95% confidence interval this leads to

$$1.96 \times \frac{0.5}{\sqrt{n}} = \frac{0.975}{\sqrt{n}} = \frac{97.5}{\sqrt{n}}\%. \tag{5.34}$$

We plot the maximum estimate error as a function of the sample size.

```
## sample size
n ← seq(60, 10000, by = 20)
## Maximum estimation error
maxError ← 97.5/sqrt(n)
## Graphical representation
DF ← data.frame(n, maxError)
ggplot(DF, aes(x = n, y = maxError)) + geom_line() + ylab("Percent [%]") +
  xlab("Sample size") + ggtitle("Maximum estimation error")
```



Typically, 1000 people are surveyed opinion polls. The maximum estimation error in this case is at most about 3.1%. We could now calculate a number of cases by specifying the desired maximum estimation error by setting it equal to the expression (5.34), and solving the resulting equation for n.

$$\sqrt{n} = \frac{0.975}{\text{max. estimation error}} \qquad \Longleftrightarrow \qquad n = \left(\frac{0.975}{\text{max. estimation error}}\right)^2 \tag{5.35}$$

We then round the result up to the next integer number yielding our sample size. We can also calculate the sample size directly using function `ssize.propCI` from package `"MKpower"` (Kohl (2020c)). The continuity correction can also be taken into account as well as sample size for different exact intervals can be calculated. Instead of the maximum estimation error, the desired length of the confidence interval has to be specified for this function.

```
## without continuity correction
ssize.propCI(0.5, width = 0.062, method = "wald")
```

```
     Sample size calculation by method of wald

               n = 999.3389
            prop = 0.5
           width = 0.062
      conf.level = 0.95

NOTE: Two-sided confidence interval
```

```
## with continuity correction
ssize.propCI(0.5, width = 0.062, method = "wald-cc")
```

```
     Sample size calculation by method of wald-cc
```

```
              n = 1031.345
           prop = 0.5
          width = 0.062
     conf.level = 0.95

NOTE: Two-sided confidence interval
```

```
## Clopper-Pearson
ssize.propCI(0.5, width = 0.062, method = "clopper-pearson")
```

```
     Sample size calculation by method of clopper-pearson

              n = 1032
           prop = 0.5
          width = 0.062
     conf.level = 0.95

NOTE: Two-sided confidence interval
```

```
## Agresti-Coull
ssize.propCI(0.5, width = 0.062, method = "agresti-coull")
```

```
     Sample size calculation by method of agresti-coull

              n = 995.4975
           prop = 0.5
          width = 0.062
     conf.level = 0.95

NOTE: Two-sided confidence interval
```

Before important elections, pollsters survey up to 50.000 people, which in any case leads to a maximum estimation error of less than 0.5%.

```
## without continuity correction
ssize.propCI(0.5, width = 0.0088, method = "wald")
```

```
     Sample size calculation by method of wald

              n = 49605.61
           prop = 0.5
          width = 0.0088
     conf.level = 0.95

NOTE: Two-sided confidence interval
```

```
## with continuity correction
ssize.propCI(0.5, width = 0.0088, method = "wald-cc")
```

```
     Sample size calculation by method of wald-cc

               n = 49832.63
            prop = 0.5
           width = 0.0088
      conf.level = 0.95

NOTE: Two-sided confidence interval
```

These considerations and calculations are not only important for pollsters, but also play an important role in other fields such as epidemiology or medical epidemiology or medical statistics. Here, for example, it is a matter of estimating the prevalences, incidences or lethality of diseases or the success rates of treatments.

5.4 Exercises

Always briefly describe and explain the results. Use the ICU dataset for exercises 5–18 and always select appropriate functions for the computations.

1. Construct a dataset consisting of exactly five positive numbers, such that the median is equal to 5 and the arithmetic mean is equal to 7. In a second step, modify the dataset, such that the median is unchanged, but the arithmetic mean is larger than the third quartile.

2. How must a dataset look like, such that the standard deviation is equal to 0? In which situation is the standard deviation maximal? Use simple datasets to think about the questions.

3. One can study bone resorption by means of TRAP (tartrate resistant acid phosphatase), which can be measured in one's blood. In a trial of 31 young women, the arithmetic mean of TRAP was equal to 13.2 U/l (Units per liter). Assume a normal distribution model for TRAP, where the standard deviation is known to be $\sigma = 6.5$ U/l. Specify a 95% confidence interval for the mean μ of the women, who are represented by the trial. How does the interval change, if the standard deviation is not known, but was estimated as 6.5 U/l by means of the sample standard deviation (standardization $\frac{1}{n-1}$)? That is, compare the results of the formulas (5.24) and (5.26).

4. Assume 6 successes on 20 attempts. Can you use the approximative confidence interval for the probability of success p? Check the rule of thumb (5.29). Compare the Clopper-Pearson interval and the asymptotic confidence interval including continuity correction.

5. Estimate the probability that an ICU patient is male. To do this, add a binary variable to the data set that has the value 1 if the patient is a male; e.g., by.

```
ICUData$man ← as.integer(ICUData$sex == "male")
```

Use the Bernoulli model $\mathcal{P} = \{\text{Bernoulli}(p) \mid p \in (0,1)\}$ and compute the ML, the CvM-MD, and the AL estimator for p. For the AL estimator, assume 2% of erroneous data (apply function `roptest` of package `"ROptEst"` (Kohl and Ruckdeschel (2019)) with argument `eps = 0.02`). Determine the associated asymptotic confidence intervals as well as bootstrap confidence intervals. Compare the asymptotic and bootstrap confidence intervals with the Clopper-Pearson interval.

6. Assume that the log bilirubin values of ICU patients can be described by a normal distribution.

```
ICUData$logBili ← log10(ICUData$bilirubin)
```

Calculate the ML estimator and compare the result with median and MAD as well as the RMX estimator (apply function `roblox` of package `"RobLox"` (Kohl (2019))). For the RMX estimator, assume $1-5\%$ of erroneous data. Also determine the associated confidence intervals in each case applying bootstrap. Plot the data in the form of a histogram and add the three normal distribution densities with the estimated parameters. Validate the three models additionally with pp- and qq-plots.

7. Consider only patients with a length of stay greater than one day (`LOS` > 1).

```
ICUdata.LOS2 ← ICUdata[ICUdata$LOS > 1, ]
```

Assume that the maximum body temperature (column `temperature`) of these ICU patients can be described by a normal distribution. Calculate the ML estimator and compare the the result with median and MAD as well as the RMX estimator (apply function `roblox` of package `"RobLox"` (Kohl (2019))). For the RMX estimator assume $1-5\%$ of erroneous data. Also determine the corresponding confidence intervals using bootstrap. Plot the data in the form of a histogram and add the three normal distribution densities with the estimated parameters. Validate the three models additionally with pp- and qq-plots.

8. Consider only patients with a length of stay of exactly one day (`LOS` $= 1$) and also remove the outlier with the body temperature of $9.1°C$.

```
ICUData.LOS1 ← ICUData[ICUData$LOS == 1, ]
ICUData.LOS1 ← ICUData.LOS1[ICUData.LOS1$temperature > 10, ]
```

Assume the maximum body temperature (column `temperature`) of these ICU patients can be described by a normal distribution. Calculate the ML estimator and compare the result with median and MAD as well as the RMX estimator (apply function `roblox` of package `"RobLox"` (Kohl (2019))). For the RMX estimator, assume $0-5\%$ of erroneous data. Also determine the corresponding confidence intervals using bootstrap. Plot the data in the form of a histogram and add the three normal distribution densities with the estimated parameters. Validate the three models additionally with pp- and qq-plots.

9. Consider only patients with a length of stay greater than one day (`LOS` > 1). Assume the logarithmized maximum heart rates (column `heart rate`) of these ICU patients can be described by a normal distribution.

```
ICUData.LOS2 ← ICUData[ICUData$LOS > 1, ]
ICUData.LOS2$logHR ← log10(ICUData.LOS2$heart.rate)
```

   Calculate the ML estimator and compare the result with median and MAD as well as the RMX estimator (apply function `roblox` of package `"RobLox"` (Kohl (2019))). For the RMX estimator, assume $0 - 5\%$ of erroneous data. Also determine the corresponding confidence intervals using bootstrap. Plot the data in the form of a histogram, and add the three normal distribution densities with the the estimated parameters. Validate the three models additionally with pp- and qq-plots.

10. Consider only patients with a length of stay of exactly one day (`LOS` $= 1$). Assume logarithmized maximum heart rates (column `heart rate`) of these ICU patients can be described by a normal distribution.

```
ICUData.LOS1 ← ICUData[ICUData$LOS == 1, ]
ICUData.LOS1$logHR ← log10(ICUData.LOS1$heart.rate)
```

   Calculate the ML estimator and compare the result with median and MAD as well as the RMX estimator (apply function `roblox` of package `"RobLox"` (Kohl (2019))). For the RMX estimator, assume $0 - 5\%$ of erroneous data. Also determine the corresponding confidence intervals using bootstrap. Plot the data in the form of a histogram, and add the three normal distribution densities with the the estimated parameters. Validate the three models additionally with pp- and qq-plots.

11. Consider only the patients with liver failure. Assume log bilirubin values (column `bilirubin`) of these ICU patients can be described by a normal distribution.

```
ICUData.LF ← ICUData[ICUData$liver.failure == 1, ]
ICUData.LF$logBili ← log10(ICUData.LF$bilirubin)
```

   Calculate the ML estimator and compare the result to the median and MAD as well as the RMX estimator (apply function `roblox` of package `"RobLox"` (Kohl (2019))). For the RMX estimator, assume $0 - 10\%$ of erroneous data. Also determine the corresponding confidence intervals using bootstrap. Plot the data in the form of a histogram and add the three normal distribution densities with the estimated parameters. Validate the three models additionally with pp- and qq-plots.

12. Assume that the length of stay (LOS) of ICU patients can be described by a Gamma distribution. Calculate the ML, the CvM-MD, and the RMX estimator (apply function `roptest` of package `"ROptEst"` (Kohl and Ruckdeschel (2019)) and their asymptotic confidence intervals. For the RMX estimator, assume $1 - 5\%$ of erroneous data. The RMX estimator will take some time to compute. Plot the data in the form of a histogram and add the three Gamma distribution densities with the estimated parameters. Validate the three models additionally with pp- and qq-plots.

13. Consider only patients with neurologic surgery (`surgery = ‘‘neuro’’`).

```
ICUData.neuro ← ICUData[ICUData$surgery == "neuro",]
```

Consider length of stay (column `LOS`) of these ICU patients and assume that it can be described by a Gamma distribution. Determine the ML, the CvM-MD, and the RMX estimator (apply function `roptest` of package `"ROptEst"` (Kohl and Ruckdeschel (2019)) and their asymptotic confidence intervals. For the RMX estimator assume $1-5\%$ of erroneous data. The calculation of the RMX estimator will take some time. Plot the data in the form of a histogram and add the three Gamma distribution densities with the estimated parameters. Validate the three models additionally with pp- and qq-plots.

14. Select only patients with `surgery = ‘‘gastrointestinal’’` – a gastrointestinal tract surgery.

```
ICUData.gastro ← ICUData[ICUData$surgery == "gastrointestinal",]
```

Consider the length of stay (column `LOS`) of these ICU patients and assume that it can be can be described by a Gamma distribution. Determine the ML, the CvM-MD, and the the RMX estimator (apply function `roptest` of package `"ROptEst"` (Kohl and Ruckdeschel (2019)) and their asymptotic confidence intervals. For the RMX estimator assume $1-5\%$ of erroneous data. The calculation of the RMX estimator will take some time. Plot the data in the form of a histogram and add the three Gamma distribution densities with the estimated parameters. Validate the three models additionally with pp- and qq-plots.

15. Consider only patients with cardiothoracic surgery (`surgery = ‘‘cardiothoracic’’`).

```
ICUData.cardio ← ICUData[ICUData$surgery == "cardiothoracic",]
```

Consider the length of stay (column `LOS`) of these ICU patients and assume that it can be described by a gamma distribution. Determine the ML, the CvM-MD, and the RMX estimator (apply function `roptest` of package `"ROptEst"` (Kohl and Ruckdeschel (2019)) and their asymptotic confidence intervals. For the RMX estimator assume $1-5\%$ of erroneous data. The calculation of the RMX estimator will take some time. Plot the data in the form of a histogram and add the three Gamma distribution densities with the estimated parameters. Validate the three models additionally with pp- and qq-plots.

16. Examine the age of ICU patients in more detail. Assume you can describe the age by a Weibull distribution. Determine the ML, CvM-MD and the RMX estimator (apply function `roptest` of package `"ROptEst"` (Kohl and Ruckdeschel (2019)) and their asymptotic confidence intervals. You will need package `"RobExtremes"` (Ruckdeschel et al. (2019)) for this. For the RMX estimator, assume $1-5\%$ of erroneous data. The calculation of the RMX estimator will take some time. Plot the data in the form of a histogram and add the three Weibull distribution densities with the estimated parameters. Validate the three models additionally with pp- and qq-plots.

17. Consider patients with a length of stay greater than one day (`LOS` > 1).

```
ICUdata.LOS2 ← ICUdata[ICUdata$LOS > 1, ]
```

Examine the maximum SAPS-II score (column `SAPS.II`) of these ICU patients and assume that this can be described by a Weibull distribution. Determine the ML-, the CvM-MD and the RMX estimators (apply function `roptest` of package `"ROptEst"` (Kohl and Ruckdeschel (2019)) and their asymptotic confidence intervals. You will need package `"RobExtremes"` (Ruckdeschel et al. (2019)) for this. For the RMX estimator, assume $1 - 5\%$ of erroneous data. The calculation of the RMX estimator will take some time. Plot the data in the form of a histogram and add the three Weibull distribution densities with the estimated parameters. Validate the three models additionally with pp- and qq-plots.

18. Consider patients with a length of stay of one day (`LOS` = 1).

```
ICUdata.LOS1 ← ICUdata[ICUdata$LOS == 1, ]
```

Examine the maximum SAPS-II score (column `SAPS.II`) of these ICU patients and assume that this can be described by a Weibull distribution. Determine the ML-, the CvM-MD and the RMX estimators (apply function `roptest` of package `"ROptEst"` (Kohl and Ruckdeschel (2019)) and their asymptotic confidence intervals. You will need package `"RobExtremes"` (Ruckdeschel et al. (2019)) for this. For the RMX estimator, assume $1 - 5\%$ of erroneous data. The calculation of the RMX estimator will take some time. Plot the data in the form of a histogram and add the three Weibull distribution densities with the estimated parameters. Validate the three models additionally with pp- and qq-plots.

19. Load https://github.com/stamats/COVID-19/blob/master/COVID-19.Rmd and https://github.com/stamats/COVID-19/blob/master/Bevoelkerung2019.RData and save both files in a common directory. Using RStudio, generate the html file to the R Markdown file and read it. Attention: You need to install additional R packages for this!
Use recent figures for Germany and use them to recalculate the number of unreported cases (as well as 95% confidence interval).

20. In the European Union, a disease is said to be rare when the incidence is less than 1 in 2000 persons. The incidence rate per year of sarcoidosis is about 5-60 cases per 100.000. We assume that this is a symmetrical 95% confidence interval for the incidence rate. The expected value (midpoint of the interval) is 32.5 cases per 100.000 inhabitants. The length of the confidence interval is 55 cases per 100.000 population. Determine the sample size of how many subjects would have to be included in a study to confirm this asymptotic confidence interval. Use the same values for calculating the number of cases using the Clopper-Pearson and Agresti-Coull interval.

21. View https://clinicaltrials.gov/ct2/show/study/NCT03976804, a recent study on the prevalence of lung cancer. There are 500 high-risk patients to be studied for prevalence of lung cancer. In various other prevalence studies in Europe, a prevalence of 2% has been determined.

The prevalence is expected to be more than 2%. What is the minimum prevalence in this study, so that the lower limit of the associated (two-sided) 95% confidence interval is greater than 2%; so that one can assume a significant result and therefore a success of the study? Answer this question using function `ssize.propCI` from `"MKpower"` package (Kohl (2020c)). Try combinations of prevalences and lengths of the confidence interval, so that the lower limit of the corresponding 95% confidence interval is greater than 2% and at the same time results in a number of cases of roughly 500.

22. Examine the relationship between survival (i.e. `outcome` $\neq$ ‘‘`died`’’) and the SAPS-II score, by looking for an optimal cut-off for survival. First, define a new variable.

```
ICUData$died ← as.integer(ICUData$outcome == "died")
```

We assume that the probability of dying is monotone increasing with increasing SAPS-II score. Use Youdens J statistics and back up your results using a stratified Bootstrap. Calculate sensitivity and specificity for the calculated optimal cut-off. Also determine the AUC and Brier score for the SAPS-II score.

23. Examine the relationship between survival (i.e. `outcome` $\neq$ ‘‘`died`’’) and maximum heart rate, by looking for an optimal cut-off for survival. First, define a new variable.

```
ICUData$died ← as.integer(ICUData$outcome == "died")
```

We assume that the probability of dying is monotone increasing with increasing maximum heart rate. Use Youdens J statistics and back up your results using a stratified Bootstrap. Calculate sensitivity and specificity for the calculated optimal cut-off. Also determine the AUC and Brier score for the SAPS-II score.

24. You want to study the probability of trash in a production process and for this purpose draw a representative sample of the produced parts. You estimate the unknown probability of trash and determine the corresponding 95% confidence interval. You repeat this procedure every month for five months, where each month you draw a new independent sample. Then, the probability that all five intervals cover the true unknown parameter is smaller than 95%. How large is this probability exactly? How likely is it, that at least four of the five confidence intervals will cover the true unknown parameter?

6 Statistical Tests

In this chapter we introduce statistical tests. In detail, it covers the following topics:

- Hypotheses
- Test decisions, power, sensitivity, type I and type II error
- Order for the correct conduct of a test
- Calculation of sample size and power analysis
- 1-sample binomial test (exact and asymptotic)
- 1-sample multinomial test
- 2-sample binomial test (asymptotic)
- Fisher's exact test, χ^2 test, Cramér's V test
- McNemar χ^2-test
- Cochran-Mantel-Haenszel χ^2-test
- t-test: 1-sample, paired, 2-sample, Welch, Hsu
- Wilcoxon signed rank test, Wilcoxon rank sum test / Mann-Whitney U test.
- F-test, Ansari-Bradley test
- One-way ANOVA, Kruskal-Wallis test
- One-way ANOVA with repeated measures, Friedman test, Quade test
- Testing correlations (Pearson, Spearman, Kendall)
- Testing for normality: Shapiro-Wilk test, Lilliefors (Kolmogorov-Smirnov) test, Cramér-von Mises test, Shapiro-Francia test
- Bootstrap and Permutation tests
- Post hoc tests

The R code for this chapter is in the file `Testen.Rmd`, which you can download from my GitHub account (Link: https://github.com/stamats/ISDR/blob/main/Tests.Rmd). Right-click on `Raw`. Then you can save the `target as...`. Save the R Markdown files in the same folder as the data which will be used in the respective chapters.
We first install the packages needed in this chapter.

```
install.packages(c("coin", "exactRankTests", "ggpubr", "datarium"))
```

Make sure that you have already installed the packages of the previous Chapter 2–5. We load all packages required in this chapter.

```
library(DescTools)
library(ggplot2)
library(ggsci)
library(gridExtra)
library(MKinfer)
library(coin)
library(exactRankTests)
library(ggpubr)
library(datarium)
```

As explained in Section 2.4, running `library` repeatedly is not problematic.

6.1 Introduction

Empirical investigations and studies usually start with a new idea, a conjecture about a certain often open problem. This conjecture is usually postulated on the basis of empirical observations and/or subject-specific theoretical considerations. It facilitates the verification of an assumption, if it may be formulated in a precise and quantifiable way. In this case, one speaks of a **hypothesis**. First, one should collect all available information about the problem and elaborate the theoretical background to verify whether the hypothesis is generally plausible. Frequently, one hereby already realizes that the hypothesis can not be true, which saves work (and money).

In many fields, direct proofs of hypotheses are not possible and they can not be verified directly by a single experiment. At this point, statistics comes into play. We collect representative and relevant data for the problem and subject the data to a statistical analysis, where the results can be ensured by so-called **statistical tests**. In the following example, the general approach is described by means of a dice game.

Example 6.1. We consider a dice game, where it is important to throw “6”. After some time of playing, we realize that our dice only rarely gives “6”. Therefore, we conjecture

the frequency of “6” is too small

or more generally,

the frequency of “6” is incorrect.

In particular, this implies that not all sides of the dice occur with identical probability; that is, the dice is not fair. The precise and quantifiable formulation of the conjecture leads to the hypothesis:

$$\text{The probability } p \text{ of “6” is not equal to } \tfrac{1}{6}\text{; abbreviated: } p \neq \tfrac{1}{6}$$

In general, an answer by means of statistical tests is only possible, if there are mutually exclusive cases. For the dice either

$$\text{our hypothesis is true, i.e. } p \neq \tfrac{1}{6}$$

or

$$\text{our hypothesis is not true, i.e. } p = \tfrac{1}{6}$$

In the present case, one collects information (evidence) for the hypothesis by throwing the dice n times and by counting the number of “6”. The open questions we can answer by means of statistical tests are:

1. How often should we throw the dice?
2. How many “6” do we need to decide in favor or against the hypothesis?

As the previous example shows, the origin of statistical tests are two mutually exclusive hypotheses. These are usually denoted as follows:

Null hypothesis H_0: Hypothesis that shall be *falsified*.

Alternative (hypothesis) H_1: Hypothesis that shall be *confirmed* (research hypothesis).

We transfer this notion to our dice example.

Example 6.2. We again consider the dice game, where it is important to throw “6”. Here, we obtain

$$\text{Null hypothesis } H_0\text{: } p = \tfrac{1}{6} \text{ versus Alternative } H_1\text{: } p \neq \tfrac{1}{6}$$

Since the alternative includes the cases $p < \frac{1}{6}$ and $p > \frac{1}{6}$, it is also called a **two-sided** hypothesis. Of course, also the **one-sided** cases

- $H_0 : \ p = \frac{1}{6}$ versus $H_1 : \ p < \frac{1}{6}$
- $H_0 : \ p \geq \frac{1}{6}$ versus $H_1 : \ p < \frac{1}{6}$

would be possible.

Note:
The decision whether to consider a one-sided or two-sided alternative, must always be made before conducting the test. In medicine, for instance, one-sided alternatives emerge only rarely. In fact, often an improvement is solely of interest, but a worsening would have far reaching consequences, thus for ethical reasons and for the safety of the patients a two-sided alternative has to be chosen.

In the framework of inferential statistics we assume (representative) samples of larger populations. All values we compute, depend on the concrete sample and are subject to uncontrollable random variations. In view of the decisions that are made based on statistical results, one has to conclude that wrong decisions can never completely avoided. It is inevitable, that we make a wrong decision with a (hopefully small) positive probability. If we transfer this situation to statistical testing, we get the situation shown in Table 6.1.

	H_0 **is true**	H_1 **is true**
Decision for H_0	correct decision $1-\alpha$ (sensitivity)	type II error β
Decision for H_1	type I error α (signifikance level)	correct decision $1-\beta$ (power, specificity)

Table 6.1: Decision situation in case of statistical tests.

Thus, the possible wrong decisions are:

Type I error: Probability of rejecting H_0 although it is true.

Type II error: Probability of not rejection H_0 although it is false.

We describe the errors and their consequences in more detail by means of an example.

Example 6.3. We consider the following situation in medicine: There is an effective and safe therapy, that is in use for many years – a so-called **gold standard**. Now, somebody is convinced, that their new therapeutic approach is even more effective.
In this case, it would be a type I error, if one decides against the gold standard and in favor of the new therapy, although the new approach is not better or perhaps even worse. As a consequence, the patients are withheld from a more effective therapy and in cases, where the therapy has adverse effects, it would even harm patients.
In contrast, a type II error would be that one keeps the gold standard, although the new approach is actually better. That is, one has missed a chance for an improvement. However, the patients still get an effective and safe therapy.
In this medical application, the type I error would be the more serious wrong decision.

We briefly summarize the essential facts about the two errors, where we start with the type I error.

Type I error:

- It is inevitable, but controllable.
- The error probability α must be set *before* conducting the test!
- α forms the basis for determining the acceptance respectively, rejection region of H_0.

- In principle, α may be arbitrarily chosen. The standard choice is $\alpha = 0.05$, sometimes also $\alpha = 0.01$ or smaller is used, but very (very) rarely $\alpha > 0.05$.
- Dependent on α the acceptance of H_1 is also called statistically significant ($\alpha = 0.05$), statistically very significant ($\alpha = 0.01$), or statistically extremely or highly significant ($\alpha = 0.001$).

Type II error:

- It is difficult to determine/estimate.
- In general, it holds: The larger α, the smaller β; that is, a small α and a small β are two competing aims.
- Furthermore, it holds: The larger the sample size n, the smaller is β. In practice, this is the only way to control β and implies the need for a detailed sample size calculation and power analysis.
- However, for sample size calculations a certain prior knowledge about the effect size, the variation of the applied estimators, the type I error and the intended power is required.
- Standard assumptions for sample size calculations are $\alpha = 0.05, 0.01$ and $1 - \beta = 0.8, 0.9$.

The following list contains the necessary steps for conducting a statistical test. In the framework of a clinical trial, one strictly has to follow the given order, as it ensures that nobody can influence the result of the test after the start of the trial.

1. Definition of the hypotheses H_0 and H_1 (one-/two-sided?)
2. Fixing of the type I error (significance level)
3. Selection of an appropriate test T
4. Sample size calculation and power analysis (selection of β-error resp. power $1-\beta$, expected effect, expected variance, etc.)
5. Determination of rejection (K_α) and acceptance ($\bar{K}_\alpha$) region of H_0
6. Conduct of the experiments and generation of relevant data $x_1, \ldots, x_n$
7. Calculation of the test statistics $t = T(x_1, \ldots, x_n)$
8. Decision for H_1 ($t \in K_\alpha$) or H_0 ($t \in \bar{K}_\alpha$)

In practice, the test decision is usually based on the so-called **p value**. By this is meant the following (conditional) probability

1-sided test: $p = P(T \geq t \mid H_0)$ resp. $p = P(T \leq t \mid H_0)$

2-sided test: $p = 2\min\{P(T \geq t \mid H_0), P(T \leq t \mid H_0)\}$, with symmetry to 0: $p = P(|T| \geq |t| \mid H_0)$

Accordingly, the probability is calculated that the value of the test statistic T takes more extreme values than the observed test statistic t under the assumption, that H_0 is correct. Thus, if p is small, it is unlikely that the data at hand have been generated under the null hypothesis and we decide in favor of the alternative. More precisely, one decides as follows:

If $p \leq \alpha$: rejection of H_0

If $p > \alpha$: acceptance of H_0, i.e. rejection of H_1

Remark 6.4. ***(a)*** *The p value is not the probability of H_0. This probability does not exist, because H_0 is either true or false. Moreover, p is also not the type I error; i.e., the probability to reject the null hypothesis H_0 even though it is correct.*

(b) *It is incorrect to speak of highly or highest significant dependent of the p value. The strength of the significance does not directly depend on the magnitude of the p-value, but on whether the p-value falls below a certain predefined significance threshold α. If, for example a level of $\alpha = 5\%$ was selected, then even if $p < 0.01$ or $p < 0.001$ it is "only" a significant result and not a highly or highest significant result.*

(c) *It is also crucial to recognize that statistical significance is not synonymous with relevance. Especially with very large samples, even the smallest differences can be significant, without that any consequences can be derived from them. It is therefore important to always keep an eye on the size of the observed effect and its variance in addition to significance. Confidence intervals are very well suited for this purpose.*

In the following example, we demonstrate using the **2-sample t-test**, which is also called **Student t-test** after the pseudonym of its inventor, how to perform a statistical test in practice.

Example 6.5. Let us assume a (well-defined) population including two (well characterized and disjoint) groups, that we want to compare. We are interested in the expectation (location parameter) of a certain attribute X. We additionally assume that the attribute is normally distributed (at least approximately) and that it has identical variances for both groups; that is, it holds for group I: $X_1 \sim \text{Norm}(\mu_1, \sigma^2)$ and for group II: $X_2 \sim \text{Norm}(\mu_2, \sigma^2)$. We conduct steps 1-8, as listed above, to compare the expectations of the two groups by means of a statistical test:

1. We consider the following hypotheses

$$H_0 : \mu_1 = \mu_2 \text{ versus } H_1 : \mu_1 \neq \mu_2$$

   That is, the alternative is two-sided.

2. We choose the standard type I error (significance level): $\alpha = 0.05$

3. Since we assume a normal distribution for both groups and want to estimate the mean, where the variance is also unknown and has to be estimated, it leads to a t distribution. Consequentially, we select the two-sample t test. Let $x_1, \ldots, x_{n_1}$ be the observations of group I and $y_1, \ldots, y_{n_2}$ the observations of group II, then the test statistics reads

$$T(x_1, \ldots, x_{n_1}; y_1, \ldots, y_{n_2}) = \sqrt{\frac{n_1 n_2}{n_1 + n_2}} \frac{\text{AM}(x_1, \ldots, x_{n_1}) - \text{AM}(y_1, \ldots, y_{n_2})}{\text{SD}(x_1, \ldots, x_{n_1}; y_1, \ldots, y_{n_2})} \tag{6.1}$$

where

$$\mathrm{SD}\,(x_1,\ldots,x_{n_1};y_1,\ldots,y_{n_2}) = \sqrt{\frac{(n_1-1)\tilde{S}(x_1,\ldots,x_{n_1})+(n_2-1)\tilde{S}(y_1,\ldots,y_{n_2})}{n_1+n_2-2}} \tag{6.2}$$

and $\tilde{S}$ is the sample variance with standardization $\frac{1}{n-1}$.

4. For sample sice calculation and power analysis we additionally need the (expected) effect size $\delta = |\mu_1 - \mu_2|$, the (expected) variance σ^2 and the wanted power $1-\beta$.
The influence of the effect size on the sample size is displayed in Figure 6.1, where we consider the standard setup $\beta = 0.2$ and assume $\sigma = 1$ without restriction. The computations were performed by applying function `power.t.test`. As we see, the required sample size clearly decreases with

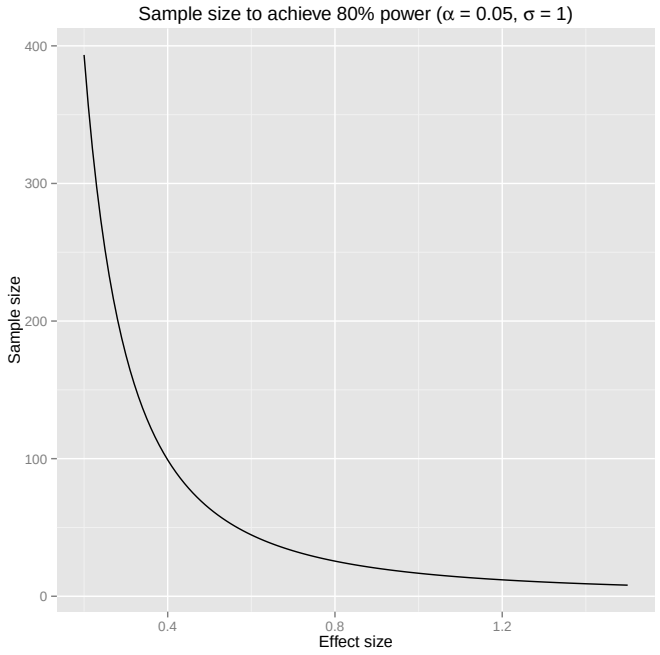


Figure 6.1: Sample size dependent on effect size.

increasing effect size; that is, the larger the effect, the smaller the sample size or we can also put it the other way round: with very large samples we may even verify small (irrelevant) effects.
Figure 6.2 shows the dependence of the sample size on the variance, where we assumed an effect size of 1 without restriction. The computations were again performed by means of function `power.t.test`. Thus, the larger the variance, the larger the sample size has to be chosen. In particular, the (expected) ratio $\frac{\delta}{\sigma}$ is of crucial importance, which is also called the (expected) **standardized effect**. Other often used terms for this are **Cohens d** or also **SMD** (standardized mean difference).
We perform the case number calculation with $\delta = 1$, $\sigma = 1$, $\alpha = 0.05$ and $1-\beta = 0.8$

```
1 power.t.test(delta = 1, sd = 1, sig.level = 0.05, power = 0.8)
```

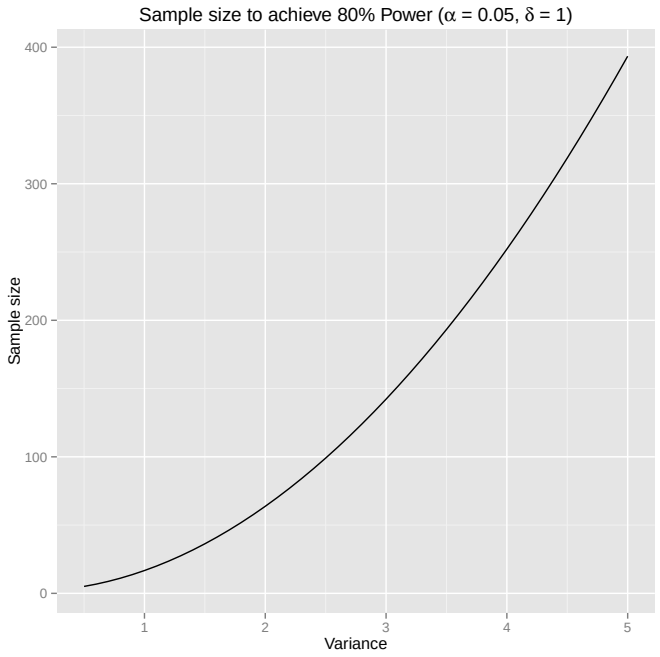


Figure 6.2: Sample size dependent on variance.

```
     Two-sample t test power calculation

              n = 16.71477
          delta = 1
             sd = 1
      sig.level = 0.05
          power = 0.8
    alternative = two.sided

NOTE: n is number in *each* group
```

Under these assumptions, a case number of 17 per group is sufficient to produce the expected difference with a power of 80% and an type I error of 5%. In studies, the number of cases is often adjusted upward to compensate for possible dropouts. We use a case number of $n = 20$. With complete data this leads to a power of approx 87%, as the following calculation with `power.t.test` shows.

```
1 power.t.test(n = 20, delta = 1, sd = 1, sig.level = 0.05)
```

```
     Two-sample t test power calculation

              n = 20
          delta = 1
```

```
              sd = 1
       sig.level = 0.05
           power = 0.8689528
     alternative = two.sided

NOTE: n is number in *each* group
```

5. If we assume the null hypothesis H_0 is true, the test statistic T follows a t distribution with n_1+n_2-2 degrees of freedom. This fact we can use to determine the acceptance region $\bar{K}_\alpha$ of H_0. Because of the symmetry of the situation, we obtain $\bar{K}_\alpha = [-c, c]$, where it must hold

$$P(-c \leq T \leq c \mid H_0) = 1 - \alpha \tag{6.3}$$

i.e. c is the $(1-\alpha/2)$ quantile of the $t_{n_1+n_2-2}$ distribution. c is also called **critical value** of the test. Under the assumption $n_1 = n_2 = 20$, we get

```
qt(0.975, df = 38)
```

```
[1] 2.024394
```

6. We conduct the experiment and generate random numbers by means of function `rnorm`. More precisely, we use $X_1 \sim \text{Norm}(0.5, 1)$ for group 1 and $X_2 \sim \text{Norm}(1.5, 1)$ for group 2.

```
## random numbers for demonstration
X1 <- rnorm(n = 20, mean = 0.5, sd = 1)
X2 <- rnorm(n = 20, mean = 1.5, sd = 1)
```

7. We compute the test statistic by means of function `t.test`.

```
t.test(X1, X2, var.equal = TRUE)$statistic
```

```
        t
-2.822995
```

In two-sided testing, the absolute value is to be compared with the critical value. If it is smaller, we decide for H_0; if not, we decide for H_1. In the present case we therefore decide for H_1 and we could thus confirm our claim.

8. We can alternatively calculate the p-value, the probability, that the above calculated or a more extreme value of the test statistic occurs assuming that H_0 is true. This extreme value can occur in the two-sided test either below or above. Due to the symmetry of the t-distribution it is sufficient to calculate one of the two probabilities and to double it.

```
2*pt(abs(t.test(X1, X2, var.equal = TRUE)$statistic), df = 38,
     lower.tail = FALSE)
```

```
          t
0.00753125
```

With the help of the function `t.test` we get the same result a little easier

```
t.test(X1, X2, var.equal = TRUE)$p.value
```

```
[1] 0.00753125
```

The p-value is consequently smaller than the specified significance level 0.05 and we can speak of a significant result. The p-value is even smaller than 0.01. In such cases, many scientific publications refer to the difference as highly significant. Strictly speaking, this is not correct, since this designation does not refer to the p-value, but the significance level.

The complete output of function `t.test` reads

```
t.test(X1, X2, var.equal = TRUE)
```

```

        Two Sample t-test

data:  X1 and X2
t = -2.823, df = 38, p-value = 0.007531
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -1.3859110 -0.2283271
sample estimates:
mean of x mean of y
0.6416238 1.4487428
```

The printed 95% confidence interval is an interval for $\mu_1 - \mu_2$ and thus represents the expected effect. One can also use it for the test decision. In the present case the null hypothesis is represented by $\mu_1 - \mu_2 = 0$. Since the reference value 0 does not lie in the 95% confidence interval, the null hypothesis is to be rejected at the significance level of 5%. More generally speaking, if the corresponding reference value for the null hypothesis does not lie in the $1 - \alpha$ confidence interval, then the null hypothesis can be rejected at the α significance level.
Because the interval reflects the expected effect, it shows even more than the p-value and gives us an impression of the relevance of the results. In the best case, based on the available data, there would be an effect of up to about -1.38. But also an effect of only about -0.23 would not be possible based on the available data. This aspect, that no significant difference is found, will be examined again in exercise 1 in Section 6.4.

Note:
For some years now, "hunting" for significance ("p-hacking") has been viewed in scientific research as very problematic. It has on the one hand, led to the assumption that many, if not most, results are false positives (Ioannidis (2005), Colquhoun (2014)). On the other hand, it has also had the positive effect of more interest and examination of the strengths and weaknesses of statistical testing (Cohen (1994), Sterne and Davey Smith (2001), Head et al. (2015), Wasserstein and Lazar (2016), Amrhein et al. (2017)). Moreover, as a result, confidence intervals have justifiably gained in importance (Rigby (1999), du Prel et al. (2009), Ranstam (2012), Sedgwick (2013)).

For sample size calculation and power analysis there are a number of functions in R, which mostly start with "`power.`". Besides `power.t.test`, there are functions `power.prop.test` and `power.anova.test` in base package `"stats"` (R Core Team (2022a)). In addition, there are a number of extension packages, which provide functions for sample size calculations and power analysis for various tests and models.

Note:
It is common practice, to check the assumptions of statistical tests in pre-tests. This includes the verification of distributional assumptions, especially the normal distribution, or the assumption of equal variances (homogeneity of variances). In addition to the methodological problems that several interdependent questions are tested on the same data set and that the sequence of steps that is necessary to keep the type I error under control, pre-tests often have a lower power than the main test, which reflects the scientific question and is the basis for case and was the basis for sample size calculation. Thus, in the case of small numbers of cases deviations are only detected with a low probability (many false negative results). On the other hand, with large case numbers, the pre-tests will detect small, often irrelevant deviations for the main test. Rasch et al. (2011) show by the example of the t-test that the practice of pre-tests does not pay off. On the one hand, this leads to unknown type I and type II errors. On the other hand, their simulation study did neither show a gain in (empirical) power nor a reduction of the (empirical) type I error. Therefore, they recommend not to perform any pre-tests and to use the Welch t-test instead of the Student t-test. Moreover, from their point of view the Wilcoxon-Mann-Whitney test should only be used for ordinal data. These tests are discussed in more detail in Section 6.3.

In the following two sections, I will briefly introduce a number of statistical tests briefly.

6.2 Nominal variables

This section is primarily concerned with statistical tests for nominal variables. Of course, these tests can also be used for ordinal variables by ignoring the information of the order. In this section the tests shown in Figure 6.3 are discussed in more detail. In the following we will work our way through the diagram step by step. We start again with the simplest model namely the Bernoulli model, which is used in the case of binary variables, and we want to perform statistical tests for the investigation of the probability of success. We assume that only data from one sample are available and we want to test the probability for the presence of 1. For the comparison of the relative frequency with a given value, we can use the

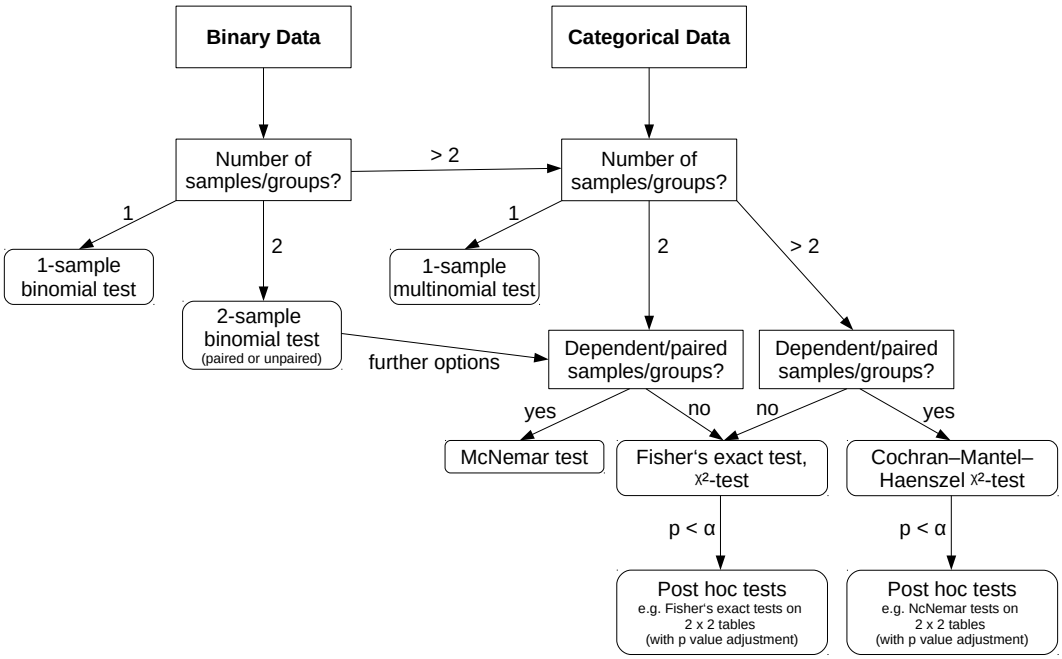


Figure 6.3: Baumdiagramm zur Auswahl des geeignetsten Tests im Fall nomialer Merkmale.

1-sample binomial test, which is described in the function `binom.test`. In the case of the confidence intervals (cf. Example 5.13) we can use an approximation with the help of the normal distribution (with and without continuity correction). This is available in the form of the function `prop.test`. We use the data set of ICU patients and examine the prevalence for liver failure. We would like to investigate whether we can assume a prevalence of less than 5%;

$$H_0 : \; p >= 0.05 \quad \text{versus} \quad H_1 : \; p < 0.05$$

We assume a significance level of 0.05 and use both the exact and the asymptotic test, so the `binom.test` and `prop.test` functions. We specify the alternative using `alternative = "less"`.

```
ICUData ← read.csv(file = "ICUData.csv", fileEncoding = "utf8",
                   stringsAsFactors = TRUE)
table(ICUData$liver.failure)
```

```
  0   1
480  20
```

```
## exact test
binom.test(20, 500, p = 0.05, alternative = "less")
```

```
        Exact binomial test
```

```
data:  20 and 500
number of successes = 20, number of trials = 500, p-value = 0.1789
alternative hypothesis: true probability of success is less than 0.05
95 percent confidence interval:
 0.00000000 0.05759556
sample estimates:
probability of success
                  0.04
```

```
## asymptotic test with continuity correction
prop.test(20, 500, p = 0.05, alternative = "less")
```

```

        1-sample proportions test with continuity correction

data:  20 out of 500, null probability 0.05
X-squared = 0.85263, df = 1, p-value = 0.1779
alternative hypothesis: true p is less than 0.05
95 percent confidence interval:
 0.00000000 0.05822552
sample estimates:
   p
0.04
```

```
## asymptotic test without continuity correction
prop.test(20, 500, p = 0.05, alternative = "less", correct = FALSE)
```

```

        1-sample proportions test without continuity correction

data:  20 out of 500, null probability 0.05
X-squared = 1.0526, df = 1, p-value = 0.1525
alternative hypothesis: true p is less than 0.05
95 percent confidence interval:
 0.00000000 0.05706325
sample estimates:
   p
0.04
```

In all three cases we get very similar results, which is primarily due to the high sample size, although the asymptotic test is significantly better due to the continuity correction. We also see that by testing one-sided we also get corresponding one-sided confidence intervals. In all three cases the p-value is greater than 0.05, so we must retain the null hypothesis; the prevalence of liver failure could also be greater than or equal to 5%. As we can see from the confidence intervals, on the basis of the available data, a prevalence of up to approximately 5.8% would also be possible. Since we can also derive the test decision from the corresponding confidence interval, we can alternatively use the function `binomCI` from the package package `"MKinfer"` (Kohl (2022b)) and thus obtain exact, asymptotic or bootstrap confidence intervals; for example

```
binomCI(20, 500, method = "boot", alternative = "less")
```

```
        boot confidence interval

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 9999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, conf = 1 - alpha, type = bootci.type)

Intervals :
Level      Normal              Basic              Studentized
95%   ( 0.0000,  0.0544 )   ( 0.0000,  0.0540 )   ( 0.0000,  0.0572 )

Level     Percentile            BCa
95%   ( 0.000,  0.054 )   ( 0.000,  0.054 )
Calculations and Intervals on Original Scale

sample estimate:
prob
0.04

additional information:
          standard error of prob bootstrap standard error of prob
                     0.008763561                       0.008761857
```

All calculated bootstrap confidence intervals intersect the probability range given by the Null hypothesis of [0.05, 1]. Thus the null hypothesis on the significance level of 5% cannot be rejected.

With this example we can also nicely demonstrate “p-hacking”. If one had assumed a prevalence of 6% instead of 5%, then the result would be significant (6% is outside the confidence intervals); for example

```
## exact test
binom.test(20, 500, p = 0.06, alternative = "less")
```

```
        Exact binomial test

data:  20 and 500
number of successes = 20, number of trials = 500, p-value =
0.03129
alternative hypothesis: true probability of success is less than 0.06
95 percent confidence interval:
 0.00000000 0.05759556
sample estimates:
probability of success
                  0.04
```

In clinical trials, the problem that assumptions or even the entire statistical analysis are changed after the data are available is approached by the requirement that each study should be entered into a corresponding registry before it is conducted. All major scientific journals require this when the results of the study are submitted for publication. At least in the case of the so-called primary hypotheses, on which the success or failure of a study are determined, expected results and the statistical methods have to be specified in detail in the study protocol. In many other areas of research this additional protection does not exist, which is why it is never really certain that the statistical analysis has not been adapted to the available data in order to produce significant results, which is then equated with the success of the study.

If a sample has more than two characteristic values, this can be examined more closely with the help of a multinomial test. For this purpose the function `multinomial.test` from the package `"EMT"` (Menzel (2021)) can be used. In the case of the 1-sample test, it is necessary to specify for each possible category its expected probability. Since the computation of the exact multinomial test is very complex, we do not give an example.

If two or more groups are to be compared with respect to a binary characteristic, it is important whether the groups are dependent on each other or not. First of all, we assume that there are two independent groups. In this case, we can compare the data in the form of a 2×2 contingency table (table 6.2). A first

	0	1	Row sum
Group 1	a	b	a + b
Group 2	c	d	c + d
Col. sum	a + c	b + d	a + b + c + d

Table 6.2: Example of a 2×2 contingency table

possible test in this situation is the asymptotic **2-sample binomial test**, which is also implemented in the function `prop.test`. We compare the prevalence of liver failure in women and men.

```
## 2x2 contingency table
cont.table ← table(ICUData$sex, ICUData$liver.failure)
cont.table
```

```
          0   1
  female 168   7
  male   312  13
```

```
prop.test(c(7, 13), c(175, 325))
```

```
        2-sample test for equality of proportions without continuity
        correction
```

```
data:  c(7, 13) out of c(175, 325)
X-squared = 0, df = 1, p-value = 1
alternative hypothesis: two.sided
95 percent confidence interval:
 -0.03601123  0.03601123
sample estimates:
prop 1 prop 2
  0.04   0.04
```

Accordingly, we do not obtain significantly different prevalences for women and men. For the asymptotic calculation we can also use function `binomDiffCI` from package `"MKinfer"` (Kohl (2022b)).

```
binomDiffCI(a = 7, b = 13, c = 168, d = 312)
```

```
        wilson confidence interval (independent proportions)

95 percent confidence interval:
                                             2.5 %     97.5 %
difference of independent proportions -0.03407428 0.04349435

sample estimate:
difference of proportions
                        0

additional information:
proportion of group 1 proportion of group 2
                 0.04                  0.04
```

Since the confidence interval of the difference contains zero, that means that there are no significantly different prevalences for women and men. Bootstrap confidence intervals can also be calculated with this function.

```
binomDiffCI(a = 7, b = 13, c = 168, d = 312, method = "boot")
```

```
        boot confidence interval (independent proportions)

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 9999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, conf = 1 - alpha, type = bootci.type)

Intervals :
Level      Normal              Basic              Studentized
95%   (-0.0361,  0.0361 )   (-0.0378,  0.0347 )   (-0.0334,  0.0407 )

Level     Percentile            BCa
```

```
95%   (-0.0347,  0.0378 )   (-0.0356,  0.0360 )
Calculations and Intervals on Original Scale

sample estimate:
difference of proportions
                         0

additional information:
proportion of group 1 proportion of group 2
                 0.04                  0.04
```

Again, there are no significant differences. An alternative way of looking at contingency tables leads to hypergeometric distributions and **Fisher's exact test**. It is implemented in the function `fisher.test`. Another asymptotic possibility of the analysis is a χ^2**-test**, which can be performed using the function `chisq.test`. In case of the χ^2-test the counting frequencies in the cells should not be too small. Depending on the source, the minimum count varies between 1 and 5 per cell of the contingency table. We apply the two tests to the above situation.

```
1 ## Fisher's exact test
2 fisher.test(cont.table)
```

```
        Fisher's Exact Test for Count Data

data:  cont.table
p-value = 1
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
 0.3625184 3.0194493
sample estimates:
odds ratio
         1
```

```
1 ## chi^2 test
2 chisq.test(cont.table)
```

```
        Pearson's Chi-squared test

data:  cont.table
X-squared = 0, df = 1, p-value = 1
```

Again, there is no significance in either case. In the case of the Fisher's exact test, this means that the odds ratio for liver failure is not significantly different from 1. Hence, neither for women nor for men a significantly increased risk can be assumed. In the case of the χ^2 test, we can conclude that there is no significant (stochastic) dependence between sex and the prevalence of liver failure.

A very interesting package with a lot of statistical tests is package `"coin"` (Hothorn et al. (2006)). It contains exact, asymptotic and approximate (based on Monte Carlo simulations) tests. An approximative χ^2 test can be computed with the function `chisq_test`.

```
chisq_test(cont.table, distribution = "approximate")
```

```
        Approximative Pearson Chi-Squared Test

data:  Var2 by Var1 (female, male)
chi-squared = 0, p-value = 1
```

Also with this approximative test method based on simulations, we do not obtain a significant (stochastic) dependence. All applied methods agree that the null hypothesis is to be maintained.
The different possibilities we have introduced to study 2×2-contingency tables are based on different hypotheses that are not equivalent. In the first asymptotic approaches. the probabilities are compared directly,

$$H_0 : p_1 - p_2 = 0 \quad \text{versus} \quad H_1 : p_1 - p_2 \neq 0$$

in which p_1 and p_2 represent the true probabilities of the two groups. This is the easiest to interpret and also clearly shows, which group has a lower or higher probability.
In the case of Fisher's exact test, the result gives information about the **chance (odds)** or the **chance ratio (OR = odds ratio)**

$$H_0 : \text{OR} = 1 \quad \text{versus} \quad H_1 : \text{OR} \neq 1$$

that means if $p_1 = p_2$ resp $p_1 - p_2 = 0$, then also $\frac{p_1}{1-p_1} = \frac{p_2}{1-p_2}$ and

$$\text{OR} = \frac{\frac{p_1}{1-p_1}}{\frac{p_2}{1-p_2}} = 1$$

Unfortunately, without knowing p_1 or p_2, it is not possible to calculate the **relative risk** $\frac{p_1}{p_2}$ or p_2 resp. p_1 based on the odds ratio.
In the case of the χ^2 test, the hypotheses are based on the χ^2 statistic, which compares the observed counts with the expected counts (see equation (2.6))

$$H_0 : \chi^2 = 0 \quad \text{versus} \quad H_1 : \chi^2 > 0$$

Accordingly, there is a relation to the contingency coefficients in Definition 2.9. With the help of the function `CramerV` from the package `"DescTools"` (Andri et mult. al. (2022)) can be used to calculate a confidence interval. We choose `method = "fisheradj"` because the calculation with the standard method in this example causes problems.

```
CramerV(x = cont.table, conf.level = 0.95, method = "fisheradj")
```

```
  Cramer V     lwr.ci     upr.ci
0.00000000 0.00000000 0.08769059
```

The confidence interval contains zero, from which we can conclude that the contingency coefficient is not significantly different from zero.
Fisher's exact test and the χ^2 test can also be applied to $r \times s$ contingency tables. For large values of r and s Fisher's exact test becomes computationally intensive and it may be necessary to set the argument `workspace` to a higher value or to use p-values which are obtained by simulation. We investigate the relationship between the type of surgery and the outcome. In this case it would be necessary to clearly increase the `workspace` of Fisher's exact test. We therefore use simulated p-values instead.

```
1 cont.table ← table(ICUData$surgery, ICUData$outcome)
2 cont.table
```

```
                died home other hospital secondary care/rehab
  cardiothoracic    10   42             13                158
  gastrointestinal  18   51              7                  3
  neuro              7    6             22                 11
  other             32   52             26                 11
  trauma             2   19              4                  6
```

```
1 ## Fisher's exact test
2 fisher.test(cont.table, simulate.p.value = TRUE, B = 1e5)
```

```
        Fisher's Exact Test for Count Data with simulated p-value (based
        on 1e+05 replicates)

data:  cont.table
p-value = 1e-05
alternative hypothesis: two.sided
```

```
1 ## chi^2 test
2 chisq.test(cont.table)
```

```
        Pearson's Chi-squared test

data:  cont.table
X-squared = 259.48, df = 12, p-value < 2.2e-16
```

```
1 chisq_test(cont.table, distribution = "approximate")
```

```
        Approximative Pearson Chi-Squared Test
```

```
data:  Var2 by
         Var1 (cardiothoracic, gastrointestinal, neuro, other, trauma)
chi-squared = 259.48, p-value < 1e-04
```

```
CramerV(x = cont.table, conf.level = 0.95)
```

```
 Cramer V    lwr.ci    upr.ci
0.4159155 0.3557842 0.4581223
```

We obtain a significant association of moderate strength between the type of surgery and the outcome. By dichotomization, one could extract a 2×2 contingency tables from the contingency table and further investigate the associations with downstream tests (post hoc tests).
There are also a number of alternative tests of contingency tables; for more details see Campbell (2007).

In the case where two dependent groups are present, **McNemar's test** can be applied. It is a χ^2-test, which can be calculated with the help of the function `mcnemar.test`. Since in the ICU data set does not contain dependent measurements (e.g. before – after), we use the data from the function example for demonstration. These are data on approval of the president's performance in two consecutive successive polls, each surveying 1600 Americans of voting age.

```
Performance ← matrix(c(794, 86, 150, 570), nrow = 2,
                  dimnames = list("First" = c("Approve", "Disapprove"),
                                  "Second" = c("Approve", "Disapprove")))
Performance
```

```
            Second
First        Approve Disapprove
  Approve        794         150
  Disapprove      86         570
```

We perform the McNemar test.

```
mcnemar.test(Performance)
```

```

        McNemar's Chi-squared test with continuity correction

data:  Performance
McNemar's chi-squared = 16.818, df = 1, p-value = 4.115e-05
```

The significant result shows that the performance of the president changed significantly between the two surveys. An exact, asymptotic, and approximate McNemar test can also be performed using the function `mh_test` from the package `"coin"` (Hothorn et al. (2006)).

```
mh_test(as.table(Performance), distribution = "exact")
```

```
        Exact Marginal Homogeneity Test

data:  response by
         conditions (First, Second)
         stratified by block
chi-squared = 17.356, p-value = 3.716e-05
```

```
mh_test(as.table(Performance))
```

```
        Asymptotic Marginal Homogeneity Test

data:  response by
         conditions (First, Second)
         stratified by block
chi-squared = 17.356, df = 1, p-value = 3.099e-05
```

```
mh_test(as.table(Performance), distribution = "approximate")
```

```
        Approximative Marginal Homogeneity Test

data:  response by
         conditions (First, Second)
         stratified by block
chi-squared = 17.356, p-value < 1e-04
```

All three tests agree and also indicate a significant change in agreement. This situation can also be analyzed with the function `binomDiffCI` from the package `"MKinfer"` (Kohl (2022b)).

```
binomDiffCI(a = 794, b = 150, c = 86, d = 570, paired = TRUE)
```

```
        wilson-cc confidence interval (paired data)

95 percent confidence interval:
                                              2.5 %     97.5 %
difference of proportions (paired data) 0.02123042 0.05870387

sample estimate:
difference of proportions
                     0.04

additional information:
proportion of group 1 proportion of group 2
                 0.59                  0.55
```

The advantage here is that a statement about the direction and magnitude of the change can be made. The result is therefore a significant change (decrease) of the agreement by more than 2.1%. Instead of the asymptotic confidence intervals, bootstrap intervals can be calculated.

```
binomDiffCI(a = 794, b = 150, c = 86, d = 570, paired = TRUE, method = "boot")
```

```
        boot confidence interval (paired data)

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 9999 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, conf = 1 - alpha, type = bootci.type)

Intervals :
Level      Normal              Basic              Studentized
95%   ( 0.0213,  0.0586 )   ( 0.0212,  0.0587 )   ( 0.0214,  0.0587 )

Level     Percentile            BCa
95%   ( 0.0213,  0.0587 )   ( 0.0212,  0.0581 )
Calculations and Intervals on Original Scale

sample estimate:
difference of proportions
                     0.04

additional information:
proportion of group 1 proportion of group 2
                 0.59                  0.55
```

Accordingly, all calculated Boostrap confidence intervals yield very similar results and confirm a significant change (decrease) in agreement by at least 2.1%.

For more than two dependent groups, the **Cochran-Mantel-Haenszel χ^2 test** can be applied, which is implemented in the function `mantelhaen.test`. We again use data from the examples on the function for demonstration. It is the data of rabbits injected with streptococci, which are either immediately or after 1.5 h treated with penicillin.

```
Rabbits ← array(c(0, 0, 6, 5,
                  3, 0, 3, 6,
                  6, 2, 0, 4,
                  5, 6, 1, 0,
                  2, 5, 0, 0), dim = c(2, 2, 5),
                dimnames = list(Delay = c("None", "1.5h"),
                                Response = c("Cured", "Died"),
                                Penicillin.Level = c("1/8", "1/4", "1/2",
                                                     "1", "4")))
Rabbits
```

```
, , Penicillin.Level = 1/8

      Response
Delay  Cured Died
  None     0    6
  1.5h     0    5

, , Penicillin.Level = 1/4

      Response
Delay  Cured Died
  None     3    3
  1.5h     0    6

, , Penicillin.Level = 1/2

      Response
Delay  Cured Died
  None     6    0
  1.5h     2    4

, , Penicillin.Level = 1

      Response
Delay  Cured Died
  None     5    1
  1.5h     6    0

, , Penicillin.Level = 4

      Response
Delay  Cured Died
  None     2    0
  1.5h     5    0
```

Besides the asymptotic test, there is also an exact conditional test.

```
1 mantelhaen.test(Rabbits)
```

```

        Mantel-Haenszel chi-squared test with continuity correction

data:  Rabbits
Mantel-Haenszel X-squared = 3.9286, df = 1, p-value = 0.04747
alternative hypothesis: true common odds ratio is not equal to 1
95 percent confidence interval:
  1.026713 47.725133
sample estimates:
common odds ratio
                7
```

```
mantelhaen.test(Rabbits, exact = TRUE)
```

```
        Exact conditional test of independence in 2 x 2 x k tables

data:  Rabbits
S = 16, p-value = 0.03994
alternative hypothesis: true common odds ratio is not equal to 1
95 percent confidence interval:
   1.077401 529.837399
sample estimates:
common odds ratio
         10.36102
```

In both cases, we obtain an odds ratio that is significantly different from 1. The risk of death is significantly increased with the delay of 1.5 h in the treatment with penicillin. We can also use the function `cmh_test` from the package `"coin"`(Hothorn et al. (2006)) and obtain an asymptotic, exact, and approximative Cochran-Mantel-Haenszel test.

```
cmh_test(as.table(Rabbits))
```

```
        Asymptotic Generalized Cochran-Mantel-Haenszel Test

data:  Response by
         Delay (None, 1.5h)
         stratified by Penicillin.Level
chi-squared = 5.6571, df = 1, p-value = 0.01738
```

```
cmh_test(as.table(Rabbits), distribution = "exact")
```

```
        Exact Generalized Cochran-Mantel-Haenszel Test

data:  Response by
         Delay (None, 1.5h)
         stratified by Penicillin.Level
chi-squared = 5.6571, p-value = 0.03994
```

```
cmh_test(as.table(Rabbits), distribution = "approximate")
```

```
        Approximative Generalized Cochran-Mantel-Haenszel Test

data:  Response by
         Delay (None, 1.5h)
         stratified by Penicillin.Level
chi-squared = 5.6571, p-value = 0.0376
```

The results of the tests again indicate that immediate treatment with penicillin leads to better treatment success. Here, both a small (OR = 1.03 resp. OR = 1.08) and probably clinically less relevant as well as an enormous (OR = 47.7 resp. OR = 529.8) difference is possible. Again, similar to the case of independent groups, downstream tests ((post hoc tests) could be used to further investigate the associations by, for example, applying the corresponding tests to selected 2×2 contingency tables. This could also be used, for example, to investigate the dependence on the penicillin dose in more detail.

6.3 Ordinal and quantitative variables

In this section, I will first discuss the tests shown in Figure 6.4 in more detail. The dendrogram should also serve as an orientation to select the most appropriate test.

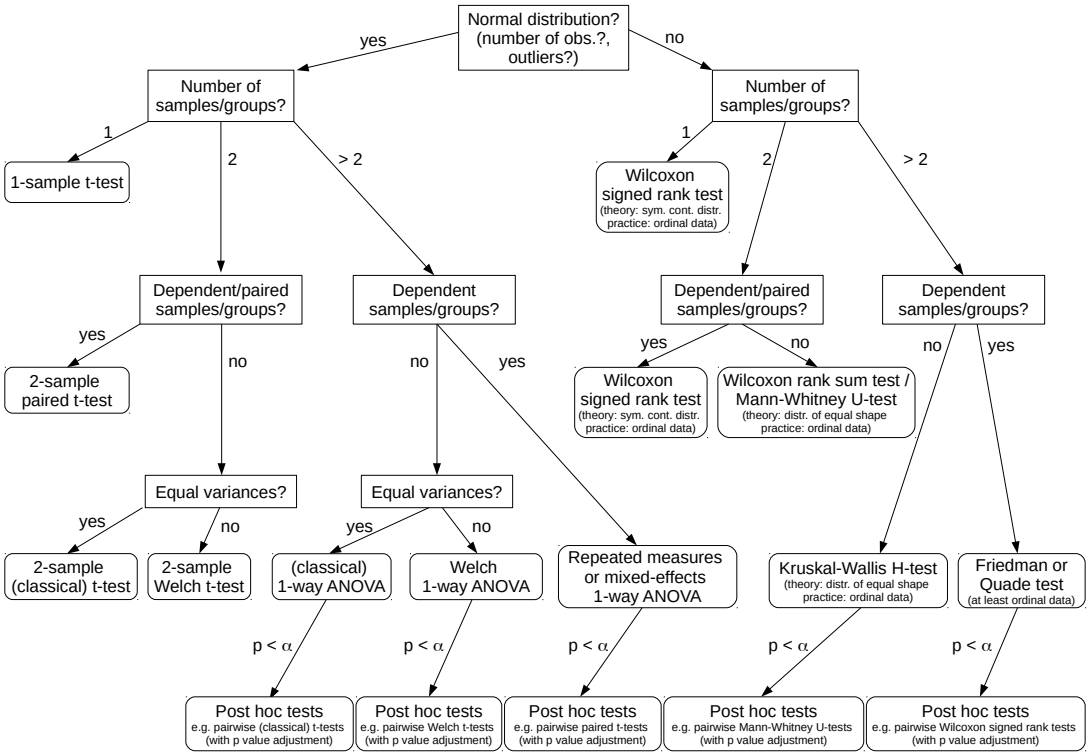


Figure 6.4: tree diagram for selecting the most appropriate test in the case of ordinal or quantitative characteristics

In the following, we will work our way through the diagram step by step. The first question to ask when selecting an appropriate test is, whether the characteristic to be examined follows a normal distribution at least approximately. If yes, this argues for the t-test (left half of the tree), but only if no outliers are expected. If outliers are to be expected, it is better to choose a rank test (right half of the tree), which has a certain robustness against outliers due to the transition to ranks. If the planned number of cases per group is very small (clearly smaller than 10), one must, to a certain extent, "believe" in a normal distribution, since in this case the non-parametric rank tests may not be significant at all, not even if the groups are perfectly separated. The larger the number of cases, the less deviations from the normal distribution are

relevant (due to the central limit theorem). With a planned number of cases of 50 or more, the normal approximation of the distribution of the arithmetic mean is so good, that even larger deviations from the normal distribution (not outliers!) are no problem for the application of t-tests.

We start with the left side of the tree and probably the most commonly used statistical test, the t-test and its variants. In the simplest case a single sample whose values are realizations of independent and Norm (μ, σ^2)-distributed random variables. One studies the unknown location parameter μ, where the variance σ^2 is unknown and thus also has to be estimated. Possible null hypotheses are for instance $\mu = \mu_0$ or $\mu \leq \mu_0$, where $\mu_0 \in \mathbb{R}$ is known and must be specified before performing the test. The corresponding test is called **one-sample t test** and can be computed by function `t.test`.
We use our ICU dataset, which is in more detail explained in Section 2.3. As we have seen in the previous sections, the maximum body temperature of ICU patients can be quite well described by a normal distribution. We investigate the hypothesis, whether the average ICU patient has an increased body temperature during their stay on the ICU, i.e.

$$H_0 : \mu \leq 37.5 \text{ versus } H_1 : \mu > 37.5$$

We apply the one-sample t test, where we omit patient 398. We can specify the one-sided alternative by argument `alternative = 'greater'` of function `t.test`. By argument `mu = 37.5`, we define the value, which we want to use for comparison.

```
1 t.test(ICUData$temperature[-398], mu = 37.5, alternative = "greater")
```

```
        One Sample t-test

data:  ICUData$temperature[-398]
t = 4.1973, df = 498, p-value = 1.599e-05
alternative hypothesis: true mean is greater than 37.5
95 percent confidence interval:
 37.63389      Inf
sample estimates:
mean of x
 37.72044
```

Based on a significance level of 5%, we can assume that in mean ICU patients have an increased body temperature during their stay on the ICU. The p value clearly increases (factor 1000 (!)), if we add patient 398, but the test still favors the alternative.

```
1 t.test(ICUData$temperature, mu = 37.5, alternative = "greater")
```

```
        One Sample t-test

data:  ICUData$temperature
t = 2.1027, df = 499, p-value = 0.01799
```

```
alternative hypothesis: true mean is greater than 37.5
95 percent confidence interval:
 37.5353     Inf
sample estimates:
mean of x
  37.6632
```

The left endpoint of the confidence interval has shifted to the left by approx. 0.1 and is now only slightly above 37.5. Although we obtain a significant result in both cases, this does not say anything about the clinical relevance. Since the mean value may be only minimally greater than 37.5°C, the clinical relevance is likely to be only small.
Because omitting patients in a clinical trial is not permitted under normal circumstances and data manipulation is quickly suspected in other contexts as well, in this case an appropriate 1-sample rank test might be recommendable. In the 1-sample case, it is the **Wilcoxon signed rank test**, which is used to examine the location of the data. In the case of symmetry, the location corresponds to the median, which in this case also coincides with the arithmetic mean.

```
wilcox.test(ICUData$temperature, mu = 37.5, alternative = "greater",
            conf.int = TRUE)
```

```
        Wilcoxon signed rank test with continuity correction

data:  ICUData$temperature
V = 69173, p-value = 0.0002349
alternative hypothesis: true location is greater than 37.5
95 percent confidence interval:
 37.59999      Inf
sample estimates:
(pseudo)median
      37.69991
```

This appears to be the asymptotic variant of the test with an additional continuity correction. Here no confidence interval is calculated by default, but additionally `conf.int = TRUE` must be set. We remove the outlier (patient 398) and test again.

```
wilcox.test(ICUData$temperature[-398], mu = 37.5, alternative = "greater",
            conf.int = TRUE)
```

```
        Wilcoxon signed rank test with continuity correction

data:  ICUData$temperature[-398]
V = 69173, p-value = 0.0001671
alternative hypothesis: true location is greater than 37.5
95 percent confidence interval:
 37.60003      Inf
```

```
sample estimates:
(pseudo)median
      37.69997
```

We see that this changes the result only very little, whereby we get a significant result in both cases; i.e., we can say that due to the (at least approximately) symmetrical situation, the median of the maximum body temperature is significantly greater than 37.5°C. However, the clinical relevance of the result remains questionable.

Two so-called connected or paired samples are present when we get two values of the same characteristic from one person at different points in time. We thus obtain pairs of values (x_i, y_i), which we define as realizations of independent and identically distributed random variables (X_i, Y_i) $(i = 1, \ldots, n, n \in \mathbb{N})$. Furthermore it holds: $D_i = X_i - Y_i \sim \mathcal{N}(\mu, \sigma^2)$, where the unknown location parameter μ is of interest and additionally the variance σ^2 is unknown. Possible null hypotheses are for example $\mu = \mu_0$ or $\mu \leq \mu_0$ for a given $\mu_0 \in \mathbb{R}$. The test is called **paired t-test** and can be computed applying function `t.test` with argument `paired = TRUE`. The paired t-test just corresponds to the 1-sample t-test for the difference of pairs of values. The same is true for the 2-sample rank test for paired samples. The test is equivalent to the Wilcoxon signed rank test. Since in the ICU data set no repeated measurements and therefore no paired data is included, we use the `sleep` dataset from the package `"datasets"` (R Core Team (2022a)), which we can load using the `data` function. The subjects were treated with two different sleep drugs and the question is whether the sleep time differs with and without medication.

```
1 data(sleep)
2 t.test(sleep$extra[1:10], mu = 0)
```

```
        One Sample t-test

data:  sleep$extra[1:10]
t = 1.3257, df = 9, p-value = 0.2176
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 -0.5297804  2.0297804
sample estimates:
mean of x
     0.75
```

```
1 t.test(sleep$extra[11:20], mu = 0)
```

```
        One Sample t-test

data:  sleep$extra[11:20]
t = 3.6799, df = 9, p-value = 0.005076
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
```

```
 0.8976775 3.7623225
sample estimates:
mean of x
     2.33
```

Apparently, only the second drug significantly increases sleep time compared to the control. The increase is at least 0.90 h. We now compare the two drugs (data pairs).

```
t.test(sleep$extra[1:10], sleep$extra[11:20], paired = TRUE)
```

```

        Paired t-test

data:  sleep$extra[1:10] and sleep$extra[11:20]
t = -4.0621, df = 9, p-value = 0.002833
alternative hypothesis: true mean difference is not equal to 0
95 percent confidence interval:
 -2.4598858 -0.7001142
sample estimates:
mean difference
          -1.58
```

The second drug is significantly better than the first drug and leads to a sleep extension of at least 0.70 h. If we consider the difference between the two drugs and perform the 1-sample t-test, we indeed obtain the identical result.

```
t.test(sleep$extra[1:10]-sleep$extra[11:20])
```

```

        One Sample t-test

data:  sleep$extra[1:10] - sleep$extra[11:20]
t = -4.0621, df = 9, p-value = 0.002833
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 -2.4598858 -0.7001142
sample estimates:
mean of x
    -1.58
```

We can also alternatively use the Wilcoxon signed rank test in this situation.

```
wilcox.test(sleep$extra[1:10], sleep$extra[11:20], paired = TRUE,
            conf.int = TRUE)
```

```

        Wilcoxon signed rank test with continuity correction
```

```
data:  sleep$extra[1:10] and sleep$extra[11:20]
V = 0, p-value = 0.009091
alternative hypothesis: true location shift is not equal to 0
95 percent confidence interval:
 -2.949921 -1.050018
sample estimates:
(pseudo)median
     -1.400031
```

Again, we get a significant result, but also a number of warnings that are suppressed here in the book. The trigger for the warnings are **bindings** in the data set; i.e., there is at least one pair of values consisting of two identical values. Since for the Wilcoxon signed rank test one assumes a continuous distribution, this should not happen and it is unclear, whether the p-value and the confidence interval are erroneous. An exact variant of the paired test, which yields correct results even in the presence of ties, is provided by function `wilcoxsign_test` of package `"coin"`(Hothorn et al. (2006)). Unfortunately, the function does not have an option for also computing the respective confidence interval. Alternatively one can use function `wilcox.exact` from package `"exactRankTests"` (Hothorn and Hornik (2022)), which is no longer actively developed, but still provides reliable results.

```
1 wilcox.exact(sleep$extra[1:10], sleep$extra[11:20], paired = TRUE,
2              conf.int = TRUE)
```

```
        Exact Wilcoxon signed rank test

data:  sleep$extra[1:10] and sleep$extra[11:20]
V = 0, p-value = 0.003906
alternative hypothesis: true mu is not equal to 0
95 percent confidence interval:
 -3.00 -1.05
sample estimates:
(pseudo)median
         -1.45
```

The values have changed slightly. There is still a significant change and a location shift (in case of symmetry: difference of the medians) of at least 1.05 h.

In the next step, we consider the situation where two unconnected (independent) samples of lengths n_1 and n_2 from $\mathcal{N}(\mu_1, \sigma^2)$ and $\mathcal{N}(\mu_2, \sigma^2)$ are present, where the unknown location parameters and at the same time the variance σ^2 (identical for both samples!) are unknown. This so-called (classical) **2-sample t-test** or **Student t-test** was discussed in detail in Example 6.5. It can be tested with the help of the function `t.test` and the argument `var.equal = TRUE`.
If additionally the variances of the two groups are different, this will lead to the so-called **Behrens-Fisher problem**, which can be solved approximately with the **Welch t-test** or with the **Hsu t-test**. Further approximations can be found with the help of permutations or bootstrap. The Welch t-test can

also be computed using function `t.test`, the Hsu t-test for example can be determined with function `hsu.t.test` from package `"MKinfer"` (Kohl (2022b)). The corresponding permutation and bootstrap test are available for example in the functions `perm.t.test` and `boot.t.test` of package `"MKinfer"` (Kohl (2022b)). It is also possible to use these functions to calculate the corresponding 1-sample and Student t-test via permutations and bootstrap, respectively.

In the following, we want to investigate, whether the maximum body temperature of females (μ_1) and males (μ_2) is significantly different. As we consider two independent groups and as we may assume a normal distribution for both groups, we can apply the two-sample t test. It is an open questions, if we may assume equal variances or not. We compute the variances of females and males.

```
## females
sd(ICUData$temperature[ICUData$sex == "female"])
```

```
[1] 1.258335
```

```
## males
sd(ICUData$temperature[ICUData$sex == "male"])
```

```
[1] 1.946092
```

The results of both groups are clearly different. However, we did not take care of the male patient 398. Hence, we recompute the standard deviations of males, where we omit patient 398.

```
## males without patient 398
sd(ICUData$temperature[-398][ICUData$sex[-398] == "male"])
```

```
[1] 1.123373
```

Again, the value of this patient has a strong impact on the result. We will once include patient 398 and once omit patient 398 in our computations. In both cases, we choose the more conservative approach and apply the Welch t-test, where the separation of the sexes is done by means of the following R formula: `temperature ~ sex`.

```
## with patient 398
t.test(temperature ~ sex, data = ICUData)
```

```
        Welch Two Sample t-test

data:  temperature by sex
t = -0.37638, df = 481.71, p-value = 0.7068
alternative hypothesis: true difference in means between group female
                        and group male is not equal to 0
95 percent confidence interval:
 -0.3368620  0.2285543
```

```
sample estimates:
mean in group female   mean in group male
            37.62800             37.68215
```

```
## without patient 398
t.test(temperature ~ sex, data = ICUData[-398,])
```

```
        Welch Two Sample t-test

data:  temperature by sex
t = -1.2514, df = 323.73, p-value = 0.2117
alternative hypothesis: true difference in means between group female
                        and group male is not equal to 0
95 percent confidence interval:
 -0.36618695  0.08144621
sample estimates:
mean in group female   mean in group male
            37.62800             37.77037
```

The body temperature of women is slightly lower on average. The results of the test (with and without patient 398), however, support the hypothesis that these are only random fluctuations. We can/must consequently assume that the maximum body temperature of men and women does not differ on average (null hypothesis). For comparison we calculate the Hsu t-test and the respective permutation and bootstrap Welch t-tests.

```
hsu.t.test(temperature ~ sex, data = ICUData[-398,])
```

```
        Hsu Two Sample t-test

data:  temperature by sex
t = -1.2514, df = 174, p-value = 0.2125
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.3669119  0.0821712
sample estimates:
mean of x mean of y   SD of x   SD of y
37.628000 37.770370  1.258335  1.123373
```

```
perm.t.test(temperature ~ sex, data = ICUData[-398,])
```

```
        Permutation Welch Two Sample t-test

data:  temperature by sex
(Monte-Carlo) permutation p-value = 0.1992
95 percent (Monte-Carlo) permutation percentile confidence interval:
 -0.34902989  0.07432804
```

```
Results without permutation:
t = -1.2514, df = 323.73, p-value = 0.2117
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.36618695  0.08144621
sample estimates:
mean in group female   mean in group male
            37.62800             37.77037
```

```
1 boot.t.test(temperature ~ sex, data = ICUData[-398,])
```

```
        Bootstrapped Welch Two Sample t-test

data:  temperature by sex
bootstrapped p-value = 0.209
95 percent bootstrap percentile confidence interval:
 -0.3686133  0.0814806

Results without bootstrap:
t = -1.2514, df = 323.73, p-value = 0.2117
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.36618695  0.08144621
sample estimates:
mean in group female   mean in group male
            37.62800             37.77037
```

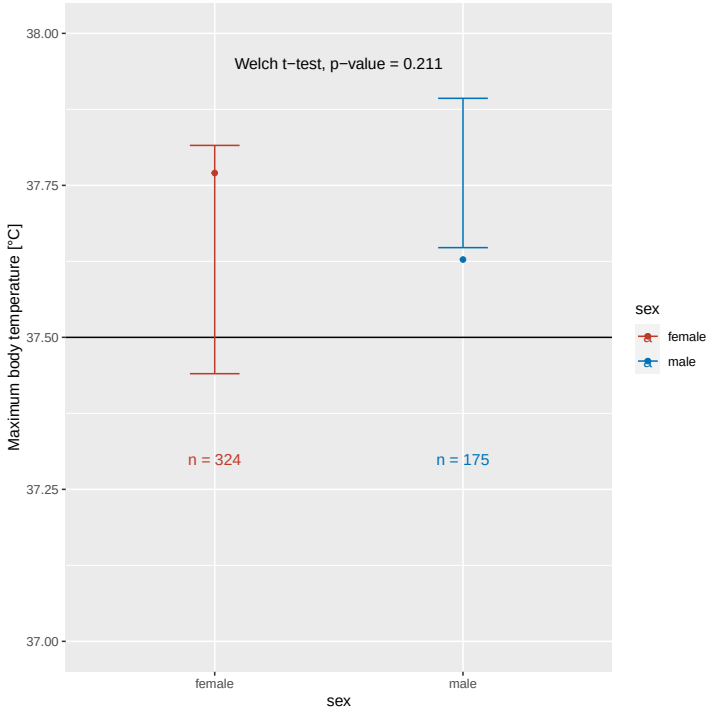
Accordingly, we obtain very similar results in all cases. We plot the result of the Welch t-test with the help of the package "ggplot2" (Wickham (2009)). One should always make sure that the graph really reflects the test. We therefore plot mean values in combination with the corresponding 95% confidence intervals. For this we first prepare a data.frame with the mean values and the upper and lower bounds of the confidence intervals. We use the function tapply to split the data according to sex and apply the respective function to it.

```
1  AM <- tapply(ICUData$temperature[-398], ICUData$sex[-398], mean)
2  CI <- tapply(ICUData$temperature[-398], ICUData$sex[-398], meanCI)
3  n <- tapply(ICUData$temperature[-398], ICUData$sex[-398], length)
4  DF <- data.frame(AM = AM,
5                   CI.lo = c(CI$male$conf.int[1],
6                             CI$female$conf.int[1]),
7                   CI.up = c(CI$male$conf.int[2],
8                             CI$female$conf.int[2]),
9                   sex = c("male", "female"),
10                  n = n)
11 DF$sex <- factor(DF$sex, levels = c("female", "male"))
12 DF
```

```
              AM    CI.lo    CI.up    sex   n
female 37.62800 37.64759 37.89315   male 175
male   37.77037 37.44026 37.81574 female 324
```

We plot the result, choosing the NEJM palette from the package "ggsci" (Xiao (2018)) and add additional text to the graph using the functions geom_text and annotate.

```
ggplot(DF, aes(x = sex, y = AM, color = sex)) +
  scale_color_nejm() + geom_point() + geom_hline(yintercept = 37.5) +
  geom_errorbar(aes(ymin = CI.lo, ymax = CI.up), width = 0.2) +
  geom_text(aes(y = c(37.3, 37.3), label = paste("n =", n))) +
  annotate(geom = "text", x = 1.5, y = 37.95,
           label = "Welch t-test, p-value = 0.211") +
  ylab("Maximum body temperature [°C]") + ylim(37.0, 38.0)
```



Note:
In practice, it is better to abandon the methodically questionable testing of the presence of equal variances and use the Welch t-test in most – if not all – cases (already in the planning phase!). The power of the Welch t-test is only marginally lower than the power of the Student t-test and conversely protects from an increase of the type I error, if unequal variances are present (Ruxton (2006), Rasch et al. (2011)).

If there are outliers or we cannot assume a normal distribution and at the same time the number of cases is small (not too small!) to moderate, then we should switch to the **Wilcoxon rank sum test**, which is also known as the **Mann-Whitney U-test**. We will abbreviate it by **WMW-test**. Strictly speaking, this test is applicable for two continuous distributions of the same shape. This implies in particular that the

variances of the two distributions should be the same as in case of the Student t-test. Empirical results, however, show that a slight violation of variance homogeneity does not affect the WMW-test. Due to the fact that the implementation of the test involves the transition to ranks, it is also considered to be adequate for ordinal data in spite of the above mentioned conditions. The test is available in function `wilcox.test`. In addition, the test is implemented in function `wilcox_test` from package `"coin"` (Hothorn et al. (2006)) and function `wilcox.exact` from package `"exactRankTests"` (Hothorn and Hornik (2022)).

We further compare the maximum body temperature of men and women.

```
## without patient 398
wilcox.test(temperature ~ sex, data = ICUData[-398,],
            conf.int = TRUE)
```

```
        Wilcoxon rank sum test with continuity correction

data:  temperature by sex
W = 26212, p-value = 0.1643
alternative hypothesis: true location shift is not equal to 0
95 percent confidence interval:
 -0.39995318  0.09997909
sample estimates:
difference in location
             -0.199997
```

```
## with patient 398
wilcox.test(temperature ~ sex, data = ICUData, conf.int = TRUE)
```

```
        Wilcoxon rank sum test with continuity correction

data:  temperature by sex
W = 26388, p-value = 0.1833
alternative hypothesis: true location shift is not equal to 0
95 percent confidence interval:
 -0.39997835  0.09992453
sample estimates:
difference in location
            -0.1000091
```

```
wilcox_test(temperature ~ sex, data = ICUData, conf.int = TRUE,
            distribution = "approximate")
```

```
        Approximative Wilcoxon-Mann-Whitney Test

data:  temperature by sex (female, male)
```

```
Z = -1.3309, p-value = 0.1845
alternative hypothesis: true mu is not equal to 0
95 percent confidence interval:
 -0.4  0.1
sample estimates:
difference in location
                 -0.1
```

```
1 wilcox.exact(temperature ~ sex, data = ICUData, conf.int = TRUE)
```

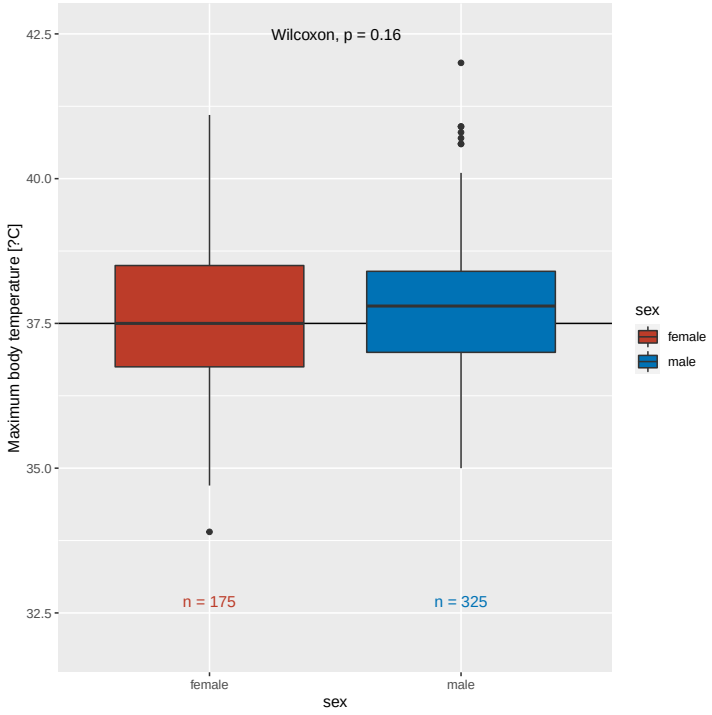
```

        Asymptotic Wilcoxon rank sum test

data:  temperature by sex
W = 26388, p-value = 0.1832
alternative hypothesis: true mu is not equal to 0
95 percent confidence interval:
 -0.39993283  0.09995573
sample estimates:
difference in location
           -0.1000092
```

The results are in agreement with the Welch t-test, although the effect of patient 398 is clearly lower than in the 1-sample case. We want to present the results in the best agreement with the test result as possible. The WMW-test is based on the **Hodges-Lehmann estimator** (HL-estimator). It is the median of all pairwise differences. In the case of the Wilcoxon signed rank test (1-sample case) one speaks of the **pseudo median**, which can be computed as the median of the mean of all pairwise sums. The pseudomedian is identical to the median under symmetry. Since we want to display the two groups side by side and the distribution of the data is almost symmetrical, we choose the median, respectivley box and whisker plots for the representation. For the addition of the test result we use the function `stat_compare_means` from the package `"ggpubr"` (Kassambara (2020)) in combinations with functions from the package `"ggplot2"` (Wickham (2009)).

```
1 ggplot(data = ICUData, aes(x = factor(sex, levels = c("female", "male")),
2                            y = temperature,
3                            fill = factor(sex, levels = c("female", "male")))) +
4   geom_hline(yintercept = 37.5) + scale_fill_nejm(name = "sex") +
5   geom_boxplot() + stat_compare_means(label.y = 42.4, label.x = 1.35) +
6   ylab("Maximum body temperature [?C]") + ylim(32, 42.5) +
7   annotate(geom = "text", x = c(1, 2), y = c(32.7, 32.7),
8            label = c("n = 175", "n = 325"), color = pal_nejm()(2)) +
9   xlab("sex")
```



As in most cases, creating a meaningful graph of the results requires some work. For example, in the definition of the factor we explicitly set the order of the levels.

Of course, there are studies in which we want to compare more than two groups with each other. We start by assuming ***k* independent groups**, which we want to examine with respect to one normally distributed characteristic. The goal is to find out whether there are differences in the mean. This is called a **1-way ANOVA**, where ANOVA stands for "ANalysis Of VArience". As in the case of the 2-sample t-test and equal variances one speaks of the classical 1-way ANOVA and calls it a **Welch-1-way ANOVA**, if the variances are allowed to be different. Both variants are implemented in the function `oneway.test`. The rank-based counterpart of the 1-way ANOVA is the **Kruskal-Wallis test**, which is sometimes called non-parametric 1-way ANOVA. The addition of non-parametric indicates that no specific distribution assumption, especially no normal distribution, is required. By the transition to ranks this analysis again offers a certain robustness against outliers. The test can be performed in R with the help of the function `kruskal.test`.

We investigate the maximum body temperature of our ICU patients with respect to their outcome. In case of the one-way ANOVA, we compare the results with and without patient 398 as well as with and without assuming equal variances. That is, we four times have to apply function `oneway.test`.

```
1 ## with patient 398, classical
2 oneway.test(temperature ~ outcome, data = ICUData, var.equal = TRUE)
```

```
        One-way analysis of means
```

```
data:  temperature and outcome
F = 1.9572, num df = 3, denom df = 496, p-value = 0.1195
```

```
## with patient 398, Welch
oneway.test(temperature ~ outcome, data = ICUData)
```

```

        One-way analysis of means (not assuming equal variances)

data:  temperature and outcome
F = 4.5435, num df = 3.00, denom df = 182.28, p-value = 0.004262
```

```
## without patient 398, classical
oneway.test(temperature ~ outcome, data = ICUData[-398,], var.equal = TRUE)
```

```

        One-way analysis of means

data:  temperature and outcome
F = 5.2203, num df = 3, denom df = 495, p-value = 0.001481
```

```
## without patient 398, Welch
oneway.test(temperature ~ outcome, data = ICUData[-398,])
```

```

        One-way analysis of means (not assuming equal variances)

data:  temperature and outcome
F = 5.3574, num df = 3.00, denom df = 189.68, p-value = 0.001458
```

Assuming equal variances and at the same time including patient 398, leads to no significant differences between the groups. In the three other cases, the mean of at least one group is significantly different from the others; that is, we can expect that the influence of outliers is largest in case of the classical one-way ANOVA.

We apply the Kruskal-Wallis test by means of function `kruskal.test`, where we compute the results with and without patient 398.

```
## with patient 398
kruskal.test(temperature ~ outcome, data = ICUData)
```

```

        Kruskal-Wallis rank sum test

data:  temperature by outcome
Kruskal-Wallis chi-squared = 15.259, df = 3, p-value = 0.001608
```

```
## without patient 398
kruskal.test(temperature ~ outcome, data = ICUData[-398,])
```

```
        Kruskal-Wallis rank sum test

data:  temperature by outcome
Kruskal-Wallis chi-squared = 15.869, df = 3, p-value = 0.001206
```

As with the other rank tests, it is once again confirmed that these procedures react only slightly to single outliers. The package "coin" (Hothorn et al. (2006)) contains the functions kruskal_test, which among other things includes an approximative version of the Kruskal-Wallis test.

```
kruskal_test(temperature ~ outcome, data = ICUData,
             distribution = "approximate")
```

```
        Approximative Kruskal-Wallis Test

data:  temperature by
         outcome (died, home, other hospital, secondary care/rehab)
chi-squared = 15.259, p-value = 0.0016
```

Overall, we can assume a significant difference between the outcome groups. Since we are dealing with more than two groups, the following questions remain: which groups differ from each other, how exactly does this difference look like and how big are the differences at all (effect sizes).

We assume that we are comparing the means, or more generally the location parameters, of more than two groups by means of a variant of 1-way ANOVA and the test yielded a significant result; i.e., the calculated p-value is smaller than the specified significance level. In this situation usually, in a second step, so-called **post hoc tests** in combination with calculations of the effect sizes and graphical representations are applied to highlight the differences between the groups. In the case of 1-way ANOVA there are a number of possibilities of post hoc tests. In my opinion the most obvious and simplest possibility is the calculation of pairwise t-tests. This is possible with the help of function pairwise.t.test. Accordingly, in the case of the Kruskal-Wallis test it is straight forward to compute pairwise WMW-tests, which can be done with function pairwise.wilcox.test. Since in this analysis several tests are performed simultaneously, one should additionally use corrections for the significance level or the p-values, to keep the type I error under control. The situation is also known as **multiple testing**, which I will discuss in more detail in Chapter 7. In R, the correction of (Bonferroni-) Holm is applied by default (Holm (1979)).

We compare the result groups pairwise with respect to the maximum body temperature using both function pairwise.t.test as well as function pairwise.wilcox.test. In the case of the t-tests we assume unequal variances (Welch t-test) and exclude patient 398 from the calculations.

```
## pairwise Welch t-tests without patient 398
pairwise.t.test(ICUData$temperature[-398], ICUData$outcome[-398],
                pool.sd = FALSE)
```

```
        Pairwise comparisons using t tests with non-pooled SD

data:  ICUData$temperature[-398] and ICUData$outcome[-398]

                     died   home   other hospital
home                 0.0410 -      -
other hospital       1.0000 0.0459 -
secondary care/rehab 1.0000 0.0036 1.0000

P value adjustment method: holm
```

```
## pairwise Wilcoxon-Mann-Whitney U tests
pairwise.wilcox.test(ICUData$temperature, ICUData$outcome)
```

```
        Pairwise comparisons using Wilcoxon rank sum test
        with continuity correction

data:  ICUData$temperature and ICUData$outcome

                     died   home   other hospital
home                 0.0534 -      -
other hospital       1.0000 0.0534 -
secondary care/rehab 1.0000 0.0024 1.0000

P value adjustment method: holm
```

The output corresponds to the pairwise (FWER-)adjusted p-values, which can be compared with the originally chosen signficance level α. A comparison with $\alpha = 0.05$ shows that in the case of the pairwise Welch t-tests, the "home" group is significantly different from all other groups. In the case of the pairwise WMW tests, two of the three comparisons with the "home" group are just slightly not significant and only the comparison with the "secondary care/rehab" group is significant. The nature of the difference (significantly smaller or larger) can be determined by calculating the arithmetic means or Hodges-Lehmann estimators, respectively. For the computations we use function `tapply` from package `"base"` (R Core Team (2022a)) and function `pairwise.fun` from package `"MKinfer"` (Kohl (2022b)).

```
## Mean values and SDs
tapply(ICUData$temperature[-398], ICUData$outcome[-398], mean)
```

```
               died                home       other hospital
           37.96176            37.44118            37.80694
secondary care/rehab
           37.85185
```

```
tapply(ICUData$temperature[-398], ICUData$outcome[-398], sd)
```

```
                died                 home       other hospital
           1.4404632            1.0674633            0.9932627
secondary care/rehab
           1.1822532
```

```
## Hodges-Lehmann estimator
pairwise.fun(ICUData$temperature[-398], ICUData$outcome[-398],
             function(x, y) wilcox.test(x, y, conf.int = TRUE)$estimate)
```

```
                           died vs home
                            -0.49997784
                 died vs other hospital
                            -0.10007520
          died vs secondary care/rehab
                            -0.09994734
                 home vs other hospital
                             0.39996128
          home vs secondary care/rehab
                             0.40003222
other hospital vs secondary care/rehab
                             0.09995271
```

According to this, the “home” group has the lowest mean and also the results for the HL estimator mean that the “home” group has the lower values. Thus, in the case of significance, we can assume that the values of the “home” group are significantly lower. For the graphical representation of the results of the Welch 1-way ANOVA we select mean values and corresponding 95% confidence intervals. This is similar to the calculations performed in case of the Welch t-test. More precisely, one would have to plot the difference of the means and the corresponding 95% confidence intervals where, strictly speaking, the confidence level would also have to be adjusted. However, the values of the individual groups, which we would like to visualize, cannot displayed this way. We prepare a `data.frame` for this with the mean values and the upper and lower limits of the respective confidence intervals. We use function `tapply`.

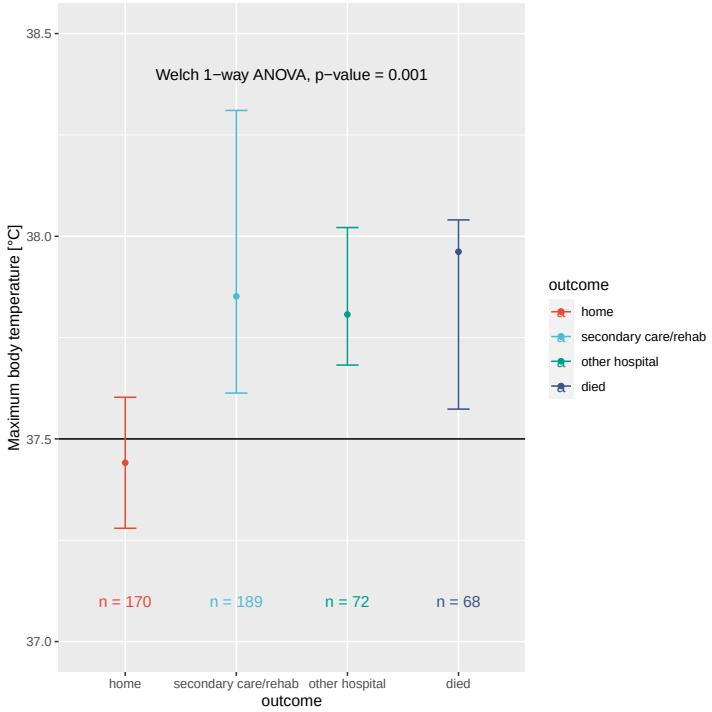
```
AM ← tapply(ICUData$temperature[-398], ICUData$outcome[-398], mean)
CI ← tapply(ICUData$temperature[-398], ICUData$outcome[-398], meanCI)
n ← tapply(ICUData$temperature[-398], ICUData$outcome[-398], length)
DF ← data.frame(AM = AM,
                CI.lo = c(CI$`other hospital`$conf.int[1],
                          CI$`home`$conf.int[1],
                          CI$`secondary care/rehab`$conf.int[1],
                          CI$died$conf.int[1]),
                CI.up = c(CI$`other hospital`$conf.int[2],
                          CI$`home`$conf.int[2],
                          CI$`secondary care/rehab`$conf.int[2],
                          CI$died$conf.int[2]),
                outcome = names(CI),
                n = n)
DF$outcome ← factor(DF$outcome, levels = c("home", "secondary care/rehab",
```

```
                                        "other hospital", "died"))
DF
```

	AM	CI.lo	CI.up	outcome	n
died	37.96176	37.57354	38.04035	died	68
home	37.44118	37.27956	37.60280	home	170
other hospital	37.80694	37.68221	38.02149	other hospital	72
secondary care/rehab	37.85185	37.61310	38.31043	secondary care/rehab	189

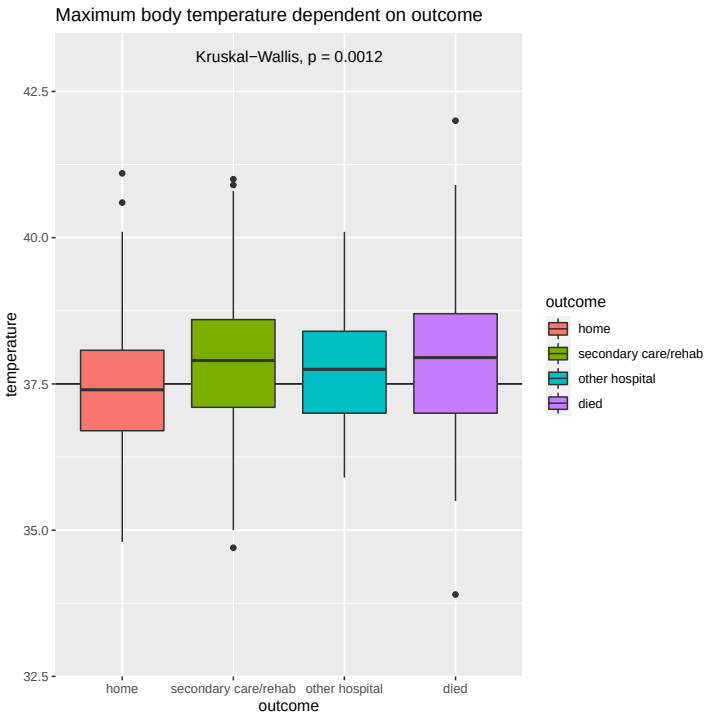
We see that the confidence intervals of the individual groups can overlap slightly and the difference is nevertheless still significant. Consequently, the presentation does not fully reflect the carried out analysis. It is also possible to conclude from this that the equal analysis of two groups has a somewhat higher power than if the two groups were considered separately. We plot the result and choose the NEJM palette from the package `"ggsci"` (Xiao (2018)) for the colors and annotate the plot with functions `geom_text` and `annotate`.

```
ggplot(DF, aes(x = outcome, y = AM, color = outcome)) +
  scale_color_npg() + geom_point() + geom_hline(yintercept = 37.5) +
  geom_errorbar(aes(ymin = CI.lo, ymax = CI.up), width = 0.2) +
  geom_text(aes(y = rep(37.1, 4), label = paste("n =", n))) +
  annotate(geom = "text", x = 2.5, y = 38.4,
           label = "Welch 1-way ANOVA, p-value = 0.001") +
  ylab("Maximum body temperature [°C]") + ylim(37.0, 38.5)
```



Assuming approximate symmetry, we plot the result of the Kruskal-Wallis test similarly to the WMW-test by means of Box-and-Whisker plots. We use function `ggplot` from package `"ggplot2"` (Wickham (2009)). We do not show the value of patient 398, who belongs to the died group.

```
ICUtmp ← ICUData[,c("outcome", "temperature")]
ICUtmp$outcome ← factor(ICUtmp$outcome, levels = c("home",
                                                   "secondary care/rehab",
                                                   "other hospital",
                                                   "died"))
ggplot(data=ICUtmp, aes(x=outcome, y=temperature, fill=outcome)) +
  geom_hline(yintercept = 37.5) + geom_boxplot() +
  stat_compare_means(label.y = 43, label.x = 2.0) +
  ggtitle("Maximum body temperature dependent on outcome") +
  ylim(c(33, 43))
```



The median of the maximum body temperature of the group that was discharged home is obviously lower than for the other groups.

> Note:
> When comparing the location of two or more independent groups (samples) that have similar variance, the power of the comparison is essentially dictated by the smallest group. Therefore, in studies, care should be taken to ensure, as far as possible, that groups of equal size are ideally compared with each other. Groups of equal size are also referred to as a **balanced design**.

In the case of groups that are dependent (connected) (e.g. measurements of the same individuals at different time points) we speak of a **1-way ANOVA with repeated measures**. We also speak of within- and between-subject factors. The within-subject factor corresponds in many cases to different time-points. While the between-subject factor describes the treatment. In addition to this classical approach, so-called mixed-effects models are used, in which the subject (within-subject factor), of which several measurements are available, is modeled as a so-called random effect. In addition, there are again rank-based nonparametric approaches that may be applied, among others the Quade or the Friedman test. The

Quade test can be regarded as an extension of the Wilcoxon signed rank test to more than two dependent groups. The **Friedman test** is more conservative and corresponds to the extension of the sign/median test (a special binomial test) to more than two related groups.

Since there are no variables with measurement repetitions in the ICU dataset, we use dataset `selfesteem` from package `"datarium"` (Kassambara (2019)), which contains data on the self-esteem of 10 individuals. For these individuals, that participate in a specific diet, self-esteem scores were measured at three different time points

```
data(selfesteem)
selfesteem
```

```
# A tibble: 10 x 4
      id    t1    t2    t3
   <int> <dbl> <dbl> <dbl>
 1     1  4.01  5.18  7.11
 2     2  2.56  6.91  6.31
 3     3  3.24  4.44  9.78
 4     4  3.42  4.71  8.35
 5     5  2.87  3.91  6.46
 6     6  2.05  5.34  6.65
 7     7  3.53  5.58  6.84
 8     8  3.18  4.37  7.82
 9     9  3.51  4.40  8.47
10    10  3.04  4.49  8.58
```

We bring the data from the given “wide” to a “long” format required for the analysis and then perform the 1-way ANOVAs presented above. Using the functions `head` and `tail`, we show the first and last six lines of the “long” data set.

```
SE.long ← data.frame(id = rep(selfesteem$id, 3),
                     score = c(selfesteem$t1, selfesteem$t2,
                               selfesteem$t3),
                     time = rep(c("t1", "t2", "t3"), each = 10))
head(SE.long)
```

```
  id    score time
1  1 4.005027   t1
2  2 2.558124   t1
3  3 3.244241   t1
4  4 3.419538   t1
5  5 2.871243   t1
6  6 2.045868   t1
```

```
tail(SE.long)
```

```
   id    score time
25  5 6.457287   t3
```

```
26  6 6.653224   t3
27  7 6.840157   t3
28  8 7.818623   t3
29  9 8.471229   t3
30 10 8.581100   t3
```

We can perform all calculations using the function `rm.oneway.test` from the package `"MKinfer"` (Kohl (2022b)).

```
## Classic repeated-measures 1-way ANOVA
rm.oneway.test(x = SE.long$score, g = SE.long$time, id = SE.long$id)
```

```

        Repeated measures 1-way ANOVA

data:  SE.long$score , SE.long$time and SE.long$id
F = 55.469, num df = 2, denom df = 18, p-value = 2.014e-08
```

```
## Mixed-effects 1-way ANOVA
rm.oneway.test(x = SE.long$score, g = SE.long$time, id = SE.long$id,
               method = "lme")
```

```

        Mixed-effects 1-way ANOVA

data:  SE.long$score , SE.long$time and SE.long$id
F = 65.261, num df = 2, denom df = 18, p-value = 5.641e-09
```

```
## Quade test
rm.oneway.test(x = SE.long$score, g = SE.long$time, id = SE.long$id,
               method = "quade")
```

```

        Quade test

data:  x, g and id
Quade F = 24.673, num df = 2, denom df = 18, p-value = 6.96e-06
```

```
## Friedman test
rm.oneway.test(x = SE.long$score, g = SE.long$time, id = SE.long$id,
               method = "friedman")
```

```

        Friedman rank sum test

data:  x, g and id
Friedman chi-squared = 18.2, df = 2, p-value = 0.0001117
```

It appears that the various methods are in agreement and there is a significant change over time. We want to investigate this change in more detail using paired t-tests and Wilcoxon signed rank tests.

```
## Pairwise paired t-tests.
pairwise.t.test(SE.long$score, SE.long$time, paired = TRUE)
```

```
        Pairwise comparisons using paired t tests

data:  SE.long$score and SE.long$time

   t1      t2
t2 0.0015  -
t3 1e-06   0.0015

P value adjustment method: holm
```

```
## Pairwise Wilcoxon signed rank tests.
pairwise.wilcox.test(SE.long$score, SE.long$time, paired = TRUE)
```

```
        Pairwise comparisons using Wilcoxon signed rank exact test

data:  SE.long$score and SE.long$time

   t1      t2
t2 0.0059  -
t3 0.0059  0.0059

P value adjustment method: holm
```

We get pairwise significant differences between all time points. Since we are dealing with paired values and we know that the paired tests are identical to the 1-sample test of the differences, we calculate the means, SD and medians for the paired differences of the time points. To do this, we use function `pairwise.fun` from package `"MKinfer"` (Kohl (2022b)), which can be used to apply any given function to pairs of groups.

```
## arithmetic means
pairwise.fun(SE.long$score, SE.long$time, function(x, y) mean(x-y))
```

```
t1 vs t2 t1 vs t3 t2 vs t3
 1.79382  4.49622  2.70240
```

```
## SDs
pairwise.fun(SE.long$score, SE.long$time, function(x, y) sd(x-y))
```

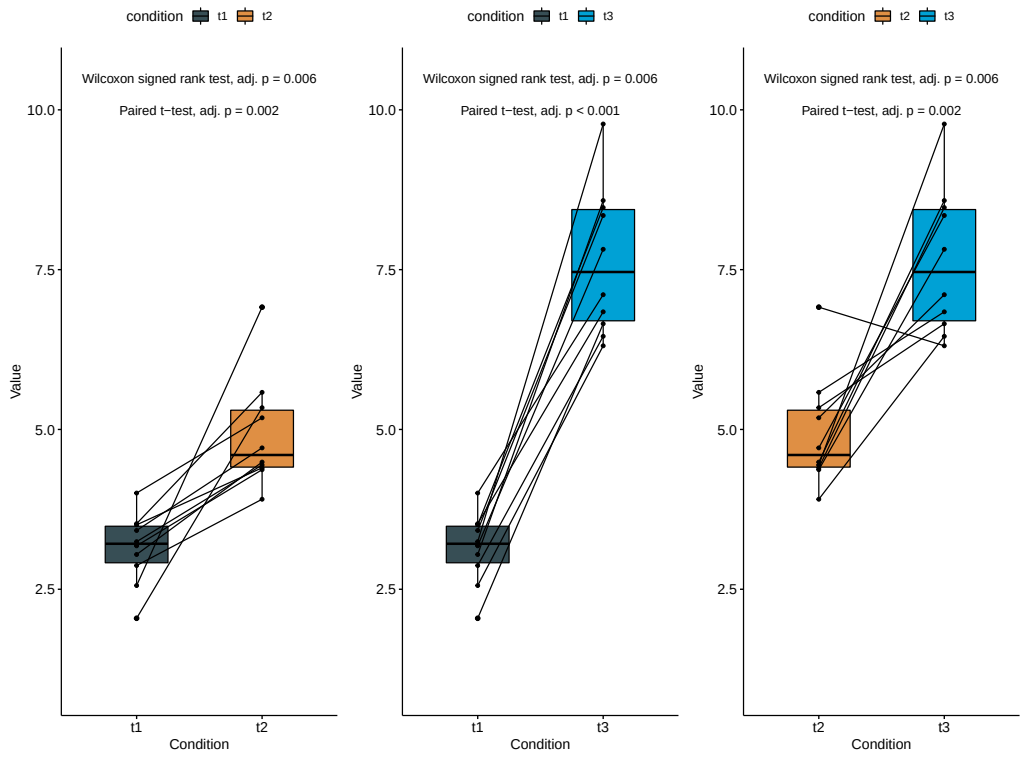
```
t1 vs t2 t1 vs t3 t2 vs t3
1.141907 1.074852 1.755559
```

```
## medians
pairwise.fun(SE.long$score, SE.long$time, function(x, y) median(x-y))
```

```
t1 vs t2 t1 vs t3 t2 vs t3
1.245675 4.623277 2.998624
```

We obtain increasing scores from t1 to t3. We plot the pairwise differences applying function `ggpaired` from package `ggpubr` (Kassambara (2020)). In addition, we use function `grid.arrange` from package `"gridExtra"` (Auguie (2017)) to arrange the three plots side by side.

```
gg1 ← ggpaired(data = selfesteem, cond1 = "t1", cond2 = "t2",
         fill = "condition") + ylim(1, 10.5) +
  scale_fill_manual(values = pal_jama()(3)[1:2]) +
  annotate(geom = "text", x = 1.5, y = 10.5,
           label = "Wilcoxon signed rank test, adj. p = 0.006") +
  annotate(geom = "text", x = 1.5, y = 10,
           label = "Paired t-test, adj. p = 0.002")
gg2 ← ggpaired(data = selfesteem, cond1 = "t1", cond2 = "t3",
         fill = "condition") + ylim(1, 10.5) +
  scale_fill_manual(values = pal_jama()(3)[c(1,3)]) +
  annotate(geom = "text", x = 1.5, y = 10.5,
           label = "Wilcoxon signed rank test, adj. p = 0.006") +
  annotate(geom = "text", x = 1.5, y = 10,
           label = "Paired t-test, adj. p < 0.001")
gg3 ← ggpaired(data = selfesteem, cond1 = "t2", cond2 = "t3",
         fill = "condition") + ylim(1, 10.5) +
  scale_fill_manual(values = pal_jama()(3)[c(2,3)]) +
  annotate(geom = "text", x = 1.5, y = 10.5,
           label = "Wilcoxon signed rank test, adj. p = 0.006") +
  annotate(geom = "text", x = 1.5, y = 10,
           label = "Paired t-test, adj. p = 0.002")
grid.arrange(gg1, gg2, gg3, nrow = 1)
```



There are also clear differences graphically between the time points, with scores clearly increasing over time for the vast majority of subjects.

With this, we have discussed all the tests of Figure 6.4 and we turn to other important statistical tests below.

In some cases, we are not interested in the means but in the variances of two independent groups. If a characteristic is present that is normally distributed in both groups, the corresponding test is based on the quotient of the two variances. According to Remark 4.28 this leads to a F-distribution and, accordingly, we speak of an **F-test**. We can perform this test with the help of the function `var.test`.
The non-parametric counterpart to the F-test, which does not require a normal distribution assumption and is based on ranks, is the **Ansari-Bradley test**. It can be applied using the function `ansari.test`.
As we have seen above, the variances of the maximum body temperature are different for females and males. We investigate, whether this is a random variation or not. We perform the test with and without patient 398.

```
## with patient 398
var.test(temperature ~ sex, data = ICUData)
```

```
        F test to compare two variances

data:  temperature by sex
F = 0.41809, num df = 174, denom df = 324, p-value = 6.066e-10
alternative hypothesis: true ratio of variances is not equal to 1
```

```
95 percent confidence interval:
 0.3236546 0.5457195
sample estimates:
ratio of variances
         0.4180863
```

```
## without patient 398
var.test(temperature ~ sex, data = ICUData[-398,])
```

```

        F test to compare two variances

data:  temperature by sex
F = 1.2547, num df = 174, denom df = 323, p-value = 0.08265
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.9711595 1.6379524
sample estimates:
ratio of variances
          1.254713
```

In the case of the variance the influence of the outlier (patient 398) is even stronger than in the case of the mean. Depending on whether we test with or without patient 398, we get a significant difference or not. The p-value changes by a factor of 10^8!
We want to investigate whether this is also true for the Ansari-Bradley test and use function `ansari.test`. In contrast to the F-Test the Ansari-Bradley test compares the scale values themselves and not their squares (variance). That means that the ratio of the scale values has to be squared in order to compare it directly with the ratio of the variances. For the calculation of the ratio of the scale values as well as the confidence interval, the position parameters of the two groups must be the same. Therefore, it is recommended to center the data at the median before the calculation. Besides the function `ansari.text` we use the function `tapply` for the calculations.

```
ICUtmp ← ICUData[,c("temperature", "sex")]
tapply(ICUtmp$temperature, ICUtmp$sex, median)
```

```
female   male
  37.5   37.8
```

```
male ← ICUtmp$sex == "male"
ICUtmp$temperature[male] ← ICUtmp$temperature[male] - 37.8
ICUtmp$temperature[!male] ← ICUtmp$temperature[!male] - 37.5
ansari.test(temperature ~ sex, data = ICUtmp, conf.int = TRUE)
```

```

        Ansari-Bradley test

data:  temperature by sex
```

```
AB = 21296, p-value = 0.388
alternative hypothesis: true ratio of scales is not equal to 1
95 percent confidence interval:
 0.9090881 1.2857440
sample estimates:
ratio of scales
       1.000041
```

```
ansari.test(temperature ~ sex, data = ICUtmp[-398,], conf.int = TRUE)
```

```
        Ansari-Bradley test

data:  temperature by sex
AB = 21210, p-value = 0.3577
alternative hypothesis: true ratio of scales is not equal to 1
95 percent confidence interval:
 0.9166452 1.3000351
sample estimates:
ratio of scales
       1.055589
```

The results show that the influence of the outlier (patient 398) is clearly smaller, as we have already seen in case of the other rank tests. In summary, we have to conclude that the hypothesis of an equal variance/scale cannot be rejected.

Another important test is to see if there is a significant correlation between two variables. If we assume that there are two normally distributed characteristics, it follows that the correlation between these two variables is linear. Consequently, in this case we can determine the strength of the correlation with the help of the Pearson correlation ρ. The associated test statistic follows a t-distribution with $n - 2$ degrees of freedom, the null hypothesis is $\rho = 0$. If we cannot assume normal distributions and must more generally assume a monotone relationship, the correlations according to Spearman and Kendall are appropriate alternatives. The corresponding three tests can be performed using the function `cor.test`. We investigate whether the correlation between maximum body temperature and maximum heart rate is significantly different from 0. We use the function `cor.test` and examine Pearson, Spearman and Kendall correlation, whereby in the case of the Pearson correlation we exclude patient 398 from the calculations.

```
## Pearson without patient 398
cor.test(ICUData$temperature[-398], ICUData$heart.rate[-398])
```

```
        Pearson's product-moment correlation

data:  ICUData$temperature[-398] and ICUData$heart.rate[-398]
t = 6.9546, df = 497, p-value = 1.118e-11
alternative hypothesis: true correlation is not equal to 0
```

```
95 percent confidence interval:
 0.2156624 0.3757592
sample estimates:
      cor
0.2978033
```

```
## Spearman
cor.test(ICUData$temperature, ICUData$heart.rate, method = "spearman")
```

```

        Spearman's rank correlation rho

data:  ICUData$temperature and ICUData$heart.rate
S = 15291695, p-value = 1.521e-09
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.2659957
```

```
## Kendall
cor.test(ICUData$temperature, ICUData$heart.rate, method = "kendall")
```

```

        Kendall's rank correlation tau

data:  ICUData$temperature and ICUData$heart.rate
z = 6.0032, p-value = 1.935e-09
alternative hypothesis: true tau is not equal to 0
sample estimates:
      tau
0.1826903
```

In all three cases we get a significant correlation. Unfortunately, we can only test whether the correlation is significantly different from 0 or is positive or negative, and not, for example, whether the correlation is is significantly different from a given value. This can be done in the case of the Pearson correlation by using the respective confidence interval. In case of the Spearman and Kendall correlation, however, no confidence intervals are calculated by function `cor.test`. For the confidence interval of the Spearman correlation we can use the function `SpearmanRho` of package `"DescTools"` (Andri et mult. al. (2022)), for the confidence interval of the Kendall correlation function `KendallTauB` of package `"DescTools"` (Andri et mult. al. (2022)).

```
## Spearman
SpearmanRho(ICUData$temperature, ICUData$heart.rate, conf.level = 0.95)
```

```
      rho    lwr.ci    upr.ci
0.2659957 0.1825635 0.3456245
```

```
## Kendall
KendallTauB(ICUData$temperature, ICUData$heart.rate, conf.level = 0.95)
```

```
    tau_b    lwr.ci    upr.ci
0.1826903 0.1254649 0.2399157
```

As we have already seen in Section 2.6.2, the Spearman correlation just matches the Pearson correlation of ranks. We can use this for the calculation of a corresponding confidence interval and get the same results as with function `SpearmanRho` with the help of function `cor.test`.

```
cor.test(rank(ICUData$temperature), rank(ICUData$heart.rate))
```

```

        Pearson's product-moment correlation

data:  rank(ICUData$temperature) and rank(ICUData$heart.rate)
t = 6.1578, df = 498, p-value = 1.521e-09
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.1825635 0.3456245
sample estimates:
      cor
0.2659957
```

Overall, there is a significant, but also only low to moderate correlation between the maximum body temperature and the maximum heart rate, which is unlikely to be of practical use. We can test with the help of the confidence intervals, whether the correlation is significantly less than 0.5, which makes the correlation of little practical use for many applications. We want to test at a significance level of $\alpha = 0.01$. Accordingly, we need to consider 99% confidence intervals. Since function `SpearmanRho` does not compute one-sided confidence intervals, we calculate a two-sided $1 - 2\alpha$ confidence interval. The upper limit of this interval is then equal to the upper limit of the corresponding one-sided interval.

```
## Pearson
cor.test(ICUData$temperature[-398], ICUData$heart.rate[-398],
         alternative = "less", conf.level = 0.99)$conf.int
```

```
[1] -1.0000000  0.3897994
attr(,"conf.level")
[1] 0.99
```

```
## Spearman
SpearmanRho(ICUData$temperature, ICUData$heart.rate, conf.level = 0.98)
```

```
      rho    lwr.ci    upr.ci
0.2659957 0.1666303 0.3600128
```

```
cor.test(rank(ICUData$temperature), rank(ICUData$heart.rate),
         alternative = "less", conf.level = 0.99)$conf.int
```

```
[1] -1.0000000  0.3600128
attr(,"conf.level")
[1] 0.99
```

```
## Kendall
KendallTauB(ICUData$temperature, ICUData$heart.rate, conf.level = 0.98)
```

```
    tau_b    lwr.ci    upr.ci
0.1826903 0.1147675 0.2506131
```

In all three cases we could say that the tests are highly significant ($\alpha = 0.01$!), since the upper limit of the confidence interval is smaller than 0.5. Indeed, the upper limits are clearly smaller than 0.5.

In the last part of this section we will briefly discuss distribution tests. We want to find out, whether we can assume that the data at hand follow a certain predefined distribution P or not. We are therefore dealing with the following hypotheses

$$H_0 : P \in \mathcal{P} \quad \text{versus} \quad H_1 : P \notin \mathcal{P}$$

Probably the most common question is whether data follows a normal distribution or not. We will restrict ourselves to this case here; that is, if

$$P \in \mathcal{P} = \left\{ \text{Norm}\,(\mu, \sigma^2) \mid \mu \in \mathbb{R}, \sigma \in (0, \infty) \right\} \tag{6.4}$$

There are several possible tests for this situation, because the alternative is very large (infinite-dimensional) and no single test can completely cover the alternative. Accordingly, it is also unclear which test has the overall highest power in this situation. Known tests implemented in R are for example: Shapiro-Wilk test, Kolmogorov-Smirnov test or Lilliefors test, Anderson-Darling test, Cramér-von-Mises test, Shapiro-Frankia test, Jarque-Bera test, D'Agostino test, etc. Besides the function `shapiro.test` from package `"stats"` (R Core Team (2022a)), we apply the functions `LillieTest`, `CramerVonMisesTest` and `ShapiroFranciaTest` from the package `"DescTools"` (Andri et mult. al. (2022)) to calculate some known tests for normal distribution for the maximum body temperature. Since these tests can react to outliers, which is quite justified, if one considers such values as a correct part of the data set, we perform the calculations without patient 398.

```
## Shapiro-Wilk test
shapiro.test(ICUData$temperature[-398])
```

```
        Shapiro-Wilk normality test

data:  ICUData$temperature[-398]
W = 0.99189, p-value = 0.008013
```

```
## Lilliefors (Kolmogorov-Smirnov) test
LillieTest(ICUData$temperature[-398])
```

```
        Lilliefors (Kolmogorov-Smirnov) normality test

data:  ICUData$temperature[-398]
D = 0.048498, p-value = 0.007001
```

```
## Cramer-von Mises Test
CramerVonMisesTest(ICUData$temperature[-398])
```

```
        Cramer-von Mises normality test

data:  ICUData$temperature[-398]
W = 0.16686, p-value = 0.01426
```

```
## Shapiro-Francia Test
ShapiroFranciaTest(ICUData$temperature[-398])
```

```
        Shapiro-Francia normality test

data:  ICUData$temperature[-398]
W = 0.99131, p-value = 0.006149
```

In this case, all tests agree and reject the normal distribution assumption. This is probably due to the rather large sample size and the fact that these tests at high sample sizes are able to detect even the smallest deviations from the given distribution with a high degree of certainty. Since the deviation tends to be small as the analysis in Section 5.2 shows, these results can be ignored with regard to the central limit theorem and one can perform a t-test or a 1-way ANOVA without restriction, at least if one omits the obvious outlier (patient 398). This is also confirmed by the fact that t-test, 1-way ANOVA, WMW-test and Kruskal-Wallis test consistently yield very similar results when patient 398 is not considered.

Note:
The use of distribution tests is not recommended in most cases, for example as pre-tests (Rasch et al. (2011)). In particular, the Kolmogorov-Smirnov test should be avoided (Schoder et al. (2006); Ghasemi and Zahediasl (2012)). These tests in most cases do not or just insufficiently solve the problem at hand. For small and medium sample sizes, where arguing with limit theorems is questionable, the tests have a low power and reliably detect only clear deviations from the given distribution. For large or very large samples, even the smallest deviations are detected. In many cases, however, such small deviations are irrelevant for the applied statistical methods, since corresponding limit theorems apply. In summary, my recommendation is therefore to use the diagnostic plots presented in Section 5.2 for model diagnostics and validation instead of various statistical tests.

6.4 Exercises

Always describe the steps of your analysis and interpret the results. In case of exercises 4–15, 17 and 18 use the ICU dataset and choose appropriate functions for the computations.

1. In a randomized and controlled trial, a new treatment to avoid the infection with HIV was studied. There were no significant differences between the new treatment and a control group. The ratio between newly occurring infections was 1.0, where the 95% confidence interval was [0.63, 1.58]. Based on this result, can you be sure that the new treatment has no effect? In this context, what could be the meaning of "Absence of Evidence Is Not Evidence of Absence"?

2. Those responsible for the FranSO study (Kemmler et al. (2017)) assumed for the sample size calculation of the primary endpoint (sarcopenia z-score), a difference of means of 1.0 and a standard deviation of 1.4. Calculate the sample size using Student's t-test. Use a significance level of 5% and a power of 80%. How much does the sample size increase if you increase the power to 90%? The study ultimately included 33 respectively, 34 subjects per group. How large is the power of the study if you assume a sample size of 33 per group?

3. For a study that shall be planned, we expect a difference in means of 0.70 for the selected primary endpoint, assuming that the standard deviation of the control group will be 0.75 and the standard deviation of the treatment group will be 1.40. For the sample size calculation, use the Welch t-test with a significance level of 5% and a power of 90%. How does the sample size change if you use the Hsu t-test instead? What sample size do you get for the WMW-test? Also, examine how the sample size changes, if the difference between the means is actually slightly smaller or larger than expected. To do this, consider the interval [0.60, 0.80].
   Use the functions `power.welch.t.test`, `power.hsu.t.test` and `sim.ssize.wilcox.test` from the package `"MKpower"` (Kohl (2020c)). Use `sim.ssize.wilcox.test` in combination with function `rnorm`. For more details on how to use these functions, look at the help pages for the functions and in the vignette of the package, which you can open via `vignette("MKpower")`.

4. Investigate the assumption that more men than women are treated at the ICU. Formulate the hypotheses and use both an exact test and an asymptotic test to verify this. Plot the data appropriately with the test result.

5. Investigate, whether males die more frequently on the ICU than females. Compute an exact as well as an asymptotic test to check this. As starting point, use the 2×2 contingency table generated by

   ```
   1 table(ICUData$sex, ICUData$outcome == "died")
   ```

   Plot the data appropriately with the test result.

6. Investigate whether patients in an ICU are, on average, significantly older than 60 years of age. Use the 1-sample t-test as well as the Wilcoxon signed rank test for the analysis. Which test seems more appropriate? Plot the data appropriately with the test result.

7. Assume a normal distribution for the logarithmized bilirubin values and compare the mean log-concentrations of bilirubin for the ICU patients with and without liver failure applying t-tests. Do you think the classical or the Welch t-test is more appropriate? In a second step, apply the Wilcoxon-Mann-Whitney U-test and compute the test once with and once without taking the logarithm of the bilirubin values. Compare the two results. What do you detect? What is the reason for it? Plot the data appropriately with the test result.

8. Compare the average length of stay of females and males by applying the Wilcoxon-Mann-Whitney U-test. Plot the data appropriately with the test result.

9. Assume that the maximum body temperature of the ICU patients can be described by a normal distribution. Investigate, whether the mean values of the different surgical groups differ significantly. Use the Welch 1-way ANOVA for the analysis and disregard patient 398. If you obtain a significant result, examine the difference in more detail using appropriate post hoc tests. Plot the data appropriately with the test result.

10. Consider only patients with a length of stay greater than one day (`LOS` > 1).

```
ICUData.LOS2 ← ICUData[ICUData$LOS > 1, ]
```

    Assume that the log bilirubin values of ICU patients can be described by a normal distribution. Use the Welch 1-way ANOVA to find out whether the mean values of the logarithmized bilirubin values are different for the surgical groups. If you obtain a significant result, investigate the difference in more detail using appropriate post hoc tests. Plot the data appropriately with the test result.

11. Assume that the maximum measured heart rate of the ICU patients can approximately be described by a normal distribution. Investigate, using a Welch 1-way ANOVA, whether the mean heart rates of the different outcome groups differ significantly. If you obtain a significant result, examine the differences in more detail using post hoc tests. Plot the data appropriately with the test result.

12. Compare the mean length of stay of the different surgical groups. Use the Kruskal-Wallis test for this purpose. If you obtain a significant result, examine the differences more closely using post hoc tests. Plot the data appropriately with the test result.

13. Consider only patients with a length of stay greater than one day (`LOS` > 1).

```
ICUData.LOS2 ← ICUData[ICUData$LOS > 1, ]
```

    Apply the Kruskal-Wallis test to find out whether the outcome groups differ significantly with respect to their mean SAPS-II scores. If you obtain a significant result, examine the differences in more detail using post hoc tests. Plot the data appropriately with the test result.

14. Consider the SAPS II scores of the ICU patients and investigate, whether the mean scores differ between the different surgical groups. Use the Kruskal-Wallis test for this purpose. If you obtain a significant result, examine the differences in more detail using post hoc tests. Plot the data appropriately with the test result.

15. Consider only patients with a length of stay greater than one day (`LOS` > 1).

```
ICUData.LOS2 ← ICUData[ICUData$LOS > 1, ]
```

Apply the Kruskal-Wallis test to find out whether the outcome groups differ significantly with respect to their mean ages. If you obtain a significant result, examine the differences in more detail using post hoc tests. Plot the data appropriately with the test result.

16. Consider the following data set:

```
id:  1,    2,    3,    4,    5,    6,    7,    8,    9,    10
t0:  8.8,  8.9,  9.4,  8.7,  9.2, 10.9,  9.5, 11.2, 10.6,  9.2
t1: 10.6, 11.3, 11.8, 11.1, 11.8, 12.6, 11.7, 13.0, 12.9, 10.6
t2: 12.9, 13.1, 13.2, 13.2, 13.8, 14.7, 13.8, 15.0, 14.7, 13.1
```

Measurements were taken from 10 subjects at three time points. Create a `data.frame` with the data. Investigate with the help of a repeated-measures 1-way ANOVA to see if there are significant differences between the time points. Also, use a mixed-effects 1-way ANOVA as well as the Quade and the Friedman test for the analysis. If significant results are found, perform post hoc tests with paired t-tests and Wilcoxon signed rank tests, respectively. Plot the data appropriately with the test result.

17. Investigate whether the log bilirubin values of men and women differ with respect to their scale. Use both the F-test and the Ansari-Bradley test. Plot the data appropriately with the test result.

18. Using an appropriate test to investigate whether there is a significant correlation between age and SAPS II score. Which correlation coefficient seems appropriate for this situation and why? Plot the data appropriately with the test result.

19. Use the `chem` dataset of package `"MASS"` (Venables and Ripley (2002)), which can be loaded by the following R code

```
library(MASS)
data(chem)
```

Use the Shapiro-Wilk test as well as the Lilliefors (Kolmogorov-Smirnov), the Cramer-von Mises and the Shapiro-Francia test of package `"DescTools"` (Andri et mult. al. (2022)) to verify, whether the data follows a normal distribution. What do you observe? In addition, use a qq-plot to check the assumption of normality, which can be generated by functions `qqnorm` and `qqline`. Do you think the plot confirms the tests? Repeat the tests of normality, but this time omit observation 17. Compare and interpret all results.

7 Multiple testing

In this chapter, we discuss the multiple testing problem. In detail the following topics are discussed:

- Family-wise error rate (FWER)
- Bonferroni method
- (Bonferroni-) Holm method
- Simes-Hochberg method
- sample size calculations for multiple primary endpoints
- False discovery rate (FDR)
- Benjamini-Hochberg method
- Benjamini-Yekutieli method
- Volcano plot

The R code for this chapter is contained in file `MultiplesTesten.Rmd`, which is downloadable from my GitHub account (link: https://github.com/stamats/ISDR/blob/main/MultipleTesting.Rmd). Right-click on `Raw` and save the file. The least difficulties arise if you save my R Markdown files in the same folder as the data used in the respective chapters.
We first install the packages needed in this chapter.

```
BiocManager::install("multtest", update = FALSE)
BiocManager::install("limma", update = FALSE)
install.packages("MKomics")
```

Make sure that you have already installed the packages from the previous Chapters 2–6. We load all packages required in this chapter.

```
library(ggplot2)
library(MKpower)
library(multtest)
library(MKomics)
library(MKinfer)
```

As explained in Section 2.4, running `library` repeatedly is unproblematic.

7.1 Introduction

In Chapter 6 we already learned about some of the pitfalls of statistical testing. However, the assumption there was that you just run one statistical test and the result of this test describes the success or failure of a study. In practice, this is still often the case. In the context of clinical trials, for example, one speaks of the **primary endpoint**, which is still often described by a single parameter. However, there are also more and more situations where this is no longer considered adequate. For example, in studies of migraine drugs, the following four parameters should be used: pain, nausea, sensitivity to light, and sensitivity to sound (Offen et al. (2007)). Accordingly, an increasing number of studies are investigating not only one but several (primary) hypotheses simultaneously.

In addition, high-throughput methods have been developed in recent years and decades, for example in biology and medicine, that allow the simultaneous measurements of hundreds or thousands of parameters (e.g. genes, proteins or metabolites). A large number of variables are also determined simultaneously on the internet or by smartphone apps and their relevance, for e.g. the placement of advertisements or shopping behavior is being investigated. In this context, we speak of so-called **"big data"**, which is then analyzed by a bioinformatician or a **"data scientist"** and visualized.

In many fields today not only one but $N > 1$ ($N \in \mathbb{N}$) hypotheses are considered simultaneously and accordingly not only one but N statistical tests have to be performed simultaneously. We already know:

- for a single test we have an type I error of $\alpha \in (0, 1)$
- If we test $N > 1$ hypotheses, each at the same significance level α, we have for each of the N tests again a type I error of α.

However, we do not know how large the error probability is for all N tests together, or, more fundamentally, how the concept of type I error can be canonically applied to N tests. In the following we will discuss the two most important approaches in multiple testing.

7.2 Family-wise Error Rate (FWER)

In the case of the **family-wise error rate (FWER)** one considers the probability of making at least one type I error in $N \in \mathbb{N}$ tests

$$\text{FWER} = P(\text{“at least 1 type I error”}) \tag{7.1}$$

The exact calculation of this probability is generally difficult. In the case, that the N hypotheses are (stochastically) independent from each other and all N statistical tests are performed with the same significance level $\alpha \in (0, 1)$, exact calculations are possible. We first use the counter probability

$$\text{FWER} = P(\text{“at least 1 type I error”}) = 1 - P(\text{“no type I error”}) \tag{7.2}$$

where in the case of a single test

$$N = 1: \qquad P(\text{“no type I error”}) = \text{sensitivity} = 1 - \alpha \tag{7.3}$$

In the case of N (stochastic) independent tests, it follows that

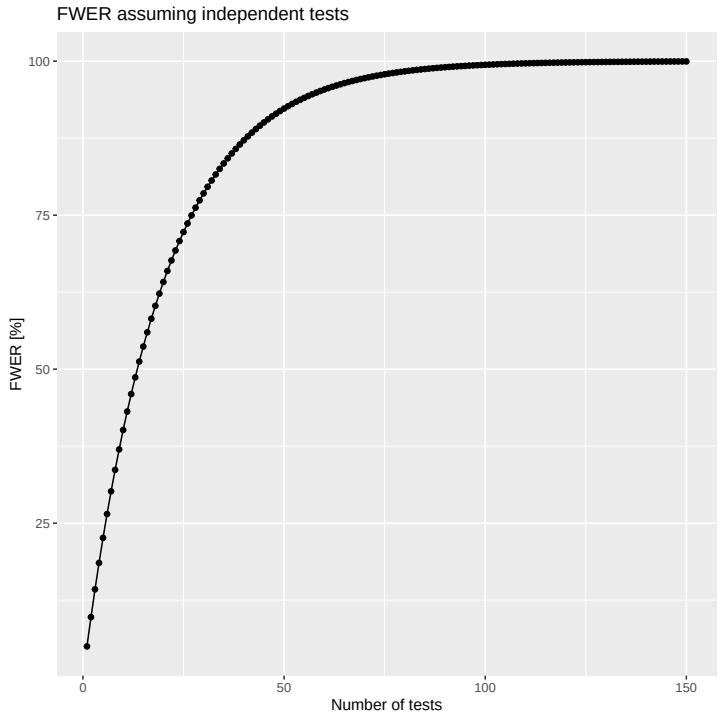
$$P(\text{“no type I error”}) = (1-\alpha)^N \tag{7.4}$$

This gives

$$\text{FWER} = 1-(1-\alpha)^N \tag{7.5}$$

We plot the situation for $\alpha = 0.05$.

```
N ← 1:150
FWER ← 100*(1 − (1−0.05)^N)
DF ← data.frame(N = N, FWER = FWER)
ggplot(DF, aes(x = N, y = FWER)) +
  geom_point() + geom_line() + xlab("Number of tests") + ylab("FWER [%]") +
  ggtitle("FWER assuming independent tests")
```



The FWER thus increases quite rapidly with the number of tests and we get for example

```
round(DF[c(1,2,3,5,10,14,45,59,90),], 1)
```

```
    N FWER
1   1  5.0
2   2  9.8
3   3 14.3
5   5 22.6
10 10 40.1
14 14 51.2
45 45 90.1
59 59 95.2
90 90 99.0
```

Unfortunately, the assumption of independent hypotheses and tests is not fulfilled in practice in the vast majority of cases. In addition, the exact dependency structure between the hypotheses is difficult to describe in practice or is not known. Thus, the exact calculation of the FWER is usually not possible and one can only use estimations (by upper bounds). The simplest and also generally most valid estimation follows directly from the property of the so-called sub-additivity of probability measures

$$P\left(\bigcup_{i=1}^{N} A_i\right) \leq \sum_{i=1}^{N} P(A_i) \tag{7.6}$$

for any (measurable) events $A_1, \ldots, A_N$. From this we get with

$$P(A_i) = P(\text{“type I error of test } i\text{”}) = \alpha \qquad \forall i = 1, \ldots, N \tag{7.7}$$

the following upper bound for the FWER

$$\text{FWER} \leq \min\{N\alpha,\ 1\} \tag{7.8}$$

Since $N\alpha > 1$ for $N > \frac{1}{\alpha}$, the minimum of $N\alpha$ and 1 is considered to obtain a valid probability ($\in [0, 1]$) in each case. Instead of α we use an **adjusted significance level** $\tilde{\alpha} < \alpha$ for each test. In particular, by defining $\tilde{\alpha} = \frac{\alpha}{N}$, we get

$$\text{FWER} \leq \min\{N\tilde{\alpha},\ 1\} = N\tilde{\alpha} = N\frac{\alpha}{N} = \alpha \tag{7.9}$$

This approach can also be applied to the p values $p_1, \ldots, p_n$ of the tests, since it is equivalent to adjust the p values instead of the significance levels. That is, instead of comparing p_i with $\tilde{\alpha}$, one compares $\tilde{p}_i = \min\{Np_i,\ 1\}$ with α.

Note:
In practice, one usually adjusts p values in multiple testing situations (not the significance level). The only important exception is the case of sample size calculation. Here it is most of the time easier to work with the adjustment of the significance level.

The approach outlined above is known as the **Bonferroni method**

Algorithm 7.1. *Let there be $N \in \mathbb{N}$ statistical tests at significance level $\alpha \in (0, 1)$ with null hypotheses $H_{0,1}, \ldots, H_{0,N}$ and corresponding alternatives $H_{1,1}, \ldots, H_{1,N}$, which lead to the p values $p_1, \ldots, p_N$. The **Bonferroni method** is the following one-step method*

Replace p_i by $\tilde{p}_i := \min\{Np_i, 1\}$ $(\forall i = 1, \ldots, N)$.

If $\tilde{p}_i \leq \alpha$, choose $H_{1,i}$, otherwise keep $H_{0,i}$ $(\forall i = 1, \ldots, N)$.

Note:
The Bonferroni method does not require knowledge about the dependence structure of the hypotheses. This advantage on the other hand makes the Bonferroni method very conservative; i.e., one has a high certainty that one does not make any false positive decisions. However, this also means that many correct positive tests may be classified as non-significant (false negative tests). The power of this method is therefore comparatively small and the type II error comparatively large, respectively.

In addition to the Bonferroni method, there are somewhat less conservative so-called multi-step methods to control the FWER. For these methods the p values have to be sorted in ascending order in a first step

$$p_{(1)} \leq p_{(2)} \leq \cdots \leq p_{(N)}$$

The main alternative to the Bonferroni method is the method of Holm (1979), which is also known as **Bonferroni-Holm method**.

Algorithm 7.2. *Let there be $N \in \mathbb{N}$ statistical tests at significance level $\alpha \in (0, 1)$ with null hypotheses $H_{0,1}, \ldots, H_{0,N}$ and corresponding alternatives $H_{1,1}, \ldots, H_{1,N}$ which lead to the p values $p_1, \ldots, p_N$. Let $p_{(1)}, \ldots, p_{(N)}$ be the p values sorted in ascending order and $H_{0,(j)}$ and $H_{1,(j)}$ $(j = 1, \ldots, N)$ be the corresponding hypotheses. The **(Bonferroni-) Holm method** is applied for $j = 1, \ldots, N$ in a stepwise manner starting with the smallest p value. The adjusted p value is calculated as follows*

Replace $p_{(j)}$ by

$$\tilde{p}_{(j)} = \max_{k=1,\ldots,j} \left\{ \min \left\{ (N - k + 1) p_{(k)}, 1 \right\} \right\} \tag{7.10}$$

If $\tilde{p}_{(j)} \leq \alpha$, choose $H_{1,(j)}$, otherwise keep $H_{0,(j)}$.

It would also be possible to integrate an early stop into the algorithm.

Remark 7.3. *If one obtains an adjusted p value that is greater than the predefined significance level, one could also stop the (Bonferroni-) Holm procedure, since all subsequent adjusted p values will in any case be at most even larger. In practice, however, all p values are usually adjusted. Therefore, this optional early stop was not included in the above algorithm.*

Note:
The (Bonferroni-) Holm method is a so-called **step-down method**: starting from the smallest p value, larger p values are considered step by step. This is referred to as step-down, because the test statistic becomes smaller with greater p value, that means the test statistic steps down. This method is somewhat less conservative (i.e., has greater power) than the Bonferroni method.

We consider a simple example.

Example 7.4. Let there be p values of six tests at a significance level of 5% with null hypotheses $H_{0,1}, \ldots, H_{0,6}$ and alternatives $H_{1,1}, \ldots, H_{1,6}$. The corresponding p values are

$$p_1 = 0.004,\ p_2 = 0.011,\ p_3 = 0.039,\ p_4 = 0.012,\ p_5 = 0.001,\ p_6 = 0.480$$

Without an adjustment of the p values, one would choose the alternative in five of the six cases at a significance level of 5%.

(a) The Bonferroni method yields

$$\begin{aligned}
\tilde{p}_1 &= 6p_1 = \mathbf{0.024} \\
\tilde{p}_2 &= 6p_2 = 0.066 \\
\tilde{p}_3 &= 6p_3 = 0.234 \\
\tilde{p}_4 &= 6p_4 = 0.072 \\
\tilde{p}_5 &= 6p_5 = \mathbf{0.006} \\
\tilde{p}_6 &= 6p_6 = 2.880 \Rightarrow 1.000
\end{aligned}$$

We compare our results with the results of function `p.adjust`.

```
pval ← c(0.004, 0.011, 0.039, 0.012, 0.001, 0.480)
p.adjust(pval, method = "bonferroni")
```

```
[1] 0.024 0.066 0.234 0.072 0.006 1.000
```

i.e., "only" $H_{0,1}$ and $H_{0,5}$ can be rejected at the 5% level.

(b) We first sort the p values

$$p_5 \leq p_1 \leq p_2 \leq p_4 \leq p_3 \leq p_6$$

We calculate the adjusted p values based on this sequence using the (Bonferroni-) Holm method

$$\begin{aligned}
\tilde{p}_5 &= 6p_5 = \mathbf{0.006} \\
\tilde{p}_1 &= 5p_1 = \mathbf{0.020} \\
\tilde{p}_2 &= 4p_2 = \mathbf{0.044} \\
\tilde{p}_4 &= 3p_4 = 0.036 \Rightarrow \mathbf{0.044} \\
\tilde{p}_3 &= 2p_3 = 0.078 \\
\tilde{p}_6 &= 1p_6 = 0.480
\end{aligned}$$

In the case of $\tilde{p}_4$, the calculated value of 0.036 would be below the the previous value of $\tilde{p}_2$. This is not allowed and is compensated for by the maximum in equation (7.10). We verify the results again using function `p.adjust`, where a sorting of the p values is not necessary, but is done within the function.

```
## sorting not necessary
p.adjust(pval, method = "holm")
```

```
[1] 0.020 0.044 0.078 0.044 0.006 0.480
```

Thus, we can reject the null hypotheses $H_{0,5}$, $H_{0,1}$, $H_{0,2}$ and $H_{0,4}$ at the 5% level. This shows that the (Bonferroni-) Holm method is less conservative than the Bonferroni method.

Another well-known adjustment method and in a certain sense a counterpart to the (Bonferroni-) Holm method is the **Simes-Hochberg method**.

Algorithm 7.5. *Let there be $N \in \mathbb{N}$ **independent** statistical tests at a significance level $\alpha \in (0,1)$ with null hypotheses $H_{0,1}, \ldots, H_{0,N}$ and associated alternatives $H_{1,1}, \ldots, H_{1,N}$ which lead to the p values $p_1, \ldots, p_N$. Let $p_{(1)}, \ldots, p_{(N)}$ be the p values sorted in ascending order and $H_{0,(j)}$ and $H_{1,(j)}$ $(j = 1, \ldots, N)$ be the corresponding hypotheses. The **Simes-Hochberg method** for $j = N, \ldots, 1$ consists of the following calculation rule*

Replace $p_{(j)}$ by

$$\tilde{p}_{(j)} = \min_{k=j,\ldots,N} \left\{ \min\{(N-k+1)p_{(k)}, 1\} \right. \tag{7.11}$$

If $\tilde{p}_{(j)} \leq \alpha$, choose $H_{1,(j)}$, otherwise keep $H_{0,(j)}$.

Again, it would be possible to include an early stop in the algorithm.

Remark 7.6. *If, starting with the largest p value, one obtains an adjusted p value that is smaller than the given significance level, one could also stop the procedure, since all following adjusted p values will in any case be at most even smaller. In practice, however, all p values are usually adjusted. Therefore, this optional early stop was not included in the above algorithm.*

Note:
The Simes-Hochberg Method is a so-called **step-up procedure**. One starts with the largest p value and steps up the test statistic. In general, step-up procedures are less conservative than the corresponding step-down procedures. This is also true for the Simes-Hochberg method, which is somewhat less conservative than its step-down counterpart, the (Bonferroni-) Holm method. However, it requires independent tests, which is a rather strong requirement that in most of the cases is not fulfilled in practice.

We consider the situation of Example 7.4 in the following example.

Example 7.7. We additionally have to assume that the tests are independent. We apply the Simes-Hochberg procedure to calculate the adjusted p values, where as a reminder

$$p_5 \leq p_1 \leq p_2 \leq p_4 \leq p_3 \leq p_6$$

We get

$$\begin{aligned}
\tilde{p}_6 &= 1p_6 = 0.480 \\
\tilde{p}_3 &= 2p_3 = 0.078 \\
\tilde{p}_4 &= 3p_4 = \mathbf{0.036} \\
\tilde{p}_2 &= 4p_2 = 0.044 \Rightarrow \mathbf{0.036} \\
\tilde{p}_1 &= 5p_1 = \mathbf{0.020} \\
\tilde{p}_5 &= 6p_5 = \mathbf{0.006}
\end{aligned}$$

In the case of $\tilde{p}_2$ the calculated value of 0.044 would be above the the previous value of $\tilde{p}_4$. This is not allowed and is prevented by the outer minimum in equation (7.11). We check our calculations again using function `p.adjust`.

```
## sorting not necessary
p.adjust(pval, method = "hochberg")
```

```
[1] 0.020 0.036 0.078 0.036 0.006 0.480
```

Thus, as in case of the (Bonferroni-) Holm method, we can reject the null hypotheses $H_{0,5}$, $H_{0,1}$, $H_{0,2}$ and $H_{0,4}$ at the 5% level. However, $\tilde{p}_2$ and $\tilde{p}_4$ are slightly smaller than in the case of (Bonferroni-) Holm. This shows that the Simes-Hochberg method is is somewhat less conservative than the (Bonferroni-) Holm method. However, this is quite expensively bought by the additional requirement of independent tests.

Note:
In practice, it is in most cases recommended to use the (Bonferroni-) Holm method for the adjustment of p values in order to reliably control the FWER. It is less conservative than the Bonferroni's method and, in contrast to the Simes-Hochberg method, can also be used for can also be used for dependent tests.

Finally, we consider an example of sample size calculation in a situation with multiple primary endpoints.

Example 7.8. We assume that we want to consider three (primary) endpoints simultaneously, comparing two groups. The study is successful, if we obtain a significant difference for all three endpoints. For simplicity, we further assume that the three parameters reflecting the primary endpoints follow a normal distribution. The groups can each have different standard deviations. Consequently, we decide to use the Welch t-test for the sample size calculation. We expect the following situations for the three endpoints:

endpoint 1: $|\mu_1 - \mu_2| = 0.5, \sigma_1 = 1, \sigma_2 = 1.2$

endpoint 2: $|\mu_1 - \mu_2| = 0.75, \sigma_1 = 1.5, \sigma_2 = 1.2$

endpoint 3: $|\mu_1 - \mu_2| = 1.0, \sigma_1 = 1.5, \sigma_2 = 1.75$

We want to achieve a power of at least 90% in the study. Furthermore the FWER should be at most 5%. For the sample size calculation, the Bonferroni method is usually easiest to use in such cases. Consequently, we should perform the tests with a type I error of $\alpha = 0.05/3$. This leads to

```
## endpoint 1
power.welch.t.test(delta = 0.5, sd1 = 1.0, sd2 = 1.2,
                   sig.level = 0.05/3, power = 0.9)
```

```
     Two-sample Welch t test power calculation

              n = 133.3424
          delta = 0.5
            sd1 = 1
            sd2 = 1.2
      sig.level = 0.01666667
          power = 0.9
    alternative = two.sided

NOTE: n is number in *each* group
```

```
## endpoint 2
power.welch.t.test(delta = 0.75, sd1 = 1.5, sd2 = 1.2,
                   sig.level = 0.05/3, power = 0.9)
```

```
     Two-sample Welch t test power calculation

              n = 90.13938
```

```
           delta = 0.75
             sd1 = 1.5
             sd2 = 1.2
       sig.level = 0.01666667
           power = 0.9
     alternative = two.sided

NOTE: n is number in *each* group
```

```
## endpoint 3
power.welch.t.test(delta = 1.0, sd1 = 1.5, sd2 = 1.75,
                   sig.level = 0.05/3, power = 0.9)
```

```
     Two-sample Welch t test power calculation

              n = 73.25381
          delta = 1
             sd1 = 1.5
             sd2 = 1.75
       sig.level = 0.01666667
           power = 0.9
     alternative = two.sided

NOTE: n is number in *each* group
```

We obtain the highest number of cases for the first endpoint. Hence, the study should be conducted with a sample size of 134 subjects per group, to achieve the desired power of at least 90% and to keep an FWER of at most 5%. We can use the function `power.welch.t.test` to calculate not only the sample size, but also, starting from a sample size the corresponding power. For a sample size of 134, we calculate the power for the second and third endpoints.

```
## endpoint 2
power.welch.t.test(delta = 0.75, sd1 = 1.5, sd2 = 1.2,
                   sig.level = 0.05/3, n = 134)
```

```
     Two-sample Welch t test power calculation

              n = 134
          delta = 0.75
             sd1 = 1.5
             sd2 = 1.2
       sig.level = 0.01666667
           power = 0.9821366
     alternative = two.sided

NOTE: n is number in *each* group
```

```
## endpoint 3
power.welch.t.test(delta = 1.0, sd1 = 1.5, sd2 = 1.75,
                   sig.level = 0.05/3, n = 134)
```

```
     Two-sample Welch t test power calculation

              n = 134
          delta = 1
            sd1 = 1.5
            sd2 = 1.75
      sig.level = 0.01666667
          power = 0.995346
    alternative = two.sided

NOTE: n is number in *each* group
```

In the case of endpoints 2 and 3, a sample size of 134 subjects per group leads leads to a power of 98.2% and 99.5%, respectively. This approach ignores a possible correlation between the variables. If also the correlations between the variables are known, a sample size calculation with multivariate normal distributions, for example by Monte Carlo simulations could be done.

Note:
There is the general problem, that all known approaches to control the FWER are quite conservative, as they involve a rather strong adjustment of the p values. Therefore, especially in situations with a lot of tests (> 10), these methods lead to many true differences remaining undetected.

7.3 False Discovery Rate (FDR)

The conservatism of the FWER or more precisely of the known FWER procedures has led to the search for alternatives for situations with many or very many simultaneous tests. We first consider the possible **testing decisions in multiple testing**.

	decision for		
i = 1,...,N	for H_{0i}	against H_{0i}	Sum
H_{0i} true	U	V	$N_0(= U + V)$
H_{0i} false	W	S	$N_1(= W + S)$
Sum	$N - R(= U + W)$	$R(= V + S)$	$N(= U + V + W + S)$

Table 7.1: Testing decisions in multiple testing.

With these terms we get FWER $= P(V \geq 1)$. A possible alternative to a strict control of the probability of false positive test results is to look for a control "in the mean" (i.e., in the expected value). This led to

the criterion of the expected proportion of false positive tests in all positive tests, which is referred to as the **false discovery rate (FDR)**

$$\mathrm{FDR} = \mathrm{E}\left(\frac{V}{R}\right) = \mathrm{E}\left(\frac{V}{V+S}\right) \tag{7.12}$$

In general, FDR $\leq$ FWER, which means that FDR is less conservative than FWER.

Probably the most commonly used procedure for calculating the FDR adjusted p values is the method of Benjamini and Hochberg (1995).

Algorithm 7.9. *Let $N \in \mathbb{N}$ **independent** or **positive dependent** statistical tests be given at significance level $\alpha \in (0,1)$ with null hypotheses $H_{0,1}, \ldots, H_{0,N}$ and alternatives $H_{1,1}, \ldots, H_{1,N}$, leading to the p values $p_1, \ldots, p_N$. Let $p_{(1)}, \ldots, p_{(N)}$ be the p values in ascending order and $H_{0,(j)}$ and $H_{1,(j)}$ $(j = 1, \ldots, N)$ be the corresponding hypotheses. The **Benjamini-Hochberg method** for $j = N, \ldots, 1$ consists of the following calculations*

Replace $p_{(j)}$ with

$$\tilde{p}_{(j)} = \min_{k=j,\ldots,N} \left\{ \min \left\{ \frac{N}{k} p_{(k)}, 1 \right\} \right\} \tag{7.13}$$

If $\tilde{p}_{(j)} \leq \alpha$, choose $H_{1,(j)}$, otherwise keep $H_{0,(j)}$.

Remark 7.10. ***(a)*** *Starting with the largest p value if one obtains an adjusted p value, which is smaller than the given significance level, one could also stop the procedure, since all following adjusted p values will in any case be at most even smaller. In practice, however, all p values are usually adjusted. Therefore, this optional early stop was not included in the above algorithm.*

(b) *The exact definition of the assumed positive dependence is described in Benjamini and Yekutieli (2001) and, as shown there, captures many practical situations.*

(c) *The Benjamini-Hochberg method is, as the method of Simes-Hochberg, a **step-up method**. We can compare these two methods directly, since they only differ in the correction factor. In the following we want to examine whether the Benjamini-Hochberg method indeed leads to smaller p values.*

$$\begin{aligned}
\frac{N}{k} p_{(k)} &\leq (N-k+1) p_{(k)} \qquad \text{for } k = 1, \ldots, N \\
\frac{N}{k} &\leq (N-k+1) \qquad (p_{(k)} > 0) \\
N &\leq Nk - k^2 + k \\
0 &\leq N(k-1) - k^2 + k \\
0 &\leq N(k-1) - k(k-1) \\
0 &\leq (N-k)(k-1)
\end{aligned}$$

In the last equation, with the exception of the cases $k = 1$ and $k = N$, the right hand side is always greater than 0; that means the adjusted p values in case of Benjamini-Hochberg are smaller in all steps except the first and the last (where they are equal) than in case of Simes-Hochberg.

We continue example 7.4.

Example 7.11. We additionally assume that the tests are independent or at least only positively dependent. Remember that

$$p_1 = 0.004,\ p_2 = 0.011,\ p_3 = 0.039,\ p_4 = 0.012,\ p_5 = 0.001,\ p_6 = 0.480$$

and

$$p_5 \leq p_1 \leq p_2 \leq p_4 \leq p_3 \leq p_6$$

With the Benjamini-Hochberg method we obtain

$$\begin{aligned}
\tilde{p}_6 &= 1p_6 = 0.480 \\
\tilde{p}_3 &= \frac{6}{5}p_3 = \mathbf{0.0468} \\
\tilde{p}_4 &= \frac{6}{4}p_4 = \mathbf{0.018} \\
\tilde{p}_2 &= \frac{6}{3}p_2 = 0.022 \Rightarrow \mathbf{0.018} \\
\tilde{p}_1 &= \frac{6}{2}p_1 = \mathbf{0.012} \\
\tilde{p}_5 &= 6p_5 = \mathbf{0.006}
\end{aligned}$$

In the case of $\tilde{p}_2$ the calculated value of 0.022 would be above the previous value of $\tilde{p}_4$. This is not allowed and is compensated for by the outer minimum in equation (7.13). Again, we can verify these calculations using function `p.adjust`.

```
## Sorting not necessary
p.adjust(pval, method = "fdr")
```

```
[1] 0.0120 0.0180 0.0468 0.0180 0.0060 0.4800
```

Thus, we can reject the null hypotheses $H_{0,5}$, $H_{0,1}$, $H_{0,2}$, $H_{0,4}$ and also $H_{0,3}$ at the 5% level. The adjusted p values are strikingly smaller than in the case of the FWER methods.

> Note:
> In practice, the method of Benjamini and Hochberg (1995) is used to control the FDR in most of the cases. If it is unclear, whether the assumption of positive-dependent tests is fulfilled, one can switch to the more conservative **Benjamini-Yekutieli method**, which is described in Benjamini and Yekutieli (2001) as a modification of the Benjamini-Hochberg method. It controls the FDR in any case.

In the following example, we will apply the presented methods to a real and very well-known gene expression dataset.

Example 7.12. We use the gene expression data from Golub et al. (1999), which is available in the Bioconductor package `"multtest"` (Pollard et al. (2012)). This is one of the first publications in which gene expression was used for the prediction (diagnosis) of a disease. Namely, samples were taken from patients with acute lymphoblastic leukemia (ALL) and patients with acute myeloid leukemia (AML) and patients were classified by their molecular gene signature. The package `"multtest"` contains several methods that can be used to adjust p values. We first load the dataset `golub`.

```
data(golub)
str(golub)
```

```
num [1:3051, 1:38] -1.458 -0.752 0.457 3.135 2.766 ...
- attr(*, "dimnames")=List of 2
..$ : NULL
..$ : NULL
```

For clarity, we convert the existing variable `golub.cl` into a factor.

```
golub.cl ← factor(golub.cl, labels = c("ALL", "AML"))
table(golub.cl)
```

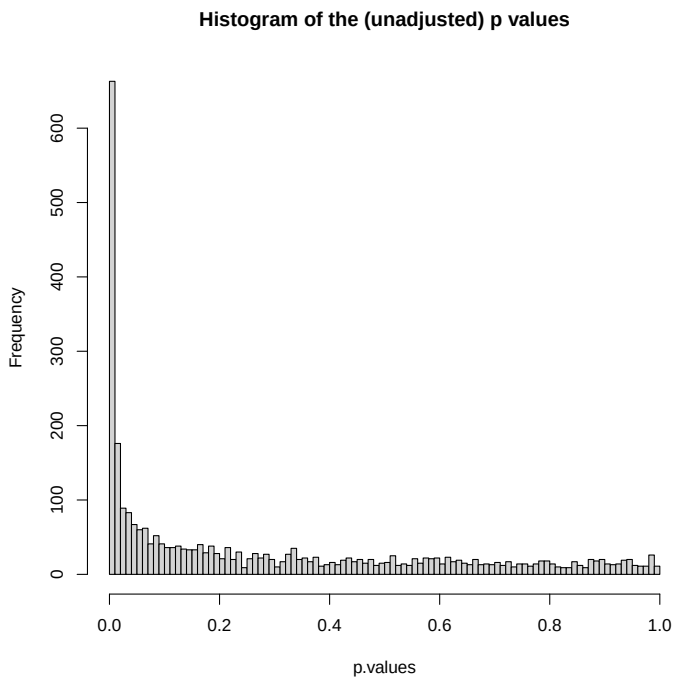
```
golub.cl
ALL AML
 27  11
```

We apply the Welch t-test to compare the two types of leukemia. To do this, we first implement a simple function `ttest` that returns only the p value of the test. With the help of the function `apply`, we then apply this function to the rows (`MARGIN = 1`) of the dataset.

```
ttest ← function(x, g) t.test(x~g)[["p.value"]]
p.values ← apply(X = golub, MARGIN = 1, FUN = ttest, g = golub.cl)
```

We plot the (unadjusted) p values in the form of a histogram plot.

```
hist(p.values, nclass = 101,
     main = "Histogram of the (unadjusted) p values")
```

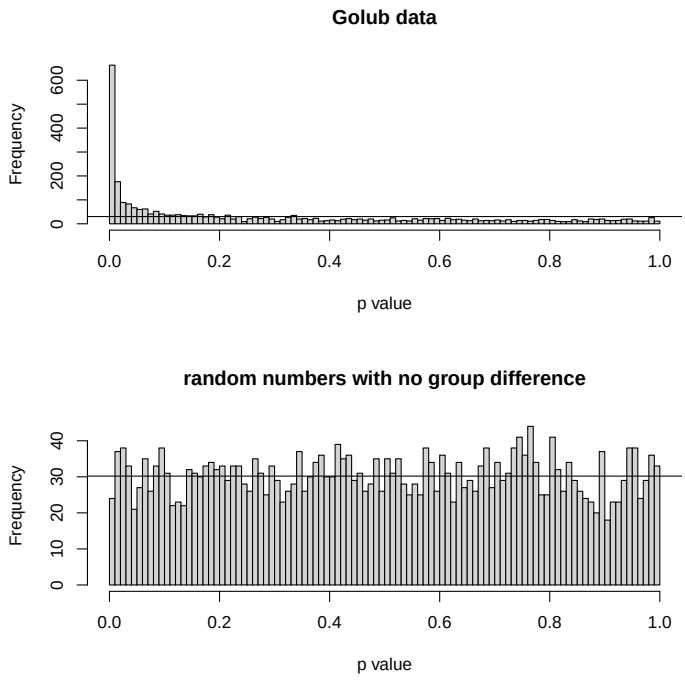


With a clear peak at small p values the histogram strongly indicates that there are many significant differences between the two groups. To illustrate this, we generate (pseudo-)random numbers of two groups without differences in the next step and calculate the corresponding p values. We use a random data set, which has the same dimension as the gene expression data set.

```
M ← matrix(rnorm(nrow(golub)*ncol(golub)), nrow = nrow(golub))
p.values.cf ← apply(M, 1, ttest, g = golub.cl)
```

We compare the histograms of the p values of the two analyses.

```
par(mfrow = c(2, 1))
hist(p.values, nclass = 101, xlab = "p value", main = "Golub data")
abline(h = 30.2)
hist(p.values.cf, nclass = 101, xlab = "p value",
     main = "random numbers with no group difference")
abline(h = 30.2)
```



In case of no significant differences, the p values of the Welch t-test follow a uniform distribution. In particular, it is expected by chance that in about 5% of the cases we will find a p value less than 5% (i.e. false positive test results).

```
## expected number
0.05*nrow(M)
```

```
[1] 152.55
```

```
## actual number
sum(p.values.cf < 0.05)
```

```
[1] 153
```

We see that the expected number of false positive tests in the case of the random number data set agrees very well with the actual number. Also in the case of the real data set, which has the same dimension, one has to expect a corresponding number of false positive test results. In total one obtains the following number of positive tests.

```
1 sum(p.values < 0.05)
```

```
[1] 1078
```

How many of these positive test results are true positives and how many false positives?
The exact answer to this question is not known in practice. We calculate the adjusted p values to approximate the answer to this question, where we use the various methods implemented in function `mt.rawp2adjp` of package `"multtest"` (Pollard et al. (2012)).

```
1 p.values.adj ← mt.rawp2adjp(p.values)
```

For each of the methods, we calculate the number of p values that are less than 5%.

```
1 colSums(p.values.adj[["adjp"]] < 0.05)
```

```
      rawp Bonferroni       Holm   Hochberg    SidakSS    SidakSD
      1078        103        103        103        103        104
        BH         BY        ABH  TSBH_0.05
       695        293        824        807
```

The first result comes from the unadjusted p values. The procedures `Bonferroni` to `SidakSD` are procedures that control the FWER. We see that we would be able to identified only slightly more than 100 of the more than 1000 significant genes as true positives, if we want to control the FEWR. In this group of about 100 genes, however, the probability of one or more false positive tests is at most 5%. The remaining four results are from methods that control the FDR. In the case of the Benjamini-Hochberg method (`BH`) we obtain nearly 700 genes that we can still consider as significant after adjustment. The expected proportion of false positives among these nearly 700 genes is therefore at most 5%, which corresponds to about $0.05 \times 695 \approx 35$ genes. Within the original 1078 genes, we have to assume that this fraction is about $\frac{152.55}{1078} \approx 14\%$. Hence, instead of about one in seven positive results in the unadjusted list of significant genes only about one in 20 is false positive in the `BH` list.
The results of the Benjamini-Yekutieli method (`BY`) is also interesting, because it does not require any additional assumption about the form of the dependence. We see that this method gives a result that is between the FWER methods and the method of Benjamini-Hochberg. For more details on the methods, we refer to the help page of the of function `mt.rawp2adjp` and the literature given there.

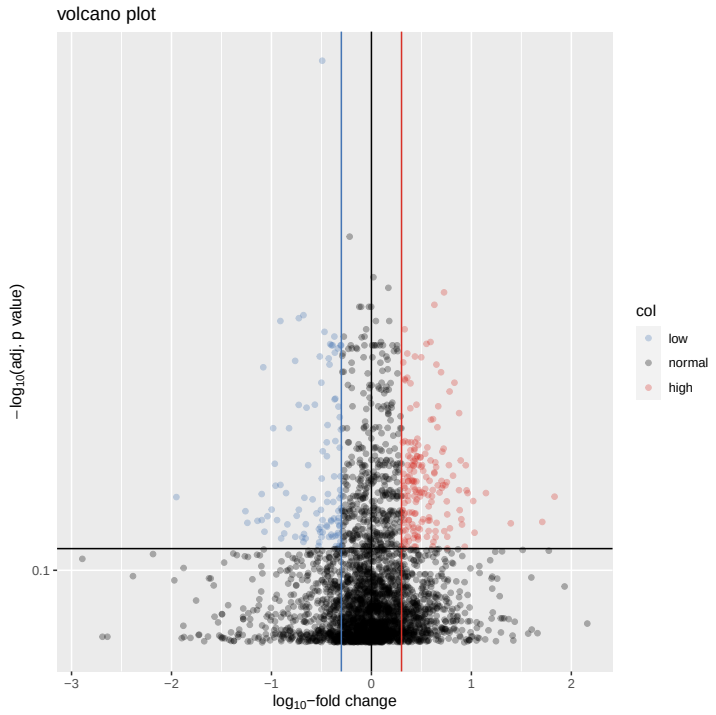
Finally, we would like to point out the main result of phase I of the Microarray Quality Control (MAQC-I) project, which was initiated by the FDA (U.S. Food and Drug Administration). In this project, which was about the reproducibility of the results of gene expression analyses with the aid of so-called microarrays, it was shown that the reproducibility of the results can be increased by combining adjusted p values with the log-fold change; see MAQC-Consortium (2006). We continue with the above example and take a look at this combination with the help of a so-called **volcano plot**.

Example 7.13. To generate the volcano plot, we first need to calculate the log-fold changes. We use the function `pairwise.logfc` from the package `"MKomics"` (Kohl (2020b)).

```
logFC ← apply(golub, 1, pairwise.logfc, g = golub.cl)
```

We can create the volcano plot with the function `volcano` from the package `"MKinfer"` (Kohl (2022b)). In addition to the log-fold change, we will plot the p values adjusted by the Benjamini-Hochberg method. It is common in gene expression analyses to consider an adjusted p value of less than 0.05 as significant and a change of 2-fold or more as relevant. The values of the data set are $\log_{10}$-transformed. Consequently, the limit for the log-fold change is $\pm\log_{10}(2) \approx \pm 0.3$.

```
volcano(x = logFC, pval = p.values.adj[["adjp"]][, "BH"], effect.low = -log10(2),
        effect.high = log10(2), alpha = 0.3,
        xlab = expression(paste(log[10], "-fold change")),
        ylab = expression(paste(-log[10], "(adj. p value)")),
        title = "volcano plot")
```



As can be seen in the plot, the p values are first $\log_{10}$-transformed for the plot. By this transformation very small p values become large negative numbers. Due to the additional change of the sign, these large negative numbers then become large positive numbers; i.e., the smallest p values are at the top of the

figure. The limit for the transformed p values is thus at $-\log_{10}(0.05) \approx 1.3$. The most important genes are therefore located at the top left (blue) and top right (red) in the plot.
With the help of the functions `expression` and `paste` it is also possible to use simple mathematical expressions in the labeling of plots. A more detailed description can be found on the help page `plotmath`.

Note:
The result of the MAQC-I project is not very surprising from a statistical point of view, since the log-fold change in this case is a measure of the effect and accordingly also represents the relevance of the results. Independently of gene expression data, it can be assumed that in case of single as well as multiple tests, a combination of significance and relevance will increase the probability of selecting true positive test results and thus will increase their reproducibility.

7.4 Exercises

Explain the steps of your analysis and interpret the results.

1. Adjust the following p values manually.

   ```
   0.001, 0.760, 0.550, 0.001, 0.002, 0.271, 0.005, 0.007, 0.210, 0.008
   ```

   Use the methods of

   a) Bonferroni

   b) Bonferroni-Holm

   c) Simes-Hochberg

   d) Benjamini-Hochberg

   Compare your results with the results you obtain using function `p.adjust`. Interpret the results of the different methods and compare them with each other. Does it confirm that the FWER methods are more conservative than the FDR method?

2. Adjust the following p values manually.

   ```
   0.001, 0.820, 0.950, 0.001, 0.001, 0.011, 0.012, 0.001, 0.009, 0.001
   ```

   Use the methods of

   a) Bonferroni

   b) Bonferroni-Holm

   c) Simes-Hochberg

   d) Benjamini-Hochberg

   Compare your results with the results you obtain using function `p.adjust`. How do the different adjustment methods behave when several p-values are identical? Compare and interpret the results of the different methods.

3. Perform a sample size calculation for the study of Dodick et al. (2019). Two primary endpoints were considered. First, the proportion of patients free of pain after two hours and second, the proportion of patients free of pain and rid of the most other symptoms (noise sensitivity, odor sensitivity, nausea) after two hours. According to the study protocol, which can be found at https://clinicaltrials.gov/ct2/show/NCT02828020, the assumptions for the sample size case calculation were

   endpoint 1: $p_1 = 0.1, p_2 = 0.24$

   endpoint 2: $p_1 = 0.267, p_2 = 0.377$

   Apply function `power.prop.test` of package `"stats"` (R Core Team (2022a)) for determining the required sample size. The FWER should be at most 5% and the power should be at least 90%. Based on your results, do you consider a sample size of 450 cases per group, as used by Dodick et al. (2019), as sufficient for the study?

4. Use the ICU dataset and compare patients with and without liver failure with respect to sex, age, maximum heart rate, maximum body temperature, SAPS-II score, length of stay, and outcome. Use appropriate tests for each. Store the p values and adjust them using Holm's method. Use the function `p.adjust`. For which variables do you obtain a significant result even after adjusting the p-values?

5. Use the ICU dataset and compare female and male patients with respect to age, maximum heart rate, maximum body temperature, SAPS-II score, liver failure, length of stay, and outcome. Use appropriate tests for each. Store the p values and adjust them using Holm's method. Use the function `p.adjust`. For which variables do you obtain a significant result even after adjusting the p-values?

6. Use the ICU dataset and compare patients who died with patients who didn't die with respect to sex, age, maximum heart rate, maximum body temperature, SAPS-II score, liver failure, and length of stay.

   ```
   ICUData$died ← as.integer(ICUData$outcome == "died")
   ```

   Use appropriate tests for each. Store the p values and adjust them using Holm's method. Use the function `p.adjust`. For which variables do you obtain a significant result even after adjusting the p-values?

7. Download the file `normData.RData` from my GitHub account (link: https://github.com/stamats/ISDR/blob/main/normData.RData).

   a) Apply the Welch t-test to the rows of the dataset `normData`, where the two groups are given by the factor `group` also included in the `RData`-file. Draw a histogram of the (unadjusted) p values. Do you obtain more significant results than would be expected by chance?

   b) Calculate the adjusted p values using function `mt.rawp2adjp` of package `"multtest"` (Pollard et al. (2012)). Compare the results of the different adjustment methods by calculating, how many of the adjusted p values are smaller than 0.05 in each case.

c) The data in `normData` are $\log_2$-transformed. Calculate the $\log_2$-fold changes for the rows and display them together with the Benjamini-Hochberg adjusted p values in a volcano plot.

Software versions

For generating this book the following software versions have been used:

- R version 4.2.1 Patched (2022-09-25 r82918), `x86_64-pc-linux-gnu`
- Locale: `LC_CTYPE=de_DE.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=de_DE.UTF-8`, `LC_COLLATE=de_DE.UTF-8`, `LC_MONETARY=de_DE.UTF-8`, `LC_MESSAGES=de_DE.UTF-8`, `LC_PAPER=de_DE.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=de_DE.UTF-8`, `LC_IDENTIFICATION=C`
- Running under: `Linux Mint 20.3`
- Matrix products: default
- BLAS: `/usr/lib/x86_64-linux-gnu/libf77blas.so.3.10.3`
- LAPACK: `/home/kohlm/RTOP/Rbranch/lib/libRlapack.so`
- Base packages: base, datasets, graphics, grDevices, methods, parallel, stats, stats4, utils
- Other packages: Biobase 2.56.0, BiocGenerics 0.42.0, boot 1.3-28, coin 1.4-2, datarium 0.1.0, DescTools 0.99.46, distr 2.8.0, distr6 1.6.11, distrEx 2.8.0, distrMod 2.8.5, evd 2.3-6.1, exactRankTests 0.8-35, ggplot2 3.3.6, ggpubr 0.4.0, ggsci 2.9, gridExtra 2.3, knitr 1.40, MASS 7.3-58.1, MKclass 0.3, MKdescr 0.8, MKinfer 0.8, MKomics 0.8, MKpower 0.5, multtest 2.52.0, qqplotr 0.0.5, RandVar 1.2.1, RColorBrewer 1.1-3, rmx 0.8, RobAStBase 1.2.2, RobExtremes 1.2.0, RobLox 1.2.0, robustbase 0.95-0, ROptEst 1.2.1, rrcov 1.7-1, scales 1.2.1, sfsmisc 1.1-13, startupmsg 0.9.6, survival 3.4-0
- Loaded via a namespace (and not attached): abind 1.4-5, actuar 3.3-0, arrangements 1.1.9, assertthat 0.2.1, backports 1.4.1, broom 1.0.1, car 3.1-0, carData 3.0-5, cellranger 1.1.0, checkmate 2.1.0, circlize 0.4.15, class 7.3-20, cli 3.4.1, clue 0.3-61, cluster 2.1.4, codetools 0.2-18, colorspace 2.0-3, compiler 4.2.1, ComplexHeatmap 2.12.1, crayon 1.5.2, data.table 1.14.2, DBI 1.1.3, DEoptimR 1.0-11, dictionar6 0.1.3, digest 0.6.29, doParallel 1.0.17, dplyr 1.0.10, e1071 1.7-11, ellipsis 0.3.2, evaluate 0.17, Exact 3.2, expint 0.1-7, expm 0.999-6, fansi 1.0.3, farver 2.1.1, foreach 1.5.2, generics 0.1.3, GetoptLong 1.0.5, ggsignif 0.6.4, gld 2.6.5, GlobalOptions 0.1.2, glue 1.6.2, gmp 0.6-6, grid 4.2.1, gtable 0.3.1, httr 1.4.4, IRanges 2.30.1, iterators 1.0.14, labeling 0.4.2, lattice 0.20-45, libcoin 1.0-9, lifecycle 1.0.3, limma 3.52.4, lmom 2.9, magrittr 2.0.3, Matrix 1.5-1, matrixStats 0.62.0, matrixTests 0.1.9.1, mgcv 1.8-40, modeltools 0.2-23, multcomp 1.4-20, munsell 0.5.0, mvtnorm 1.1-3, nlme 3.1-160, ooplah 0.2.0, param6 0.2.4, pcaPP 2.0-2, pillar 1.8.1, pkgconfig 2.0.3, png 0.1-7, proxy 0.4-27, purrr 0.3.5,

R6 2.5.1, Rcpp 1.0.9, readxl 1.4.1, rjson 0.2.21, rlang 1.0.6, RobAStRDA 1.2.0, rootSolve 1.8.2.3, rstatix 0.7.0, rstudioapi 0.14, S4Vectors 0.34.0, sandwich 3.0-2, set6 0.2.5, shape 1.4.6, splines 4.2.1, stringi 1.7.8, stringr 1.4.1, TH.data 1.1-1, tibble 3.1.8, tidyr 1.2.1, tidyselect 1.2.0, tools 4.2.1, utf8 1.2.2, vctrs 0.4.2, withr 2.5.0, xfun 0.33, zoo 1.8-11

Bibliography

Almeida, A., Loy, A., and Hofmann, H. (2018). *ggplot2 Compatible Quantile-Quantile Plots in R*. (Cited on pages 131, 133 and 136)

Amrhein, V., Korner-Nievergelt, F., and Roth, T. (2017). The earth is flat (p > 0.05): significance thresholds and the crisis of unreplicable research. *PeerJ*, 5:e3544. (Cited on page 200)

Andri et mult. al., S. (2022). *DescTools: Tools for Descriptive Statistics*. R package version 0.99.46. (Cited on pages 18, 19, 20, 35, 36, 43, 46, 47, 49, 207, 240, 242 and 246)

Armbruster, D. A. and Pry, T. (2008). Limit of blank, limit of detection and limit of quantitation. *Clin Biochem Rev*, 29 Suppl 1:49–52. (Cited on page 146)

Auguie, B. (2017). *gridExtra: Miscellaneous Functions for "Grid" Graphics*. R package version 2.3. (Cited on pages 144 and 236)

Benjamini, Y. and Hochberg, Y. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, 57(1):289–300. (Cited on pages 257 and 258)

Benjamini, Y. and Yekutieli, D. (2001). The control of the false discovery rate in multiple testing under dependency. *Ann. Stat.*, 29. (Cited on pages 257 and 258)

Bernal, E. (2014). Limit of detection and limit of quantification determination in gas chromatography. In Guo, X., editor, *Advances in Gas Chromatography*, chapter 3. IntechOpen, Rijeka. (Cited on page 147)

Box, G. and Draper, N. (1987). *Empirical model-building and response surfaces*. Wiley series in probability and mathematical statistics: Applied probability and statistics. Wiley. (Cited on page 9)

Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. *Mon. Wea. Rev.,*, 78:1–3. (Cited on page 152)

Brodersen, K. H., Ong, C. S., Stephan, K. E., and Buhmann, J. M. (2010). The balanced accuracy and its posterior distribution. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 3121–3124. IEEE. (Cited on page 147)

Broman, K. W. and Woo, K. H. (2018). Data organization in spreadsheets. *The American Statistician*, 72(1):2–10. (Cited on page 11)

Campbell, I. (2007). Chi-squared and Fisher-Irwin tests of two-by-two tables with small sample recommendations. *Stat Med*, 26(19):3661–3675. (Cited on page 209)

Canty, A. and Ripley, B. D. (2021). *boot: Bootstrap R (S-Plus) Functions*. R package version 1.3-28. (Cited on pages 122, 158, 159, 167 and 171)

Chambers, J. (2000). Stages in the Evolution of S. http://ect.bell-labs.com/sl/S/history.html [Last access 29.08.2015]. (Cited on pages 1 and 2)

Chambers, J. (2008). *Software for Data Analysis: Programming with R*. Springer. (Cited on pages 1 and 2)

Chernick, M. R. and LaBudde, R. A. (2011). *An Introduction to Bootstrap Methods with Applications to R*. Wiley Publishing, 1st edition. (Cited on pages 158, 159 and 160)

Cohen, J. (1994). The earth is round ($p < .05$). 49. (Cited on page 200)

Colquhoun, D. (2014). An investigation of the false discovery rate and the misinterpretation of p-values. *R Soc Open Sci*, 1(3):140216. (Cited on page 200)

Csárdi, G., Hester, J., Wickham, H., Chang, W., Morgan, M., and Tenenbaum, D. (2021). *remotes: R Package Installation from Remote Repositories, Including 'GitHub'*. R package version 2.4.2. (Cited on page 86)

Dalgaard, P. (2010). [R] R 2.11.0 is released. https://stat.ethz.ch/pipermail/r-help/2010-April/236141.html [Last access 2015 Aug 29]. (Cited on page 2)

Dalgaard, P. (2013). [R] R 3.0.0 is released. https://stat.ethz.ch/pipermail/r-help/2013-April/350751.html [Last access 2015 Aug 29]. (Cited on page 2)

Dalgaard, P. (2015). R 3.2.1 liftoff. https://stat.ethz.ch/pipermail/r-announce/2015/000586.html [Last access 2015 Aug 29]. (Cited on page 2)

Dalgaard, P. (2020). R 4.0.0 is released. https://stat.ethz.ch/pipermail/r-announce/2020/000653.html [Last access 2020 Apr 24]. (Cited on page 2)

DataCamp Inc. (2022). Search all 25.258 R packages on CRAN and Bioconductor. https://www.rdocumentation.org/ [Last access 2022 Oct 08]. (Cited on page 3)

Davison, A. C. and Hinkley, D. V. (1997). *Bootstrap Methods and Their Applications*. Cambridge University Press, Cambridge. ISBN 0-521-57391-2. (Cited on page 159)

Delignette-Muller, M. L. and Dutang, C. (2015). fitdistrplus: An R package for fitting distributions. *Journal of Statistical Software*, 64(4):1–34. (Cited on page 127)

Dodick, D. W., Lipton, R. B., Ailani, J., Lu, K., Finnegan, M., Trugman, J. M., and Szegedi, A. (2019). Ubrogepant for the treatment of migraine. *New England Journal of Medicine*, 381(23):2230–2241. (Cited on page 264)

du Prel, J. B., Hommel, G., Röhrig, B., and Blettner, M. (2009). Confidence interval or p-value?: part 4 of a series on evaluation of scientific publications. *Dtsch Arztebl Int*, 106(19):335–339. (Cited on page 200)

Fisher, R. A. (1936). Has mendel's work been rediscovered? *Annals of science*, 1(2):115–137. (Cited on page 131)

Gentleman, R. and Temple Lang, D. (2007). Statistical analyses and reproducible research. *Journal of Computational and Graphical Statistics*, 16(1):1–23. (Cited on page 124)

Gentleman, R. C., Carey, V. J., Bates, D. M., Bolstad, B., Dettling, M., Dudoit, S., Ellis, B., Gautier, L., Ge, Y., Gentry, J., Hornik, K., Hothorn, T., Huber, W., Iacus, S., Irizarry, R., Leisch, F., Li, C., Maechler, M., Rossini, A. J., Sawitzki, G., Smith, C., Smyth, G., Tierney, L., Yang, J. Y., and Zhang, J. (2004). Bioconductor: open software development for computational biology and bioinformatics. *GenomeBiology*, 5:R80. http://www.bioconductor.org. (Cited on page 3)

Ghasemi, A. and Zahediasl, S. (2012). Normality tests for statistical analysis: a guide for non-statisticians. *Int J Endocrinol Metab*, 10(2):486–489. (Cited on page 243)

Globus, A. (1994). Principles of information display for visualization practitioners. (Cited on page 77)

Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., Loh, M. L., Downing, J. R., Caligiuri, M. A., Bloomfield, C. D., and Lander, E. S. (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537. (Cited on page 258)

Hagemann, O. (2014). TSH-basal. http://www.laborlexikon.de/Lexikon/Infoframe/t/TSH-basal.htm [Last access 2015 Aug 29]. (Cited on page 107)

Hamilton, T. E., Davis, S., Onstad, L., and Kopecky, K. J. (2008). Thyrotropin levels in a population with no clinical, autoantibody, or ultrasonographic evidence of thyroid disease: implications for the diagnosis of subclinical hypothyroidism. *J. Clin. Endocrinol. Metab.*, 93(4):1224–1230. (Cited on page 107)

Hanley, J. A. and McNeil, B. J. (1982). The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143(1):29–36. (Cited on page 152)

Harjutsalo, V., Sund, R., Knip, M., and Groop, P. (2013). Incidence of type 1 diabetes in Finland. *JAMA*, 310(4):427–428. (Cited on page 99)

Harrell, F. (2014). *Regression Modeling Strategies*. Springer Series in Statistics, 2. edition. (Cited on page 152)

Harrower, M. and Brewer, C. A. (2003). Colorbrewer.org: An online tool for selecting color schemes for maps. *The Cartographic Journal*, 40(1):27–37. (Cited on pages 67 and 79)

Head, M. L., Holman, L., Lanfear, R., Kahn, A. T., and Jennions, M. D. (2015). The extent and consequences of p-hacking in science. *PLOS Biology*, 13(3):1–15. (Cited on page 200)

Hedderich, J. and Sachs, L. (2018). *Angewandte Statistik*. Springer Berlin Heidelberg. (Cited on page 39)

Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70. (Cited on pages 228 and 251)

Hornik, K. (2008). *The Past, Present, and Future of the R Project*. http://www.statistik.uni-dortmund.de/useR-2008/slides/Hornik.pdf [Last access 2015 Aug 29]. (Cited on pages 1 and 2)

Hornik, K. (2022). R FAQ. http://cran.r-project.org/doc/manuals/R-FAQ.html [Last access 2022 Oct 08]. (Cited on page 3)

Hothorn, T. and Hornik, K. (2022). *exactRankTests: Exact Distributions for Rank and Permutation Tests*. R package version 0.8-35. (Cited on pages 219 and 224)

Hothorn, T., Hornik, K., van de Wiel, M. A., and Zeileis, A. (2006). A Lego system for conditional inference. *The American Statistician*, 60(3):257–263. (Cited on pages 207, 209, 213, 219, 224 and 228)

Howson Ian (2022). R Package Documentation. https://rdrr.io/ [Last access 2022 Oct 08]. (Cited on page 3)

Huber, W., Carey, V. J., Gentleman, R., Anders, S., Carlson, M., Carvalho, B. S., Bravo, H. C., Davis, S., Gatto, L., Girke, T., Gottardo, R., Hahne, F., Hansen, K. D., Irizarry, R. A., Lawrence, M., Love, M. I., MacDonald, J., Obenchain, V., Ole's, A. K., Pag'es, H., Reyes, A., Shannon, P., Smyth, G. K., Tenenbaum, D., Waldron, L., and Morgan, M. (2015). Orchestrating high-throughput genomic analysis with Bioconductor. *Nature Methods*, 12(2):115–121. (Cited on page 122)

Iacus, S. M., Urbanek, S., Goedman, R. J., and Ripley, B. (2022). R for Mac OS X FAQ. https://cran.r-project.org/bin/macosx/RMacOSX-FAQ.html [Last access 2022 Oct 08]. (Cited on page 3)

Iba, W. and Langley, P. (1992). Induction of one-level decision trees. In Sleeman, D. and Edwards, P., editors, *Machine Learning Proceedings 1992*, pages 233 – 240. Morgan Kaufmann, San Francisco (CA). (Cited on page 151)

Ihaka, R. (1997). R-beta: New R Version for Unix. https://stat.ethz.ch/pipermail/r-help/1997-December/001929.html [Last access 2020 Sep 06]. (Cited on page 2)

Ihaka, R. (1998). R : Past and Future History. Technical report, Statistics Department, The University of Auckland. https://www.stat.auckland.ac.nz/~ihaka/downloads/Interface98.pdf [Last access 2020 Sep 06]. (Cited on page 2)

Ihaka, R. (2003). Colour for Presentation Graphics. http://www.stat.auckland.ac.nz/~ihaka/downloads/DSC-Color-Slides.pdf [Last access 2015 Aug 29]. (Cited on page 65)

Ioannidis, J. P. A. (2005). Why most published research findings are false. *PLOS Medicine*, 2(8). (Cited on pages 124 and 200)

Kassambara, A. (2019). *datarium: Data Bank for Statistical Analysis and Visualization*. R package version 0.1.0. (Cited on page 233)

Kassambara, A. (2020). *ggpubr: 'ggplot2' Based Publication Ready Plots*. R package version 0.3.0. (Cited on pages 225 and 236)

Kemmler, W., Weissenfels, A., Teschler, M., Willert, S., Bebenek, M., Shojaa, M., Kohl, M., Freiberger, E., Sieber, C., and von Stengel, S. (2017). Whole-body electromyostimulation and protein supplementation favorably affect sarcopenic obesity in community-dwelling older men at risk: the randomized controlled franso study. 12.2017:1503 – 1513. (Cited on page 244)

Kohl, M. (2005). *Numerical contributions to the asymptotic theory of robustness*. Dissertation, Universität Bayreuth, Bayreuth. (Cited on page 142)

Kohl, M. (2019). *RobLox: Optimally robust influence curves and estimators for location and scale*. R package version 1.2.0. (Cited on pages 122, 142, 145, 169, 185 and 186)

Kohl, M. (2020a). *MKclass: Statistical Classification*. R package version 0.3. (Cited on pages 148, 149, 150, 151 and 152)

Kohl, M. (2020b). *MKomics: Functions for Omics Data Analysis*. R package version 0.5. (Cited on page 262)

Kohl, M. (2020c). *MKpower: Power Analysis and Sample Size Calculation*. R package version 0.4. (Cited on pages 182, 189 and 244)

Kohl, M. (2022a). *MKdescr: Descriptive Statistics*. R package version 0.8. (Cited on pages 18, 25, 27, 28, 45 and 59)

Kohl, M. (2022b). *MKinfer: Inferential Statistics*. R package version 0.8. (Cited on pages 155, 156, 159, 161, 164, 165, 202, 205, 210, 220, 229, 234, 235 and 262)

Kohl, M. (2022c). *rmx: Radius-Minimax Estimators*. R package version 0.8. (Cited on pages 122, 142, 143, 145 and 169)

Kohl, M. and Ruckdeschel, P. (2010). R package distrMod: S4 classes and methods for probability models. *Journal of Statistical Software*, 35(10):1–27. (Cited on pages 127, 128, 129, 140, 156, 163 and 167)

Kohl, M. and Ruckdeschel, P. (2019). *ROptEst: Optimally robust estimation*. R package version 1.2.1. (Cited on pages 145, 172, 185, 186, 187 and 188)

Kohl, M., Ruckdeschel, P., and Rieder, H. (2010). Infinitesimally robust estimation in general smoothly parametrized models. *Statistical Methods & Applications*, 19(3):333–354. (Cited on page 142)

Limpert, E. and Stahel, W. A. (2011). Problems with using the normal distribution–and ways to improve quality and efficiency of data analysis. *PLoS ONE*, 6(7):e21403. (Cited on page 109)

Limpert, E., Stahel, W. A., and Abbt, M. (2001). Log-normal distributions across the sciences: Keys and clues. *BioScience*, 51:341–352. (Cited on page 109)

MAQC-Consortium (2006). The MicroArray Quality Control (MAQC) project shows inter- and intraplatform reproducibility of gene expression measurements. *Nature Biotechnology*, 24:1151–1161. (Cited on page 262)

Mendel, G. (1866). Versuche über plflanzenhybriden. verhandlungen des naturforschenden vereines in brünn, bd. iv für das jahr 1865. *Abhandlungen*, pages 3–47. (Cited on page 131)

Menzel, U. (2021). *EMT: Exact Multinomial Test: Goodness-of-Fit Test for Discrete Multivariate Data*. R package version 1.2. (Cited on page 204)

Morgan, M. (2019). *BiocManager: Access the Bioconductor Project Package Repository*. R package version 1.30.10. (Cited on page 122)

Muenchen, R. A. (2022). The Popularity of Data Science Software. http://r4stats.com/articles/popularity/ [Last access 2022 Oct 08]. (Cited on page 2)

Neuwirth, E. (2014). *RColorBrewer: ColorBrewer palettes*. R package version 1.1-2. (Cited on pages 67, 69, 72 and 83)

Novitski, C. E. (2004). Revision of fisher's analysis of mendel's garden pea experiments. *Genetics*, 166(3):1139–1140. (Cited on page 131)

Offen, W., Chuang-Stein, C., Dmitrienko, A., Littman, G., Maca, J., Meyerson, L., Muirhead, R., Stryszak, P., Baddy, A., Chen, K., Copley-Merriman, K., Dere, W., Givens, S., Hall, D., Henry, D., Jackson, J. D., Krishen, A., Liu, T., Ryder, S., Sankoh, A. J., Wang, J., and Yeh, C.-H. (2007). Multiple co-primary endpoints: Medical and statistical solutions: A report from the multiple endpoints expert team of the pharmaceutical research and manufacturers of america. *Drug Information Journal*, 41(1):31–46. (Cited on page 248)

Pedersen, T. L. and Robinson, D. (2022). *gganimate: A Grammar of Animated Graphics*. R package version 1.0.8. (Cited on page 76)

Pollard, K. S., Gilbert, H. N., Ge, Y., Taylor, S., and Dudoit, S. (2012). *multtest: Resampling-based multiple hypothesis testing*. R package version 2.12.0. (Cited on pages 258, 261 and 264)

Powers, D. M. (2011). Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness and Correlation. *Journal of Machine Learning Technologies*, 1(2):37–63. (Cited on page 147)

R Consortium (2022). R Consortium. https://www.r-consortium.org/ [Last access 2022 Oct 08]. (Cited on page 2)

R Core Team (2022a). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. (Cited on pages vi, 1, 74, 122, 127, 178, 200, 217, 229, 242 and 264)

R Core Team (2022b). *R Data Import/Export*. http://cran.r-project.org/doc/manuals/r-release/R-data.pdf. (Cited on page 12)

R Core Team (2022c). R Developer Page. https://developer.r-project.org/ [Last access 2022 Oct 08]. (Cited on page 2)

R Core Team (2022d). *R Installation and Administration*. R Foundation for Statistical Computing, Vienna, Austria. http://cran.r-project.org/doc/manuals/R-admin.pdf. (Cited on page 4)

Ranke, J. (2022). Index of /bin/linux/debian. https://cran.r-project.org/bin/linux/debian/ [Last access 2022 Oct 08]. (Cited on page 3)

Ranstam, J. (2012). Why the p-value culture is bad and confidence intervals a better alternative. *Osteoarthritis and Cartilage*, 20(8):805–808. (Cited on page 200)

Rasch, D., Kubinger, K. D., and Moder, K. (2011). The two-sample t test: pre-testing its assumptions does not pay off. *Stat. Papers*, 52:219–231. (Cited on pages 200, 223 and 243)

Rieder, H. (1994). *Robust asymptotic statistics*. Springer Series in Statistics. Springer. (Cited on page 142)

Rigby, A. S. (1999). Getting past the statistical referee: moving away from P-values and towards interval estimation. *Health Education Research*, 14(6):713–715. (Cited on page 200)

Ripley, B. D. and Murdoch, D. J. (2022). R for Windows FAQ. http://cran.r-project.org/bin/windows/base/rw-FAQ.html [Last access 11.10.2022]. (Cited on page 3)

Ruckdeschel, P., Kohl, M., and Horbenko, N. (2019). *RobExtremes: Optimally robust estimation for extreme value distributions*. Contributions by S. Desmettre, G. Kroisandt, E. Massini, D. Pupashenko and B. Spangl; R package version 1.2.0. (Cited on pages 145, 187 and 188)

Ruckdeschel, P., Kohl, M., Stabla, T., and Camphausen, F. (2006). S4 Classes for Distributions. *R News*, 6(2):2–6. http://CRAN.R-project.org/doc/Rnews/Rnews_2006-2.pdf. (Cited on pages 85, 91, 94, 98, 101, 105, 107, 110, 115, 118, 119, 128 and 135)

Rutter, M. (2022). Index of /bin/linux/ubuntu. https://cran.at.r-project.org/bin/linux/ubuntu/ [Last access 2022 Oct 08]. (Cited on page 3)

Ruxton, G. D. (2006). The unequal variance t-test is an underused alternative to Student's t-test and the Mann–Whitney U test. *Behavioral Ecology*, 17(4):688–690. (Cited on page 223)

Schoder, V., Himmelmann, A., and Wilhelm, K. P. (2006). Preliminary testing for normality: some statistical aspects of a common concept. *Clin. Exp. Dermatol.*, 31(6):757–761. (Cited on page 243)

Sedgwick, P. (2013). P values or confidence intervals? *BMJ*, 346. (Cited on page 200)

Shim, E., Mizumoto, K., Choi, W., and Chowell, G. (2020). Estimating the risk of covid-19 death during the course of the outbreak in korea, february-march, 2020. *medRxiv*. (Cited on page 119)

Sonabend, R. and Kiraly, F. (2022). *distr6: The Complete R6 Probability Distributions Interface*. https://alan-turing-institute.github.io/distr6/, https://github.com/alan-turing-institute/distr6/. (Cited on pages 86, 95, 98, 101, 106, 108, 111, 112, 114 and 116)

Sterne, J. A. and Davey Smith, G. (2001). Sifting the evidence-what's wrong with significance tests? *BMJ*, 322(7280):226–231. (Cited on page 200)

Steuer, D. (2022). Index of /bin/linux/suse. https://cran.at.r-project.org/bin/linux/suse/ [Last access 2022 Oct 08]. (Cited on page 3)

The R Foundation (2022a). Contributors. https://www.r-project.org/contributors.html [Last access 2022 Oct 08]. (Cited on page 2)

The R Foundation (2022b). The R Foundation. https://www.r-project.org/foundation/ [Last access 2022 Oct 08]. (Cited on page 2)

Úcar, I. (2022). Fedora Packages of R Software. https://cran.at.r-project.org/bin/linux/fedora/ [Last access 2022 Oct 08]. (Cited on page 3)

Venables, W. N. and Ripley, B. D. (2002). *Modern Applied Statistics with S*. Springer, New York, fourth edition. ISBN 0-387-95457-0. (Cited on pages 122, 127, 128, 142, 156 and 246)

Wald, A. (1980). *A method of estimating plane vulnerability based on damage of survivors.* Center for Naval Analyses, crc 432 edition. (Cited on page 10)

Wasserstein, R. L. and Lazar, N. A. (2016). The asa statement on p-values: Context, process, and purpose. *The American Statistician*, 70(2):129–133. (Cited on page 200)

WHO (2015a). Country and regional data on diabetes – WHO European Region. http://www.who.int/diabetes/facts/world_figures/en/index4.html [Last access 29.08.2015]. (Cited on page 93)

WHO (2015b). Diabetes. http://www.who.int/mediacentre/factsheets/fs312/en/ [Last access 29.08.2015]. (Cited on pages 88 and 96)

Wickham, H. (2009). *ggplot2: Elegant graphics for data analysis*. Springer New York. http://had.co.nz/ggplot2/book. (Cited on pages 18, 19, 22, 28, 34, 37, 40, 49, 52, 54, 57, 63, 64, 71, 79, 81, 83, 84, 130, 131, 133, 134, 136, 222, 225 and 231)

Wickham, H. and Seidel, D. (2022). *scales: Scale Functions for Visualization*. R package version 1.2.1. (Cited on pages 18 and 22)

Wikipedia (2015). Körpergröße — Wikipedia, Die freie Enzyklopädie. https://de.wikipedia.org/w/index.php?title=K%C3%B6rpergr%C3%B6%C3%9Fe&oldid=145235809 [Last access 30.08.2015]. (Cited on page 104)

Wikipedia contributors (2021). Saps ii — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=SAPS_II&oldid=1047694912 [last access 1-November-2022]. (Cited on page 17)

Wikipedia contributors (2022a). Andora — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Andora&oldid=1113739296 [last access 1-November-2022]. (Cited on page 93)

Wikipedia contributors (2022b). Bilirubin — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Bilirubin&oldid=1116981450 [last access 1-November-2022]. (Cited on page 17)

Wikipedia contributors (2022c). Finland — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Finland&oldid=1118943471 [last access 1-November-2022]. (Cited on page 99)

Wikipedia contributors (2022d). Intelligence quotient — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Intelligence_quotient&oldid=1119126577 [last access 1-November-2022]. (Cited on page 105)

Wikipedia contributors (2022e). Markdown — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Markdown&oldid=1114072172 [Online; accessed 11-October-2022]. (Cited on page 5)

Wikipedia contributors (2022f). Reference range — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Reference_range&oldid=1119241635 [last access 1-November-2022]. (Cited on page 107)

Xiao, N. (2018). *ggsci: Scientific Journal and Sci-Fi Themed Color Palettes for 'ggplot2'*. R package version 2.9. (Cited on pages 72, 83, 223 and 231)

Xie, Y. (2013). animation: An R package for creating animations and demonstrating statistical methods. *Journal of Statistical Software*, 53(1):1–27. (Cited on page 76)

Xie, Y. (2015a). *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition. ISBN 978-1498716963. (Cited on pages v and 6)

Xie, Y. (2015b). *knitr: A general-purpose package for dynamic report generation in R*. R package version 1.10.5, http://CRAN.R-project.org/package=knitr. (Cited on page vi)

Xie, Y., Allaire, J., and Grolemund, G. (2018). *R Markdown: The Definitive Guide*. Chapman and Hall/CRC, Boca Raton, Florida. ISBN 9781138359338. (Cited on page 6)

Youden, W. J. (1950). Index for rating diagnostic tests. *Cancer*, 3:32–35. (Cited on page 147)

Zeileis, A., Hornik, K., and Murrell, P. (2009). Escaping RGBland: Selecting Colors for Statistical Graphics. *Computational Statistics & Data Analysis*, 53:3259–3270. (Cited on page 66)

Index

Index of R Objects